Dynamic Flexible Queueing Network Models for the Design and Control of High Performance Operational Systems

by

Hoda Parvin

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Industrial and Operations Engineering)
in The University of Michigan
2012

Doctoral Committee:

Associate Professor Mark P. Van Oyen, Chair
Professor Hilbrand E. Romeijn
Associate Professor Hyun-Soo Ahn
Associate Professor Amy M. Cohn

ACKNOWLEDGEMENTS

I would like to use these few pages to thank all the people who helped me during my time in graduate school.

First, I would like to thank Professor Van Oyen for his support and for allowing me the opportunity to work with him and learn from him. I cannot imagine going through this process without his thoughtful support, encouragement, and kindness. In addition to all I learned from him academically, Professor Van Oyen reminded me to value people and to always keep others' interests in mind. These were valuable lessons and I will hold on to them for the rest of my life.

I would also like to thank my teachers for making me see the world differently and equipping me with the knowledge I need to be successful in my career. Many thanks to Professors Goker, Epelman, Babich, Chao, Tilman, Smith, Babayan, and Windfuhr. Many thanks to Professor Cohn for teaching me linear programming and advanced mathematical modeling. I should say that after taking linear programming course for several time, thanks to Professor Cohn's amazing teaching skills, I developed a comprehensive understanding of the subject. I also should thank Professor Cohn for being a supportive member of my PhD committee; I learned a lot from her. I also greatly appreciate the help and support I received from my dissertation committee, Professor Romeijn and Professor Ahn. I appreciate Professor Ahn's great input and help with Chapter II, the most challenging chapter of my dissertation.

Thank you to Professor Dimitris Pandelis for his encouragement and input on

Chapter III. It would have not been possible without his extensive knowledge of stochastic processes and his passion for teaching. Even though I never got a chance to meet him in person, I hope to do so in near future. I learned so much from him. He is truly a great teacher and an outstanding researcher.

I should thank Professor Tenektzis for being such an inspiring role model. Learning from him in his class as well as outside the classroom was one the most valuable learning experiences during my time in Ann Arbor. He and his wife, Barbara, are truly among the best people I have known in my life.

I appreciate Professor Larry Seiford and Professor Daskin for the important role they had as department chairs throughout my graduate studies in Michigan.

I am grateful for all the assistance I received from the IOE department's staff since I started as a PhD student in Michigan. Special thanks to Tina Blay for always supporting me. Also thanks to Mary Winter, Candy Ellis, Gwen Brown, Matt Irelan, and Wanda Dobberstein for helping me with the administrative work. Many thanks to Chris Konrad and Rod Capps for providing all kinds of hardware and software related help. I should thank Mint for not only always being available to help me with all possible hardware issue but for being a true friend. His help and support throughout my time in Ann Arbor was invaluable.

I believe friendship is the most important pillar of life. I feel so blessed and fortunate for having great friends who always supported me and helped me achieve my goals. Many thanks to Maryam Bahar, Mohsen Sharifani, Hamed Shafeian, Sara Sarkhili, Hazhir Rahmandad, Elnaz Jalilipour, Mandis Beigi, Maryam Abolfazli, and Jennie Lane for their unlimited kindness and support. Thank to Eiman Haj Abasi for being a supportive and great friend and also for her much-needed help in editing Chapter II of my dissertation. There are many friends in Ann Arbor whom

I had the opportunity of spending time with. They truly made my time in graduate school an amazing adventure. Many thanks to Arleigh Waring, Katrina Appell, Ada Barlatte, Stanko Dimitrov, Tara Terry, Warren Sutton, Ashkan Malek, Fang Dong, Pete Larson, and Marcial Lapp. I should thank Matthew Philson for his great help in implementing the ideas introduced in Chapter V. Many thanks to Sharon and Bob Ongaro for being the greatest hosts on earth for allowing me to stay with them when I had to commute between Washington DC and Ann Arbor. I appreciate Jonathan Helm's support and friendship and specially his valuable collaboration with me on Chapter IV. I would like to thank Dr. Damon Williams for sharing the initial results of his dissertation research with me and helping me develop Chapter III as an extension of his research.

I am deeply grateful for the support and encouragement of Shervin Ahmadbeygi. Being PhD student is not easy; I would not have been able to be successful without endless support of my best friend and my husband, Shervin. Every day with him is a new beginning.

I am also grateful of my parents, Faranak Hamzeh and Mohammadnabi Parvin who have played an inspiring role throughout my life. I owe all my achievements to them. I should thank my brother Amin Parvin for his encouragement. Without him life would have been missing something seriously. I should acknowledge the important role of my grandparents Gity Bazyaran and Ghasem Hamzeh for their kindness and support from the day one of my life till now. They are the true meaning of happiness in my life.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

ABSTRACT

In this dissertation, we study the impact of efficient resource allocation policies on the performance of a variety of systems including service centers, manufacturing systems, and pharmaceutical distribution centers.

In Chapter II, we investigate the optimal server scheduling policy in service industries such as call centers and off-line information technology service centers. We model the system as a Markov Decision Process and analytically characterize the optimal server allocation and scheduling policy. We also propose an efficient heuristic to improve server scheduling with no need to solve the MDP formulation. Our computational results confirm the effectiveness of our heuristic as compared to other well-studied routing algorithms from the literature.

In Chapter III, we propose a new production line design framework for a U-shaped production system consisting of several stations and cross-trained workers. We address efficient line design principles to enhance the system's throughput while keeping the number of required skills per worker significantly lower. We design an extensive test suite and use simulation to show that the system we designed can achieve nearly the same level of throughput as a fully cross-trained system.

In Chapter IV, we present two-stage and three-stage stochastic network flow formulations to address the problem of deploying disease treatment in developing countries when the demand is uncertain. We find efficient distribution strategies that improve access to treatments at a minimum cost. We use demand data on the

facility-based malaria treatment distribution provided by the Malawian Ministry of Health. We show that the proposed stochastic approaches can effectively reduce shortages and lower transportation costs.

Finally in Chapter V, we address the problem of routing incoming calls in an information technology service center with cross-trained servers and heterogeneous demand. Another important characteristic of the studied model is the fixed task completion deadline associated with each incoming service request. If the deadline is not met, a relatively large deadline violation penalty will be charged. To design and implement the proposed routing algorithm, we use real data from an industrial research partner. The simulation results confirm the effectiveness of the proposed routing heuristic in improving customer satisfaction by avoiding deadline violation.

CHAPTER I

Introduction

In this dissertation, we study problems drawn from a large domain of optimization problems addressing the complex issue of resource allocation under uncertainty. The resources in a manufacturing environment could be workers and the source of uncertainty could be the manufacturing process variability that causes randomness in processing times. In a service environment, the resources are human agents who respond to incoming requests from customers, and the service process variability (often driven in part by the unique needs of the customer) causes randomness in processing times. Arrivals typically follow a random pattern and also the time that requests spend in the system is often uncertain. In our fixed task zone chaining model of production systems, the models are closed queueing network models for which the random arrival times are driven by the system design and operating policies as well as intrinsic process time randomness. In a distribution network such as the one in Chapter IV, resources are the commodities that are transported and temporal and spatial demand fluctuations are among the key uncertainties to be modeled.

The methodologies investigated in this dissertation include: simulation, statistical analysis, Markov Decision Processes, dynamic programming, and stochastic programming. The studied application domains include call centers, service industries,

manufacturing systems, and medication distribution centers.

In Chapter II, we consider the dynamic scheduling of cross-trained servers to multiple classes of customers, who are impatient while waiting in finite queues or being served. We model the scheduling problem as an "N" network with two classes of customers and two servers: one dedicated server and one cross-trained server. Since each queue has a finite buffer size, a new arrival will be blocked if the queue is full, incurring a rejection penalty. When a customer waits longer than she is willing to, she might renege and generate a reneging cost. The system also incurs a class-specific holding cost per unit time for each customer in the system. We provide sufficient conditions under which static priority policies are optimal. We also show, however, that the optimal policy can in general be complected and lacks a monotone switching curve. For these cases, we develop a dynamic allocation heuristic called the comparative expected reward rate index (CERRI) that effectively incorporates the primary drivers of cost. A performance comparison to the optimal policy and two well-studied server allocation policies from the literature show that our heuristic performs well and is robust to parameter changes.

In Chapter III, we study the optimal zone design for a flexible assembly line with limited work in progress level. The new paradigm of Fixed Task Zone Chain (FTZC) as a special type of zone-based cross-training was introduced by Williams [93]. Based on his initial results we develop a heuristic dynamic control policy to maximize the line's throughput given a zone structure. We also prove several useful properties of the optimal policy and extrapolate from them to devise a heuristic control policy that yields high throughput. The performance of a FTZC system is contingent upon the choice of zone structure; therefore, we devise the zone assignment (ZonA) algorithm to design the zone structure to achieve high throughput levels. We then

derive sufficient conditions that guarantee that the line is balanceable through ZonA. Benchmarking over a test suite supports the effectiveness of our proposed heuristic worker control policy as well as the ZonA algorithm, and we compare it to the performance of other paradigms.

In Chapter IV, we address the problem of distributing drugs to treat malaria through the centralized, multi-tiered health system model of the Malawian Ministry of Health, given temporal and spatial uncertainty in malaria infection and demand for treatment services. Malaria poses a serious threat to society in many developing world countries. The efficient and effective distribution of malaria treatments is a key challenge in resource constrained countries such as Malawi. Chapter IV develops a practical solution that integrates strategic level and operational level models to better manage treatment distribution through the centralized, multi-tiered health system model of the Malawian Ministry of Health. At the strategic level, we develop a two-stage stochastic programming approach to address the problem of demand uncertainty. In the first stage, before the malaria season starts, an initial round of shipments of Artemesinin Combination Threrapies (ACTs) are sent to each local clinic (third tier) from district hospitals (second tier), which receive medications from regional warehouses (first tier). When the malaria season begins and demand is realized, a recourse action is triggered. We analyze two different implementations: (1) a transshipment model, in which clinics facing a shortage can receive ACTs from clinics that have supply surpluses, and (2) a delayed shipment model, in which a safety inventory is stocked at the district hospitals to resupply individual clinics. The first strategic model indicates that the optimal policy involves the creation of *clinic clusters* with exclusive transshipment policies. This insight enables us to reconstruct the problem at the operational level, solving each clinic cluster independently, where

the clusters are identified using the original stochastic programming models. We then solve the operational problem using Markov decision process (MDP) approach to determine optimal periodic transshipment policies. Finally, we demonstrate a potential for a 16% reduction of shortage cost by using our proposed distribution system based on a case study using historical data from 290 facilities under the control of the Malawi Ministry of Health.

In Chapter V, we present a simulation-based approach to study alternative dynamic assignment policies in an information technology (IT) service delivery environment. Our overarching goal is to find the most cost-effective assignment of service requests to cross-trained agents in a large-scale network. We present a novel heuristic algorithm that assigns an analytically described allocation index to each service request that has arrived. It incorporates factors such as variability in agents capabilities, uncertainty in request inter-arrival times, and complex service level agreements (SLA). We investigate the effectiveness of our proposed assignment algorithm using real world data from an IT service environment on a small problem instance. We discuss how the results of this simulation can help improve the terms of service level contracts as well as agent training programs.

CHAPTER II

Control Policies for the "N" Network with Impatient Customers and Finite Buffers

2.1 Introduction

In many service and make-to-order (MTO) manufacturing environments, the ability to have adequate capacity is critical to avoid delays and lost demand. Many of these systems, however, operate with finite capacity that cannot be changed as quickly as demand changes, making adaptive resource allocation (i.e., doing the right job with the right resource at the right time) more critical. One way to efficiently utilize the resources is to increase operational flexibility through cross-training of workers/agents.

Cross-training has the potential to help quickly respond to the needs from a variety of consumers without adding more workers, but can only be achieved by effectively allocating the right resource at the right time (dynamic resource allocation). Furthermore, because of the high cost of training and maintaining a flexible workforce, many firms use a combination of cross-trained and traditional/dedicated servers (possessing narrow skill sets). In such environments, using more expensive flexible resources efficiently is critical to reducing the system's operating costs.

The "N" network structure has been an excellent research model for gaining fundamental insights for flexible workforce (see Bell and Williams [18], Harrison [44],

Bell and Williams[19], and Ahn et al. [2]). A typical "N" network model (including ours) assumes one of the two servers, assumes server 1 is a specialist and can only serve jobs/customers in queue 1 (type-1 jobs) and the other server, server 2, is a generalist and can serve either in queue 1 or queue 2. There are two job types: a type-1 job is a "shared" task which can be processed by both servers and a type-2 job is a "fixed" task which can only be processed by the cross-trained server.

We assume that buffers for both queues are finite. Consequently, after filling a buffer of size (N_i), a new type i arrival to the system is blocked. We assume that jobs that are waiting in the system are impatient; jobs can leave the system before service completion. We allow heterogeneous jobs; hence costs for waiting, blocking, and reneging all depend on the job type.

A good example of a system with flexible workforce, finite buffer, and reneging customers is a MTO system production with "general" and "customized" products. In such production processes customers can cancel their orders if not served or they may find another supplier. In this case, the cost associated with reneging represents the lost revenue plus the loss of goodwill. In the MTO context, the finite order buffer can model the phenomenon in which a firm stops taking orders (or turns away customers) if there are too many jobs in the system to avoid the loss of goodwill for failure to fulfill the orders in a timely manner.

Another example of systems where reneging and blocking occur in a system with flexible workforce is IT service centers where agents perform a variety of tasks, including remote monitoring and management of hardware and software, developing new applications, applying security patches, etc. At these IT service centers, a dispatcher first categorizes the service requests based on their type and assigns them to job-specific queues. Servers (also called "agents" to avoid confusion with "computer

servers") pick the requests from these queues and begin to address them. IT service centers hire and train agents with a variety of skills. While some agents are more narrowly trained to resolve high-volume, low complexity problems, other agents can have a wider set of skills. These cross-trained agents not only resolve low complexity problems, but they can also handle more complex service requests. IT service delivery centers often need to deal with the issue of impatient customers, because customers can simultaneously attempt to resolve the problems themselves and/or hire a competitor in an effort to expedite the service. This will impose direct and indirect reneging costs on the system, including the loss of revenue and goodwill as well as the penalty imposed by the service agreement.

Despite its deceptively simple appearance, our model is complex because of the cost structure and the fact that has an intermediate amount of cross-training, half way between fully dedicated and fully cross-trained systems. To achieve full benefits in our model, efficient job-to-server assignment policies must be employed.

An effective policy will judiciously apply efforts of the generalist to the type-1 task of the inflexible (specialist) worker so as to help without inducing starvation of the specialist. By investigating the structure of the optimal assignment policy in an "N" network, we provide sufficient conditions under which a static priority policy is optimal. However, we illustrate that those sufficient conditions are not always met. Therefore, static priority rules may not necessarily be effective in general. Using insights derived from the system dynamics and from numerical experimentation, we then develop a heuristic algorithm for general cases and test it in an extensive numerical study to establish its effectiveness.

This chapter is organized as follows: The remainder of Section 2.1 surveys the literature. A Markov Decision Process (MDP) formulation of the dynamic job-to-

server allocation problem is presented in Section 2.3, followed by a discussion of the structure of the optimal allocation policy. To gain insight into the characteristics of the optimal policy, we present numerical results for a limited test suite in Subsection 2.3.3. In Section 2.4 we present sufficient conditions that guarantee the optimality of a strict priority policy. For other cases, where the sufficient conditions are not satisfied, finding an optimal dynamic allocation of jobs to servers may not be practical in real time. Therefore, we present a heuristic, index-based dynamic allocation algorithm in Section 2.5. The heuristic algorithm is described as an analytical function of the basic model parameters. It also clearly presents intuition so that the dispatcher can gain better insight into the dynamic assignment procedure. Final remarks and future research are explained in Section 2.6.

2.2 Literature Survey

The research in this chapter is related to two streams of research: allocation of flexible servers and reneging of impatient customers and/or blocking arising from finite capacity.

Flexibility and Cross-training

A number of papers have shown that effective control policies are critical to improving system performance and realizing the full value of flexible resources. These papers include Iravani et al. [52], Iravani et al. [51], and Gurumurthi and Benjaafar [43]. Buyukkoc et al. [24] and Warland [89] show that in a simple cross-training framework with one server and several problem types, the $c\mu$ rule can be optimal. Van Oyen et al. [86], Brusco et al. [23], Van Mieghem [85], Sennott et al. [79], and Ahn et al. [1] consider optimal allocation of a flexible workforce on more complex network topologies.

Cross-training often increases the complexity of the system as well as training costs. Hopp et al. [48], McClain et al. [66], Jordan and Graves [54], and Parvin et al. [71] evaluate the performance of the system when a limited degree of flexibility is added. They all support the notion that a modest level of flexibility can significantly improve the system performance, suggesting that limited flexibility is a good design that reaps the benefits of flexibility while keeping the training costs (or other overhead costs) low.

Our model is a special case of "N" network, studied in a number of papers. Harrison [44] discusses how $c\mu$ results can perform poorly in an "N" network when service times are deterministic. Bell and Williams [18] discuss the optimality of a threshold policy for an "N" network in heavy traffic with continuous review. Ahn et al. [2] study an "N" network model with no external arrivals. They characterize different forms of optimal control policies and the necessary and sufficient conditions resulting in optimality of the policies. Veatch [87] argues for the optimality of $c\mu$ rule in an "N" network with preemption. However, under more complex models such as finite buffer capacity or customer impatience, the $c\mu$ rule may not necessarily perform optimally and this rule does not hold in every cross-trained setting. For instance, the model of Down and Lewis [31] is an "N" network with upgrades, and they prove the optimality of a threshold policy.

For an "N" network with a more complex model, we argue that, unlike the above-mentioned cases, the optimal policy may not be easy to characterize. Our research is different from the existing literature since we include several other realistic features such as reneging, blocking, and finite buffers in our model. We should also mention that while most of these papers focus on partial characterization of optimal (or asymptotically optimal) policies, our research focuses on finding sufficient conditions

when a static rule is optimal as well as developing a reasonable heuristic for all other cases.

Reneging and Blocking

Customer abandonment has been studied in the context of queueing networks. The reneging phenomenon is an important part of many real world systems. Gayon et al. [39] studies reneging to address lead time quotation by customers in inventory models. Garnet et al. [37] study the same phenomenon in call centers. Several papers studied the issue of reneging within the context of inflexible systems and showed that even for rather simple models, reneging still imposes difficulty in analyzing the system performance measures. See Baccelli et al. [11], Bae et al. [12], Finch [34], Gavish and Schweitzer [38], Jennings and Reed [53], Panwar et al. [70], Stanford [82], and Barrer [14] for more information on the reneging effect in single server systems. Boots and Tijms [21] study the effect of reneging on system performance measures in multiple-server queueing systems. Kim et al. [56] characterize the optimal assignment policy in a production system with homogenous demand. In their model, demand is realized in a multistage process and customers may leave the system at each given stage. Down et al. [30] study the optimality of $c\mu$-type policies in a system consisting of one flexible server and two job types with linear renege rates. Note that the models proposed by Kim et al. [56] and Down et al. [30] assume infinite buffer capacity and consider only one server. Our model addresses a system with two servers and finite buffer capacity.

Ghamami and Ward [41] model an "N" network structure with holding costs and reneging penalties under heavy traffic. Garnet et al. [37] develop an asymptotic approach to designing large call centers when customers' impatience is exponentially distributed. Zohar et al. [99] study the dynamic nature of customers' impatience in

both online customer contact centers and conventional telephone call centers. These aforementioned papers only focus on systems with heavy traffic. In such settings, asymptotic-based approximate approaches work well. Our research, however, differs from the above work as it develops an approach applicable to low, medium, and high traffic systems and more complex cost structures.

Finite buffer capacity often complicates a queueing network model. [45] study the design of tandem queues with a finite buffer. Andradóttir et al. [5] study flexible servers in a finite buffer system with the objective of maximizing throughput. They also address an extension of this research problem where the flexible server can be subject to failure (see Andradóttir et al. [8]). Andradóttir and Ayhan [4] study the problem of an intermediate finite buffer with several flexible servers in a tandem queue with two stations. They characterize the optimal policy to maximize the long run average throughput. Kim and Van Oyen [55] develop a heuristic approach for allocating a flexible server to two job types with a finite queue capacity. See Perros [72] for an extensive literature study of the queueing systems with finite buffer capacities.

To the best of our knowledge, there is no existing work that addresses the optimal allocation in "N" network structure with heterogeneous demand, holding costs, impatient customers, and finite buffer capacities.

2.3 Model

We consider an "N" network model with a dedicated server (server 1) and a cross-trained server (server 2). Jobs are categorized into two types. Type-1 jobs are *shared* task that can be assigned to both servers; and type-2 jobs are termed *fixed* and can only be assigned to the cross-trained server (server 2). We assume that

interarrival times and service times are exponentially distributed. Type-i jobs arrive at the system with the rate of λ_i'. The dedicated server serves type-1 jobs at the rate of μ_{11}'. The cross-trained server can serve both type-1 and type-2 jobs at the rate of μ_{21}' and μ_{22}' respectively. Type-i jobs waiting in the queue incur holding costs h_i' per unit time. We assume a finite buffer (N_i) for each job type. At the event of an arrival to a full buffer, a blocking penalty, b_i for the rejected type-i job is charged. Each job may renege from the system (independently) with a type-dependent rate of r_i'. With x_i type-i jobs in the system, a reneging event of type-i jobs happens with the rate of $r_i'x_i$. At the event of a type-i job reneging from the system, a lump sum penalty of π_i is incurred.

Jobs can be *preempted* at any time. Our proposed model allows *collaboration* between servers, i.e. they can team up to serve a type-1 job more quickly.

Figure 2.1 illustrates the "N" network representation of this system. It is called an "N" network because it resembles a rotated letter "N" (see Figure 2.1). Here we should note that throughout this chapter, terms such as service requests, orders, jobs and customers are often used interchangeably to refer to "arrivals" to the queueing network. Similarly, servers, agents and workers are equivalently used to refer to the classical notion of servers in a queueing model. Next, the notation is presented in Table 5.1.

2.3.1 MDP Formulation

We first model the problem as a continuous-time stochastic control problem. The vector $\mathbf{x}(t) = (\mathbf{x}_1(t), \mathbf{x}_2(t))$, represents the state of the system at time t and contains the number of type-1 and type-2 jobs, respectively. Since our model is a controlled continuous time Markov chain and therefore the future is conditionally independent of the past given the current state. Thus, without loss of generality we only focus

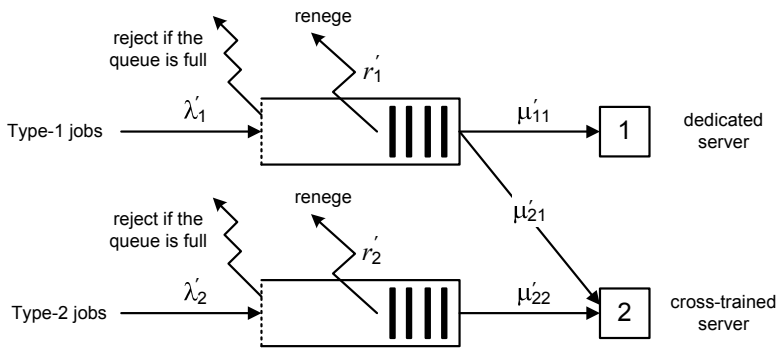


Figure 2.1: "N" network model with one fully cross-trained server.

i	Index for job types, $i \in \{1,2\}$
j	Index for servers, $j \in \{1,2\}$
N_i	Maximum buffer size of type-i jobs in the system, $N_i \in \mathbb{N}$, and $i \in \{1,2\}$
μ'_{ji}	Service rate of server j of type-i job, $\mu_{ji} \in \mathbb{R}^+$, and $j,i \in \{1,2\}$
a_{ji}	An indicator function for server j's skill set, $a_{ji} = 1$ if server j is capable of solving type-j job and zero otherwise, $j,i \in \{1,2\}$
λ'_i	The arrival rate of type-i job, $i \in \{1,2\}$
r'_i	The renege rate of type-i job, $i \in \{1,2\}$
π_i	Lump sum cost of type-i job that reneges; $\pi'_i \geq 0 \quad \forall i \in \{1,2\}$
h'_i	Linear holding cost per unit time associated with a type-i job; $h'_i \geq 0 \quad \forall i \in \{1,2\}$
b_i	Lump sum blocking cost of type-i; $b'_i \geq 0 \quad \forall i \in \{1,2\}$
β	Discount rate in continuous-time
$R_i(t)$	Cumulative number of type-i jobs reneged in interval $(0,t]$, $i \in \{1,2\}$
$\Lambda_i(t)$	Cumulative number of type-i jobs rejected/blocked in interval $(0,t]$, $i \in \{1,2\}$.

Table 2.1: Notation.

on Markovian policies (see Ross [76]). A policy is Markovian if it depends only on the current state of the system, $(\mathbf{x}_1(t), \mathbf{x}_2(t))$. Then $J_g(\mathbf{x}(0))$, the *expected total β-discounted cost* of the system with initial state $x(0)$ is:

$$J_g(\mathbf{x}(0)) = \lim_{T\to\infty} E\Big\{ \int_0^T e^{(-\beta t)} \sum_{i=1}^{2} \Big((h'_i x_i(t) + \pi_i dR_i(t) + b_i d\Lambda_i(t)) \Big) dt \Big\}. \tag{2.1}$$

To conveniently capture the dynamics of the system, we discretize the model based on the uniformization technique introduced by Lippman [63]. The uniformization rate, ψ, is the maximum transition rate across all the states:

$$\psi = \sum_{i=1}^{2} (\lambda'_i + N_i r'_i + \mu'_{2i}) + \mu'_{11}.$$

We then define τ as an exponential random variable with mean $\frac{1}{\psi}$ corresponding to the length of one period in the discrete Markov chain. Let α be the discrete-time discount factor corresponding to β:

$$\alpha = E(e^{-\tau\beta}) = \int_0^\infty e^{-\beta y}\psi e^{-\psi y}dy = \frac{\psi}{\beta+\psi}.$$

We define the discrete-time state vector as (x_1, x_2), denoting the numbers of type-1 and type-2 jobs in the system. We calculate the holding cost per period as:

$$h_i = \frac{h_i'}{\beta+\psi}. \qquad\qquad i = 1, 2$$

Let $\lambda_i = \frac{\lambda_i'}{\psi}$, $r_i = \frac{r_i'}{\psi}$, and $\mu_{ji} = \frac{\mu_{ji}'}{\psi}$ for all $i, j \in \{1, 2\}$. Let $V_n(\mathbf{x})$ denote the *value function* for n-period problem with terminal value $V_0(\mathbf{x}) = 0$ for all $\mathbf{x}$. To simplify the notation, we introduce two *operators*. $A_i\mathbf{x}$ denotes the *arrival operator* and $D_i\mathbf{x}$ denotes the *departure operator*, both of which are defined as follows:

$$A_1\mathbf{x} = (x_1 + 1 \wedge N_1, x_2), \qquad A_2\mathbf{x} = (x_1, x_2 + 1 \wedge N_2).$$

$$D_1\mathbf{x} = (x_1 - 1 \vee 0, x_2), \qquad D_2\mathbf{x} = (x_1, x_2 - 1 \vee 0).$$

In the above notation, $a \wedge b = min\{a, b\}$ and $a \vee b = max\{a, b\}$. Also $D_i^2\mathbf{x}$ is defined as $D_iD_i\mathbf{x}$, and $A_i^2\mathbf{x}$ defined similarly. Let u_{ji} be an indicator function representing the assignment of server j to a type-i job. We do not allow one server working on two different jobs simultaneously; however, type-1 jobs can be served collaboratively by both servers. To be specific, $U(\mathbf{x})$ is the set of feasible actions in state $\mathbf{x}$:

$$U(\mathbf{x}) = \{u_{ji} : u_{ji} \le a_{ji}, \sum_{i=1}^{2} u_{ji} \le 1, \forall i, j\}.$$

Then, $V_{n+1}(\mathbf{x})$ is recursively defined in the following optimality equation:

(2.2)
$$V_{n+1}(\mathbf{x}) = \sum_{i=1}^{2} h_i x_i + \alpha \left\{ \begin{array}{l} \sum_{i=1}^{2} [\lambda_i V_n(A_i\mathbf{x}) + r_i x_i V_n(D_i\mathbf{x}) + \pi_i r_i x_i + \lambda_i b_i I_{\{x_i=N_i\}}] \\ + \min_{\mathbf{u}\in U(\mathbf{x})} \{\sum_{i=1}^{2} \sum_{j=1}^{2} u_{ji}\mu_{ji}(V_n(D_i\mathbf{x}) - V_n(\mathbf{x}))\} \\ +(1 - \sum_{i=1}^{2}(\lambda_i + r_i x_i))V_n(\mathbf{x}) \end{array} \right\}.$$

In Equation (2.2), the terms in the first line in the braces represent the one-stage cost associated with state $\mathbf{x}$ (including holding costs, reneging penalties, and blocking penalties) and the terms associated with transitions to the next stage due to a new arrival of type-i job with probability λ_i and the renege of a type-i job with probability $r_i x_i$. The second and third lines include terms associated with the service completion and self-loop terms due to uniformization. We define an operator Δ_i as the difference in value function between states $A_i x$ and x, that is,

(2.3)
$$\Delta_i V_n(\mathbf{x}) = V_n(A_i\mathbf{x}) - V_n(\mathbf{x}) \quad \forall i, n.$$

Using Equation (2.3) and converting minimization into maximization, the value function in Equation (2.2) is written as:

(2.4)
$$V_{n+1}(\mathbf{x}) = \sum_{i=1}^{2} h_i x_i + \alpha \left\{ \begin{array}{l} \sum_{i=1}^{2} [\lambda_i V_n(A_i\mathbf{x}) + r_i x_i V_n(D_i\mathbf{x}) + \pi_i r_i x_i + \lambda_i b_i I_{\{x_i=N_i\}}] \\ - \max_{\mathbf{u}\in U(\mathbf{x})} \{\sum_{i=1}^{2} \sum_{j=1}^{2} u_{ji}\mu_{ji}\Delta_i V_n(D_i\mathbf{x})\} \\ +(1 - \sum_{i=1}^{2}(\lambda_i + r_i x_i))V_n(\mathbf{x}) \end{array} \right\}.$$

2.3.2 Structure of an Optimal Policy

To investigate and characterize optimal assignment policies over a class of static polices. We first analyze the finite horizon model, and then extend the results to the

infinite horizon model by applying a convergence result from Sennott [78], Proposition 4.3.1, given that all the costs are positive and finite and the state is finite as well.

The next theorem shows that an optimal policy is non-idling and the value function is in the increasing queue lengths. The proof uses induction and a sample path argument (see Section 2.7.1).

Theorem II.1. *In this system, we have: (P0) the optimal policy is a non-idling policy and (P1)* $\Delta_i V_n(\mathbf{x}) \geq 0$ *for all* $\mathbf{x}$*,* n*, and* $i \in \{1, 2\}$*.*

As a direct result of this theorem, the original value function presented in (2.2) can be simplified by eliminating all of the idling actions to:

(2.5)

$$V_{n+1}(\mathbf{x}) = \sum_{i=1}^{2} h_i x_i + \alpha \left\{ \begin{array}{l} \sum_{i=1}^{2} [\lambda_i V_n(A_i\mathbf{x}) + r_i x_i V_n(D_i\mathbf{x}) + \pi_i r_i x_i + \lambda_i b_i I_{\{x_i = N_i\}}] \\ - \max\{\mu_{21}\Delta_1 V_n(D_1\mathbf{x}), \mu_{22}\Delta_2 V_n(D_2\mathbf{x})\} \\ -\mu_{11}\Delta_1 V_n(D_1\mathbf{x}) + (1 - \sum_{i=1}^{2}(\lambda_i + r_i x_i))V_n(\mathbf{x}) \end{array} \right\}.$$

2.3.3 Complexity of Optimal Policies

Illustrative numerical examples can provide insight into the structure of an optimal policy. As shown by Ahn et al. [2], the optimal job-to-server assignment policy, even in the absence of reneging and finite queue capacity assumptions, will not have a simple structure in general. We gain insight into conditions under which we obtain a specific form of optimal policies (i.e. strict priority, threshold, etc) by numerically solving the MDP model for different parameters. Table 2.2 illustrates the parameter settings used for each numerical experiment. Note that in all cases, we set $\lambda_1 = 1$, $\lambda_2 = 1$, $\mu_{22} = 0.5$, and $N_1 = N_2 = 20$ (buffer capacity).

Numerical example	μ		r		h		π		b		Optimal policy
	μ_{11}	μ_{21}	r_1	r_2	h_1	h_2	π_1	π_2	b_1	b_2	
1	0.5	0.5	1	0.4	3	4	5	10	2	6	Strict priority (1)
2	0.5	0.5	0.75	2	3	3	16	5	4	2	Strict priority (2)
3	1	0.5	0.5	0.05	1.5	1	1	1	50	50	Switching curve
4	1	1	1	1	1	1	1	1	1	50	No pattern

Table 2.2: Parameters used for the illustrative examples.

We implemented the MDP value iteration algorithm in C++ and used a convergence criterion of 10^{-6}. The optimal action plots are presented in Figure 2.2. In each, the horizontal (vertical) axis represents the number of type-1(2) jobs in the system. An optimal action (represented by dark gray shading) assigns server 2 (the cross-trained server) to jobs of type-1 (the shared task). Similarly, the feasible action assigns server 2 to type-2 jobs (the fixed task) is denoted by a light gray shading.

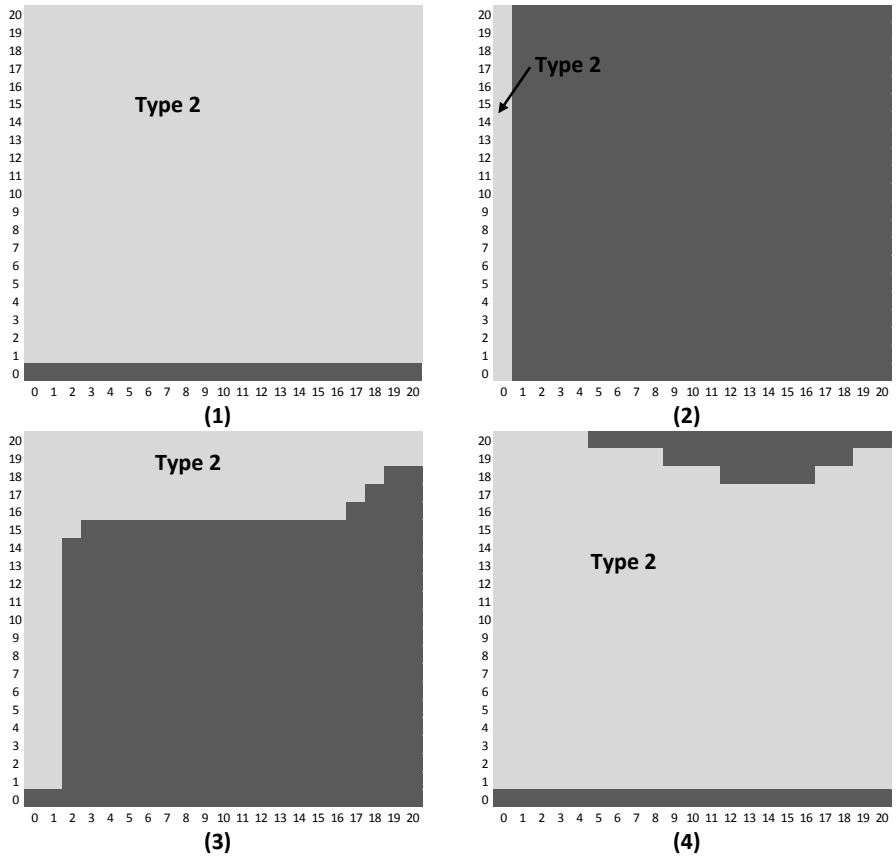


Figure 2.2: Optimal action plots associated with each numerical example; x_1 (x_2) is on the horizontal (vertical) axis.

In Figure 2.2(1), the priority is given to type-2 jobs. In this example, the cross-trained server works on shared tasks only when there is no fixed task. A justification for this observed pattern can be derived from a "modification" to the $c\mu$ type rule: the priority could be given to a job with the highest "expected instantaneous cost saving rate" per unit time possibly defined as: $(h_i + \pi_i r_i)\mu_{2i}, i \in \{1, 2\}$. Note that on the boundaries, this condition could also include an expected blocking penalty cost $(\lambda_i b_i)$. We prove the sufficient conditions for optimality of strict priority policy in favor of job 2 in Theorem II.3.

In Figure 2.2(2), we observe the optimality of a strict priority rule that assigns server 2 to type-1 jobs. The sufficient conditions to achieve such control policies are investigated in Theorem II.4. In Figure 2.2(3), we observe that the optimal policy is a switching curve that is monotone in queue lengths. Finally, in Figure 2.2(4) we observe a non-monotone switching curve, which intuitively seems to be caused by the blocking penalty for queue 2.

In practice, it is easy to implement an optimal strict priority server scheduling policy for cases such as Figures 2.2(1) and 2.2(2). For these cases, we characterize conditions which guarantee such strict priority policies in Section 2.4. On the other hand, implementing an optimal dynamic assignment algorithm for cases such as Figure 2.2(3) requires a solution to the MDP. This may not be practical in real world settings, so we present in Section 2.5 a good heuristic approach for a wide range of parameters.

2.4 Optimality of Strict Priority Policies

In this section, we present a set of conditions which guarantee the optimality of a strict priority policies. Such a policy is easy to implement. A strict priority policy in

favor of type-2 jobs is defined for our purposes such that server 2 (the cross-trained server) works on jobs of type-2 until there are no type-2 jobs remaining in the system. Then, as long as there is no type-2 jobs in the system, server 2 collaborates with server 1 on serving type-1 jobs (collaboration is required only if $\mathbf{x}_1 = 1$; otherwise the modeling of collaboration or independent service is the same). As soon as a new type-2 job enters the queue, server 2 preempts the service of the type-1 job and switches to serving a type-2 job.

Intuition, analysis, and numerical experiments led us to speculate that Conditions $(C1)-(C3)$, as stated in Theorem (II.3), guarantee the optimality of a strict priority policy in favor of type-2 jobs. Similarly, Conditions $(C4)-(C6)$, as stated in Theorem (II.4), guarantee the optimality of a strict priority policy in favor of type-1 jobs.

In Theorem II.1 we proved the optimality of non-idling policies. In the next theorem we exploit the results of Theorem II.1 to find an upper bound on the value of $\Delta_i V_n(\mathbf{x})$, which is necessary to prove a strict priority policy in Theorem II.3. This proof is presented in Section 2.7.2.

Theorem II.2. *For all* $\mathbf{x}$ *and* n, $\Delta_i V_n(\mathbf{x}) = V_n(A_i\mathbf{x}) - V_n(\mathbf{x})$, $i \in \{1,2\}$, *is bounded above by a positive constant* $\mathcal{B}_i^* = \max\left\{\frac{h_i}{1-\alpha}, b_i + h_i, \pi_i + h_i\right\}$.

Theorem II.3 provides sufficient conditions under which a strict priority policy in favor of type-2 jobs is optimal (see the proof in Section 2.7.3). A similar argument for type-1 jobs is provided in Theorem II.4.

Theorem II.3. *If the following conditions hold:*

$$\mu_{22}(h_2 + \alpha\pi_2 r_2) \geq \mu_{21}(h_1 + \alpha(\pi_1 r_1 + \lambda_1 b_1)), \qquad (C1)$$

$$r_1 \geq r_2 + \mu_{22}, \qquad (C2)$$

$$\mu_{22} b_2 \geq \mu_{21}\mathcal{B}_1^*, \qquad (C3)$$

then the value function has the following properties:

$$\mu_{22}\Delta_2 V_n(D_2\mathbf{x}) \geq \mu_{21}\Delta_1 V_n(D_1\mathbf{x}) \quad \forall \mathbf{x} : x_2 \geq 1 \quad \forall n, \qquad (P2)$$

$$\mu_{22}\Delta_2 V_n(\mathbf{x}) \geq \mu_{21}\Delta_1 V_n(\mathbf{x}) \quad \forall \mathbf{x}, n, \qquad (P3)$$

and as a result, a strict priority policy in favor of type-2 jobs is optimal.

To understand the conditions in Theorem II.3, observe that Condition $(C1)$ captures the "reward rate" of serving a type-2 job on the left-hand-side and compares it with the reward rate of serving a type-1 job on the right-hand-side. The reward rate of serving a type-2 job is the average rate at which the total system cost is reduced when the cross-trained server is assigned to a type-2 job, accounting for the fact that by completing the service of a type-2 job, we save a unit of holding cost (h_2) and also prevent in expectation some amount of renege penalty in the future periods $(\pi_2 r_2)$. On the right-hand-side, we account for the same cost elements with respect to type-1 jobs. In addition to the holding cost and the renege penalty, we include $\lambda_1 b_1$ to account for the blocking penalty of type-1 jobs that will occur as a result of prioritizing type-2 jobs if $\mathbf{x}_1 = N$.

On the other hand, Condition $(C2)$ implies that once entered, type-2 jobs are less likely to be removed from the system (either by service completion or reneging). In other words, the duration that a type-2 job is in the queue is long and imposes a great negative externalities on other jobs (both currently in the queue and future arrivals). As a result compared to compare to type-1 job, a job type-2 has higher overall cost.

To understand the intuition behind Condition $(C3)$, consider that the highest system cost occurs at the maximum queue level. At this state (N_1, N_2), the system faces the largest holding cost, reneging penalty (caused by the high renege rate), and

blocking penalty. Serving a type-2 job reduces the queue length with the rate of μ_{22} and thus reduces the blocking cost by $\mu_{22}b_2$. This decision, however, will cause queue 1 to grow at a relatively increased rate, but $\mathcal{B}_1^*$ limits the marginal cost of an additional type-1 job ($\Delta_1 V_n(x)$), thus $(C3)$ guarantees that type-2 service provides a greater instantaneous rate of cost saving.

These three conditions together imply that assigning priority to queue 2 is optimal when the cost reduction (in terms of rate) achieved by serving a job in queue 2 is higher than that by serving in queue 1.

Next, in Theorem II.4, we use an argument similar to that of Theorem II.3 to prove that under certain sufficient conditions, a strict priority policy in favor of type-1 jobs is optimal (see the proof in Section 2.7.4).

Theorem II.4. *If the following conditions hold:*

$$\mu_{21}(h_1 + \alpha\pi_1 r_1) \geq \mu_{22}(h_2 + \alpha(\pi_2 r_2 + \lambda_2 b_2)) \qquad (C4)$$

$$r_2 \geq r_1 + \mu_{11} + \mu_{21} \qquad (C5)$$

$$\mu_{21} b_1 \geq \mu_{22}\mathcal{B}_2^*, \qquad (C6)$$

then the value function has the following properties:

$$\mu_{21}\Delta_1 V_n(D_1\mathbf{x}) \geq \mu_{22}\Delta_2 V_n(D_2\mathbf{x}) \quad \forall \mathbf{x} : x_1 \geq 1 \ \ \forall n, \qquad (P4)$$

$$\mu_{21}\Delta_1 V_n(\mathbf{x}) \geq \mu_{22}\Delta_2 V_n(\mathbf{x}) \quad \forall \mathbf{x}, n, \qquad (P5)$$

and as a result, a strict priority policy in favor of type-1 jobs is optimal.

The conditions in Theorem II.4 are similar to those in Theorem II.3. Condition $(C4)$ captures the "reward rate" of serving a type-1 job and compares it with the reward rate of serving a type-2 job. The reward rate of serving a type-1 job is the average rate at which the total system cost is reduced when the cross-trained server

is assigned to a type-1 job. By removing a type-1 job, we save a unit of holding cost (h_1) and also reduce the expectation of incurring renege penalty in future periods ($\pi_1 r_1$). On the right-hand-side, we have the same cost elements for type-2 jobs. However, to guarantee that serving a type-1 job is saving more cost in average, we need to include $\lambda_2 b_2$ to account for the blocking penalty of type-2 jobs.

$(C5)$ states that $r_1 + \mu_{11} + \mu_{21}$ is larger than r_2. Thus it implies that once entered, on average, type-1 jobs stay in the system for a longer period. This increased time imposes negative externalities on other jobs. Thus, $(C5)$ further emphasizes that type-1 jobs have a higher system-wide impact than type-2 jobs.

Condition $(C6)$ is similar to Condition $(C3)$. Condition $(C6)$ consider that the highest system cost occurs at the maximum queue level. Choosing to serve a type-1 job, reduces the queue length with the rate of μ_{21} and thus reduces the expected blocking cost by $\mu_{21} b_1$. This decision, however, will cause queue 2 to grow at a relatively increased rate. Condition $(C6)$ guarantees that the cost of an extra type-2 jobs is bounded above by $\mathcal{B}_2^*$. Thus, $(C6)$ guarantees that serving a type-1 job provides a greater instantaneous rate of cost saving.

These three conditions together imply that assigning the cross-trained server to type-1 jobs, on average, results in more cost reduction. Thus, it is optimal to prioritize type-1 jobs.

2.5 Dynamic Index Heuristic Policy

When neither of the sufficient conditions presented in $(C1) - (C3)$ nor $(C4) - (C6)$ hold, the optimal scheduling policy can be both complex and computationally difficult. The literature lacks a server scheduling heuristic for moderate to low traffic systems that addresses all the features of heterogeneous demand and service rates,

customer reneging with penalties, and finite buffers with costs for the overflow of arrivals to a full queue.

For this reason we develop a heuristic based on an approximate *reward rate index*. This problem can be viewed as a restless multi-armed bandit model, a class for which general solutions are unknown (see Whittle [91] and Gittins et al. [42] for details). Although we cannot identify an optimal stopping time, it is still effective to create a myopic reward rate maximizing policy in the spirit of a Gittins Index. We find that while not always optimal, our proposed reward rate index-based heuristic policy often generates near-optimal results. This suggests that the simpler models and intuition that were used to derive this heuristic are sound. For ease of reference, we call it the Comparative Expected Reward Rate Index (CERRI).

The heuristic calculates an index which estimates the expected reward rate (i.e. rate of cost savings) of assigning each job type to the cross-trained server. After calculating the index for both job types, the cross-trained server will be assigned to the first job of the queue with the highest reward rate index. Thus, until the next transition of the system, this assignment results in the highest myopic expected cost saving. It is important to note that holding and blocking costs can be incurred in any state. However, the blocking penalty is incurred only at the boundaries. Therefore, the impact of the blocking penalty cannot be easily evaluated at each given state with a myopic approximation. We must account for the impact of reducing one type-i job from the system and thus reducing the possibility of incurring a blocking penalty, we develop an approximate measure, $f_j^k(\mathbf{x}_j)$, of the probability of reaching a buffer boundary (i.e. full queue length) before reaching the state of zero (i.e. no type-j jobs in the system).

CERRI, as presented in (2.6), estimates the comparative expected reward (or the

cost reduction) rate of assigning the cross-trained server (server 2) to job type k, where $k = 1, 2$:

$$I_k^{CE}(\mathbf{X}) =$$

$$\frac{1}{D_k}\left\{\begin{array}{l}\Big(I_{\{k=1\}}\mu_{21} + \mu_{11} + r_1x_1 - \lambda_1(1 - I_{\{x_1=N_1\}})\Big)h_1 + f_1^k(x_1)b_1 + \\ \Big(I_{\{k=2\}}\mu_{22} + r_2x_2 - \lambda_2(1 - I_{\{x_2=N_2\}})\Big)h_2 + f_2^k(x_2)b_2 + \\ \Big(I_{\{k=1\}}\mu_{21} + \mu_{11} - r_1x_1\Big)\pi_1 + \Big(I_{\{k=2\}}\mu_{22} - r_2x_2\Big)\pi_2\end{array}\right\}. \tag{2.6}$$

$$\text{where} \quad D_k = \mu_{11} + I_{\{k=1\}}\mu_{21} + I_{\{k=2\}}\mu_{22} + r_1x_1 + r_2x_2 + \lambda_1 + \lambda_2.$$

In Equation (2.6), term 1 (the first line in Equation(2.6)) includes the holding cost savings for a type-1 job (h_1) due to the service completion or reneging (with rate r_1x_1). If server 2 works on type-1 job, the service completion rate is $\mu_{11} + \mu_{21}$, otherwise it is μ_{11}. Due to the possibility of reneging from the system, a type-1 job leaves the system at the rate of $\mu_{11} + I_{\{k=1\}}\mu_{21} + r_1x_1$. If an extra type-1 job arrives to the system (with rate λ_1) the holding cost will not be saved. Therefore, we subtract λ_1h_1 if $x_1 < N_1$. As the number of jobs in the queue reaches the full buffer, the possibility of incurring a blocking penalty increases. $f_1^k(x_1)$ estimates the effect of a blocking penalty given the current state and the server assignment. In general, $f_j^k(x_j)$ may overestimate the effect of queue j reaching the boundary from state x_j if server 2 serves queue k, since it assumes that if we allocate server 2 to queue j in state x_j, the assignment will be the same for all states $x_j + 1$ to N_j unless there is a change in state of queue $3 - j$. This approximation is useful for its tractability and is satisfactory for our purpose. The formal definition of $f_j^k(x_j)$ is given in Section 2.5.1.

Term 2 (the second line in Equation(2.6)) incorporates the saving of a holding cost of a type-2 job(h_2) due to reneging or service completion if server 2 is assigned

to job type-2. But the cost can increase with a new arrival of a type-2 job (with rate λ_2 when $x_2 < N_2$). An expected blocking penalty term ($f_2^k(x_2)b_2$) is added to term 2 to represent the possibility of incurring a blocking penalty in case of an arrival.

Term 3 (the first part of the third line in Equation(2.6)) corresponds to the expected reneging penalty savings of a type-1 job (π_1) if service is completed before the renege. This results in two competing exponentials with rates $\mu_{11} + I_{\{k=1\}}\mu_{21}$ and r_1x_1.

Finally, term 4 (the second part of the third line in Equation(2.6)) corresponds to the reneging penalty of a type-2 job. When $k = 2$, there is an exponential race between a service completion (with rate μ_{22}) and a reneging event (with rate r_2x_2). In case of assigning the cross-trained sever to a type-1 job, the only event that directly contributes to the reneging penalty (π_2) is the reneging event itself.

It should be noted that a heuristic approach is generally useful when it is computationally simpler than solving the MDP. This is the case for CERRI. Furthermore, CERRI is derived from intuition and a simple analytical model that captures first order effects. CERRI estimates expected reward (cost saving) rates by measuring the first order impact of key cost elements such as holding cost, reneging penalty, and blocking penalty.

2.5.1 Estimating the Probability of Incurring the Blocking Penalty

In this section, we explain the detailed steps of computing term $f_j^k(x_j)$ used in CERRI, which is the probability that the buffer associated with job type j reaches the upper boundary before emptying when server 2 is assigned to job type k and the number of jobs of type j in the system is x_j (j=1,2 and k=1,2), given that the same control policy that was employed in state x_j is effective in states $x_{j+1}, .., N_j$. We estimate this probability by using a simplified birth and death process and recursively

computing $f_j^k(x_j)$ by solving the set of equations described by (2.7)-(2.11).

$$f_j^k(N_j) = 1, \tag{2.7}$$

$$f_j^k(x_j) = P_{x_j}^k f_j^k(x_j + 1) + Q_{x_j}^k f_j^k(x_j - 1), \qquad 0 < x_j < N_j \tag{2.8}$$

$$f_j^k(0) = 0, \tag{2.9}$$

where:

$$P_{x_j}^k = \frac{\lambda_j}{\lambda_j + r_j x_j + \mu_{11} I_{\{j=1\}} + \mu_{21} I_{\{k=1\}} + \mu_{22} I_{\{k=2\}}}, \text{and} \tag{2.10}$$

$$Q_{x_j}^k = \frac{r_j x_j + \mu_{11} I_{\{j=1\}} + \mu_{21} I_{\{k=1\}} + \mu_{22} I_{\{k=2\}}}{\lambda_j + r_j x_j + \mu_{11} I_{\{j=1\}} + \mu_{21} I_{\{k=1\}} + \mu_{22} I_{\{k=2\}}}. \tag{2.11}$$

In (2.10) and (2.11), $P_{x_j}^k$ is the transition probability from state x_j to $x_j + 1$ and $Q_{x_j}^k$ is the transition probability from state x_j to $(x_j - 1)$. This way we can compute $f_j^k(x_j)$ for all $x_j \in \{0, 1, .., N_j\}$ by solving a system of linear equations. Note that $f_j^k(x_j)$ simplifies the probability of reaching boundaries before reaching the state of zero by assuming that a decision made at state x_j will be valid for the consequent states. Therefore, $f_j^k(x_j)$ overestimates the positive impact of the employed control policy in reducing the length of the queue. Our computational experiments confirm the efficacy of this approximation.

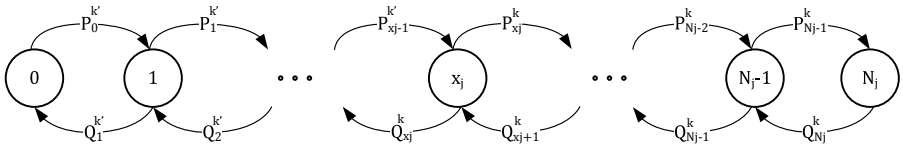


Figure 2.3: The simplified birth and death process

2.5.2 Computational Experiments

This section analyzes the performance of CERRI under a test suite. Before doing so, it is imperative to compare the actions prescribed by CERRI and the optimal

actions for the same four problem instances described in Table 2.2.

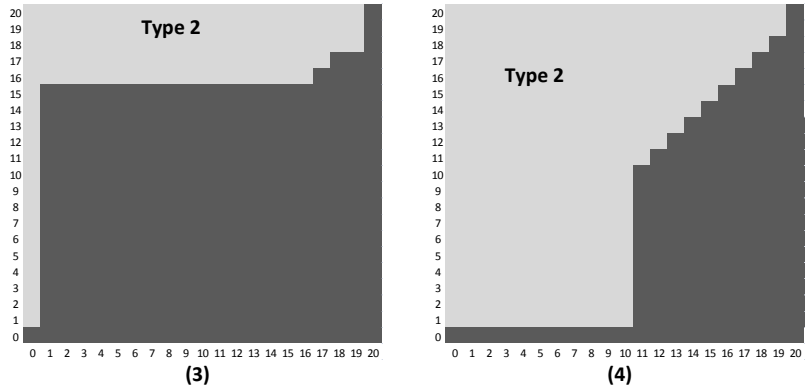


Figure 2.4: CERRI action plot associated with numerical examples 3 and 4. In the dark shaded states, server 2 serves queue 1.

In examples (1) and (2) of Figure 2.2, CERRI generates actions identical to the optimal actions; hence, there is no sub-optimality. In example (3), we observe that CERRI's actions are *almost* identical to the optimal actions – both follow a threshold-type policy (the sub-optimality is only 1.2%). Although an optimal policy can be complex, CERRI closely resembles it. However, as observed in example (4), an optimal policy may violate the threshold structure, which occurs for states $\mathbf{x}_2 \in \{18, 19, 20\}$. As one might expect, in cases where the optimal action is more complex, CERRI's actions, as illustrated in Figure 2.4, tend to deviate from the optimal actions. Nevertheless, CERRI is only 2.54% suboptimal in example (4), which is surprising at first but can be explained because CERRI matches the optimal policy in the most probable states (i.e., $x_1 \leq 10$ and $x_2 \leq 17$).

In the next step, we analyze the sensitivity of CERRI with respect to changes in the model parameters: (1) arrival rates, (2) cost components (i.e. holding cost, reneging penalty and blocking penalty), (3) the cross-trained server's skill levels (i.e. service rates), (4) renege rates, and (5) buffer sizes. To do so, we generate an extensive benchmarking test suite which includes 2,196 problem instances. These

problem instances are generated by combining three *cost regimes* defined with respect to the relative cost of job types. Each cost regimes is evaluated across 122 *parameter scenarios* focusing on each parameter in the system and six buffer levels. Additional details on how we generated the test suite problem instances are found in Section 2.7.5. Each problem instance is then solved using CERRI and the MDP approach presented in Section 2.3, with average cost per unit time performance values denoted AC^H and AC^* respectively. The *optimality gap* of CERRI is calculated as:

$$gap^H = \frac{AC^H - AC^*}{AC^*} \times 100(\%). \tag{2.12}$$

We also compare the results of our proposed heuristic with two heuristic policies that have proven effective in a variety of models in the literature utilizing holding costs: the longest queue (LQ) and the $c\mu$ algorithms. Overall, the results of the test suite illustrate that CERRI significantly outperforms LQ and $c\mu$ as illustrated in Figure 2.5. Figure 2.5 depicts the overall performance of CERRI across all test suite instances in the form of relative cumulative frequency of observations (depicted by the vertical axis) that fall in each bin. As illustrated in Figure 2.5, CERRI solves more than 80% of problem instances to an optimality gap of 4% or less versus about 53% and 18% for $c\mu$ and LQ respectively. Table 2.3 demonstrates the statistical performance measures for the test suite.

Next we present detailed sensitivity analysis results. Note that the results illustrated in the following sections are obtained by taking the average optimality gap across the three cost regimes as defined in Table 2.4 in Section 2.7.5. The averaging across these regimes usually reveals the same pattern as each specific one of the three cases. Averaging allows us to summarize the behavior of the heuristic across all the parameter regimes in one figure.

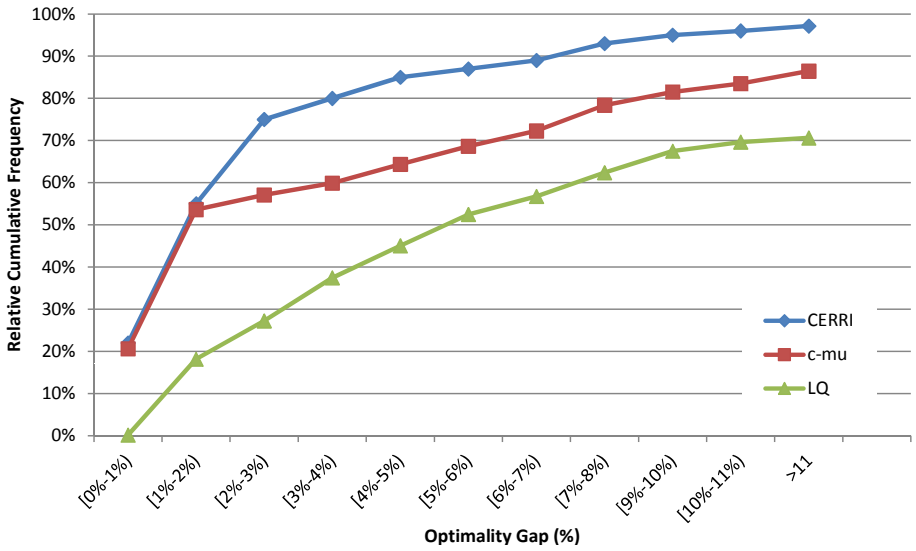


Figure 2.5: The overall performance of CERRI.

Heuristic	$c\mu$	LQ	$CERRI$
Mean(%)	6.45	8.91	3.18
Standard Deviation(%)	6.73	7.44	2.29
25th Percentile(%)	0.22	2.21	0.00
Median(%)	1.91	5.35	1.54
75th Percentile(%)	9.95	15.98	2.31
Min(%)	0	0	0
Max(%)	58.26	41.24	16.85

Table 2.3: Statistics on percentage optimality gap for all heuristic algorithms.

Impact of Arrival Rate

The impact of arrival rates is illustrated in Figure 2.6. As observed in this figure, CERRI outperforms the other heuristics at all levels of λ_1 and λ_2 (λ_2 ranging from 0.1 to 2 when $\lambda_1 = 1$) with a maximum queue capacity (N) of 10.

Note that neither LQ nor $c\mu$ account for arrival rates. This might not be an issue in cases where the buffer size is very large or the blocking penalty is negligible, but in our test suite it is. Furthermore, LQ and $c\mu$ have a tendency to over-assign server 2 to type-1 jobs as compared to the optimal action. Server 2 is more versatile, so if an algorithm ignores arrival rates, it tends to overload server 2, increasing congestion as λ_2 increases (leaving little time for helping in queue 1). Thus, with a higher type-

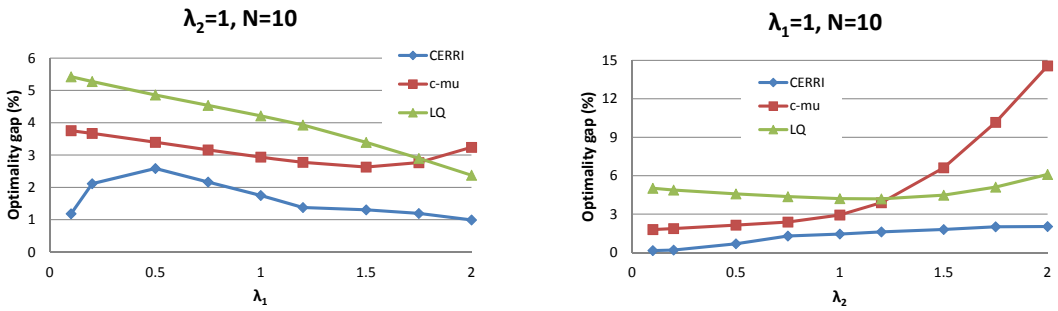


Figure 2.6: Impact of arrival rate on optimality gap.

2 job arrival rate, a heuristic that over-assigns server 2 to type-1 jobs causes the incoming type-2 jobs to wait longer. It will also have a higher chance of incurring reneging or blocking penalties in queue 2.

Impact of Blocking Penalty

The blockage of arrivals and the associated blocking penalty are key features of our model that are investigated in Figures 2.7 and 2.8. Note that as the buffer size increases, the probability of incurring blocking penalty will drop significantly. To better illustrate this effect, we present two graphs with a buffer size of 10 (Figure 2.7) and another case where buffer size is only 5 (Figure 2.8). Both figures show that CERRI always outperforms the LQ and $c\mu$ heuristics. The benefit of employing CERRI is more significant when the buffer size is small ($N = 5$), as one would expect. Fortunately, CERRI is insensitive to buffer size, indicating that the model of buffer rejection penalties is accurate.

Impact of Servers' Skill Level

To illustrate the impact of service rates (μ_{21} and μ_{22}), we let μ_{21} and μ_{22} vary in the left and right plots of Figure 2.9, respectively. We observe significantly better and more robust performance from CERRI compared to LQ and $c\mu$.

In Figure 2.9, we observe that when the value of μ_{21} is relatively small, LQ has

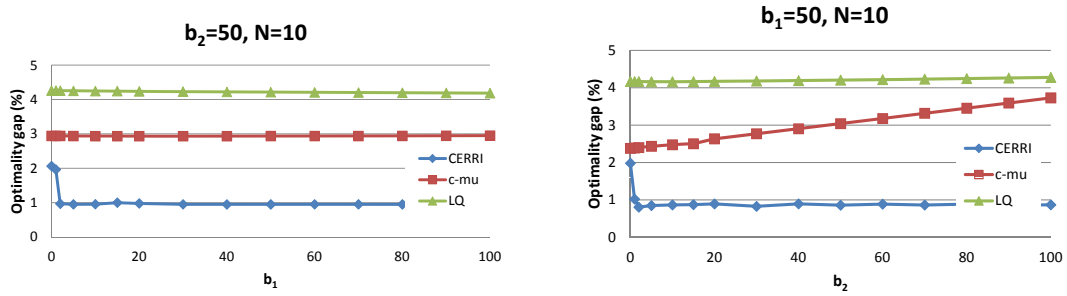


Figure 2.7: Impact of blocking penalty on optimality gap for $N = 10$.

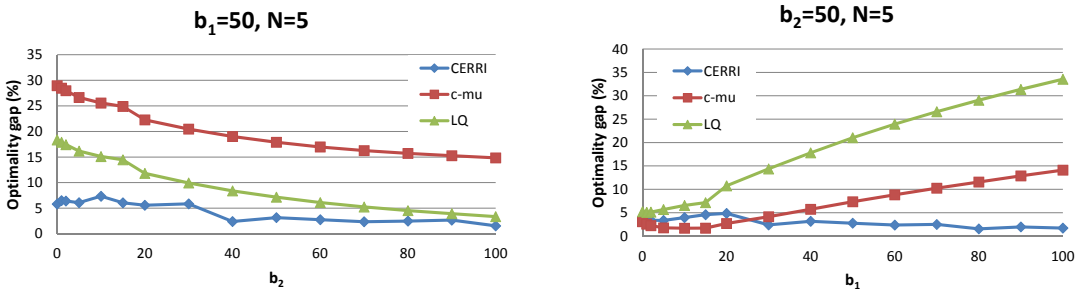


Figure 2.8: Impact of blocking penalty on optimality gap for $N = 5$.

a high optimality gap (the figure on the left), and the opposite occurs as the value of μ_{22} increases. This is because as μ_{21} decreases or μ_{22} increases (relative to μ_{11}), server 2 becomes better at handling type-2 jobs, which undermines the flexibility in the system. In the extreme case where $\mu_{21} = 0$, the system completely loses flexibility, and LQ tends to over-assign server 2 to type-1 jobs by using only queue length without considering other parameters.

Impact of Renege Rate

To analyze the impact of renege rates on the performance of the heuristics, we consider two cases. First we fix the reneging rate of type-1 jobs at the default value (0.05) and let the reneging rate of type-2 job vary between 0 to 0.4. In the second run, we fix the reneging rate of type-2 jobs at the default value (0.05) and let the reneging rate of type-1 job vary between 0 to 0.4. The results are illustrated in

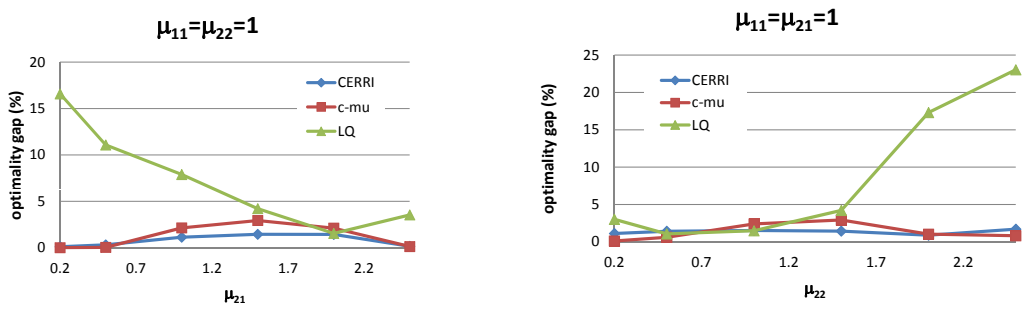


Figure 2.9: Impact of service rate on optimality gap.

Figure 2.10. Again, in both cases, we observe that CERRI outperforms the other heuristics.

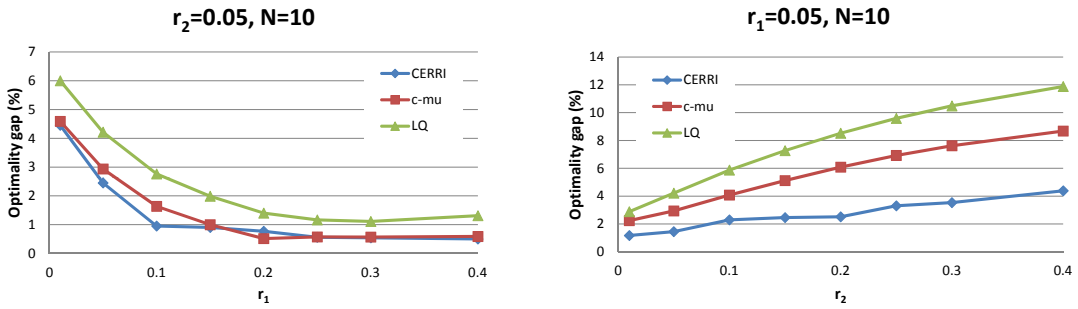


Figure 2.10: Impact of renege rates on optimality gap.

Interestingly, we observe different behaviors in the two cases illustrated in Figure 2.10. In the right figure, we see a rather intuitive pattern: as the reneging rate for type-2 jobs increases, the performance of all heuristics tends to deteriorate. However, in the left figure, we see the opposite pattern. As the reneging rate for type-1 job increases, the optimality gap closes for all heuristics. The main reason for this behavior is the tendency of all three heuristic approaches to over-assign server 2 to queue 1. We have already argued that this over-assignment is detrimental in most cases. However, as the reneging rate for job 1 increases, the type-1 jobs that are assigned to server 2 have a greater rate of leaving the system. Therefore, the high reneging rate of type-1 jobs reduces the workload of server 2 and mitigates the adverse

effects of this over-assignment simply because it keeps queue 1 small. Therefore, the flexible server spends less time there. The opposite occurs as r_2 increases. We see this complexity of interacting effects obviously in Figure 2.10. Note that it is hard to make simple modification to the $c\mu$ rule to account for reneging because the rate is sensitive to queue length.

Impact of Renege Penalty

To analyze the impact of reneging penalty on the performance of the heuristics, we consider two cases. First, we fix the reneging penalty of type-2 jobs at the default value (5) and let the reneging penalty of type-1 jobs change between 0 to 120. In the second run, we fix the reneging penalty of type-1 jobs at the default value (5) and let the reneging penalty of type-2 jobs range from 0 to 120. The results of this analysis are illustrated in Figure 2.11. Once again we observe that CERRI performs much better compared to LQ and $c\mu$. Furthermore, we observe that the performance of CERRI significantly improves as reneging penalties increase for either job type. Neither of the other two policies has an effective way of capturing reneging in its decisions.

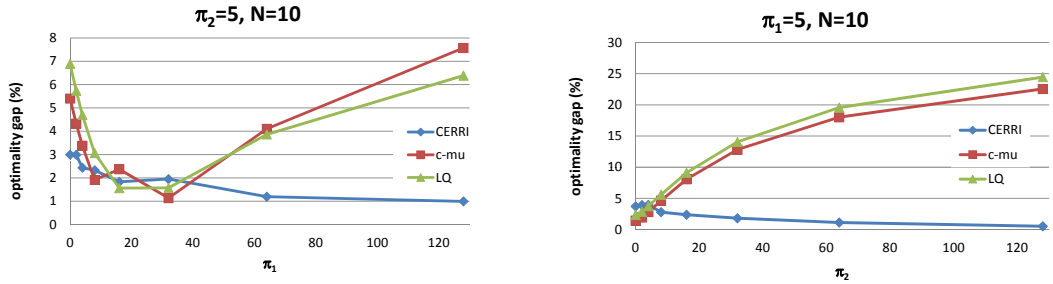


Figure 2.11: Impact of renege penalty on optimality gap.

Impact of Holding Cost

To analyze the impact of holding cost, we first set the holding cost of type-1 jobs at the default value of 1 and let the holding cost of type-2 jobs range from 0 to 5. Then we repeat with the reverse scenario. The results are illustrated in Figure 2.12.

We observe that CERRI maintains its good performance for all levels of holding cost and achieves a very low error across the range. We also note that the performance of LQ deteriorates as h_2 increases, while CERRI and $c\mu$ have similar performances for very high values of h_2. Interestingly, when h_1 is very small, $c\mu$ performs better as it does not tend to over-assign server 2 to type-1 jobs. However for smaller values of h_2, $c\mu$ performs significantly worse than CERRI, since it ignores the effect of reneging and blocking. In both cases, for larger holding cost values, the performance gap of $c\mu$ closes. This is rather intuitive: as the holding cost increases, other cost components in the system (such as blocking and reneging penalties) lose their impact, simplifying the problem for $c\mu$. LQ has regions where it is nearly optimal; however, the extreme values of the h_2/h_1 ratio call for a new priority rule (except $h_2 \simeq 0$), which LQ cannot achieve. Therefore, the full benefits of employing CERRI can be observed in the "middle ranges" of holding cost.

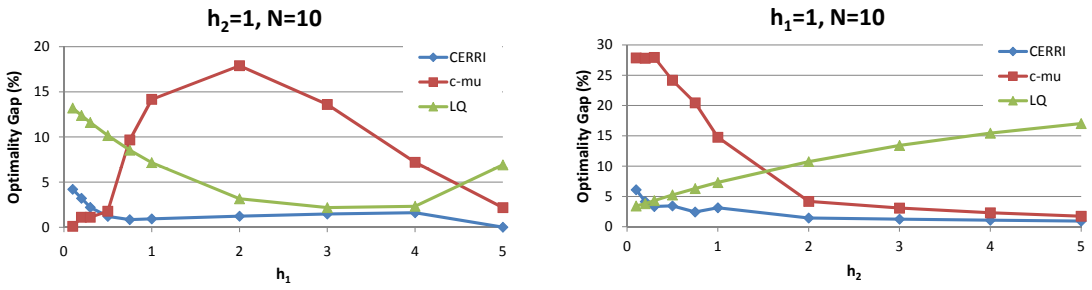


Figure 2.12: Impact of holding cost on optimality gap.

Impact of Buffer Capacity

Finally, we present the impact of the buffer capacity in Figure 2.13. We set all the parameters to their default values (see Table 2.6) and set the buffer capacity to 5, 10, 15, 20, 25, and 30. As the buffer capacity increases, the impact of blocking decreases; therefore LQ and $c\mu$ tend to perform better for higher buffer capacities, whereas CERRI is very good at all levels. This is because at lower buffer sizes, there is a large probability of reaching the full buffer and incurring blocking cost increases.

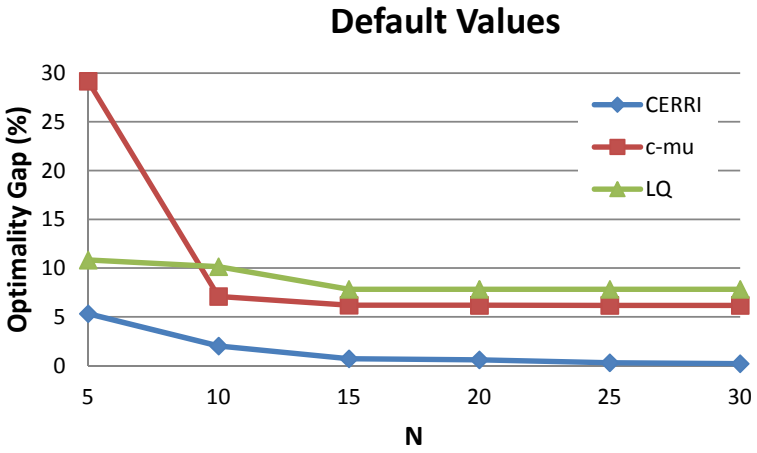


Figure 2.13: Impact of buffer size on optimality gap.

2.6 Conclusions and Future Research

This research analyzes the problem of assigning heterogeneous customer demand to servers in an "N" network of two servers with two job types, a fixed task, and a shared task. The cross-trained (flexible) server can handle both job types, while the dedicated server can only serve the shared jobs, type 1. The system is complex, possessing three cost components: holding costs per unit time per job in the system, blocking penalties, and reneging penalties. One interesting feature critical to many applications is finite buffers, where a new arrival to a full buffer is blocked and incurs a cost. Also jobs may renege after an exponentially distributed time in the system.

We use a Markov Decision Process (MDP) approach to model the problem. We first develop a continuous-time model and then use uniformization technique to convert it into a discrete model. We analyze the dynamic server scheduling (or equivalently, job routing problem) and develop sufficient conditions which guarantee the optimality of a strict priority policy to assign the flexible server (server 2) to the fixed task (job type 1). Similarly, we study sufficient conditions that guarantee the optimality of a strict priority policy to assign the flexible server (server 2) to the shared task (job type 2). We then use value iteration to solve the MDP model numerically and show that in the absence of the above-mentioned sufficient conditions, the optimal job routing policy may not be easily characterized as monotonic threshold functions.

We develop CERRI, an index-based heuristic algorithm that generates near-optimal policies for cases where the above sufficient conditions are absent. The index captures the main cost components of the problem such as holding costs, reneging penalties, and blocking penalties through a simplified birth and death process. We test the performance of CERRI using a comprehensive test-suite and compare its results against the optimal assignment achieved by MDP and other assignments computed by two widely-used heuristics, the $c\mu$ and the LQ rules. We show that CERRI finds efficient (and often optimal) policies in for a vast majority of problem structures. CERRI can be viewed as a simplified model of the dynamics that dominate the control of this system. The excellent performance of CERRI reveals that the simplified model contains valid insights. We also use the results of CERRI to understand the key characteristics of the problem by analyzing its sensitivity as a function of elements such as buffer size, arrival, service and reneging rates and cost components such as holding cost, blocking and reneging penalty.

The insights gained from our research can help better characterize larger networks as long as they can be broken down into smaller modules which resemble a similar "N" network structure. Further investigations are required to develop insights and heuristics for more complex queueing networks.

2.7 Proofs of Theorems in Chapter II

2.7.1 Proof of Theorem II.1

Proof. To prove these we use induction. Note that the results hold when $n = 0$, since $V_0(\mathbf{x}) = 0$ for all $\mathbf{x}$. Now suppose that the results hold up to some $n \geq 0$. We now suppose that the claims hold for $n + 1$ as well.

Optimality of a ***non-idling policy*** *in period* $n + 1$: We use sample path argument to show optimality of non-idling policy at stage $n + 1$. Without loss of generality we show optimality of a non-idling policy for sever server 2 (the cross-trained server). A similar argument will treat server 1. Consider the following two policies: policy g idles server 2 at stage $n + 1$ and follows the optimal policy afterward. Policy $\tilde{g}$ does not idle server 2 at stage $n+1$ and follows the optimal policy afterward. Let $J^g_{n+1}(\mathbf{x})$ represent the expected total discounted cost function under policy g at stage $n + 1$ and state $\mathbf{x}$. Therefore, by Equation 2.2, we have:

$$J^g_{n+1}(\mathbf{x}) - J^{\tilde{g}}_{n+1}(\mathbf{x}) = \mu_{2i}\Delta_i V_n(D_i\mathbf{x}) \geq 0 \quad \text{(by induction hypothesis)}. \tag{2.13}$$

As a result, the expected cost under policy $\tilde{g}$ is less than the expected cost under g. Therefore, the optimal control belongs to the class of non-idling policies. $V_{n+1}(x_1, x_2)$ *is* ***non-decreasing*** *in* x_1 *and* x_2: To show this property, we first expand $V_{n+1}(\mathbf{x})$ and $V_{n+1}(D_i\mathbf{x})$ and use a term-by-term comparison to show that $V_{n+1}(\mathbf{x}) - V_{n+1}(D_i\mathbf{x}) \geq 0$. Using (2.2) we expand $V_{n+1}(\mathbf{x})$ as the following:

$$V_{n+1}(\mathbf{x}) =$$

$$\sum_{i=1}^{2} h_i x_i + \alpha \left\{ \begin{array}{l} \sum_{i=1}^{2}[\lambda_i V_n(A_i\mathbf{x}) + r_i x_i V_n(D_i\mathbf{x}) + \pi_i r_i x_i + \lambda_i b_i I_{\{x_i = N_i\}}] + \\ \min\{\mu_{21}\Delta_1 V_n(D_1\mathbf{x}) + [1 - \sum_{i=1}^{2}(\lambda_i + r_i x_i) - \mu_{11} - \mu_{21}]V_n(\mathbf{x}), \\ \mu_{22}V_n(D_2\mathbf{x}) + [1 - \sum_{i=1}^{2}(\lambda_i + r_i x_i) - \mu_{11} - \mu_{22}]V_n(\mathbf{x})\} \end{array} \right\}.$$

$V_{n+1}(D_i\mathbf{x})$ can be expanded in a similar fashion. Subtracting $V_{n+1}(\mathbf{D_i x})$ from $V_{n+1}(\mathbf{x})$, and setting $Q = 1 - \sum_{i=1}^{2}(\lambda_i + r_i x_i) - \mu_{11}$, we have:

$$V_{n+1}(\mathbf{x}) - V_{n+1}(D_i\mathbf{x}) =$$
$$h_i + \alpha \left\{ \begin{array}{l} \sum_{i=1}^{2}[\lambda_j V_n(A_j D_i\mathbf{x}) + r_i(x_j - I_{\{j=i\}})V_n(D_j D_i\mathbf{x})] + \\ \pi_i r_i x_i + \lambda_i b_i I_{\{x_i=N_i\}} + \min\{\mu_{21}V_n(D_1\mathbf{x}) + \\ (Q-\mu_{21})V_n(\mathbf{x}), \mu_{22}V_n(D_2\mathbf{x}) + (Q-\mu_{22})V_n(\mathbf{x})\} - \\ \min\{\mu_{21}V_n(D_1 D_i\mathbf{x}) + (Q-\mu_{21})V_n(D_i\mathbf{x}), \\ \mu_{22}V_n(D_2 D_i\mathbf{x}) + (Q-\mu_{22})V_n(D_i\mathbf{x})\} \end{array} \right\}.$$

The terms in the first and second lines of the above equation, are all non-negative by non-negativity of costs and by the induction hypothesis. Subtracting line four from line three will result is a non-negative term, since by induction hypothesis we know that $V_n(D_i\mathbf{x}) \geq V_n(D_i^2\mathbf{x})$ and $V_n(\mathbf{x}) \geq V_n(D_i\mathbf{x})$. The result follows since for any four numbers, if $a \geq c$ and $b \geq d$, then $\min\{a,b\} - \min\{c,d\} \geq 0$.

Therefore $V_{n+1}(\mathbf{x})$ is non-decreasing in $\mathbf{x}$ and the proof is complete. □

2.7.2 Proof of Theorem II.2

Proof by a sample path argument. The case of $x_i = 0$ is trivial, because the upper-bound is zero. $\Delta_i V_n(\mathbf{x})$ can be thought of as the incremental expected discounted n-stage cost due to the additional job of type-1. For clarity, we mark this extra job as the *red* job. To find an upper-bound on $\Delta_i V_n(\mathbf{x})$, we evaluate the cost impact of the *red* job along the possible sample paths. Assume Ω is the set of all possible sample paths. We partition Ω into four sets $(\Omega_1, ..., \Omega_4)$ along which four distinct cases apply.

In case 1, the *red* job is not cleared from the system during the n stages to go. In case 2, the *red* job leaves the system after receiving service. In case 3, the *red* job

causes a blockage of a new arrival and in case 4, the *red* job reneges.

For further clarification we call the scenario in which the system follows an optimal policy starting with $A_i\mathbf{x}$ jobs (i.e. the system includes the *red* job), "scenario A" and the scenario in which the system has $\mathbf{x}$ jobs (i.e. the system excludes the *red* job), "scenario B". The policy in scenario B is mimicked by scenario A, until the two systems couple (from which time on both systems will follow the optimal policy). Let $J_n(A_i\mathbf{x})^\omega$ represent the expected cost for a given sample path ω under the policy that mimics B until two systems couple. Note that in all cases except for case 1, the two scenarios will couple after the event associated with each case occurs. We consider each case and the costs associated with it in more detail.

Case 1: In this case, scenarios A and B never couple within n stages for following reasons: (a) there is never a blockage of any job of type-1 under scenario A at any stage prior to stage n, and (b) the *red* job's service is not completed prior to the last stage and so the reneging events match for both scenarios. In this case, we have:

$$J_n(A_i\mathbf{x})^\omega - V_n^\omega(\mathbf{x}) = \sum_{t=0}^{n-1} \alpha^t h_i = h_i \sum_{t=0}^{n-1} \alpha^t \leq \frac{h_i}{1-\alpha} \quad \forall \omega \in \Omega_1 \quad \text{since } \alpha < 1.$$

Case 2: In this case the *red* job's service will complete at $\tau_2(\omega)$ stages into the future where $1 \leq \tau_2(\omega) \leq n$, and scenarios A and B are coupled after that time. In this case, similar to case 1, we only incur holding costs. However, as the *red* job in this case leaves the system prior to the time that it leaves the system in case 1, the expected cost of this case is less than the one in case 1.

$$J_n(A_i\mathbf{x})^\omega - V_n^\omega(\mathbf{x}) = \sum_{t=0}^{\tau_2(\omega)} \alpha^t h_i = h_i \sum_{t=0}^{\tau_2(\omega)} \alpha^t \leq \frac{h_i}{1-\alpha} \quad \forall \omega \in \Omega_2. \quad \text{since } \alpha < 1.$$

Case 3: A job is blocked at a time $\tau_3(\omega)$ stages into the future because queue one has N_i jobs at stage $\tau_3(\omega)$ when the arrival occurs. Therefore, at stage $\tau_3(\omega)$ both

systems couple. Therefore, we have:

$$(2.14)\quad J_n(A_i\mathbf{x})^\omega - V_n^\omega(\mathbf{x}) = b_i\alpha^{\tau_3(\omega)} + \sum_{t=0}^{\tau_3(\omega)} \alpha^t h_i \leq \max\big\{\frac{h_i}{1-\alpha}, b_i + h_i\big\} \quad \forall\omega \in \Omega_3.$$

To find the upper-bound for the right hand side in (2.14), consider the sequence $s_k = b_i\alpha^k + \sum_{t=0}^{k} \alpha^t h_i$. Note that in this sequence, $s_{k+1} - s_k = \alpha^k(\alpha(b_i + h_i) - b_i)$. If $\alpha(b_i + h_i) - b_i \geq 0$, the sequence s_k is an increasing sequence. Then $\frac{h_i}{1-\alpha}$ will be an upper bound on all the elements of the sequence defined in (2.14). Otherwise, $b_i + h_i$, can be an upper bound.

Case 4: In this case the *red* job reneges at time $\tau_4(\omega)$ and this causes the scenarios to couple. Of course, having any job ahead of the *red* job does not result in coupling of the processes. Therefore, we have:

(2.15)

$$J_n(A_i\mathbf{x})^\omega - V_n^\omega(\mathbf{x}) = \pi_i\alpha^{\tau_4(\omega)} + \sum_{t=0}^{\tau_4(\omega)} \alpha^t h_1 \leq \max\big\{\frac{h_i}{1-\alpha}, \pi_i + h_i\big\} \quad \forall\omega \in \Omega_4.$$

The argument here is similar to the one used in Case 3. We consider the right hand side of (2.15) as a sequence. In this sequence, we have, $s'_{k+1} - s'_k = \alpha^k(\alpha(\pi_i + h_i) - \pi_i)$. If $\alpha(\pi_i + h_i) - \pi_i \geq 0$, the sequence s'_k is an increasing sequence. In this case, $\frac{h_i}{1-\alpha}$ will be an upper bound on all the elements of the sequence defined in (2.15). Otherwise, $\pi_i + h_i$, can be an upper bound.

Considering all the above cases, we define $p_i = P(\omega \in \Omega_i)$ which is the probability of each case. This way we have:

$$J_n(A_i\mathbf{x})^\omega - V_n^\omega(\mathbf{x}) \leq p_1\frac{h_i}{1-\alpha} + p_2\frac{h_i}{1-\alpha} +$$
$$p_3\max\big\{\frac{h_i}{1-\alpha}, b_i + h_i\big\} + p_4\max\big\{\frac{h_i}{1-\alpha}, \pi_i + h_i\big\}.$$

We define ${\mathcal{B}_i}^* = \max\{\frac{h_i}{1-\alpha}, b_i + h_i, \pi_i + h_i\}$. Thus $J_n(A_i\mathbf{x})^\omega - V_n^\omega(\mathbf{x}) \leq \mathcal{B}_i^*$. Therefore, $\Delta_i V_n(\mathbf{x}) \leq J_n(A_i\mathbf{x})^\omega - V_n^\omega(\mathbf{x}) \leq \mathcal{B}_i^*$. This completes the proof. □

2.7.3 Proof of Theorem II.3

Proof of Theorem 3. *The proof is by induction.*

As the base of induction, we need to prove properties $(P2)$ and $(P3)$ of the value function for the stage 0. Note that $V_0(\mathbf{x}) = 0, \forall \mathbf{x}$ by definition, and as a result properties $(P2)$ and $(P3)$ hold trivially.

Induction Hypothesis (IH): We assume that all the properties $(P2)$ and $(P3)$ hold for $n-1$ and for all $x_2 \geq 1$ for $(P2)$ and for all $x_2 \geq 0$ for $(P3)$. We next prove these properties for n as our inductive step.

Proof of $(P2)$: To prove this property, we first break down the both sides of $(P2)$. From the definition of D_i and Δ_i, we have the following expressions:

$$\mu_{22}\Delta_2 V_n(D_2\mathbf{x}) = \mu_{22}\big[V_n(x_1, x_2) - V_n(x_1, x_2 - 1)\big]. \tag{2.16}$$

$$\mu_{21}\Delta_1 V_n(D_1\mathbf{x}) = \mu_{21}\big[V_n(x_1, x_2) - V_n((x_1 - 1)^+, x_2)\big]. \tag{2.17}$$

In the next step, we expand $V_n(x_1, x_2)$ and $V_n(x_1, x_2 - 1)$ using Equation (2.5). Note that, based on the induction hypothesis, we know that it is optimal to serve job type-2 when $x_2 > 0$. Therefore, the second term of the maximization function in Equation (2.5) (i.e. $\mu_{22}\Delta_2 V_{n-1}(D_2\mathbf{x})$) is always greater or equal to the first term in the equation (i.e. $\mu_{21}\Delta_1 V_{n-1}(D_1\mathbf{x})$). For $2 \leq \mathbf{x}_2 < N_2$, we can expand the left hand side to get:

$$\begin{aligned}
&\mu_{22}\Delta_2 V_n(D_2\mathbf{x}) = \mu_{22}h_2 + \\
&\alpha\mu_{22}\left\{\begin{array}{ll}
\pi_2 r_2 + \lambda_1\Delta_2 V_{n-1}(A_1 D_2\mathbf{x}) + \lambda_2\Delta_2 V_{n-1}(\mathbf{x}) + & (i)\\
r_1 x_1\Delta_2 V_{n-1}(D_1 D_2\mathbf{x}) + \mu_{11}\Delta_2 V_{n-1}(D_1 D_2\mathbf{x}) + & (ii)\\
r_2(x_2 - 1)\Delta_2 V_{n-1}(D_2^2\mathbf{x}) + \mu_{22}\Delta_2 V_{n-1}(D_2^2\mathbf{x}) + & (iii)\\
(1 - \lambda_1 - \lambda_2 - r_1 x_1 - r_2 x_2 - \mu_{11} - \mu_{22})\Delta_2 V_{n-1}(D_2\mathbf{x}) & (iv)
\end{array}\right\}.
\end{aligned} \tag{2.18}$$

Similarly, the right hand side can be expanded:

$$\mu_{21}\Delta_1 V(D_1\mathbf{x}) = \mu_{21}h_1 +$$

(2.19)

$$\alpha\mu_{21}\left\{\begin{array}{ll} \pi_1 r_1 + \lambda_1 b_1 I_{\{x_1=N_1\}} + \lambda_1\Delta_1 V_{n-1}(\mathbf{x}) + \lambda_2\Delta_1 V_{n-1}(A_2 D_1\mathbf{x}) + & (i)\\ (r_1(x_1-1)^+)\Delta_1 V_{n-1}(D_1^2\mathbf{x}) + \mu_{11} I_{\{(x_1-1)^+\neq 0\}}\Delta_1 V_{n-1}(D_1^2\mathbf{x}) + & (ii)\\ r_2 x_2\Delta_1 V_{n-1}(D_2 D_1\mathbf{x}) + \mu_{22}\Delta_1 V_{n-1}(D_2 D_1\mathbf{x}) + & (iii)\\ (1-\lambda_1-\lambda_2-r_1x_1-r_2x_2-\mu_{11}-\mu_{22})\Delta_1 V_{n-1}(D_1\mathbf{x}) & (iv) \end{array}\right\}.$$

In the next step, we use a line-by-line comparison between the terms on the left hand side with their counterpart terms to show that $(P2)$ holds. For comparing, (2.18-i) to (2.19-i) using $(C1)$ and the induction hypothesis $(P2)$ we can show that:

$$\mu_{22}(h_2 + \alpha(\pi_2 r_2 + \lambda_1\Delta_2 V_{n-1}(A_1 D_2\mathbf{x}) + \lambda_2\Delta_2 V_{n-1}(\mathbf{x}))) \geq$$

$$\mu_{21}(h_1 + \alpha(\pi_1 r_1 + \lambda_1 b_1 I_{\{x_1=N_1\}} + \lambda_1\Delta_1 V_{n-1}(\mathbf{x}) + \lambda_2\Delta_1 V_{n-1}(A_2 D_1\mathbf{x}))).$$

The next lines of terms to compare are (2.18-ii) and (2.19-ii). We first write r_1x_1 as $r_1(x_1-1)^+ + r_1 I_{\{x_1\neq 0\}}$. This way, using the induction hypothesis $(P2)$ we get:

$$\mu_{22}((r_1(x_1-1)^+) + \mu_{11})\Delta_2 V_{n-1}(D_1 D_2\mathbf{x}) \geq$$

$$\mu_{22}((r_1(x_1-1)^+ + \mu_{11} I_{\{(x_1-1)^+\neq 0\}})\Delta_1 V_{n-1}(D_1^2\mathbf{x}).$$

To compare lines (2.18-iii) and (2.19-iii), we rewrite r_2x_2 as $r_2(x_2-1)^+ + r_2$. As a result by employing the induction hypothesis $(P2)$, we have:

$$\mu_{22}(r_2(x_2-1) + \mu_{22})\Delta_2 V_{n-1}(D_2^2\mathbf{x}) \geq \mu_{21}(r_2(x_2-1) + \mu_{22})\Delta_1 V_{n-1}(D_2 D_1\mathbf{x}).$$

Note that as a result of the substitutions, we are left with a term associated with r_1 in the left hand side, and a term associated with r_2 in the right hand side. By $(C2)$ and the induction hypothesis $(P3)$, we have:

$$\mu_{22} r_1\Delta_2 V_{n-1}(D_1 D_2\mathbf{x}) \geq \mu_{21} r_2\Delta_1 V_{n-1}(D_2 D_1\mathbf{x}).$$

Finally the term associated with the self loops can be compared using the induction hypothesis $(P2)$. This way we can show that:

$$\mu_{22}(1-\lambda_1-\lambda_2-r_1x_1-r_2x_2-\mu_{11}-\mu_{22})\Delta_2 V_{n-1}(D_2\mathbf{x})$$
$$\geq \mu_{21}(1-\lambda_1-\lambda_2-r_1x_1-r_2x_2-\mu_{11}-\mu_{22})\Delta_1 V_{n-1}(D_1\mathbf{x}).$$

In case $\mathbf{x}_2 = 1$, we have the term $\mu_{21}\Delta_1 V_{n-1}(D_1D_2\mathbf{x})$ in the left hand side instead of the term $\mu_{22}\Delta_2 V_{n-1}(D_2^2\mathbf{x})$ (by optimality of non-idling policy, i.e. instead of idling, the cross-trained server works on job type-1). In Theorem II.1, we proved the non-negativity of $\Delta_i V_n(\mathbf{x})$ for all n and $\mathbf{x}$. Therefore we have:

$$\mu_{21}\Delta_1 V_{n-1}(D_1D_2\mathbf{x}) \geq 0,$$

and by $(C2)$ and the induction hypothesis $(P3)$, we have:

$$\mu_{22}r_1\Delta_2 V_{n-1}(D_1D_2\mathbf{x}) \geq \mu_{21}(r_2+\mu_{22})\Delta_1 V_{n-1}(D_2D_1\mathbf{x}).$$

Note that, in the case where $x_2 = N_2$, as we noted before, $\lambda_2\Delta_2 V_{n-1}(\mathbf{x})$ equals zero (in the right hand side of $(P2)$. In Theorem II.2, we proved that there exists an upper bound, $\mathcal{B}_1^*$, to $\Delta_1 V_{n-1}(A_2D_1(\mathbf{x}))$, for all $\mathbf{x}$. As a result, based on Condition $(C3)$, we have $\mu_{22}b_2 \geq \mu_{21}\mathcal{B}_1^*$. Therefore we have:

$$\mu_{22}\lambda_2 b_2 \geq \mu_{21}\lambda_2\Delta_1 V_{n-1}(A_2D_1\mathbf{x}).$$

We have shown that all the terms on the left hand side of $(P2)$ are greater than or equal to all the terms on the right hand side of $(P2)$. Therefore, the proof of $(P2)$ holds for stage n and as a result, for all the other stages.

Proof of $(P3)$: We have defined $(P3)$ as:

$$\mu_{22}\Delta_2 V_n(\mathbf{x}) \geq \mu_{21}\Delta_1 V_n(\mathbf{x}) \quad \forall\mathbf{x}. \tag{2.20}$$

In the next step, we expand $V_n(x_1, x_2)$ and $V_n(x_1, x_2 - 1)$ using Equation (2.5). Note that, based on the induction hypothesis, we know that it is optimal to serve job type-2 when $x_2 \geq 1$. Therefore, the second term of the maximization function in Equation (2.5) (i.e. $\mu_{22}\Delta_2 V_{n-1}(D_2\mathbf{x})$) is always greater or equal to the first term in the equation (i.e. $\mu_{21}\Delta_1 V_{n-1}(D_1\mathbf{x})$). This way, for $1 \leq \mathbf{x}_2 < N_2$, we can expand the left hand side to get:

$$\mu_{22}\Delta_2 V_n(\mathbf{x}) = \mu_{22}h_2 +$$

(2.21)

$$\alpha\mu_{22}\left\{\begin{array}{ll} \pi_2 r_2 + \lambda_1\Delta_2 V_{n-1}(A_1\mathbf{x}) + \lambda_2\Delta_2 V_{n-1}(A_2\mathbf{x}) + & (i) \\ r_1 x_1\Delta_2 V_{n-1}(D_1\mathbf{x}) + \mu_{11}\Delta_2 V_{n-1}(D_1\mathbf{x}) + & (ii) \\ r_2 x_2\Delta_2 V_{n-1}(D_2\mathbf{x}) + \mu_{22}\Delta_2 V_{n-1}(D_2\mathbf{x}) + r_1\Delta_2 V_{n-1}(\mathbf{x}) + & (iii) \\ (1 - \lambda_1 - \lambda_2 - r_1(x_1+1) - r_2(x_2+1) - \mu_{11} - \mu_{22})\Delta_2 V_{n-1}(\mathbf{x}) & (iv) \end{array}\right\}.$$

$$\mu_{21}\Delta_1 V(\mathbf{x}) = \mu_{21}h_1 +$$

(2.22)

$$\alpha\mu_{21}\left\{\begin{array}{ll} \pi_1 r_1 + \lambda_1 b_1 I_{\{(x_1+1=N_1\}} + \lambda_1\Delta_1 V_{n-1}(A_1\mathbf{x}) + \lambda_2\Delta_1 V_{n-1}(A_2\mathbf{x}) + & (i) \\ r_1 x_1\Delta_1 V_{n-1}(D_1\mathbf{x}) + \mu_{11} I_{\{x_1\neq 0\}}\Delta_1 V_{n-1}(D_1\mathbf{x}) + & (ii) \\ r_2 x_2\Delta_1 V_{n-1}(D_2\mathbf{x}) + \mu_{22}\Delta_1 V_{n-1}(D_2\mathbf{x}) + r_2\Delta_1 V_{n-1}(\mathbf{x}) & (iii) \\ (1 - \lambda_1 - \lambda_2 - r_1(x_1+1) - r_2(x_2+1) - \mu_{11} - \mu_{22})\Delta_1 V_{n-1}(\mathbf{x}) & (iv) \end{array}\right\}.$$

In the next step, we use a line-by-line comparison between the terms on the left hand side with their counterpart terms to show that $(P3)$ holds. For comparing, (2.21-i) to (2.22-i) using $(C1)$ and the induction hypothesis $(P3)$ we can show that:

$$\mu_{22}(h_2 + \alpha(\pi_2 r_2 + \lambda_1\Delta_2 V_{n-1}(A_1\mathbf{x}) + \lambda_2\Delta_2 V_{n-1}(A_2\mathbf{x}))) \geq$$

$$\mu_{21}(h_1 + \alpha(\pi_1 r_1 + \lambda_1 b_1 I_{\{x_1+1=N_1\}} + \lambda_1\Delta_1 V_{n-1}(A_1\mathbf{x}) + \lambda_2\Delta_1 V_{n-1}(A_2\mathbf{x}))).$$

The next lines of terms to compare are (2.21-ii) and (2.22-ii). By using the induction hypothesis $(P3)$ we have:

$$\mu_{22}(r_1x_1+\mu_{11})\Delta_2V_{n-1}(D_1\mathbf{x}) \geq \mu_{21}(r_1x_1+\mu_{11}I_{\{x_1\neq 0\}})\Delta_1V_{n-1}(D_1\mathbf{x}).$$

To compare lines (2.21-iii) and (2.22-iii), first we compare the first two terms. As a result, by employing the induction hypothesis $(P3)$, we have:

$$\mu_{22}(r_2x_2+\mu_{22})\Delta_2V_{n-1}(D_2\mathbf{x}) \geq \mu_{21}(r_2x_2+\mu_{22})\Delta_1V_{n-1}(D_2\mathbf{x}).$$

Note that, we are left with a term associated with r_1 in the left hand side, and a term associated with r_2 in the right hand side. By $(C2)$ and the induction hypothesis $(P3)$, we have:

$$\mu_{22}r_1\Delta_2V_{n-1}(\mathbf{x}) \geq \mu_{21}r_2\Delta_1V_{n-1}(\mathbf{x}).$$

Finally, the term associated with the self loops can be compared using the induction hypothesis $(P3)$.

In case $\mathbf{x}_2 = 0$, we will have the term $\mu_{21}(\Delta_1V_{n-1}(D_1\mathbf{x}) + \mu_{21}\Delta_2V_{n-1}(\mathbf{x}))$ in the left hand side instead of the term $\mu_{22}\Delta_2V_{n-1}(D_2\mathbf{x})$. The reason this term appears is that when $\mathbf{x}_2 = 0$, the cross-trained server will work on the job type-1 by the optimality of non-idling policies. As a result, the term $-\mu_{21}V_{n-1}(D_1\mathbf{x})$ will appear on the left hand side. Furthermore, we need to add term $\pm\mu_{21}V_{n-1}(A_2(x))$ to be able to compare the self loops. The result of re-writing the formulation will be $\mu_{21}V_{n-1}(A_2\mathbf{x}) - \mu_{21}V_{n-1}(D_1\mathbf{x})$. By adding $\pm\mu_{21}V_{n-1}(\mathbf{x})$, we finally can see the terms $\mu_{21}(\Delta_1V_{n-1}(D_1\mathbf{x}) + \mu_{21}\Delta_2V_{n-1}(\mathbf{x}))$. Note that in Theorem II.1, we have proven the non-negativity of $\Delta_iV_n(\mathbf{x})$ for all n and $\mathbf{x}$. Therefore we have:

$$\mu_{21}\Delta_1V_{n-1}(D_1\mathbf{x}) \geq 0,$$

and by $(C2)$ and the induction hypothesis $(P3)$, we have:

$$\mu_{22} r_1 \Delta_2 V_{n-1}(\mathbf{x}) \geq \mu_{21} r_2 \Delta_1 V_{n-1}(\mathbf{x}).$$

Note that, in the case where $x_2 + 1 = N_2$, as we noted before, $\lambda_2 \Delta_2 V_{n-1}(A_2\mathbf{x})$ equals zero (in the right hand side of $(P3)$). In Theorem II.2, we proved that there exists an upper bound, $\mathcal{B}_1^*$, to $\Delta_1 V_{n-1}(A_2(\mathbf{x}))$, for all $\mathbf{x}$. As a result, based on Condition $(C3)$, we have $\mu_{22} b_2 \geq \mu_{21} \mathcal{B}_1^*$. Therefore we have:

$$\mu_{22} \lambda_2 b_2 \geq \mu_{21} \lambda_2 \Delta_1 V_{n-1}(A_2\mathbf{x}).$$

We have shown that all the terms on the left hand side of $(P3)$ are greater than or equal to all the terms on the right hand side of $(P3)$. Therefore, the proof of $(P3)$ holds for stage n and as a result, for all the other stages. □

2.7.4 Proof of Theorem II.4

Proof of Theorem 5. *The proof is by induction.*

As the base of induction, we need to prove properties $(P4)$and $(P5)$ of the value function for the stage 0. Note that $V_0(\mathbf{x}) = 0, \forall \mathbf{x}$ by definition, and as a result properties $(P4)$ and $(P5)$ will hold trivially.

Induction Hypothesis (IH): We assume that all the properties $(P4)$ and $(P5)$ hold for $n-1$ and for all $x_1 \geq 1$. We next prove these properties for n as our inductive step.

Proof of $(P4)$: To prove this property, we first break down both sides of $(P4)$. On the left hand side of $(P4)$ we have $\mu_{21} \Delta_1 V_n(D_1\mathbf{x})$ which can be expanded as the following:

$$\mu_{21} \Delta_1 V_n(D_1\mathbf{x}) = \mu_{21} \big[V_n(x_1, x_2) - V_n(x_1 - 1, x_2) \big]. \tag{2.23}$$

On the right hand side of $(P4)$ we have $\mu_{22}\Delta_2 V_n(D_2\mathbf{x})$ which can be expanded as:

$$\mu_{22}\Delta_2 V_n(D_2\mathbf{x}) = \mu_{22}\big[V_n(x_1,x_2) - V_n(x_1,(x_2-1)^+)\big]. \tag{2.24}$$

In the next step, we expand $V_n(x_1,x_2)$ and $V_n(x_1-1,x_2)$ based on (2.5). Note that, based on the induction hypothesis we know that the first term of the maximization function in Equation (2.5) (i.e. $\mu_{21}\Delta_1 V(D_1\mathbf{x})$) is always greater than or equal to the other term in the maximization (i.e. $\mu_{22}\Delta_2 V(D_2\mathbf{x})$). This way, for $2 \le \mathbf{x}_1 < N_2$, we can expand the left hand side to get:

$$\mu_{21}\Delta_1 V(D_1\mathbf{x}) = \mu_{21}h_1 +$$

(2.25)

$$\alpha\mu_{21}\left\{\begin{array}{ll} \pi_1 r_1 + \lambda_1\Delta_1 V_{n-1}(\mathbf{x}) + \lambda_2\Delta_1 V_{n-1}(A_2D_1\mathbf{x}) + & (i)\\ (r_1(x_1-1)^+)\Delta_1 V_{n-1}(D_1^2\mathbf{x}) + (\mu_{11}+\mu_{21})\Delta_1 V_{n-1}(D_1^2\mathbf{x}) + & (ii)\\ r_2x_2\Delta_1 V_{n-1}(D_2D_1\mathbf{x}) + & (iii)\\ (1-\lambda_1-\lambda_2-r_1x_1-r_2x_2-\mu_{11}-\mu_{21})\Delta_1 V_{n-1}(D_1\mathbf{x}) & (iv)\end{array}\right\}.$$

Similarly, the right hand side can be expanded:

$$\mu_{22}\Delta_2 V_n(D_2\mathbf{x}) = \mu_{22}h_2 +$$

(2.26)

$$\alpha\mu_{22}\left\{\begin{array}{ll} \pi_2 r_2 + \lambda_2 b_2 I_{\{x_2=N_2\}} + \lambda_1\Delta_2 V_{n-1}(A_1D_2\mathbf{x}) + \lambda_2\Delta_2 V_{n-1}(\mathbf{x}) + & (i)\\ r_1x_1\Delta_2 V_{n-1}(D_1D_2\mathbf{x}) + (\mu_{11}+\mu_{21})\Delta_2 V_{n-1}(D_1D_2\mathbf{x}) + & (ii)\\ r_2(x_2-1)^+\Delta_2 V_{n-1}(D_2^2\mathbf{x}) + & (iii)\\ (1-\lambda_1-\lambda_2-r_1x_1-r_2x_2-\mu_{11}-\mu_{21})\Delta_2 V_{n-1}(D_2\mathbf{x}) & (iv)\end{array}\right\}.$$

In the next step we use a line-by-line comparison between the terms on the left hand side to show that $(P4)$ holds. For the first comparison, (2.25-i) and (2.26-i), using

$(C4)$ and the induction hypothesis $(P4)$ we can show that:

$$\mu_{21}(h_1 + \alpha(\pi_1 r_1 + \lambda_1 \Delta_1 V_{n-1}(\mathbf{x}) + \lambda_2 \Delta_1 V_{n-1}(A_2 D_1 \mathbf{x}))) \geq$$

$$\mu_{22}(h_2 + \alpha(\pi_2 r_2 + \lambda_2 b_2 I_{\{x_2 = N_2\}} \lambda_1 \Delta_2 V_{n-1}(A_1 D_2 \mathbf{x}) + \lambda_2 \Delta_2 V_{n-1}(\mathbf{x}))).$$

The next lines of terms to compare are (2.25-ii) and (2.26-ii). We first write $r_2 x_2$ as $r_2(x_2 - 1)^+ + r_2 I_{\{x_2 \neq 0\}}$. This way, using the induction hypothesis $(P4)$ we get:

$$\mu_{22}((r_1(x_1 - 1)\mu_{11} + \mu_{21})\Delta_1 V_{n-1}(D_1^2 \mathbf{x}) \geq \mu_{22}((r_1(x_1 - 1) + \mu_{11} + \mu_{21})\Delta_2 V_{n-1}(D_1 D_2 \mathbf{x}).$$

To compare lines (2.25-iii) and (2.26-iii), we rewrite $r_2 x_2$ as $r_2(x_2 - 1)^+ + r_2$. As a result by employing the induction hypothesis $(P4)$, we have:

$$\mu_{21}(r_2(x_2 - 1) + \mu_{22})\Delta_1 V_{n-1}(D_2 D_1 \mathbf{x} \geq \mu_{22}(r_2(x_2 - 1) + \mu_{22})\Delta_2 V_{n-1}(D_2^2 \mathbf{x}).$$

Note that as a result of the substitutions, we are left with a term associated with r_2 in the left hand side, and a term associated with r_1 in the right hand side. By $(C5)$ and the induction hypothesis $(P5)$, we have:

$$\mu_{21} r_2 \Delta_1 V_{n-1}(D_2 D_1 \mathbf{x}) \geq \mu_{22} r_1 \Delta_2 V_{n-1}(D_1 D_2 \mathbf{x}).$$

Finally the term associated with the self loops can be compared using the induction hypothesis $(P4)$. This way we can show that:

$$\mu_{21}(1 - \lambda_1 - \lambda_2 - r_1 x_1 - r_2 x_2 - \mu_{11} - \mu_{21})\Delta_1 V_{n-1}(D_1 \mathbf{x}) \geq$$

$$\mu_{22}(1 - \lambda_1 - \lambda_2 - r_1 x_1 - r_2 x_2 - \mu_{11} - \mu_{21})\Delta_2 V_{n-1}(D_2 \mathbf{x}).$$

In case $\mathbf{x}_1 = 1$, we will have the term $\mu_{22}\Delta_2 V_{n-1}(D_2 D_1 \mathbf{x})$ in the left hand side instead of the term $(\mu_{11} + \mu_{21})\Delta_1 V_{n-1}(D_1^2 \mathbf{x})$ (by optimality of non-idling policy, i.e. instead of idling, the server will work on job type-2). In Theorem II.1, we have proven the non-negativity of $\Delta_i V_n(\mathbf{x})$ for all n and $\mathbf{x}$. Therefore we have:

$$\mu_{22}\Delta_2 V_{n-1}(D_2 D_1 \mathbf{x}) \geq 0,$$

and by $(C5)$ and the induction hypothesis $(P5)$, we have:

$$\mu_{21} r_2 \Delta_1 V_{n-1}(D_2 D_1 \mathbf{x}) \geq \mu_{22}(r_1 + \mu_{11} + \mu_{21}) \Delta_2 V_{n-1}(D_1 D_2 \mathbf{x}).$$

Note that, in the case where $x_1 = N_1$, as we noted before, $\lambda_1 \Delta_1 V_{n-1}(\mathbf{x})$ equals zero (in the right hand side of $(P4)$. In Theorem II.2, we proved that there exists an upper bound, $\mathcal{B}_2^*$, to $\Delta_2 V_{n-1}(A_1 D_2(\mathbf{x}))$, for all $\mathbf{x}$. As a result, based on Condition $(C6)$, we have $\mu_{21} b_1 \geq \mu_{22} \mathcal{B}_2^*$. Therefore we have:

$$\mu_{21} \lambda_1 b_1 \geq \mu_{22} \lambda_1 \Delta_2 V_{n-1}(A_1 D_2 \mathbf{x}).$$

We have shown that all the terms on the left hand side of $(P4)$ are greater than or equal to all the terms on the right hand side of $(P4)$. Therefore, the proof of $(P4)$ holds for stage n and as a result, for all the other stages. **Proof of $(P5)$**: We have defined $(P5)$ as:

$$\mu_{21} \Delta_1 V_n(\mathbf{x}) \geq \mu_{22} \Delta_2 V_n(\mathbf{x}) \ \forall \mathbf{x}.$$

We can expand the left hand side of the above equation to get:

$$\mu_{21} \Delta_1 V(\mathbf{x}) = \mu_{21} h_1 +$$

(2.27)

$$\alpha \mu_{21} \left\{ \begin{array}{ll} \pi_1 r_1 + \lambda_1 \Delta_1 V_{n-1}(A_1 \mathbf{x}) + \lambda_2 \Delta_1 V_{n-1}(A_2 \mathbf{x}) + & (i) \\ r_1 x_1 \Delta_1 V_{n-1}(D_1 \mathbf{x}) + (\mu_{11} + \mu_{21}) \Delta_1 V_{n-1}(D_1 \mathbf{x}) + & (ii) \\ r_2 x_2 \Delta_1 V_{n-1}(D_2 \mathbf{x}) + r_2 \Delta_1 V_{n-1}(\mathbf{x}) & (iii) \\ (1 - \lambda_1 - \lambda_2 - r_1(x_1 + 1) - r_2(x_2 + 1) - \mu_{11} - \mu_{21}) \Delta_1 V_{n-1}(\mathbf{x}) & (iv) \end{array} \right\}.$$

We can also expand the right hand side of (P_5) to get:

$$\mu_{22}\Delta_2 V_n(\mathbf{x}) = \mu_{22}h_2+$$

(2.28)

$$\alpha\mu_{22}\left\{\begin{array}{ll} \pi_2 r_2 + \lambda_2 b_2 I_{\{x_2+1=N_2\}} + \lambda_1\Delta_2 V_{n-1}(A_1\mathbf{x}) + \lambda_2\Delta_2 V_{n-1}(\mathbf{x})+ & (i)\\ r_1 x_1 \Delta_2 V_{n-1}(D_1\mathbf{x}) + (\mu_{11}+\mu_{21})\Delta_2 V_{n-1}(D_1\mathbf{x})+ & (ii)\\ r_2 x_2 \Delta_2 V_{n-1}(D_2^2\mathbf{x}) + r_1\Delta_2 V_{n-1}(\mathbf{x}) & (iii)\\ (1-\lambda_1-\lambda_2-r_1(x_1+1)-r_2(x_2+1)-\mu_{11}-\mu_{21})\Delta_2 V_{n-1}(\mathbf{x}) & (iv)\end{array}\right\}.$$

In the next step we use a line-by-line comparison between the terms on the left hand side to show that $(P5)$ holds. For the first comparison, (2.27-i) and (2.28-i), using $(C4)$ and the induction hypothesis $(P5)$ we can show that:

$$\mu_{21}(h_1 + \alpha(\pi_1 r_1 + \lambda_1\Delta_1 V_{n-1}(A_1\mathbf{x}) + \lambda_2\Delta_1 V_{n-1}(A_2\mathbf{x}))) \geq$$
$$\mu_{22}(h_2 + \alpha(\pi_2 r_2 + \lambda_2 b_2 I_{\{x_2+1=N_2\}} + \lambda_1\Delta_2 V_{n-1}(A_1\mathbf{x}) + \lambda_2\Delta_2 V_{n-1}(A_2\mathbf{x}))).$$

The next lines of terms to compare are (2.27-ii) and (2.28-ii). Using the induction hypothesis $(P5)$ we get:

$$\mu_{22}((r_1x_1+\mu_{11}+\mu_{21})\Delta_1 V_{n-1}(D_1\mathbf{x}) \geq \mu_{22}((r_1x_1+\mu_{11}+\mu_{21})\Delta_2 V_{n-1}(D_1\mathbf{x}).$$

To compare lines (2.27-iii) and (2.28-iii) by employing the induction hypothesis $(P5)$, we have:

$$\mu_{21} r_2 x_2 \Delta_1 V_{n-1}(D_2\mathbf{x}) \geq \mu_{22} r_2 x_2 \Delta_2 V_{n-1}(D_2\mathbf{x}).$$

Note that we are left with a term associated with r_2 in the left hand side, and a term associated with r_1 in the right hand side. By $(C5)$ and the induction hypothesis $(P5)$, we have:

$$\mu_{21} r_2 \Delta_1 V_{n-1}(\mathbf{x}) \geq \mu_{22} r_1 \Delta_2 V_{n-1}(\mathbf{x}).$$

Finally the term associated with the self loops can be compared using the induction hypothesis $(P5)$. This way we can show that:

$$\mu_{21}(1-\lambda_1-\lambda_2-r_1(x_1+1)-r_2(x_2+1)-\mu_{11}-\mu_{21})\Delta_1 V_{n-1}(\mathbf{x}) \geq$$
$$\mu_{22}(1-\lambda_1-\lambda_2-r_1(x_1+1)-r_2(x_2+1)-\mu_{11}-\mu_{21})\Delta_2 V_{n-1}(\mathbf{x}).$$

In case $\mathbf{x}_1 = 0$, we will have the term $\mu_{22}(V_{n-1}(A_1\mathbf{x}) - V_{n-1}(D_2\mathbf{x}))$ in the left hand side and the term $(\mu_{11}+\mu_{21})\Delta_2 V_{n-1}(\mathbf{x}) + \mu_{22}\Delta_2 V_{n-1}(D_2\mathbf{x})$ in the right hand side (by optimality of non-idling policy, i.e. instead of idling, the server will work on job type-2). By adding $\pm\mu_{22}V_{n-1}(A_1\mathbf{x})$ to the left hand side, we have:

$$\mu_{21}r_2\Delta_1 V_{n-1}(\mathbf{x}) \geq \mu_{22}(r_1+\mu_{11}+\mu_{21})\Delta_2 V_{n-1}(\mathbf{x}) \quad \text{by IH } (P5)\text{and } (C5).$$
$$\mu_{21}\mu_{22}\Delta_1 V_{n-1}(D_2\mathbf{x}) \geq \mu_{22}\mu_{22}\Delta_2 V_{n-1}(D_2\mathbf{x}) \quad \text{by IH } (P5).$$
$$\mu_{21}\mu_{22}\Delta_2 V_{n-1}(A_1D_2\mathbf{x}) \geq 0 \quad \text{by } (P1).$$

Note that, in the case where $x_1 = N_1$, as we noted before, $\lambda_1\Delta_1 V_{n-1}(A_1\mathbf{x})$ equals zero (in the right hand side of $(P5)$. In Theorem II.2, we proved that there exists an upper bound, $\mathcal{B}_2^*$, to $\Delta_2 V_{n-1}(A_1\mathbf{x})$, for all $\mathbf{x}$. As a result, based on Condition $(C6)$, we have $\mu_{21}b_1 \geq \mu_{22}\mathcal{B}_2^*$. Therefore we have:

$$\mu_{21}\lambda_1 b_1 \geq \mu_{22}\lambda_1\Delta_2 V_{n-1}(A_1\mathbf{x}).$$

We have shown that all the terms on the left hand side of $(P5)$ are greater than or equal to all the terms on the right hand side of $(P5)$. Therefore, the proof of $(P5)$ holds for stage n and as a result, for all the other stages. $\square$

2.7.5 Generating the Benchmarking Test Suite

To generate the problem instances, three main *cost regimes* are combined with a set of 122 *scenarios* and 6 levels of buffer size (5, 10, 15, 20, 25, and 30). Table 2.4 illustrates the three cost regimes. The scenarios are illustrated in Table 2.5.

Cost regimes	Description	Holding cost	Reneging penalty	Blocking penalty
1	Jobs type-1 and type-2 have similar costs	$h_2 = h_1$	$\pi_2 = \pi_1$	$b_2 = b_1$
2	Job type-1 has higher costs	$h_2 = 0.7 * h_1$	$\pi_2 = 0.7 * \pi_1$	$b_2 = 0.7 * b_1$
3	Job type-2 has higher costs	$h_2 = 1.3 * h_1$	$\pi_2 = 1.3 * \pi_1$	$b_2 = 1.3 * b_1$

Table 2.4: Cost regimes used for generating the benchmarking test suite.

Figure	Parameter	Values
2.6(L)	λ_1	0.1, 0.2, 0.5, 0.75, 1, 12, 1.5, 1.75, 2, and 2.5
2.6(R)	λ_2	0.1, 0.2, 0.5, 0.75, 1, 12, 1.5, 1.75, 2, and 2.5
2.7(L),2.8(L)	b_1	0, 1, 2, 5, 8, 10, 20, 30, 40, 50, 60, 70, 80, 90, and 100
2.7(R),2.8(R)	b_2	0, 1, 2, 5, 8, 10, 20, 30, 40, 50, 60, 70, 80, 90, and 100
2.9(L)	μ_{21}	0.2, 0.5, 0.7, 1, 1.5, 2, 2.5, and 3
2.9(R)	μ_{22}	0.2, 0.5, 0.7, 1, 1.5, 2, 2.5, and 3
2.10(L)	r_1	0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.4, and 0.5
2.10(R)	r_2	0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.4, and 0.5
2.11(L)	π_1	0, 0.01, 0.05, 0.5, 1, 2, 4, 8, 16, 32, 64, and 128
2.11(R)	π_2	0, 0.01, 0.05, 0.5, 1, 2, 4, 8, 16, 32, 64, and 128
2.12(L)	h_1	0.5, 0.75, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, and 5
2.12(R)	h_2	0.5, 0.75, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, and 5
2.13	N	5, 10, 15, 20, 25, 30, and 35

Table 2.5: Scenarios used for generating the benchmarking test suite, with the first column indicating the Figure illustrating each sensitivity analysis.

To generate each problem instance, we start with default values illustrated in Table 2.6. We then iterate over the three cost regimes (from Table 2.4) and 122 scenarios (from Table 2.5). This procedure is then repeated for other values of buffer size.

Parameter	Default value	Parameter	Default value
μ_{11}	1	h_1	1
μ_{21}	1	h_2	1
μ_{22}	1	π_1	5
λ_1	1	π_2	5
λ_2	1	b_1	50
r_1	0.05	b_2	50
r_2	0.05	N	10

Table 2.6: Default values used for generating the benchmarking test suite.

CHAPTER III

Fixed Task Zone Chaining: Worker Coordination and Zone Design for Inexpensive Cross-training in Serial CONWIP Lines

3.1 Introduction

Manufacturing and service systems are currently facing uncertainty in many aspects of their operations, including demand fluctuations, supply disruption, and processing time variability. Increasing flexibility can improve systems' ability to adapt to changes and build robust capability to nullify the disruptive impacts of uncertainty. Worker cross-training is often proposed as an effective means to achieve flexibility by improving capacity management and alleviating bottlenecks. This can result in reducing cycle time, increasing throughput, and improving workers' utilization. Cross-training can also reduce systems' reliance on excessive inventory (or buffer) levels.

Cross-training, however, can be expensive. Additional training, providing supplementary tools and the need to redesign work stations often contribute to the cost of cross-training. Furthermore, as the number of skills for each worker increases, their ability to focus on a specific task can be reduced. Thus, finding the "proper" level of cross-training is a critical step toward achieving effective flexibility.

In this chapter we analyze the effect of introducing a minimal level of cross-training

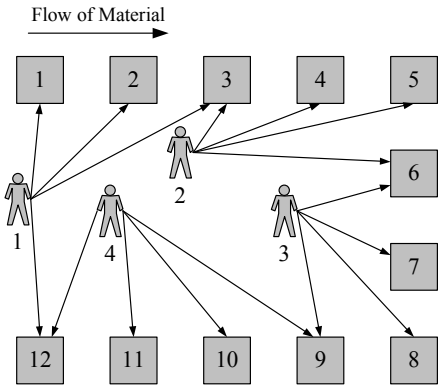


Figure 3.1: A conceptual illustration of the FTZC structure.

to U-shaped CONstant Work In Process (CONWIP) serial lines. The main idea here is to divide stations into a set of *zones* where a worker is assigned to each zone as a primary resource. To increase flexibility, neighboring zones are overlapped such that the stations at the boundaries of each zone are shared with another zone. This way, workers at each zone have a set of *fixed* stations that are assigned only to them. They are also cross-trained to work on two *shared* stations which are located at the two ends of their zones. Note that a zone may not necessarily have fixed stations, but it always have two shared stations. A conceptual representation of a FTZC structure with 12 stations and four workers is illustrated in Figure 3.1.

When the extent of cross-training is limited, three questions become very critical. First, given the choice of each worker's skill set, a worker needs to prioritize the order in which she picks jobs from each station in her skill set. Second, given the fact that two workers can pick a job from a shared station, it is important to know which worker should pick that job. Third, given the processing time at each station, we want to find the most effective zone design for the line.

To answer the first two questions, we build upon the work of Williams [93] in which the idea of "Fixed Task Zone Chaining" (FTZC) has been introduced. Williams

incorporates two main ideas of *Zoned training* introduced by McClain et al. [66] and *two-skill chaining* introduced by Jordan and Graves [54] into the new FTZC paradigm. His work, however, only addresses a very specific model in which the production line has 12 stations and 4 identical workers. He proposes two worker heuristic policies and studies the impact of those heuristics under a given symmetric zone structure and a balanceable production lines. In this chapter, we generalize the work of Williams [93] to a generic production system consisting of W workers and N stations. We extend his work to production lines that are not balanceable through a symmetric zone structure. Our work also provides insight into the dynamic line balancing (DLB) characteristics of our zone chain structure. We show how *balanceable* zone structures can help FTZC yield high throughput, while poorly designed zone structures can hinder the potential benefits of FTZC.

To address the third question, we relax the symmetric zone structure assumption and introduce a zone assignment algorithm to design balanceable lines that are customized to FTZC structure. This algorithm is based on a calculation of the optimal fraction of time that a worker should be assigned to a work station to balance the line for maximum throughput.

This chapter is organized as follows: after surveying the literature in Section 3.2, the problem description and definitions are introduced in Section 3.3. In Section 3.4, given a fixed zone structure, we analytically describe some characteristics of the optimal worker control policy in a general FTZC structure. In Section 3.5, we develop algorithms to design the zone structures to maximize throughput under FTZC. In Section 3.6, we design an extensive benchmarking test suite and compare the FTZC structure to two-skill zone chain and classic CONWIP approximation.

3.2 Literature Survey

There is extensive literature on cross-training and the value of flexibility in production systems. Hopp and Van Oyen [49] focus on developing measures to quantify the value of cross-training in any firm and address the importance of worker scheduling policies to exploit all benefits of flexibility. Treleven [83] and Andradottir et al. [9] provide thorough literature surveys on workforce agility and its benefits across different industries.

As we mentioned before, our research in this chapter is closely related to two main streams of research on cross-training: first, the concept of "overlapping zones" that was introduced by McClain et al. [66]. Second, the idea of skill chaining (cross-training workers in a line to link all task-types into a chain) which was introduced by Jordan and Graves [54]. McClain et al. [66] mainly focus on the size of overlap and its effect on systems' performance, whereas Jordan and Graves [54] conduct research to address the effect of production flexibility in systems.

Building upon the works of McClain et al. [66] and Jordan and Graves [54], our model uses both ideas of overlapping zones and two-skill chaining. But not all stations in our line are cross-trained. There are exactly two stations in each zone which are cross-trained. Any station between the neighboring stations will be fixed to only one worker.

Increased complexity can be a downside of cross-training in queueing systems. This is mostly because cross-training increases the set of feasible actions and the need to design a worker coordination policy. Another potential issue with cross-training is the increased cost of training. In addition to the obvious cost of training more servers, a server's learning curve may also start to diminish as her skill set broadens.

Therefore, in most practical settings, as studied by Hopp et al. [48], McClain et al. [66], Jordan and Graves [54], and Parvin et al. [71], a limited degree of flexibility can significantly improve a wide range of performance measures while keeping the cost of training manageable. As we have seen in moving worker modules (MWM) (see Askin and Chen [10]), there are more working stations than workers. The issue of which station a worker should pick at each given time becomes critical. The bucket brigade system (BBS) is a good example of MWM. BBS was introduced by Bartholdi and Eisenstein [16] as well as Bartholdi et al. [17]. Bartholdi and Eisenstein [16] shows that when the processing times are deterministic, sorting workers from slowest to fastest will always result in a balance line. Bartholdi [17] incorporates stochastic processing times, and with a set of mild assumptions, proves the validity of the result obtained by Bartholdi and Eisenstein [16]. A BBS is very effective in environments such as warehouses order picking where the barriers to "cross-training" are small; however, the FTZC paradigm is designed for environments not suited to a BBS (i.e., very limited cross-training). Bischak [20] considers systems with random processing times, assuming that preemption is allowed. In contrast, our research restricts attention to systems for which jobs cannot be preempted.

Andradottir et al. [6] investigate the control of two servers serving two stations with finite buffers, characterizes the optimal policy, and develops a near-optimal sever assignment policy for a more general system. It allows more than one worker to work on any single job (with additive service rates), provided there can be at most one job in service at any station. We require fewer workers than stations and allow only one worker per job but two workers can work on two jobs if the station is overlapped by them. Gel et al. [40] introduce the concept of "fixed before shared" for WIP constrained lines. This idea in that research always prioritizes a fixed station

(the one that can only be served by one worker) over a shared task-type. Later we use this principal to develop our proposed worker coordination heuristic policy.

Our main contribution is to build upon past research (especially Williams [93]) to offer a design and control methodology together with FTZC paradigm for the purpose of achieving high throughput lines while explicitly incorporating CONWIP control to limit WIP and keep cycle times short. To implement this vision, we offer an easily implementable heuristic policy that performs well, and a new zone design algorithm that improves system performance.

3.3 Problem Description

3.3.1 Problem Definition

We consider a U-shaped serial production line consisting of N stations and W workers, where $W < N$ (there are strictly less workers than stations). The only reason for this is that the FTZC paradigm will require precisely one of the W workers to serve at both the first and last stations of the line. U-shaped structure has the advantage of the first and the last station being close, so the travel time between these stations is negligible.

Raw materials enter the first station of the line (called station 1) and will proceed sequentially downstream through to the last station. As illustrated in Figure 3.1, there are strictly fewer workers than stations in the FTZC canonical system, and each worker is assigned to a zone that must overlap the zones of two neighboring workers by exactly one station on either end of the zone.

Similar to model proposed by Williams [93], each station in our model performs just one unique task-type. In this model, we do **not** allow task preemption, meaning that once a worker starts a task, it cannot be handed off until its completion. Successive processing times of task-type i at each station are i.i.d. of general distribution

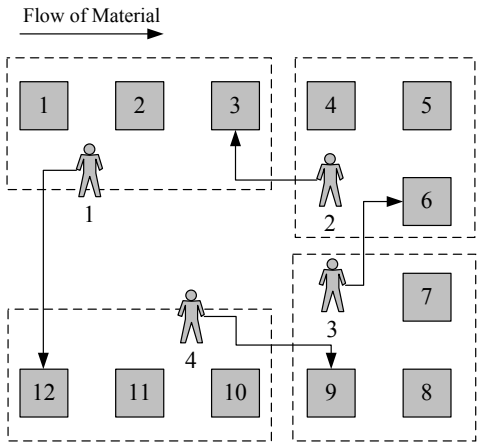


Figure 3.2: Primary zones in a symmetrical FTZC structure.

and all the workers are identical in this model. By identical and standard, we mean that the processing time in each station is independent of the worker who is assigned to the station and is based on a standard work rate of 1.

The release of raw material to the line follows CONstant Work-In-Process (CONWIP) policy. Thus, at any given time there are a fixed number of jobs in the system.

Each worker's zone is defined by two shared stations denoted by set $\mathcal{S}_k$ and, possibly, a set of fixed stations, $\mathcal{F}_k$, for worker k, in between them. Note that in FTZC structures, two adjacent zones overlap by one station. FTZC structures are designed to significantly reduce the number of skills to be cross-trained, and are intended for environments where extensive cross-training is undesirable. For example in a system of 4 workers and 12 stations, FTZC will reduce the number of skill training by almost 80% compared to full cross-training. Only one worker at a time can work on a specific job, but two workers can work simultaneously at a station provided at least two jobs are available at a shared station.

We introduce the term *primary zone* to better categorize the stations. Each worker is trained to work on a set of stations, which we define as a *zone*. We call a subset of the zone of worker k that excludes one station from one end of the zone,

the k^{th} *primary zone*, denoted $\mathcal{Z}_k$. It is convenient to exclude the "upstream" shared station from the *primary zone* to avoid over-counting the stations. Thus, the union of primary zones is the set of all stations ($\mathcal{Z} = \bigcup_{k=1}^{W} \mathcal{Z}_k$). A formal definition of upstream and downstream shared stations for each worker is presented in Definition III.1.

3.3.2 Notation and Definitions

In this section we introduce some key notation and definitions:

N	Total number of stations in a line
W	Total number of workers in a line and total number of zones in a line
$\mathcal{W}$	Set of workers in the line, $\mathcal{W} = \{1, ..., W\}$
$\mathcal{F}_k$	Set of fixed stations that worker k is trained for, where $k \in \mathcal{W}$
$\mathcal{S}_k$	Set of shared stations for which worker k is trained, where $k \in \mathcal{W}$, note that $\mathcal{F}_k \bigcup \mathcal{S}_k$ is the skill set of worker k
$\mathcal{Z}_k$	Primary zone for worker k
$\mathcal{Z}$	Set of stations in the line, $\mathcal{Z} = \bigcup_{k=1}^{W} \mathcal{Z}_k$
$\mathcal{S}$	Set of shared stations, $\mathcal{S} = \bigcup_{k=1}^{W} \mathcal{S}_k$
T_i	Mean processing time of the i^{th} station, where $i \in \mathcal{Z}$
T_0	Raw processing time of all stations, $T_0 = \sum_{i \in \mathcal{Z}} T_i$
$\oplus$	Mod N addition defined as, $i \oplus j = i + j$ if $i + j \leq N$; otherwise, $i \oplus j = i + j - N$, $\forall i, j \in \mathcal{Z}$
$\ominus$	Mod N subtraction defined as, $i \ominus j = i - j$ if $i - j \geq 1$; otherwise, $i \ominus j = i - j + N$, $\forall i, j \in \mathcal{Z}$

Definition III.1. Station i is the upstream (downstream) shared station of worker

k if, and only if, with $i \in \mathcal{F}_k \bigcup \mathcal{S}_k$ and station $i \ominus 1 (i \oplus 1)$ is not a member of set $\mathcal{F}_k \bigcup \mathcal{S}_k$.

Definition III.2. Let i be the upstream shared station of worker k. Worker k is a same-side worker if, and only if, $f > i$ for all $f \in \mathcal{F}_k$. A FTZC design has at least $W - 1$ same-side workers.

Definition III.3. Last Buffer First Served (LBFS) rule is a priority rule that defines the order of stations visited within a worker's skill zone. Under this rule, the worker checks for an available job at the most downstream station in her zone, which is the downstream shared station. Any job at the most downstream station always has first priority; otherwise the worker moves upstream (in the reverse order of material flow) and completes the task-type at the most downstream station at which a job is present (same definition as used by Williams [93]).

Definition III.4. Worker $D_s (U_s)$ is called the downstream (upstream) worker of shared station $s \in \mathcal{S}$, if and only if, station $s \oplus 1 (s \ominus 1)$ is a member of the set $\mathcal{F}_{D_s} \cup \mathcal{S}_{D_s} (\mathcal{F}_{U_s} \cup \mathcal{S}_{U_s})$.

3.3.3 Critical WIP

As defined by Hopp and Spearman [47], the critical WIP level, ω_0, is the product of the capacity of the bottleneck station times the total (raw) processing time of the entire line. Thus, it is the WIP level at which an ideal line without congestion achieves maximum throughput with minimum cycle time. Critical WIP is useful as a way of quantifying how lean a system operates via the ratio of WIP/ω_0. The following proposition applies the definition of critical WIP to the FTZC paradigm, giving us a useful WIP operating point that can be referenced.

Proposition III.5. *In a symmetric FTZC structure (i.e. one with an equal number of stations in every zone) with equal deterministic processing times at each station that operates under a LBFS protocol, the critical WIP is equal to the number of zones, $\omega_0 = W$, and the system achieves maximum throughput for any WIP level of ω_0 or greater.*

See Section 3.8 for all the proofs.

3.4 Worker Coordination Policy

We first analytically examine some characteristics of the optimal (or optimal within a class) worker coordination policies, given the zone definitions. Note that a full characterization of optimal policies is extremely difficult. Then, in Section 3.4.2, we exploit these results in designing a heuristic to control which jobs a worker picks from stations in her zone to achieve high throughput with limited WIP.

3.4.1 Analytical Results

Theorems III.6, III.7, and III.8 determine policy properties that maximize the number of job completions along any sample path. We acknowledge that very similar versions of these theorems were presented in Williams [93]. Minor improvements were contributed (e.g. checking that they apply in a more general FTZC system model) and correcting several errors. We include these theorems here for completeness. Given any realization ω of the stochastic model, the sequence of job arrival and service times is known. Let $D_t^\pi(\omega)$ be the number of completed jobs by time t under a worker allocation strategy π. We characterize a policy π^* such that $D_t^{\pi^*}(\omega) \geq D_t^\pi(\omega) \quad \forall \quad t, \pi, \omega$; i.e. a policy that is at least as good as any other policy sample pathwise. One important outcome from these theorems is the significance of the *Last Buffer First Served* (LBFS) rule to determine the priority of a worker's task-type.

Theorem III.6. *Assuming that all workers are standard, when no jobs are waiting at the downstream shared station of a same-side worker k, $k \in \mathcal{W}$, the number of job completions is maximized along every sample path by a policy that (i) does not idle worker k when fixed task-types are present, and (ii) follows the LBFS rule among the worker's set of task-types.*

This result also reinforces the intuitive rule-of-thumb that to maximize throughput within the line, priority should be given to jobs that are closer to finished goods (downstream) than raw material (upstream). In Theorem III.7, we prove that among the policies that prioritize fixed jobs over shared ones, a policy consistent with LBFS performs optimally for most of the workers.

Theorem III.7. *A policy that follows the LBFS rule maximizes the number of job completions along every sample path within the class of policies that give priority to a worker's fixed task-types versus her shared tasks, provided the worker's fixed task stations are on the same side of the line.*

In the next theorem, we consider a case where two workers are competing for one job at a shared station $s \in \mathcal{S}$, at time t. To determine the best assignment, Theorem III.8 proves that to maximize throughput, the downstream worker (as defined in Definition III.4) should pick that job, provided that the upstream worker has at least one fixed task in her training set.

Theorem III.8. *When there is only a single job available to the downstream worker D_s and the upstream worker U_s at a shared station s, a policy that gives priority to D_s over U_s maximizes the number of job completions along every sample path, if $\mathcal{F}_{U_s} \neq \emptyset$.*

3.4.2 Fixed First Max Shared (FFMS)

Based on the analytical results from Section 3.4.1, Fixed First Max Shared (FFMS) was originally developed by Williams. This policy exploits four ideas as follows. First, it gives strict priority to the worker's fixed stations over the shared stations, and will only complete a job at shared stations if no jobs are at a fixed station; hence the *fixed first* part of the name. This rule is partially justified by Theorem III.6. Intuitively, the loss of throughput is the direct result of worker starvation. Downstream workers will tend to starve when there are upstream bottlenecks at *fixed task-type stations*, because these are the stations where the system has no flexibility. Second, within the fixed stations, the LBFS rule is applied such that the downstream fixed station is preferred to the upstream fixed station. The implementation of this rule is supported by Theorem III.7 and the work of Koole and Righter [57], which focused on systems with no shared tasks and showed that the LBFS policy is optimal in many cases. In the long-run, the FFMS policy "pools" the fixed tasks for each server. Third, if no jobs are available at the fixed stations, the worker picks a job from the longest queue at the shared stations; hence the *max shared* part of the name (and note that if both queues have the same number of jobs, the worker prioritizes the most downstream station). Fourth, according to Theorem III.8, if two workers are simultaneously available at the same shared station with only one job, preference is given to the most downstream worker.

3.5 Improving Zone Structure

The FTZC approach is intended to provide sufficient flexibility to allow some variation in mean processing times. Balance is trivially achieved in a symmetrical system such as the one in Proposition III.5. Extreme variation in processing times may not

permit a "low-cost" structure such as FTZC to achieve dynamic line balancing, even with arbitrary WIP. To define *balanceable lines*, we will not reference any specific policy.

Our goal in this section is to develop a method to improve a given zone structure such that it can generate high throughput. We introduce a zone assignment algorithm called ZonA which finds a zone structure that results in a balanceable line which achieves maximum throughput of W/T_0 under FTZC. We then describe a set of conditions that guarantee this achievement.

Considering only the specified FTZC zones and mean processing times, we take the approach of the past literature that the FTZC system is balanceable if the deterministic (fluid) LP model can achieve the maximum throughput of W/T_0. We define a linear program (similar to that found in other works such as Andradottir et al. [7]) that maximizes throughput given a FTZC structure using the following notation:

Θ	Line throughput
$\mathcal{W}$	Set of workers in the line
$\mathcal{F}$	Set of fixed stations in the line
$y_{s_u k}$	Fraction of time that worker k works at her upstream shared station
$y_{s_d k}$	Fraction of time that worker k works at her downstream shared station
z_i	Fraction of time allocated to fixed station i
I_k	Fraction of idle time of worker k
$\tilde{T}_{(k)}$	Mean processing time at the k^{th} shared station

Based on the above notation, the throughput maximization linear program (LP) is

defined by (3.1)-(3.7):

LP:

$$\max \Theta \tag{3.1}$$

subject to :

$$\Theta \leq \frac{z_i}{T_i} \qquad \forall i \in \mathcal{F} \tag{3.2}$$

$$\Theta \leq \frac{y_{s_u k} + y_{s_d k \ominus 1}}{\tilde{T}_{(k)}} \qquad \forall k \in \mathcal{W} \tag{3.3}$$

$$I_k + y_{s_u k} + \sum_{i \in \mathcal{F}_k} z_i + y_{s_d k} = 1 \qquad \forall k \in \mathcal{W} \tag{3.4}$$

$$I_k, y_{s_u k}, y_{s_d k} \geq 0 \qquad \forall k \in \mathcal{W} \tag{3.5}$$

$$z_i \geq 0 \qquad \forall i \in \mathcal{F} \tag{3.6}$$

$$\Theta \geq 0. \tag{3.7}$$

In the formulation above, the objective function (3.1) is to maximize line throughput (Θ). The inequalities of (3.2) ensure that throughput is bounded above by the fraction of time allocated to the fixed station i divided by the processing time of that station. Similar to (3.2), the inequalities of (3.3) describe the upper bound on the throughput imposed by the fraction of time spent at each shared station. Note that a shared station has two workers assigned to it, the upstream and the downstream workers. Therefore, on the right-hand-side, we need to add the fraction of times each of these two workers spend on the shared station. The equations of (3.4) add the fractions of time worker k spends at each station (fixed and shared) and finds the excess capacity of worker k and assigns it to the idle time, I_k. Inequalities (3.5)-(3.7) represent non-negativity bounds on decision variables.

Note that the solution of the LP provides optimal worker allocations for the deterministic fluid model. There is no guarantee that the allocations resulting from

the application of our simple heuristic FFMS to the stochastic model will match those of the LP. It is important to keep in mind that the value of FFMS rests in its good performance, which is supported by our numerical investigation in Section 3.6. On the other hand, the main uses of the LP are to support the proof of the ZonA algorithm's ability to construct balanceable lines (Section 3.5.1) and the definition of the Classic CONWIP Approximation (Section 3.6.4). Next, we characterize optimal worker allocations for balanceable lines.

Proposition III.9. *Let* $\Theta^*, I_k^*, z_i^*, y_{s_u k}^*$,*and* $y_{s_d k}^*$ *be a solution of the linear program. If the line is balanceable, that is,* $\Theta^* = \frac{W}{T_0}$*, we have* $I_k^* = 0$*,* $y_{s_u k}^* + y_{s_d k \ominus 1}^* = \frac{W}{T_0}\tilde{T}_{(k)}$ *for all* $k \in \mathcal{W}$*, and* $z_i^* = \frac{W}{T_0}T_i$ *for all* $i \in \mathcal{F}$*.*

Let $\widehat{T}_j$ denote the effective service time of station j, so $\widehat{T}_i = \frac{T_i}{z_i^*} \forall i \in \mathcal{F}$, and $\widehat{T}_k = \frac{\tilde{T}_{(k)}}{y_{s_u k}^* + y_{s_d k \ominus 1}^*} \forall k \in \mathcal{W}$. Therefore, when the line is balanceable, Proposition III.9 implies that $\widehat{T}_j = \frac{T_0}{W}$ for all $j \in \mathcal{S} \bigcup \mathcal{F}$, a result used in the approximate model presented in Section 3.6.4.

3.5.1 Zone Assignment Algorithm (ZonA)

The idea behind this algorithm is to construct a work-sharing structure such that the workload that is fixed to any set of k consecutive workers does not exceed $k\frac{T_0}{W}$, which is a necessary condition for the line to be balanceable. We do this by first picking any station (named station one in this section only) to be a shared station. Then, with stations numbered consecutively toward the most downstream station (the direction of materials flow in Figure 3.1), the remaining $W - 1$ shared stations (named S_2, S_3,...,S_W) are given by:

$$S_i = \min\left\{k : \sum_{l=2}^{k} T_l \geq (i-1)\frac{T_0}{W}\right\}. \tag{3.8}$$

The zone boundaries for worker i are stations S_i and S_{i+1}, where $S_1 = S_{W+1} = 1$. The stations between S_i and S_{i+1}, if any, are "fixed" for worker i. Equation (3.8) ensures that $\sum_{l=2}^{S_{k+1}-1} T_l$, the total processing time of tasks that are served exclusively by workers $1, ..., k$, is less than $k\frac{T_0}{W}$.

Theorem III.10. *Having $T_i < \frac{T_0}{W}$ for all $i \in \{1, 2, ...N\}$ is a sufficient condition for the ZonA algorithm to produce a balanceable FTZC structure.*

We remark that the choice of the "first" shared station when applying the ZonA algorithm defines a FTZC structure that balances the line, but any station can be considered as station 1 to construct a FTZC solution. This way, we can have at most N different zone structures that balance the line; however, these structures may not perform identically due to variation in mean processing times. In Section 3.5.2, we develop a *Degree of Imbalance* (DOI) metric to select the particular FTZC structure that results in a minimum DOI value; DOI also serves as an indicator that correlates with low throughput for any given FTZC design.

3.5.2 Degree of Imbalance (DOI) and Full Specification of the ZonA Algorithm

The DOI metric captures the inherent variation in the mean processing times of stations *within* each zone as well as variation *across* the zones. DOI is a non-negative index for which zero corresponds to a perfectly balanced line and increasing deviations in the distribution of work content within and across zones lead to increased values. A high DOI indicates greater variation and a more significant bottleneck effect, both of which correlate with decreased throughput.

An "optimal" metric is beyond our scope; however, the results of our ZonA algorithm below are fairly insensitive to the DOI metric used. The simple and intuitive DOI metric below was found to work as well or better than several other metrics

tested.

Let $\mu_{\mathcal{Z}_k}$ and $\sigma_{\mathcal{Z}_k}$ be the average and the standard deviation of processing times of stations within primary zone $\mathcal{Z}_k$, calculated as the following:

$$\mu_{\mathcal{Z}_k} = \frac{\sum_{i \in \mathcal{Z}_k} T_i}{|\mathcal{Z}_k|}, \tag{3.9}$$

$$\sigma_{\mathcal{Z}_k} = \sqrt{\frac{\sum_{i \in \mathcal{Z}_k} (T_i - \mu_{\mathcal{Z}_k})^2}{|\mathcal{Z}_k|}}. \tag{3.10}$$

Let $\sigma_{\mu_{\mathcal{Z}}}$ capture the standard deviation of the mean processing times across the zones, and because the mean is known, $|\mathcal{W}|$ is used in the following equation.

$$\sigma_{\mu_{\mathcal{Z}}} = \sqrt{\frac{\sum_{k \in \mathcal{W}} (\mu_{\mathcal{Z}_k} - T_0/N)^2}{|\mathcal{W}|}}. \tag{3.11}$$

Next we compute two DOI metrics associated with each zone structure:

$$\text{(DOI } \underline{\text{w}}\text{ithin the zones of the structure)} \quad DOI^w = \sum_{k=1}^{W} \frac{\sigma_{\mathcal{Z}_k}}{\mu_{\mathcal{Z}_k}} \tag{3.12}$$

$$\text{(DOI } \underline{\text{a}}\text{cross the zones of the structure)} \quad DOI^a = \frac{\sigma_{\mu_{\mathcal{Z}}}}{T_0/N}. \tag{3.13}$$

Thus, the overall DOI associated with the structure is defined as:

$$\text{(DOI of the structure)} \quad DOI = DOI^w + DOI^a. \tag{3.14}$$

In order to test the effectiveness of the DOI measure, we also developed two other measures to represent the possible variability in the line, as follows:

$$\text{(Second DOI measure)} \quad DOI^1 = max\{DOI^w + DOI^a, DOI^w * DOI^a\} \tag{3.15}$$

$$\text{(Third DOI measure)} \quad DOI^2 = DOI^w + DOI^a + \sqrt{DOI^w * DOI^a} \tag{3.16}$$

Smaller DOI values represent more balanced zone structures. Therefore, given the choice among up to N ZonA structures, we pick the one with the lowest value of DOI.

We also show that there is no significant difference between employing any of the three above mentioned DOI matrices. As a result, we used the original DOI measure (not DOI^1 and DOI^2) as the performance measure throughout the test suit.

3.6 Numerical Experiments of Heuristic Policies and the Zone Assignment Algorithm

To show the effectiveness of FFMS policy as well as the zone assignment algorithm, we develop an extensive test suite employing discrete event simulation. The measure of interest is the system throughput. The system in the test suite consists of 12 stations and 4 workers. We use exponential distributions for processing time in each station. We should mention that the choice of the processing time probability distribution does not play a significant role on the result.

Each test case includes 50 simulation replications initialized with an empty line. The simulation has been run for the total of 8,000 job completions. Then the first 3,000 statistics will be considered as a warm-up period and will be discarded. We use "common random numbers" to reduce the variance in estimating the performance differences across policies.

3.6.1 Test Suite

To compare the performance of FFMS to other well-studied worker coordination policies, we test FFMS against: Random (RND), Last Buffer First Served (LBFS), and Maximum Queue (MaxQ). In all these mentioned policies (including FFMS) when two workers are simultaneously available at the same shared station with only

one job, as discussed in Theorem III.8, preference is given to the most downstream worker.

- *RND* - This policy allows each worker to choose any station with an available job in her zone at random with equal probabilities.
- *LBFS* - Based on Definition III.3.
- *MaxQ* - This policy assigns worker w to station i such that

$$i = \text{argmax}_{i \in \{\mathcal{F}_w \cup \mathcal{S}_w : Q_i(t) > 0\}} Q_i(t),$$

  where $Q_i(t)$ is the observed queue length at station i at time t. In case of a tie (same length queues), the worker prioritizes the most downstream queue (in the spirit of LBFS).

To compare the performance of the heuristic policies, we study a wide variety of problem instances, a suite of 1,024 unique cases, each having a unique set of mean processing times. To construct each instance, we begin with a standard balanced line as represented in Figure 3.2, in which all mean processing times for each of the 12 stations are set to $\frac{1}{3}$. To generate processing times for each station we use two multipliers, a set of *across primary zones (APZ) multipliers* and a set of *within primary zones (WPZ) multipliers*. In the next step, each station's mean processing time is multiplied by both the APZ and WPZ multipliers given in Tables 3.1 and 3.2. These multipliers are carefully chosen to ensure that although the means per station vary, the total raw processing time of the line stays equal to the number of workers, thus providing an upper throughput bound of 1 in every instance.

We should mention that sub-suites A to C were originally studied by Williams [93] in his dissertation. However he only focused of balanceable symmetric lines. In

our study we relax this assumption, as a result we added sub-suite D to cover this new generalization.

APZ	$\mathcal{Z}_1$	$\mathcal{Z}_2$	$\mathcal{Z}_3$	$\mathcal{Z}_4$
A	1.00	1.00	1.00	1.00
B	1.42	0.86	0.86	0.86
C	1.42	0.58	1.42	0.58
D	2.00	0.30	1.40	0.30

Table 3.1: Across primary zone (APZ) multipliers.

WPZ	**U. Fixed**	**D. Fixed**	**D. Shared**
1	0.90	1.00	1.10
2	1.00	1.00	1.00
3	0.90	1.10	1.00
4	1.40	1.40	0.20

Table 3.2: Within primary zone (WPZ) multipliers.

Each problem in the test suite is denoted by a five character code (e.g. *D1132*) that identifies the multipliers used to create that problem instance. The first character – A, B, C, or D – represents the APZ multiplier used for that problem instance. The APZ multipliers alter the line processing times across the primary zones ($\mathcal{Z}_1 - \mathcal{Z}_4$). Specifically, the ***A*** group of APZ multipliers serves as the standard line, the ***B*** group has one bottleneck in $\mathcal{Z}_1$, the ***C*** group has two bottlenecks in $\mathcal{Z}_1$ and $\mathcal{Z}_3$ with two fast zones in $\mathcal{Z}_2$ and $\mathcal{Z}_4$. ***D*** group has two bottlenecks, a very extreme bottleneck in $\mathcal{Z}_1$, and a moderate bottleneck in $\mathcal{Z}_3$. Notice that in case D the line is not generally balanceable with a symmetric FTZC. As shown in Table 3.1, the multipliers corresponding to a given letter (A, B, C, or D) apply to every station within a primary zone. We test each problem instance for eight CONWIP levels of $\{4, 6, 8, 10, 12, 24, 36, 48\}$.

3.6.2 Numerical Results for Station Assignment Heuristics

We compare the FFMS heuristic to the RND, LBFS, and MaxQ policies outlined in Section 3.6.1. However, as the design of zones can interact with the performance of

the heuristics, we isolate and clarify this effect as much as possible by later comparing sub-suites identified by the A, B, C, and D multipliers. Therefore, we consider a fixed symmetric zone structure of reasonable size in which each worker is assigned to two shared and two fixed stations (see Figure 3.2). Assuming a fixed zone structure, we ran 50 replications for each of the four policies across the 1,024 different cases as described in Tables 3.1 and 3.2. Note that cases A, B and C are balanceable under a symmetric FTZC structure; however, this is not generally true for case D. Figures 3.3 through 3.7 demonstrate the performance of the heuristics before applying ZonA. Later, in Section 3.6.3, we test how effectively the zone assignment algorithm (ZonA) can handle high DOI levels. Figures 3.9 through 3.13 illustrate the improvements achieved after employing ZonA.

Insight III.11. *On average, the FFMS policy performs the best in our policy set; i.e., it results in the highest average throughput across the entire test suite for each tested CONWIP level.*

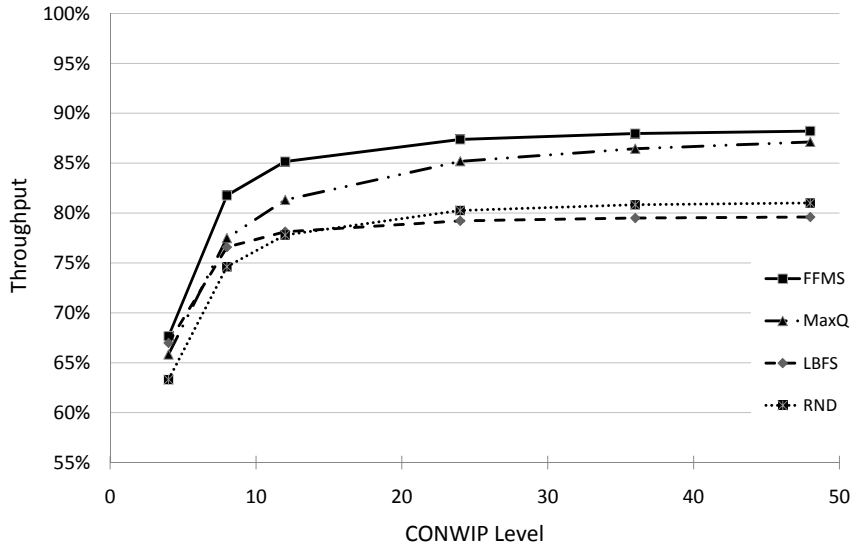


Figure 3.3: FTZC heuristic policy performance averaged over entire test suite.

Figure 3.3 shows that the FFMS consistently provides the highest average through-

put particularly for CONWIP levels above 8.

For extreme cases of high and low WIP levels, some policies perform as well as or better than FFMS. For example, in some problem instances LBFS outperforms the FFMS at a CONWIP level equal to the critical WIP value. This is intuitive, because with a low WIP level, the throughput is low, starvation is high, and the CONWIP system becomes closer to an open queueing system, for which LBFS is the optimal worker coordination policy for all workers.

In the next step we study the performance of each policy at a more detailed "sub-suite" level. In each sub-suite, we have 256 problem instances, denoted by the APZ multiplier groups (A, B, C, or D). Figures 3.4, 3.5, 3.6, and 3.7, respectively illustrate the results of these sub-suites. From Section 3.5.2, the degree of imbalance (DOI_m) denotes the magnitude of imbalance in a given problem instance, $m \in \{1, ..., 1024\}$.

A higher DOI index corresponds to greater system imbalance, which we expect will result in lower throughput. In the standard problem instance *(A2222)* in the test suite, DOI equals zero, because all station processing times are identical and the line is perfectly balanced. The DOI metric associated with each sub-suite is given in Table 3.3. Tables 3.4 and 3.5 demonstrate the average decrease in DOI as a result of employing ZonA, when the DOI^1 and DOI^2 measures are employed. The result confirms that the choice of DOI measure does not affect the performance of ZonA.

Sub-suite	Average DOI before ZonA	Average DOI after ZonA	Decrease percentage
A	1.00	0.67	33%
B	1.45	1.07	26%
C	2.28	1.58	30%
D	3.34	2.50	25%

Table 3.3: Average DOI for sub test Suites.

Figure 3.4 illustrates the performance of each policy under test sub-suite A, which has a low degree of imbalance. In this sub-suite, LBFS at a CONWIP level of 4

Sub-suite	Average DOI[1] before ZonA	Average DOI after ZonA	Decrease percentage
A	1.00	0.67	33%
B	1.45	1.07	26%
C	2.28	1.58	30%
D	3.34	2.50	25%

Table 3.4: Average DOI[1] for sub test Suites.

Sub-suite	Average DOI[2] before ZonA	Average DOI after ZonA	Decrease percentage
A	1.00	0.67	33%
B	1.45	1.07	26%
C	2.28	1.58	30%
D	3.34	2.50	25%

Table 3.5: Average DOI[2] for sub test Suites.

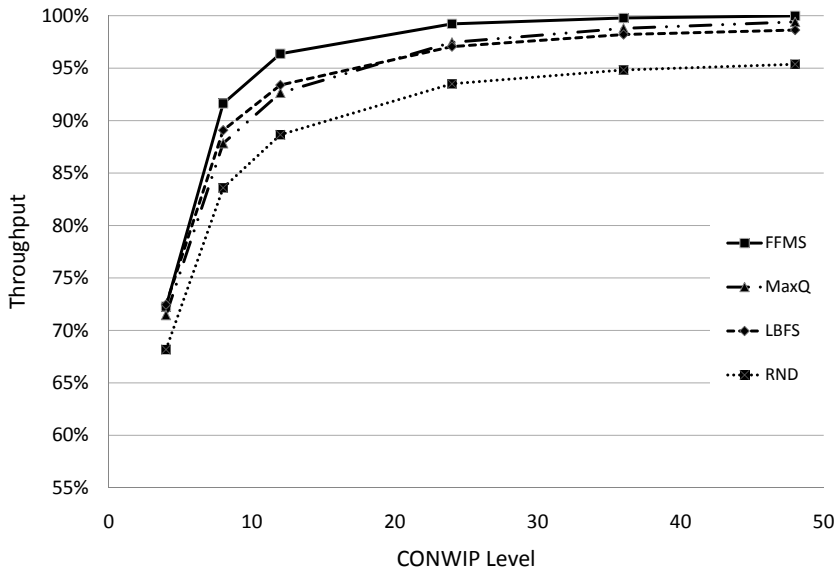


Figure 3.4: FTZC heuristic policy performance under low line imbalance (A suite).

outperforms the other policies by 0.3%. At CONWIP levels of 12 and above, FFMS outperforms the other policies. In general, low DOI values yield high throughput levels for most policies. This is expected because higher DOI yields a higher level of "line variation", and thus, increased flow variability and station starvation. For high CONWIP levels (i.e., 36 and above), the FFMS policy approaches the maximum throughput of 1 on average. Figures 3.5 and 3.6 represent moderate and high DOI levels respectively. As observed in those figures, FFMS still slightly outperforms the other policies on average. In sub-suite B, an average maximum throughput of 92%

is achieved by FFMS at the CONWIP level of 48. However, in sub-suite C, this maximum is decreased to 90%.

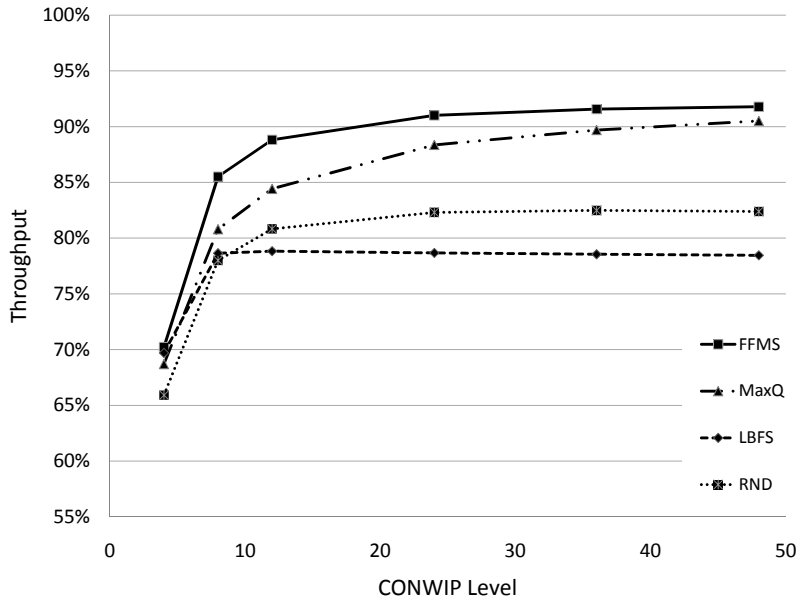


Figure 3.5: FTZC heuristic policy performance under moderate line imbalance (B suite).

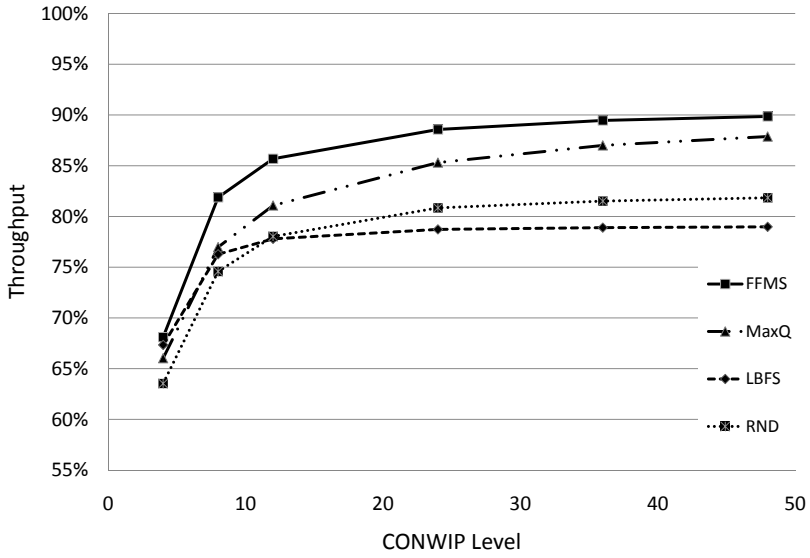


Figure 3.6: FTZC heuristic policy performance under high line imbalance (C suite).

Figure 3.7 illustrates sub-suite D, which is an extreme case where the line is not balanceable by a symmetric FTZC structure. In this sub-suite the maximum throughput level of 70% is achieved in the best case, significantly lower than sub-

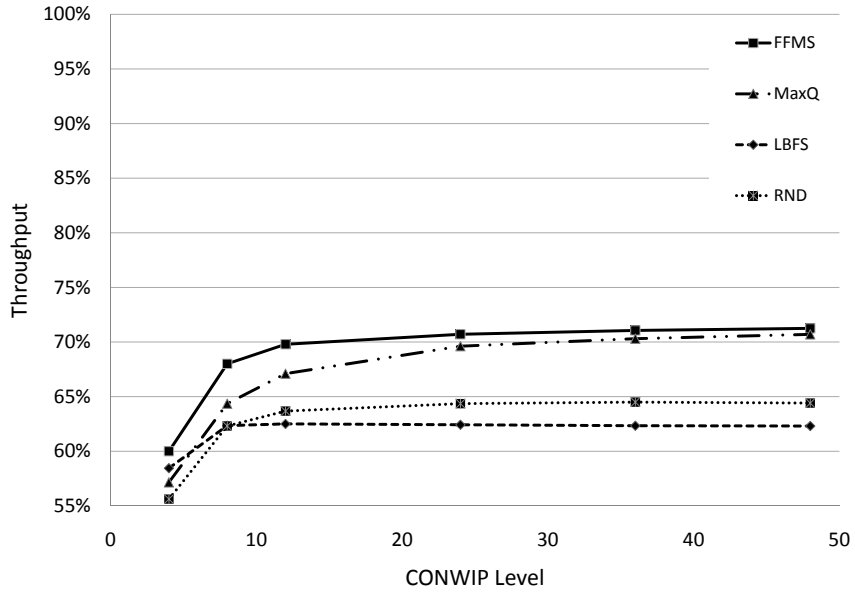


Figure 3.7: FTZC heuristic policy performance under extreme line imbalance (D suite).

suites A, B and C. Such cases of high imbalance are especially useful examples to test the value of the ZonA algorithm.

Insight III.12. *Compared to a set of reasonable intuitive policies, FFMS is achieving significantly greater performance compared to RND, LBFS, and MaxQ.*

3.6.3 Zone Assignment (ZonA) Algorithm Performance

In this section, we exploit the ZonA algorithm to improve the standard symmetrical zone structure illustrated in Figure 3.2 and compare the performance of the heuristics under the symmetrical and the improved zone structures. Note that ZonA yields zone structures that vary for each test sub-suite and each instance within a sub-suite. Figure 3.8 illustrates the ZonA structure computed for case $D1132$, where the DOI metric was reduced from 3.45 for the original symmetric zone structure as illustrated in Figure 3.2 to only 2.29. First we describe the results of the numerical experiments for the entire test suite in Figure 3.9. We can observe the positive effect of ZonA on increasing throughput levels by comparing Figures 3.3 and 3.9. As a

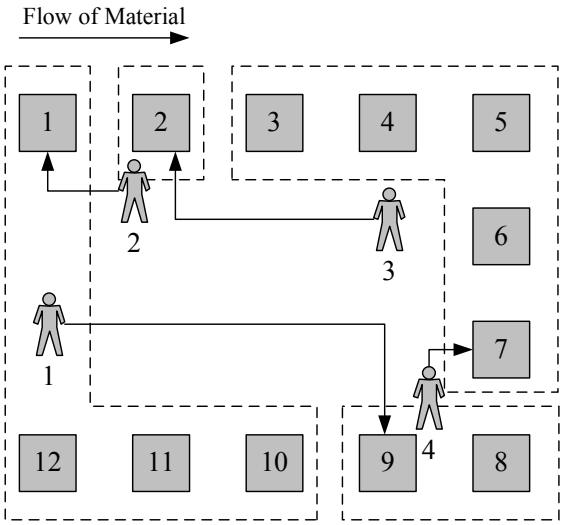


Figure 3.8: The improved zone structure obtained for case $D1132$ by ZonA algorithm.

general pattern, all policies achieve higher throughput values after redesigning their zone structures. In particular, FFMS reaches the maximum throughput of 1 for CONWIP levels of 24 and above. Next we show the details of each sub-suite A, B, C

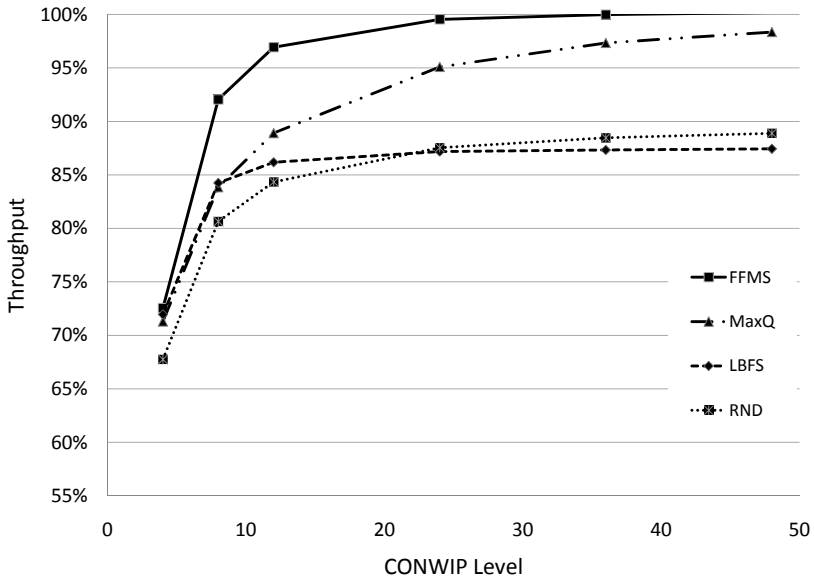


Figure 3.9: FTZC heuristic policy performance for the entire test suite using ZonA.

and D in Figures 3.10, 3.11, 3.12 and 3.13. As observed previously, sub-suite A (the balanced case) has already reached high throughput levels even with a default zone

structure. Thus it does not show a significant difference after improving its zone structures, except at WIP levels around 10. To visualize the power of ZonA algo-

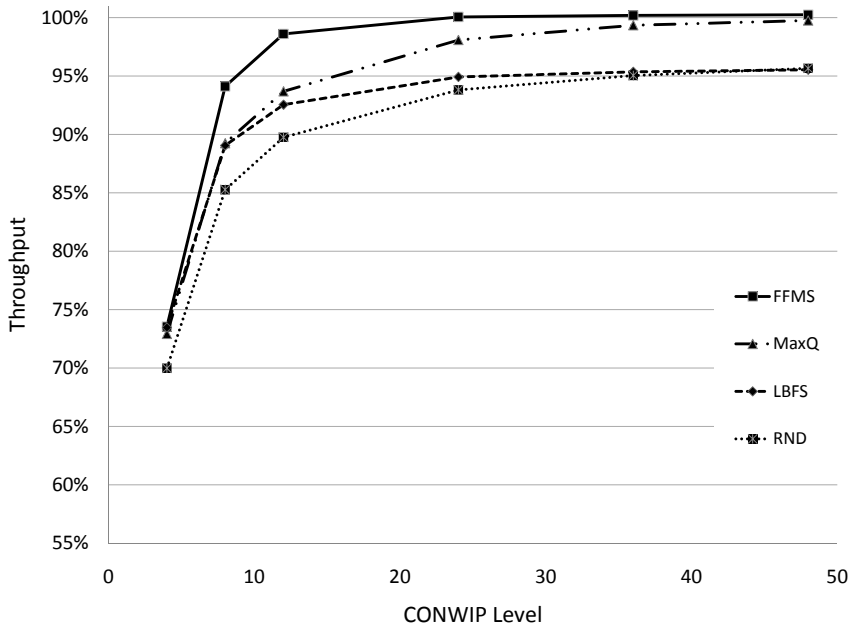


Figure 3.10: FTZC heuristic policy performance under low line imbalance using ZonA (A suite).

rithm, we can compare throughput levels of sub-suites B, C and D with the previous results. For instance, compare the throughput levels of the extremely imbalanced case (case D) before the implementation of ZonA (Figure 3.7) with throughput levels after improving the zone structure, Figure 3.13. We can clearly observe a dramatic improvement after implementing ZonA. In particular, we observe that the maximum throughput levels associated with CONWIP levels of 48 are increased from 92% (in suite B), 90% (in suite C) and 72% (in suite D) to 100% in all test suites after improving the zone structures, due to the elimination of the bottlenecks that caused these asymptotes. RND represents a reasonable policy that has no connection to the structure of cross-training. While ZonA improves the performance of RND, it is striking that ZonA has a much greater impact on improving the performance of FFMS, being well matched to the FTZC paradigm.

Another way to illustrate the improvement is by revisiting the DOI metric, which is reduced for all the sub-suites after running ZonA (see Table 3.3).

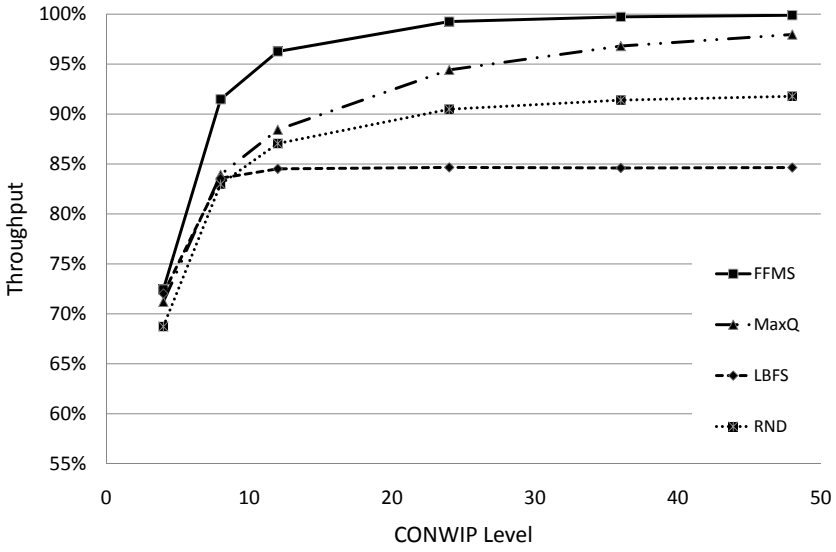


Figure 3.11: FTZC heuristic policy performance under moderate line imbalance using ZonA (B suite).

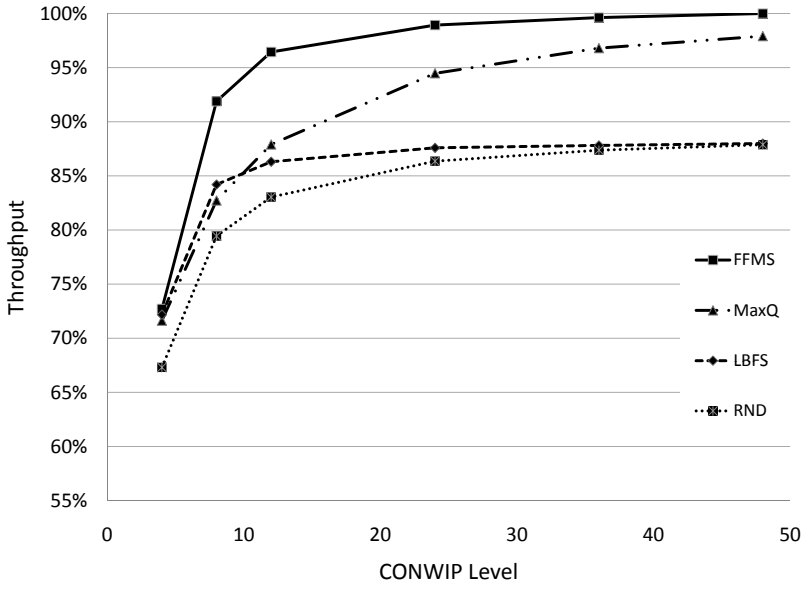


Figure 3.12: FTZC heuristic policy performance under high line imbalance using ZonA (C suite).

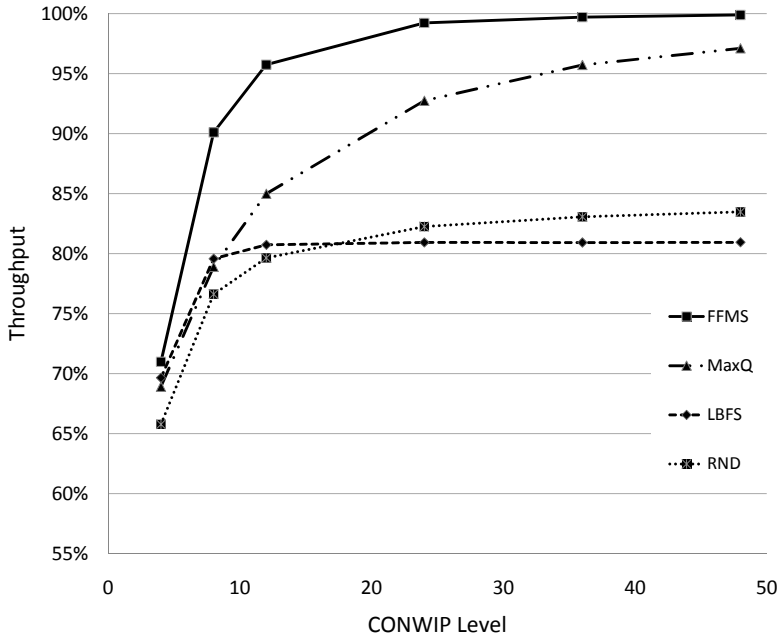


Figure 3.13: FTZC heuristic policy performance under extreme line imbalance using ZonA (D suite).

3.6.4 Comparing FTZC to Other Structures

To observe the performance of FTZC with and without ZonA algorithm and demonstrate the power of zone design, we benchmark the FTZC structure for the system of 12 stations and 4 workers against two other structures:

- Two-Skill Zone Chain (2SZC): We use this structure, which cross-trains all workers in a way that two workers cover each station in our model (see Figure 3.14). That is, two adjacent workers share the work at each station such that each of the four workers is trained on six total stations. All workers use the maximum queue policy in order to pick jobs available to them.

- Classic CONWIP Approximation (CCA): In this model, there is one worker per station and no shared stations. The workers are identical and employ the standard FCFS service discipline, so standard mean value analysis can be used to compute the performance of CCA (see Buzacott and Shanthikumar [25]).

Obviously, as CCA dedicates a worker to each station in a system with a similar number of stations, CCA always has more workers than FTZC. Therefore, to have a fair comparison between CCA and FTZC, the worker capacities in CCA are adjusted to be the same as their effective capacities for each station in FTZC. This is done by computing the fraction of time spent by workers on each station. These fractions are calculated using the linear program described in Section 3.5 and Proposition III.9.

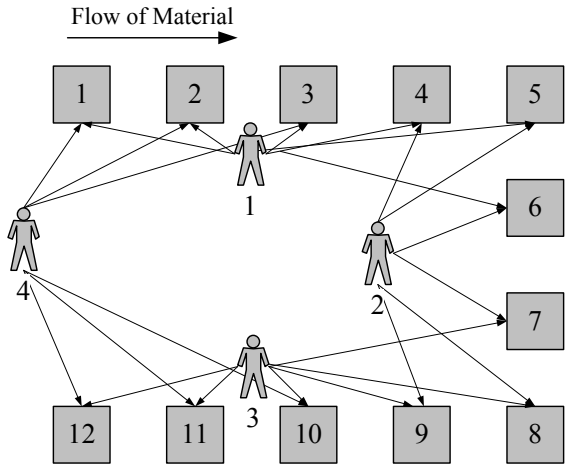


Figure 3.14: Two-skill zone chain (2SZC) structure.

Table 3.6 demonstrates the number of skills for all of these three structures.

	Total no. of skills	No. of cross-trained skills
Full XT	48	36
2SZC	24	12
FTZC	*16*	*4*
CCA	12	0

Table 3.6: Cross-training skill reduction.

We compare the throughput loss for for all three structures: 2SZC, FTZC, and CCA. Full XT achieves maximum throughput of 1 for all the WIP levels, so we do not present it in our graphs to avoid clutter.

2SZC structure achieves the highest throughput as anticipated. FTZC with ZonA archives a minimal amount of throughput loss with a fraction of skills available compared with 2SZC. FTZC without ZonA does not perform as well. However, this lack of strong performance is mainly due to the extreme imbalance in sub suite D. We measure relative efficiency using the *Percent of Throughput Loss* ($\%_{TL}$) resulting from implementing a FTZC. The results of average throughput are presented in Figure 3.15 as a function of CONWIP level.

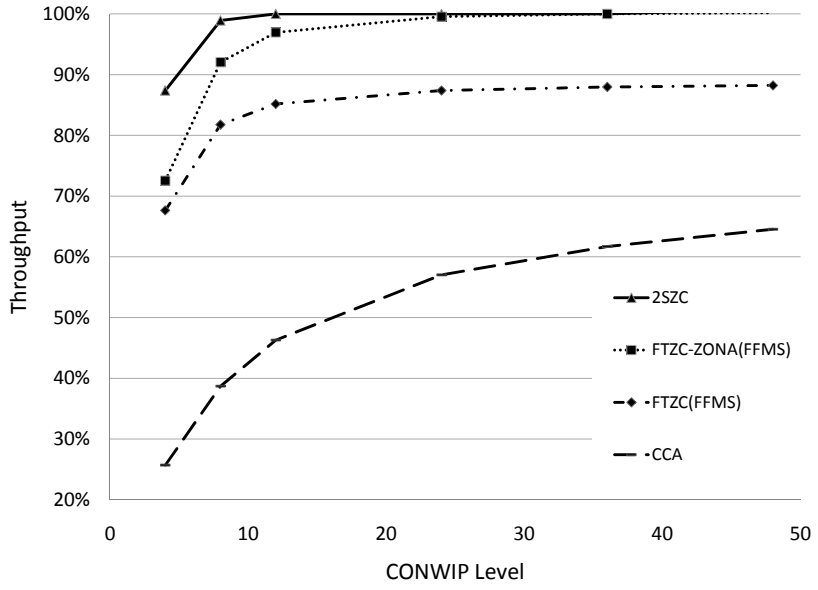


Figure 3.15: Structure performance comparison - entire suite.

As observed in Figure 3.15, the classic CONWIP structure has a weak performance due to the lack of cross-training. After implementing ZonA, FTZC rapidly approaches the system capacity for WIP levels above 8. This confirms that with ZonA almost all the systems are indeed balanceable — strong evidence that when operated under the FFMS policy, the FTZC structure is a very effective approach.

In the studied model, the FTZC structure only has 4 cross-trained skills, whereas in full cross-training, there are 36 skills to be cross-trained and 12 in 2SZC. Given

this minimal level of cross-training, FTZC performs extremely well by achieving the throughput level of 1 for WIP levels of higher than 25. This minimal reliance on cross-training makes FTZC a powerful structure that with a considerably low training cost.

Insight III.13. *Across the entire test suite, for CONWIP levels of 12 or greater, the FTZC (with ZonA) averages at least 97% of the throughput of the 2SZC. For the CONWIP levels of $2\omega_0$ to $6\omega_0$ the average $\%_{TL}$ (FTZC vs. 2SZC) is 3.6%.*

To further study the effect of line imbalance on the FTZC with FFMS, we again examine the A, B, C, and D sub-suites. Table 3.7 shows the adverse effect of line imbalance on the $\%_{TL}$. As illustrated in Table 3.7, $\%_{TL}$ (FTZC vs. 2SZC) generally increases as the DOI increases (from A suite to D suite). Low CONWIP levels allows the 2SZC to benefit more from its flexibility advantage over the FTZC. Note that in Table 3.7 the DOI measure of $DOI^w + DOI^a$ was employed.

CONWIP level	4	8	12	24	36	48
Without ZonA	21.1%	15.4%	12.9%	10.0%	8.9%	8.4%
A Suite	16.5%	6.7%	3.0%	0.0%	0.0%	0.0%
B Suite	18.4%	12.4%	10.3%	9.2%	8.6%	8.4%
C Suite	21.3%	16.9%	14.3%	12.5%	11.7%	11.3%
D suite	28.4%	25.8%	24.2%	18.4%	15.2%	14.2%
With ZonA	15.8%	5.7%	1.9%	1.1%	0.8%	0.4%
A Suite	15.5%	4.1%	0.0%	0.0%	0.0%	0.0%
B Suite	15.8%	6.2%	2.0%	0.9%	0.5%	0.3%
C Suite	15.9%	6.1%	2.5%	1.6%	1.0%	0.6%
D suite	16.0%	6.7%	3.1%	2.2%	1.9%	0.7%

Table 3.7: Comparing the percent throughput loss ($\%_{TL}$) of FTZC relative to 2SZC (with and without ZonA).

We select 5% throughput loss as a target and use Table 3.7 to make the following assertion that clarifies the robustness of the FTZC structure in various operating environments:

Insight III.14. *When coupled with ZonA and operated under the FFMS policy, the*

FTZC structure incurs a throughput loss of at most 5% when the CONWIP levels are at least approximately: (1) $2\omega_0$ for the low line imbalance (A Suite); (2) $3\omega_0$ for moderate, high and extreme line imbalances (B, C, and D Suites).

Table 3.7 also illustrates a relationship between throughput, CONWIP level and DOI. In general, $\%_{TL}$, increases as DOI index increases. Here we also measure the performance of FTZC under other DOI measures, i.e. DOI^1 and DOI^2. The results in terms of throughput loss are presented in Tables 3.8 and 3.9. As observed in these tables, the FTZC structure seems to be relatively robust against the choice of DOI. We only observe a 1% increase in $\%_{TL}$ for CONWIP level of 48 when DOI^2 was used to find "the best" zone structure.

CONWIP level	4	8	12	24	36	48
With ZonA	15.8%	5.7%	1.9%	1.1%	0.8%	0.4%
A Suite	15.5%	4.1%	0.0%	0.0%	0.0%	0.0%
B Suite	15.8%	6.2%	2.0%	0.9%	0.5%	0.3%
C Suite	15.9%	6.1%	2.5%	1.6%	1.0%	0.6%
D suite	16.0%	6.7%	3.1%	2.2%	1.9%	0.7%

Table 3.8: $\%_{TL}$ of FTZC with employing ZonA based on DOI^1 relative to 2SZC.

CONWIP level	4	8	12	24	36	48
With ZonA	15.8%	5.7%	1.9%	1.1%	0.8%	0.5%
A Suite	15.5%	4.1%	0.0%	0.0%	0.0%	0.0%
B Suite	15.8%	6.2%	2.0%	0.9%	0.5%	0.3%
C Suite	15.9%	6.1%	2.5%	1.6%	1.0%	0.7%
D suite	16.0%	6.7%	3.1%	2.2%	1.9%	0.9%

Table 3.9: $\%_{TL}$ of FTZC with employing ZonA based on DOI^2 relative to 2SZC.

3.7 Conclusions

This chapter builds upon the Fixed Task Zone Chaining (FTZC) paradigm introduced by Williams [93]. The FTZC structure is very powerful and applicable since it needs only a fraction of skills to be cross-trained compared to a fully cross-trained or two skill zone chain (2SZC) systems. FTZC divides stations into "zones" and assigns

each zone to a worker such that zones are chained together as they share a station at each end. In this chapter we generalized the original FFMS heuristic policy, which was only designed for symmetric zones and a system of 12 stations and 4 workers, to be applicable to any U-shaped CONWIP line where the number of workers in the line are strictly fewer than the stations. FFMS is an efficient worker-to-station assignment policy under the general framework of FTZC. We compare our proposed heuristic to other policies to show its effectiveness in maximizing throughput. We also develop new metrics to capture the "degree of imbalance" (DOI) to quantify the efficiency of a zone structure. We show that ZonA algorithm demonstrates a great deal of robustness to the choice of DOI. We use a comprehensive test suite to show that, on average, the FFMS policy outperforms the other policies.

The performance of the FTZC system design is tied to the efficiency of the zone structures. We use our test suite and the DOI metric to show how, in some cases, a poor (i.e. imbalanced) zone structure can result in low throughput levels. Therefore in the next step, we develop a Zone Assignment (ZonA) algorithm to improve the zone structure in a given production line. We derive sufficient conditions under which a line is balanceable with a FTZC system design. We use computational experiments to show how the ZonA algorithm can significantly improve throughput by balancing zone structures. Based on our experiments, under mild conditions, the FFMS heuristic applied to a ZonA designed structure can achieve maximum throughput for almost all practical CONWIP levels.

Finally, we benchmark the FTZC family of designs against other alternative structures ranging from a CONWIP line with a dedicated (but slow) server at every station to a "two-skill zone chain" to a fully cross-trained system. Careful numerical experiments indicate that the FTZC system (when designed using the ZonA algorithm)

can nearly reach the performance of a fully cross-trained system while requiring a far more limited set of skills and reducing the training cost.

3.8 Proofs of Propositions and Theorems of Chapter III

3.8.1 Proof of Proposition III.5

Our FTZC includes N stations and W workers, where all workers have two shared task-types, and an LBFS policy (see Definition III.3) is followed at every queue in the zone. Given the assumption of a symmetrical FTZC structure, each worker covers $\frac{N}{W}$ stations. This yields a total work content of $(\frac{N}{W})(\frac{T_0}{N}) = \frac{T_0}{W}$ for every worker within her primary zone. We have the ideal scenario (no congestion) because each station has a deterministic processing time of $(\frac{T_0}{N})$. If each worker begins at time 0 with one job at her upstream fixed-task station, all workers send one job to their "downstream" neighbor after $\frac{T_0}{W}$. Since there is no wait, the cycle time will be T_0; further each worker achieves a throughput equal to bottleneck rate, $r_b = \frac{W}{T_0}$, which is the maximum value possible. By Little's law, the critical WIP, $\omega_0 = r_b T_0 = (\frac{W}{T_0})T_0 = W$. For the deterministic system, we can see that additional WIP does not decrease throughput.

3.8.2 Proof of Theorem III.6

We prove part (ii) first. Let τ be a time instant when worker k becomes available with jobs waiting at least two stations in her zone, and i be the most downstream such non-empty station. For any policy π that assigns worker k in some other station l, where $l < i$, we construct an alternative policy $\tilde{\pi}$ that assigns worker k in station i and is such that $D_t^{\tilde{\pi}}(\omega) \geq D_t^{\pi}(\omega)$ along every sample path ω, which proves the statement of the theorem.

Because service times are task-type and worker dependent, sample path ω is defined by sequences $S_j^w(n, \omega)$, $n \in \mathbb{N}$, where $S_j^w(n, \omega)$ is the time duration of the n^{th}

service performed by worker w in station j. We construct policy $\tilde{\pi}$ to be identical to π except that at time τ worker k is assigned in station i and then follows the sequence of actions taken under π after time τ. The first action in that sequence is a service in station l, and the rest (that may include idle periods) are determined from the realization of the arrival and service processes. Let $\tau_c(\omega)$ be the time of the first service completion under π in station i. Until that time it is not possible for policy π to assign worker k in her (empty) downstream shared station, which means that $\tilde{\pi}$ can always mimic π with respect to actions taken by workers other than k, because at any time all stations where these workers can be assigned (the ones not fixed to worker k) have at least as many jobs under $\tilde{\pi}$ as under π. Therefore, $\tilde{\pi}$ is well defined in $[\tau, \tau_c(\omega)]$ and the two policies are coupled at $\tau_c(\omega)$. Time periods $[\tau + S_l^k(1,\omega), \tau_c(\omega) - S_i^k(1,\omega)]$ under π and $[\tau + S_i^k(1,\omega) + S_l^k(1,\omega), \tau_c(\omega)]$ under $\tilde{\pi}$ include the same sequence of actions by worker k. A comparison of the two policies shows that for any $t \in [\tau, \tau_c(\omega)]$, the number of jobs completed by t under $\tilde{\pi}$ is at least equal to the number completed under π.

The proof of part (i) is based on a similar construction of the alternative policy $\tilde{\pi}$. After the service completion in station i worker k idles for the same amount of time she would have idled under π, say $I(\omega)$, which can be calculated by observing the sample path realization. Then she replicates her actions under π until the two policies are coupled the first time she completes a service in station i.

3.8.3 Proof of Theorem III.7

In Theorem III.7, we only focus on the class of policies that prioritize a worker's fixed stations over shared stations. Let τ be a time instant when worker k becomes available with at least two fixed stations in her zone having jobs. Let i be the most downstream fixed station in her zone with at least one job available and let j be

another fixed station in worker k zone (i.e. $j < i$) with at least one job available. To show that LBFS is optimal among fixed tasks, for any policy π that assigns worker k to station j, we construct an alternative policy $\tilde{\pi}$ that assigns worker k to station i for which the number of completed jobs by time t ($D_t^{\tilde{\pi}}(\omega)$) is at least equal to the number of jobs completed under π ($D_t^{\pi}(\omega)$) along every sample path.

The way $\tilde{\pi}$ is defined is exactly the same as in Theorem III.6. Since π gives priority to fixed stations, it is not possible to assign worker k to her downstream shared station before station i, which ensures that $\tilde{\pi}$ can mimic π as far as the actions of the other workers are concerned. As in Theorem III.6, a comparison of $\tilde{\pi}$ and π until their coupling at the time of the first job completion under π in station i shows that $\tilde{\pi}$ performs at least as well as π.

3.8.4 Proof of Theorem III.8

Let π be a policy that assigns U_s to s. We define an alternative policy $\tilde{\pi}$ that assigns D_s to s, keeps U_s idle until job s is served at the shared task-type, and is identical to π afterwards. Since the two workers have equal speeds, the service time of task-type s is equal under policies π and $\tilde{\pi}$ along any sample path. Along any sample path, it is not possible for π to assign D_s to some task-type during the service of s. First, no job can arrive at station s from upstream because U_s is busy at s, and second, no jobs can become available for D_s downstream of s because D_s has only one shared station with her downstream neighbor (this station is either empty or occupied by the other worker). The two policies are coupled at the time of service completion at s implying that they will result in the same throughput. This shows that assigning a single job to the downstream worker (D_s) will not result in throughput loss.

3.8.5 Proof of Proposition III.9

Because $\Theta^* = \frac{W}{T_0}$, the inequality constraints (3.2) and (3.3) of the LP take the form

$$z_i^* \geq T_i \frac{W}{T_0}, \qquad \forall i \in \mathcal{F} \tag{3.17}$$

$$y_{s_u k}^* + y_{s_d k \ominus 1}^* \geq \tilde{T}_{(k)} \frac{W}{T_0}, \qquad \forall k \in \mathcal{W} \tag{3.18}$$

Summing over all constraints (3.17) and (3.18), we get

$$\sum_{i \in \mathcal{F}} z_i^* + \sum_{k \in \mathcal{W}} (y_{s_u k}^* + y_{s_d k \ominus 1}^*) \geq \frac{W}{T_0} (\sum_{i \in \mathcal{F}} T_i + \sum_{k \in \mathcal{W}} \tilde{T}_{(k)}) = \frac{W}{T_0} T_0 = W. \tag{3.19}$$

Taking the sum over all equality constraints (3.4) we get:

$$\sum_{k \in \mathcal{W}} I_k^* + \sum_{k \in \mathcal{W}} (y_{s_u k}^* + y_{s_d k}^* + \sum_{i \in \mathcal{F}_k} z_i^*) = W. \tag{3.20}$$

Because the left-hand side of Equation (3.19) is equal to the second sum in Equation (3.20), we obtain $I_k^* = 0$ for all k and $\sum_{k \in \mathcal{W}} (y_{s_u k}^* + y_{s_d k}^* + \sum_{i \in \mathcal{F}_k} z_i^*) = W$. Assuming that $z_i^* > \frac{W}{T_0} T_i$ for some i or $y_{s_u k}^* + y_{s_d k \ominus 1}^* > \frac{W}{T_0} \tilde{T}_{(k)}$ for some k, we get from (3.17) and (3.18) $\sum_{k \in \mathcal{W}} (y_{s_u k}^* + y_{s_d k}^* + \sum_{i \in \mathcal{F}_k} z_i^*) > W$, which is clearly a contradiction. Therefore, all inequalities in (3.17) and (3.18) are satisfied as equalities and the proof is complete.

3.8.6 Proof of Theorem III.10

Proof. Condition $T_i < \frac{T_0}{W}$ ensures that stations $S_1, ..., S_W$ are distinct, so that ZonA produces a FTZC structure. To prove that this structure is balanceable, we construct a feasible solution to LP such that the objective function reaches the value of $\Theta = W/T_0$. Let $F_k = \sum_{i \in \mathcal{F}_k} T_i$, which equals the sum of mean processing times at the

fixed stations of worker k and

$$y_{s_u 1} = \min_{1 \le k \le W} \left\{ k - \left[\sum_{j=1}^{k} F_j + \sum_{j=2}^{k} \tilde{T}_{(j)} \right] \frac{W}{T_0} \right\},$$

$$y_{s_u k} = \left[\sum_{j=1}^{k-1} F_j + \sum_{j=2}^{k} \tilde{T}_{(j)} \right] \frac{W}{T_0} + y_{s_u 1} - (k-1) \text{ for } k \in \mathcal{W} - \{1\},$$

$$z_i = T_i \frac{W}{T_0} \text{ for } i \in \mathcal{F},$$

$$y_{s_d k} = 1 - y_{s_u k} - \sum_{j \in \mathcal{F}_k} z_j \text{ for } k \in \mathcal{W},$$

$$I_k = 0 \text{ for } k \in \mathcal{W}.$$

We can see that the inequality constraints (3.2) are satisfied as equalities with $\Theta = W/T_0$. This is also true for inequalities (3.3). To demonstrate this we use the above equations and the fact that $T_0 = \sum_{j=1}^{W} (F_j + \tilde{T}_{(j)})$ to get

$$y_{s_u 1} + y_{s_d W} = y_{s_u 1} + 1 - y_{s_u W} - \sum_{j \in \mathcal{F}_W} z_j =$$

$$W - [\sum_{j=1}^{W-1} F_j + \sum_{j=2}^{W} \tilde{T}_{(j)}] \frac{W}{T_0} - F_W \frac{W}{T_0} = W - (T_0 - \tilde{T}_{(1)}) \frac{W}{T_0} = \tilde{T}_{(1)} \frac{W}{T_0},$$

and for $2 \le k \le W$

$$y_{s_u k} + y_{s_d k-1} = y_{s_u k} + 1 - y_{s_u k-1} - \sum_{j \in \mathcal{F}_{k-1}} z_j =$$

$$(F_{k-1} + \tilde{T}_{(k)}) \frac{W}{T_0} - F_{k-1} \frac{W}{T_0} = \tilde{T}_{(k)} \frac{W}{T_0}.$$

It remains to be shown that the non-negativity constraints for the worker allocations are also satisfied. For all $k \in \mathcal{W}$ we have:

$$\sum_{j=1}^{k} F_j + \sum_{j=2}^{k} \tilde{T}_{(j)} = \sum_{l=2}^{S_{k+1}-1} T_l < k \frac{T_0}{W} \Rightarrow y_{s_u 1} > 0,$$

and for $k \in \mathcal{W} - \{1\}$

$$\sum_{j=1}^{k-1} F_j + \sum_{j=2}^{k} \tilde{T}_{(j)} = \sum_{l=2}^{S_k} T_l \ge (k-1) \frac{T_0}{W} \Rightarrow y_{s_u k} > 0,$$

where the first inequality in each of the above equations follows from Equation (3.8). By the definition of $y_{s_u 1}$ we have

$$y_{s_u 1} + \sum_{i \in \mathcal{F}_1} z_i \leq 1 - F_1 \frac{W}{T_0} + F_1 \frac{W}{T_0} = 1,$$

and for all $k \in \mathcal{W} - \{1\}$ we have:

$$y_{s_u k} + \sum_{i \in \mathcal{F}_k} z_i \leq [\sum_{j=1}^{k-1} F_j + \sum_{j=2}^{k} \tilde{T}_{(j)}] \frac{W}{T_0} + k - [\sum_{j=1}^{k} F_j + \sum_{j=2}^{k} \tilde{T}_{(j)}] \frac{W}{T_0} - (k-1) + F_k \frac{W}{T_0} = 1.$$

Therefore, $y_{s_d k} \geq 0$. Thus a feasible solution to the LP defined by (3.1) - (3.7) can be constructed such that the maximum throughput is achieved. □

CHAPTER IV

Malaria Treatment Distribution Logistics in Developing World Health Systems and Applications to Malawi

4.1 Motivation

Despite decades of elimination and control efforts worldwide, malaria remains one of the largest killers of children worldwide and a serious threat to the health of residents in most developing countries. According to WHO [94], there were nearly 250 million suspected malaria cases in 2009, nearly all in developing countries and half the world's population lives at some risk for malaria infection. Malaney et al. [65] show that in addition to being an immense burden on human health and welfare, malaria is a major impediment to the economic development of impoverished nations. Thus, efforts to control and treat malaria are a priority of the development goals of the United Nations, various governments, and many non-governmental organizations.

According to Human Development Report [84], Malawi is one of the poorest countries in the world and ranks 153^{rd} out of 177 countries on the Human Development Index. Due to a combination of intense poverty and environmental and local weather conditions, Malawi suffers from an intense high burden of malaria. Dzinjalamala [32] indicates that all Malawians live at year round risk for malaria though incidence peaks during the December-May rainy season. The World Health Organization estimated that there were approximately 5.5 million cases of malaria out of a population

of just over 15 million people. At least a third of all medical consultations were related to malaria (WHO [94]). Multiple infections among the same individuals are common. Dzinjalamala [32] shows that though all ages are affected, symptomatic disease disproportionately strikes children under five years of age and is a major cause of childhood death. This is due in part to increasing immunity over time that develops as a result of repeated infections.

Health care in Malawi is available for free to all those who seek it, however, the availability and level of services are rudimentary. The amount Malawi spends on health care is one of the lowest in sub-Saharan Africa, $4.93 per person in 2004, and is well under the $34 per person expenditure recommended by the WHO Commission of Macroeconomics and Health (Conticini [26]). Malawi's public health system is a three tiered network consisting of a central warehouses and regional warehouses on the first tier, district hospitals on the second tier, and primary health centers and community clinics on the third. Each tier receives supplies from and answers to the tier above it with the exception of the central warehouses and regional warehouses which answer directly to the Ministry of Health (see Figure 4.1). In addition to the public providers, private health services are also available at cost for those who can afford them, though these households are primarily located in urban areas. Malaria care is provided at all levels of the Ministry of Health system, though central hospitals principally accept patients on a referral basis only. Distribution of pharmaceuticals begins at the Central Medical Stores (CMS) in Lilongwe, Malawi, the first point on the pharmaceutical supply chain in Malawi, which allocates drugs to the regional warehouses and central warehouses (first tier). Central warehouses and regional warehouses then deliver to district hospitals, which are in turn responsible for supplying primary health centers and local community clinics. The goal is to

maintain a constant and equitable supply of drugs to all levels of health facilities and to insure transparency and accountability in the system.

According to Malawian Ministry of Health [74], since 2007 the Ministry of Health of Malawi has recommended the use of ACTs as a first line treatment for patients presenting symptoms of uncomplicated malaria. As shown by Garner and Graves [36], ACTs not only treat malaria in the individual, but also reduce infectivity to mosquitoes, theoretically reducing infections in the community.

The goal of this research is to provide insights into how developing countries with highly centralized systems of health delivery can most effectively distribute these expensive and valuable treatments on a restrictive budget. In this chapter we employ stochastic programming and Markov decision models to significantly decrease the annual shortage of the medication while keeping transportation costs low and affordable.

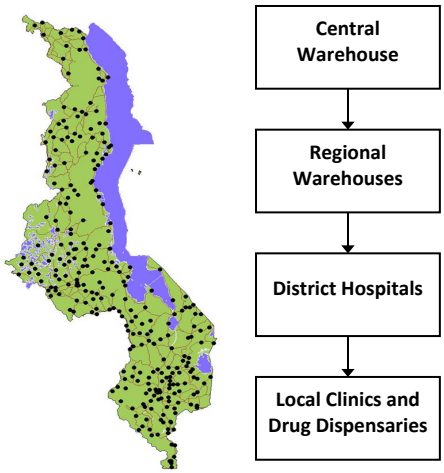


Figure 4.1: Malaria pharmaceautical distribution network in Malawi.

4.2 Literature Review

The stochastic transportation problem has been previously addressed in the literature. Williams [92] studies the problem of shipment scheduling for a single commod-

ity in the face of demand uncertainty, where there is costs associated with transportation, shortage, and excess inventory, represented by a nonlinear objective function. Holmberg [46] studies a large-scale stochastic transportation program and tests the effectiveness of a wide range of solution methods including separable programming, Benders decomposition, and Lagrangian relaxation. Romeijn and Sargut [75] study the problem of stochastic transportation with finite suppliers' capacities and demand uncertainty and use a branch-and-price algorithm to solve it.

The most relevant literature to this chapter can be divided into two categories: (1) disaster preparedness and emergency response, and (2) disease prevention resource allocation. Literature on emergency response tends to focus on broad public health needs that must be addressed rapidly and in a targeted manner after a period of prior planning. Disease prevention literature, on the other hand, focuses on long term public policy decisions, such as vaccination strategies to prevent incidence and ultimately the spread of a particular communicable disease. Our work contains components common to both types of literature. We consider a two-stage (and three-stage) response in the distribution of medications (as in disaster preparedness), but our response occurs over a longer span of time and is driven by the geographical evolution of malaria over the course of the malaria season.

Published research regarding disaster preparedness and emergency response is extensive and has been well documented by several survey papers, including Altay and Green [3], Simpson and Hancock [80], and de la Torre et al. [27]. Readers are encouraged to review these surveys for a more complete understanding of the disaster preparedness and emergency response literature.

Particularly relevant to our work is the two-stage stochastic programming literature. These approaches involve an initial allocation of resources before a disaster and

subsequent transportation to affected locations after a large emergency event. Two such approaches have been developed and researchers have performed case studies in the context of earthquake preparedness in Seattle by Mete and Zabinsky [67], Turkey by Barbarosoğlu and Arda [13] and China by Zhu et al. [98]. Additionally, Salmerón and Apte [77] develop a model that focuses on optimal acquisition and pre-positioning of assets in anticipation of a disaster, with a second stage that deploys assets to rescue victims, deliver goods and transport citizens to safety. Lee et al. [62] introduce RealOpt, a decision support tool developed to support government activities in a wide range of disasters which considers the minimum number of facilities to satisfy demand in a given region and also seeks to minimize the travel distance from population to facilities.

In addition to natural disasters, there has also been work done on terror attacks. Miller et al. [68] and Miller et al. [69] develop models to show how an existing network of care resources can respond to a bio-terror attack. The approach involves integrating two different stochastic models: (1) a model of disease progression and casualty events, and (2) a model of health care network of resources and requirements for victims at different stages of progression.

Models of disaster preparedness and emergency response share similarities with our work; however, they typically concern rare events that require a rapid response. Our system involves the disease evolution of malaria over an entire malaria season along with fixed warehousing facilities and the political sensitivity of taking supplies from one facility to another. Also, in contrast to applications of emergency response is that, rather than being a rare and catastrophic event with unknown timing, malaria follows predictable seasonal patterns. This means our model can address immediate health needs and continue to impact health demand year after year.

The other body of literature that shares common features with our work is the area of disease prevention resource allocation. This literature typically considers public health response and budget allocation to specific disease treatments to mitigate the effects or prevent the spread of a particular disease. Dimitrov and Morton [29] develop a two-stage approach where the first stage represents the one-time "design" decision followed by a second stage of optimal control of the system via MDP. It applies this approach to the prevention of the spread of malaria in Nigeria considering objectives of minimizing the number of deaths and economic impact of the disease subject to a fixed budget by considering allocation strategies based on a geographical grid. Dimitrov et al. [28] develop mathematical models for determining optimal geo-temporal stockpiles of anti-viral medication and distribution tactics to minimize the impact of an influenza pandemic using optimization combined with simulation. Due to the size of our problem, such approaches would be difficult to tractably employ.

Hutton et al. [50] develop a Markov decision model to assess the effectiveness of various treatment strategies, and shows that catch-up vaccinations have been extremely beneficial in China. Optimal allocation of resources for treating epidemics is considered in Zaric and Brandeau [95] and Zaric and Brandeau [96]. These works support our hypothesis that a dynamic policy can be more effective in treating disease outbreaks than allocating all resources at the beginning of the period. A number of models consider the spread and prevention of HIV in various populations. Zaric and Brandeau [97] develop optimization models for HIV prevention resource allocation considering aggregate level allocation to regions and local level allocation to specific clinics. Lasry et al. [60] and Lasry et al. [59] develop non-linear optimization models to determine the allocation of HIV treatment and prevention budget over a five year horizon. The disease prevention models typically focus on the effectiveness

of treatment strategies and how much budget to use to employ different types of treatments. Furthermore, those models often have no geographical or transportation component as in our models. In our work we consider a more operational level, where the quantities of treatment are fixed and the goal is to allocate them most effectively to geographical regions based on the evolution of the disease over the course of the malaria season.

In the following sections we develop our model and present some of the results based on data – that has been disguised for confidentiality – from Malawi. As a final motivation for this work, we refer the reader to Foster [35], who claims that:

1. Proper inventory management of medications and and drugs in Africa can reduce costs an estimated 15-20%.

2. Transportation of drugs and medical aid is a more critical factor in Africa than in other developing countries.

Our research addresses these problems by integrating strategic and operational level models that can provide workable on-the-ground solutions to the problem of efficient ACTs distribution in Malawi. The chapter proceeds as follows. First, we develop a strategic level stochastic programming model to address the distribution of ACTs at an aggregate level. These models provide both practical results for ACTs pre-positioning and insight into an effective management structure for operationalizing a transshipment approach – *clinic clustering*. Clinic clusters enable the effective decomposition of the 290 clinic problem into manageable clusters of about 3 clinics each on average. The output of the strategic level stochastic program can then be used to parameterize a periodic review operational model for transshipping between clinics. Clinic cluster decomposition enables tractable solutions to this periodic review model

at the cluster level using a Markov decision process (MDP) approach. From the MDP we also gain valuable insights into the structure of the optimal transshipment policy, which we show to be of threshold type.

4.3 Deterministic Model for Medication Distribution

To demonstrate the effectiveness of incorporating demand uncertainty in distribution decisions we first define a "baseline" model as a surrogate for the current state of ACTs distribution in Malawi. The goal of this model is to minimize the expected transportation costs and shortage penalties. All the distribution decisions are made upfront, before the malaria season. Current practice of ACTs distribution lacks real time updates and recourse actions. Therefore, distribution decisions are made only once, before the malaria season, based on projected demand obtained through observations of case demand. We introduce notation in Table 4.1.

$\mathcal{N}$	Set of nodes consisting of the central pharmaceutical warehouse (w), regional warehouses ($\mathcal{R}$), district hospitals ($\mathcal{D}$), and local clinics ($\mathcal{C}$)
$\mathcal{A}^{\mathcal{R}}$	Subset of arcs connecting the central warehouse to regional warehouses
$\mathcal{A}^{\mathcal{D}}$	Subset of arcs connecting regional warehouses to district hospitals
$\mathcal{A}^{\mathcal{C}}$	Subset of arcs connecting district hospitals to local clinics
$\mathcal{A}$	Set of arcs ($\mathcal{A} = \mathcal{A}^{\mathcal{R}} \cup \mathcal{A}^{\mathcal{D}} \cup \mathcal{A}^{\mathcal{C}}$)
$\mathcal{S}$	Set of demand scenarios
p_s	Probability of scenario s where $s \in \mathcal{S}$
π_i	Penalty of one unit of treatment shortage in clinic i
c_{ij}	Cost of transporting one unit of treatment on arc (i, j)
σ	Total available supply of treatments
d_i^s	Demand of local clinic i under scenario s where $i \in \mathcal{C}$ and $s \in \mathcal{S}$

Table 4.1: Distribution model notation.

The main decision variable in the baseline model, x_{ij}, corresponds to the number of malaria treatments transported on arc (i, j). In other words, all distribution decisions from the central warehouse to the regional warehouses, from the regional warehouses to the district hospitals, and from district hospitals to the local clinics are made at the same time, based on some historical estimate of the demand. Therefore, it is

quite possible that when the malaria season starts and the actual demand is realized, some local clinics will face supply shortages. An auxiliary variable, z_j^s is therefore introduced to capture the shortage of malaria treatments in clinic j under scenario s, in which a demand of d_i^s is realized for clinic i. The min-cost flow formulation introduced in (4.1)-(4.7) represents the baseline model.

$$\min \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} + \sum_{s\in\mathcal{S}}\sum_{i\in\mathcal{C}} p_s\pi_i z_i^s \tag{4.1}$$

s.t.

$$\sum_{j:(w,j)\in\mathcal{A}^{\mathcal{R}}} x_{ij} \leq \sigma \tag{4.2}$$

$$\sum_{j:(i,j)\in\mathcal{A}^{\mathcal{D}}} x_{ij} = \sum_{j:(j,i)\in\mathcal{A}^{\mathcal{R}}} x_{ji} \qquad \forall i \in \mathcal{R} \tag{4.3}$$

$$\sum_{j:(i,j)\in\mathcal{A}^{\mathcal{C}}} x_{ij} = \sum_{j:(j,i)\in\mathcal{A}^{\mathcal{D}}} x_{ji} \qquad \forall i \in \mathcal{D} \tag{4.4}$$

$$\sum_{j:(j,i)\in\mathcal{A}^{\mathcal{C}}} x_{ji} = d_i^s - z_i^s \qquad \forall i \in \mathcal{C}, \forall s \in \mathcal{S} \tag{4.5}$$

$$x_{ij} \geq 0 \qquad \forall (i,j) \in \mathcal{A} \tag{4.6}$$

$$z_i^s \geq 0 \qquad \forall i \in \mathcal{C}, \forall s \in \mathcal{S}. \tag{4.7}$$

The objective function (4.1) represents the cost minimization goal. This includes the total transportation costs from the central warehouse to the local clinics and the shortage penalty. Constraint (4.2) guarantees that no more than the available supply (σ) will be delivered to the regional warehouses. Constraints (4.3), i.e. flow conservation constraints, guarantee the distribution of treatments from regional warehouses to district hospitals. Similarly, constraints (4.4) ensure the flow of treatments from regional warehouses to local clinics. Finally, constraints (4.5) represent the flow conservation in each local clinic. The left-hand-side represents the total flow of treatments into local clinic i and the right-hand-side of (4.5) corresponds to the

total demand of clinic i under scenario s (d_i^s) minus the shortage in that clinic under scenario s (z_i^s). If the treatments in clinic i are not enough to address all the demand under the realized scenario, the unsatisfied demand is considered as shortage. Constraints (4.6) and (4.7) ensure non-negativity of the decision variables.

4.4 Two-Stage Stochastic Formulation

The models in this section contrast with the baseline model in that the demand for each clinic is considered unknown when the decisions about the transportation of treatments to each major inventory is made, but information and recourse actions are available in the second stage. In the *first stage* of these models, the Malawi Ministry of Health would decide how many treatments to send from the inventory locations to each clinic before the malaria season begins. In the *second stage*, the actual demand is realized and the Ministry can take *recourse* action to address the supply and demand mismatch. Here we consider two potential recourse actions: (1) transshipment and (2) delayed shipment.

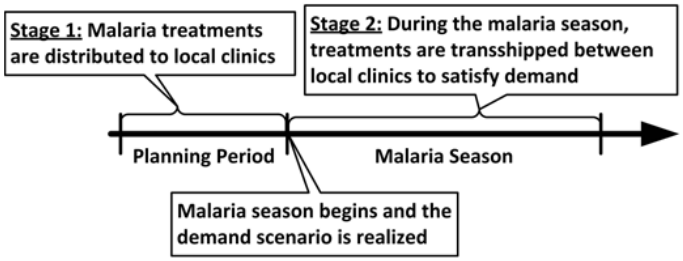
In the transshipment model, all the treatments are distributed among the clinics in the first stage. In the second stage, transshipment of treatments between clinics occurs to adjust inventories in light of new demand information. Hence clinics with high inventories can ship their extra inventory to the clinics experiencing shortages. The mathematical formulation for this case is presented in Section 4.4.1.

In the delayed shipment model, an initial delivery of treatments is distributed to the clinics, but some is held back at the higher tier. As the malaria season begins, a better estimate of the demand is realized and a second round of shipments is delivered. The mathematical formulation for this case is presented in Section 4.4.2.

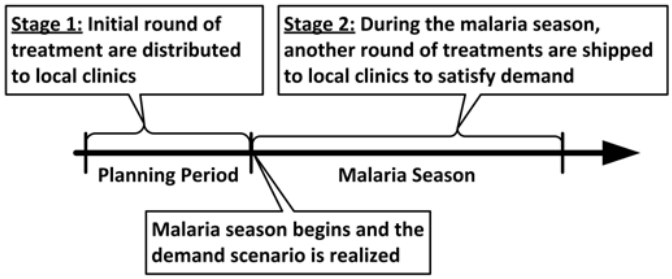
There are benefits and drawbacks to both types of models. The delayed shipment

model is less cost effective, but from an implementation standpoint it has the political benefit of not having to take stock away from one clinic to give to another.

Figure 4.2 illustrates the timeline of events for the two-stage stochastic models. Note that the recourse actions are not necessarily done all at once. Instead, the transshipment or delayed shipments are made throughout the malaria season as needed. Therefore, the recourse decisions considered here are *aggregate-level surrogates* for the actual periodic adjustments in the inventory level of each local clinic.



(a) Two-Stage Transshipment Model



(b) Two-Stage Delayed Shipment Model

Figure 4.2: Event timelines for two-stage stochastic models.

4.4.1 Case I - Transshipment between Clinics

In this strategic planning model, during the malaria season medications can be transshipped from one clinic with an excess inventory to a clinic facing a supply shortage. Under this formulation, we assume a set of scenarios ($\mathcal{S}$) where each scenario, $s \in \mathcal{S}$ is realized with probability p_s. Under scenario s, the realized value

of demand for clinic i is d_i^s. The first stage problem is formulated as follows:

$$\min \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} + \mathcal{Q} \tag{4.8}$$

s.t.

$$\sum_{j:(m,j)\in\mathcal{A}^{\mathcal{R}}} x_{ij} \leq \sigma \tag{4.9}$$

$$\sum_{j:(i,j)\in\mathcal{A}^{\mathcal{R}}} x_{ij} = \sum_{j:(j,i)\in\mathcal{A}^{\mathcal{D}}} x_{ji} \qquad \forall i \in \mathcal{R} \tag{4.10}$$

$$\sum_{j:(i,j)\in\mathcal{A}^{\mathcal{D}}} x_{ij} = \sum_{j:(j,i)\in\mathcal{A}^{\mathcal{C}}} x_{ji} \qquad \forall i \in \mathcal{D} \tag{4.11}$$

$$x_{ij} \geq 0 \qquad \forall (i,j) \in \mathcal{A}. \tag{4.12}$$

The first term in the objective function (4.8) minimizes shipping costs, similar to the deterministic objective function. The second term, $\mathcal{Q}$ captures the expected cost of the second stage (recourse) decision. Constraints (4.9) and (4.10) are the flow conservation constraints. Here we define the *expected recourse function*, $\mathcal{Q}$:

$$\mathcal{Q} = \min \sum_{s\in\mathcal{S}} p_s \Big(\sum_{(i,j)\in\mathcal{A}^{\mathcal{C}}} c_{ij}y_{ij}^s + \sum_{i\in\mathcal{C}} \pi_i z_i^s \Big) \tag{4.13}$$

s.t.

$$\sum_{j:(j,i)\in\mathcal{A}^{\mathcal{C}}} y_{ji}^s - \sum_{j:(i,j)\in\mathcal{A}^{\mathcal{C}}} y_{ij}^s + z_i^s \geq - \sum_{j:(j,i)\in\mathcal{A}^{\mathcal{C}}} x_{ji} + d_i^s \qquad \forall i \in \mathcal{C}, \forall s \in \mathcal{S} \tag{4.14}$$

$$y_{ij}^s \geq 0 \qquad \forall (i,j) \in \mathcal{A}^{\mathcal{C}}, \forall s \in \mathcal{S} \tag{4.15}$$

$$z_i^s \geq 0 \qquad \forall i \in \mathcal{C}, \forall s \in \mathcal{S}. \tag{4.16}$$

In the above formulation, the decision variable y_{ij}^s corresponds to the aggregate transshipment of treatments from clinic i to clinic j under scenario s through the malaria season. The recourse function, $\mathcal{Q}$, as defined in (4.13), captures the cost of the recourse action – the transshipment cost between the clinics plus the penalty corresponding to the shortage of medications in clinic i under scenario s at the end of

the malaria season, denoted by z_i^s. Constraints (4.14) correspond to flow conservation – the amount of treatment sent from node j to node i (y_{ji}^s) minus the number of treatments (y_{ij}^s) transported from node i to node j is greater and equal to the realized demand under scenario s, i.e. d_i^s minus the original number of treatments assigned to clinic i in the first stage ($\sum_{j:(j,i)\in\mathcal{A}^{\mathcal{D}}} x_{ji}$). The new decision variable z_i^s captures the shortage in clinic i and is subtracted from the left hand side of constraints (4.14). Note that the value of first-stage decisions x_{ij} is known in the second stage, therefore $\sum_{j:(j,i)\in\mathcal{A}^{\mathcal{D}}} x_{ji}$ is a known value.

4.4.2 Case II - Delayed Shipment

In this case, some portion of the shipment of treatments is reserved for shipment after the start of the malaria season. Similar to the transshipment model, we define d_i^s as the demand of clinic i under scenario s. Here is the first stage problem formulation:

$$\min \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} + \mathcal{Q} \tag{4.17}$$

s.t.

$$\sum_{j:(m,j)\in\mathcal{A}^{\mathcal{R}}} x_{ij} \leq \sigma \tag{4.18}$$

$$\sum_{j:(i,j)\in\mathcal{A}^{\mathcal{R}}} x_{ij} = \sum_{j:(j,i)\in\mathcal{A}^{\mathcal{D}}} x_{ji} \qquad \forall i \in \mathcal{R} \tag{4.19}$$

$$\sum_{j:(i,j)\in\mathcal{A}^{\mathcal{D}}} x_{ij} \geq \sum_{j:(j,i)\in\mathcal{A}^{\mathcal{C}}} x_{ji} \qquad \forall i \in \mathcal{D} \tag{4.20}$$

$$x_{ij} \geq 0 \qquad \forall (i,j) \in \mathcal{A}. \tag{4.21}$$

The objective function (4.17) and constraints (4.18) and (4.19) are similar to the ones in the transshipment model. However, constraints (4.20) are changed from equality to inequality. This is because we do not ship all the supply at the first stage. Thus, some reserved inventory stays at each district hospital. Here we define the *expected*

recourse function, $\mathcal{Q}$ for delayed shipment:

$$\mathcal{Q} = \min \sum_{s \in \mathcal{S}} p_s \Big(\sum_{(i,j) \in \mathcal{A}^{\mathcal{C}}} c'_{ij} w^s_{ij} + \sum_{i \in \mathcal{C}} \pi_i z^s_i \Big) \tag{4.22}$$

s.t.

$$\sum_{j:(i,j) \in \mathcal{A}^{\mathcal{C}}} w^s_{ij} \leq \sum_{j:(j,i) \in \mathcal{A}^{\mathcal{D}}} x_{ij} - \sum_{j:(i,j) \in \mathcal{A}^{\mathcal{C}}} x_{ij} \qquad \forall i \in \mathcal{D}, \forall s \in \mathcal{S} \tag{4.23}$$

$$\sum_{j:(j,i) \in \mathcal{A}^{\mathcal{C}}} w^s_{ji} + z^s_i \geq - \sum_{j:(j,i) \in \mathcal{A}^{\mathcal{C}}} x_{ji} + d^s_i \qquad \forall i \in \mathcal{C}, \forall s \in \mathcal{S} \tag{4.24}$$

$$w^s_{ij} \geq 0 \qquad \forall (i,j) \in \mathcal{A}^{\mathcal{C}}, \forall s \in \mathcal{S} \tag{4.25}$$

$$z^s_i \geq 0 \qquad \forall i \in \mathcal{C}, \forall s \in \mathcal{S}. \tag{4.26}$$

In the recourse problem defined by (4.22)-(4.26), w^s_{ij} denotes the amount of treatments shipped from district hospital i to clinic j through the malaria season. The objective function (4.22) minimizes the expected transportation costs and shortage penalties. c'_{ij} denotes the cost of delayed shipment between district hospital i and local clinic j. Due to practical reasons such as economy of scale, it is safe to assume that the initial round of shipments are less expensive than delayed shipments, i.e. $c'_{ij} > c_{ij}$.

Constraints (4.23) ensure that the second round of shipments ($\sum_{j:(m,j) \in \mathcal{A}^{\mathcal{D}}} w^s_{ij}$) will distribute all treatments left from the first stage ($\sigma - \sum_{j:(m,j) \in \mathcal{A}^{\mathcal{D}}} x_{ij}$). Constraints (4.23) ensure the conservation of flow from the central warehouse to each clinic. Constraints (4.24) capture the shortage in each clinic (z^s_i) after the second round of treatments are distributed. Constraints (4.25) and (4.26) ensure the non-negativity of shortage values.

In the next section, we discuss modeling techniques that ensure equitable distribution in the face of supply shortages.

4.4.3 Equity of Shortage

When the total supply of malaria treatments is less than the demand, shortage is inevitable. However, depending on the specific distribution scenario, some clinics may face higher shortages than others. The issue of shortage equity among clinics can be addressed such that the optimization model does not generate solutions in which some clinics have significantly higher shortage values than others.

To do so, instead of minimizing the *sum* of shortage values, we minimize the *sum of absolute differences* between the shortage of each clinic and the average shortage among all clinics. Define $\bar{z}^s$ as the average shortage of treatments in all the clinics in scenario s. Also define $\tilde{z}_i^s$ as the absolute difference between the shortage in clinic i and the average shortage. This way, we need to add the following constraints to the recourse problems.

$$\bar{z}^s = \frac{1}{|\mathcal{C}|} \sum_{i \in \mathcal{C}} z_i^s \qquad \forall s \in \mathcal{S} \tag{4.27}$$

$$\tilde{z}_i^s \geq z_i^s - \bar{z}^s \qquad \forall s \in \mathcal{S}, \forall i \in \mathcal{C} \tag{4.28}$$

$$\tilde{z}_i^s \geq -z_i^s + \bar{z}^s \qquad \forall s \in \mathcal{S}, \forall i \in \mathcal{C} \tag{4.29}$$

$$\tilde{z}_i^s \geq 0. \tag{4.30}$$

Equations (4.27) define the average shortage. Equations (4.28) - (4.30) linearize the absolute value function. Based on the above definition of equity, we can modify objective function in each model by adding a new term $\sum_{s,i} \pi_i \tilde{z}_i^s$ to the objective function.

Note that minimizing the sum of absolute differences is not the only approach to maintain equity. Other approaches such as minimizing the maximum shortage, or minimizing the difference between the minimum and the maximum shortage values can also be easily implemented and still maintain linearity of the model.

Stepwise Transportation Cost

A potential realistic extension to the stochastic model is to consider a case where treatments have to be shipped in batches. In those cases, a step-wise transportation cost function (see Figure 4.3) will represent the model better than linear cost.

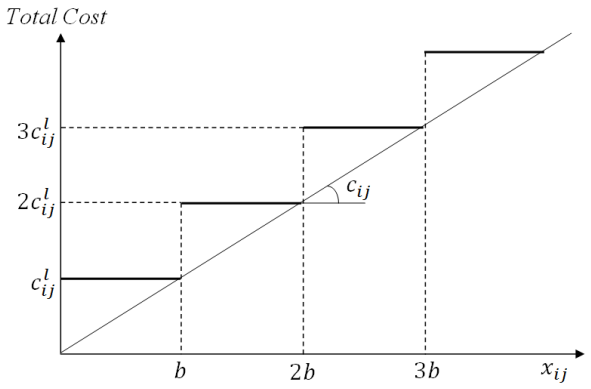


Figure 4.3: Stepwise transportation cost function.

To do so, instead of minimizing the sum of linear transportation cost, we minimize the sum of number of *truckloads* for shipping the treatment. We define u_{ij} as number of loads shipped on arc (i, j). We assume the capacity of each truck is known and fixed to b.

$$\frac{x_{ij}}{b} \leq u_{ij} \leq \frac{x_{ij}}{b} + 1 \qquad \forall (i,j) \in \mathcal{A} \tag{4.31}$$

$$u_{ij} \geq 0, \text{ integer} \qquad \forall s \in \mathcal{S}, \forall (i,j) \in \mathcal{A} \tag{4.32}$$

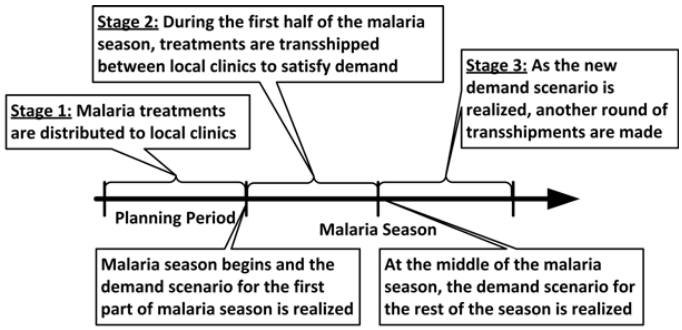
Constraints (4.31) define the decision variable u_{ij} as the number of truckloads i.e. $\lceil \frac{x_{ij}}{b} \rceil$. Constraints (4.32) ensure that the new decision variable is non-negative and integer. As a result in the objective function, instead of terms $\sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij}$, we have the new term $\sum_{(i,j)\in\mathcal{A}} c^l_{ij}u_{ij}$.

Note that the assuming a stepwise transportation cost function will turn the model into a mixed-integer program. This increases the problem complexity and solution time.

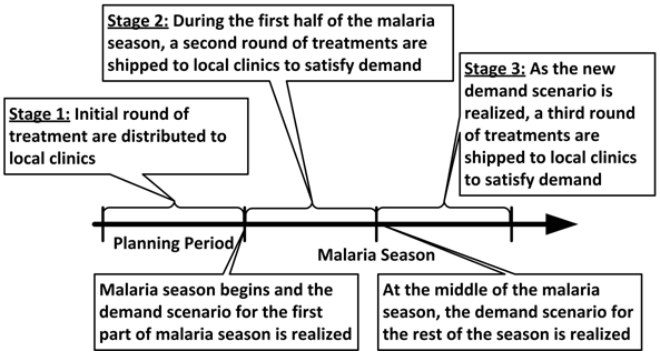
4.5 Three-Stage Stochastic Formulation

In Section 4.4, we considered two-stage models in which the demand scenario for each clinic is realized at the beginning of the malaria season. Recourse actions are then triggered to address the disparity between the realized demand and the initial inventory of treatments at each clinic - either through transshipment or using another round of delayed shipments. One major drawback of the two-stage model is that the recourse actions are aggregate-level surrogates for the actual periodic decisions. All the shipments that are made through the malaria season are aggregated as a one-shot recourse decision made for each clinic's entire demand through the season. This assumption helps simplify the problem and make the solution tractable. The downside here is that the temporal (e.g. bi-monthly, monthly, weekly, etc) fluctuations in the demand for each clinic are ignored. Therefore, when the actual temporal demand fluctuations are high, the two-stage models may underestimate the actual shortage in each period.

One potential remedy for this issue is to increase the granularity of the recourse actions. For instance, the transshipments or delayed shipments can be delivered periodically. This way, the model can better estimate the actual shortage in each period and trigger more precise actions. To implement such a model, a better understanding of the *proper* level of granularity is needed. This requires answering questions such as: should the model consider the monthly demand for each clinic and compute the monthly recourse actions, or should it happen on a weekly basis? As the granularity of the model increases, the computation time increases dramatically. Moreover, collecting and processing the demand data at a very detailed level might not be feasible in a developing nation.



(a) Three-Stage Transshipment Model



(b) Three-Stage Delayed Shipment Model

Figure 4.4: Event timelines for three-stage stochastic models.

To improve the granularity of the decisions, here we present three-stage models based on event timelines displayed in Figure 4.4. Similar to the models in Section 4.4, we consider two possible recourse actions: transshipment and delayed shipment. The only difference here is that the recourse actions are made in stages two and three. The three-stage transshipment and delayed shipment models are described in Sections 4.5.1 and 4.5.2 respectively.

4.5.1 Three-Stage Transshipment Model

For the case of three-stage clinic transshipment, the formulation for the first stage problem is identical to the two-stage problem as defined by (4.8)-(4.12). In the second stage, however, we add another term ($\mathcal{Q}'$) to the objective function (4.33) to represent the third-stage problem. We also define a new auxiliary decision variable to capture the extra inventory left in each clinic at the end of the second stage. l_i^s represents the number of malaria treatments left at clinic i under scenario s after the second stage. This way, the term $-l_i^s$ is added to the left-hand-side of the flow conservation constraints (4.38). This variable is then used in the third-stage problem as input data.

$$\mathcal{Q} = \min \sum_{s\in\mathcal{S}} p_s \Big(\sum_{(i,j)\in\mathcal{A}^{\mathcal{C}}} c_{ij} y_{ij}^s + \sum_{i\in\mathcal{C}} \pi_i z_i^s \Big) + \mathcal{Q}' \tag{4.33}$$

s.t.

$$\sum_{j:(j,i)\in\mathcal{A}^{\mathcal{C}}} y_{ji}^s - \sum_{j:(i,j)\in\mathcal{A}^{\mathcal{C}}} y_{ij}^s + z_i^s - l_i^s = - \sum_{j:(j,i)\in\mathcal{A}^{\mathcal{C}}} x_{ji} + d_i^s \qquad \forall i \in \mathcal{C}, \forall s \in \mathcal{S} \tag{4.34}$$

$$y_{ij}^s \geq 0 \qquad \forall (i,j) \in \mathcal{A}^{\mathcal{C}}, \forall s \in \mathcal{S} \tag{4.35}$$

$$z_i^s \geq 0, l_i^s \geq 0 \qquad \forall i \in \mathcal{C}, \forall s \in \mathcal{S}. \tag{4.36}$$

To formulate the third-stage problem ($\mathcal{Q}'$), we define a set of transshipment decisions made in the third stage. $y_{ij}^{'s}$ represents the number of treatment units transshipped from clinic i to clinic j under scenario s. In general, the transshipment cost in the third stage can be different from the second-stage transshipment cost. Thus we introduce a new parameter (c'_{ij}) to represent the transshipment cost in the third stage. The third-stage problem is defined by (4.37)-(4.40).

$$\mathcal{Q}' = \min \sum_{s\in\mathcal{S}} p_s \Big(\sum_{(i,j)\in\mathcal{A}^{\mathcal{C}}} c'_{ij} y_{ij}^{'s} + \sum_{i\in\mathcal{C}} \pi_i z_i^{'s} \Big) \tag{4.37}$$

s.t.

$$\sum_{j:(i,j)\in\mathcal{A}^{\mathcal{C}}} y_{ij}^{'s} - \sum_{j:(j,i)\in\mathcal{A}^{\mathcal{C}}} y_{ji}^{'s} + z_i^{'s} \geq -l_i^s + d_i^{'s} \qquad \forall i \in \mathcal{C}, \forall s \in \mathcal{S} \tag{4.38}$$

$$y_{ij}^{'s} \geq 0 \qquad \forall (i,j) \in \mathcal{A}^{\mathcal{C}}, \forall s \in \mathcal{S} \tag{4.39}$$

$$z_i^{'s} \geq 0 \qquad \forall i \in \mathcal{C}, \forall s \in \mathcal{S}. \tag{4.40}$$

Equation (4.37) defines the objective function of the third-stage transshipment problem. The goal is to minimize the expected transshipment cost ($\sum_{s\in\mathcal{S}} p_s \sum_{(i,j)\in\mathcal{A}^{\mathcal{C}}} c'_{ij} y_{ij}^{'s}$) plus the expected shortage penalty ($\sum_{s\in\mathcal{S}} p_s \sum_{i\in\mathcal{C}} \pi_i z_i^{'s}$) incurred in the third stage. The flow conservation constraints defined by (4.38) ensure that the existing inventory at clinic i under scenario s ($\sum_{j:(i,j)\in\mathcal{A}^{\mathcal{C}}} y_{ij}^{'s} - \sum_{j:(j,i)\in\mathcal{A}^{\mathcal{C}}} y_{ji}^{'s} + l_i^s$) is enough to satisfy clinic-i's demand in the third stage under that scenario ($d_i^{'s}$). If not, the unsatisfied demand will be counted toward shortage ($z_i^{'s}$). We rearranged the flow conservation constraints (4.38) so that the decisions (y' and z') are on the left-hand-side and the input data (l and d') are on the right-hand-side.

4.5.2 Three-Stage Delayed Shipment Model

For the case of three-stage delayed shipment, the formulation for the first stage problem is identical to the two-stage problem as defined by (4.17)-(4.21). Similar

to the idea presented in Section 4.5.1, we add another term ($\mathcal{Q}'$) to the objective function (4.41) to represent the third-stage problem. Similarly, a new auxiliary decision variable (l_i^s) is defined to capture the extra inventory left in each clinic at the end of the second stage.

$$\mathcal{Q} = \min \sum_{s\in\mathcal{S}} p_s \Big(\sum_{(i,j)\in\mathcal{A}^{\mathcal{C}}} c_{ij} w_{ij}^s + \sum_{i\in\mathcal{C}} \pi_i z_i^s \Big) + \mathcal{Q}' \tag{4.41}$$

s.t.

$$\sum_{j:(i,j)\in\mathcal{A}^{\mathcal{C}}} w_{ij}^s \le \sum_{j:(j,i)\in\mathcal{A}^{\mathcal{D}}} x_{ij} - \sum_{j:(i,j)\in\mathcal{A}^{\mathcal{C}}} x_{ij} \qquad \forall i \in \mathcal{D}, \forall s \in \mathcal{S} \tag{4.42}$$

$$\sum_{j:(j,i)\in\mathcal{A}^{\mathcal{C}}} w_{ji}^s + z_i^s - l_i^s = - \sum_{j:(j,i)\in\mathcal{A}^{\mathcal{C}}} x_{ji} + d_i^s \qquad \forall i \in \mathcal{C}, \forall s \in \mathcal{S} \tag{4.43}$$

$$w_{ij}^s \ge 0 \qquad \forall (i,j) \in \mathcal{A}^{\mathcal{C}}, \forall s \in \mathcal{S} \tag{4.44}$$

$$z_i^s \ge 0, l_i^s \ge 0 \qquad \forall i \in \mathcal{C}, \forall s \in \mathcal{S}. \tag{4.45}$$

To formulate the third-stage problem ($\mathcal{Q}'$), we define a set of delayed shipment decisions made in the third stage. $w_{ij}^{'s}$ represents the number of treatment units shipped from district hospital i to clinic j under scenario s in the third stage. In general, the shipment cost in the third stage can be different from that of the second-stage. Thus, similar to what we did in Section 4.4.2, we introduce a new parameter (c_{ij}'') to represent the cost of shipment in the third stage. This way, the third-stage delayed shipment problem is defined by (4.46)-(4.50).

$$\mathcal{Q}' = \min \sum_{s\in\mathcal{S}} p_s \Big(\sum_{(i,j)\in\mathcal{A}^{\mathcal{C}}} c_{ij}'' w_{ij}^{'s} + \sum_{i\in\mathcal{C}} \pi_i z_i^{'s} \Big) \tag{4.46}$$

s.t.

$$\sum_{j:(i,j)\in\mathcal{A}^{\mathcal{C}}} w_{ij}^{'s} \le \sum_{j:(j,i)\in\mathcal{A}^{\mathcal{D}}} x_{ij} - \sum_{j:(i,j)\in\mathcal{A}^{\mathcal{C}}} x_{ij} - \sum_{j:(i,j)\in\mathcal{A}^{\mathcal{C}}} w_{ij}^s \qquad \forall i \in \mathcal{D}, \forall s \in \mathcal{S} \tag{4.47}$$

$$\sum_{j:(j,i)\in\mathcal{A}^{\mathcal{C}}} w_{ji}^{'s} + z_i^{'s} \ge -l_i^s + d_i^{'s} \qquad \forall i \in \mathcal{C}, \forall s \in \mathcal{S} \tag{4.48}$$

$$w_{ij}^{'s} \ge 0 \qquad \forall (i,j) \in \mathcal{A}^{\mathcal{C}}, \forall s \in \mathcal{S} \tag{4.49}$$

$$z_i^{'s} \ge 0 \qquad \forall i \in \mathcal{C}, \forall s \in \mathcal{S}. \tag{4.50}$$

Equation (4.46) defines the objective function of the third-stage delayed shipment problem. The goal here is to minimize the expected shipment cost $(\sum_{s\in\mathcal{S}} p_s \sum_{(i,j)\in\mathcal{A}^{\mathcal{C}}} c''_{ij} w'^{s}_{ij})$ plus the expected shortage penalty $(\sum_{s\in\mathcal{S}} p_s \sum_{i\in\mathcal{C}} \pi_i z'^{s}_i)$ incurred in the third stage.

Constraints (4.47) ensures that the amount of treatments shipped out of district hospital i $(\sum_{j:(i,j)\in\mathcal{A}^{\mathcal{C}}} w'^{s}_{ij})$ is less than or equal to the amount received from all the regional warehouses in stage one $(\sum_{j:(j,i)\in\mathcal{A}^{\mathcal{D}}} x_{ij})$ minus the number of treatments shipped out to local clinics in the first stage $(\sum_{j:(i,j)\in\mathcal{A}^{\mathcal{C}}} x_{ij})$, minus the delayed shipment to the local clinics in the second stage $(\sum_{j:(i,j)\in\mathcal{A}^{\mathcal{C}}} w^{s}_{ij})$.

The flow conservation constraints defined by (4.48) ensure that the existing inventory at clinic i under scenario s $(\sum_{j:(j,i)\in\mathcal{A}^{\mathcal{C}}} w'^{s}_{ji} + l^{s}_{i})$ is enough to satisfy clinic-i's demand in the third stage under that scenario (d'^{s}_{i}). If not, the unsatisfied demand will be counted toward shortage (z'^{s}_{i}). We rearranged the flow conservation constraints so that the decisions (w' and z') are on the left-hand-side and the input data (l and d') are on the right-hand-side.

4.6 Computational Experiments

In this section, we present the results of computational experiments we performed based on actual locations of health facilities from a country-wide survey conducted by the Japanese International Cooperative Agency (JICA). We used disguised historical data (for confidentiality) from each facility based on district level malaria prevalence estimates and approximated facility catchment using Thiessen polygons and the 1998 Malawian Census. Then, for each clinic we randomly generated 10 scenarios to populate demand parameters (d^{s}_{i}). Our data includes a total of 290 facilities including 3 regional warehouses, 21 hospitals and 266 local clinics. We ran the first set of

experiments for a supply value of 1.2 million units.

To capture the transportation cost ($_{cij}$) we calculated the distance between each facility pair in kilometers. We used Lall et al. [58] to obtain an estimate for the unit transportation cost. Road quality in Malawi varies widely and the system is mostly underdeveloped. Thus, the unit transportation cost can vary depending on route. For our computational experiments, we used the average transportation cost per kilometer in Malawi reported by Lall et al. [58] which is about 4 cents (or 228.4 kwacha, the Malawian currency).

Estimating the shortage penalty is non-trivial. We considered Malawi's national income per capita, $810 reported by the WHO as a basis. We started with $20 as an initial estimate for the shortage penalty. Based on our initial computation, even a low number (such as $20) is high enough to guarantee effective distribution of medications so that all available treatment units are distributed while supplies are available. Thus the stochastic models minimize the expected shortage. For future extensions, one can perform a sensitivity analysis to quantify the impact of shortage penalty on the outcomes of the model. However, based on our initial findings, any value of shortage penalty above $20 will result in the same distribution pattern.

4.6.1 Comparing the Stochastic Models to the Baseline

Metric (Expected value)	Baseline model	Two-stage stochastic models			
		Value		Reduction	
		Del. ship.	Transship.	Del. ship.	Transship.
Transportation cost	7,727,076	7,307,568	7,230,674	5.43%	6.42%
Shortage penalty	10,024,752	8,407,020	8,407,020	16.14%	16.14%
Total cost	17,761,050	15,725,834	15,664,164	11.46%	11.81%
Shortage volume	501,238	420,351	420,351	16.14%	16.14%

Table 4.2: Comparing two-stage stochastic models to the baseline.

In Tables 4.2 and 4.3 we observe that the two approaches we have introduced (i.e. transshipment and delayed shipment) result in similar outcomes in terms of reducing

Metric (Expected value)	Baseline model	Three-stage stochastic models			
		Value		Reduction	
		Del. ship.	Transship.	Del. ship.	Transship.
Transportation cost	7,727,076	7,230,308	6,879,559	6.43%	10.97%
Shortage penalty	10,024,752	8,407,020	8,407,020	16.14%	16.14%
Total cost	17,761,050	15,654,328	15,299,457	11.86%	13.86%
Shortage volume	501,238	420,351	420,351	16.14%	16.14%

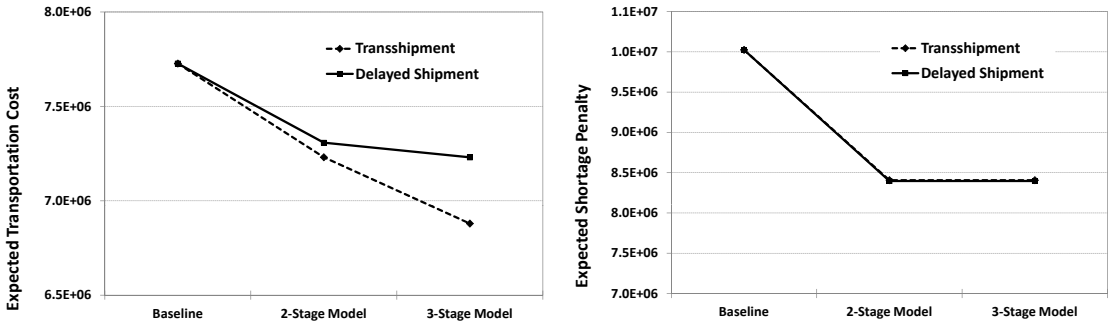
Table 4.3: Comparing three-stage stochastic models to the baseline.

the shortage penalty. However, the transshipment idea seems to be more effective in reducing transportation cost. The transshipment model can better exploit the network structure by allowing the transfer of stock between clinics in the vicinity. The delayed shipment, on the other hand, reserves some extra inventory in district hospitals and distributes it in the second stage over local clinics facing shortages. Thus, the delayed shipment model results in higher transportation costs.

4.6.2 Two-Stage vs. Three-Stage Models

Another result of the computational experiments is the marginal benefit of adding another recourse stage to the stochastic model. The three-stage stochastic model provides more opportunities to react to demand uncertainty, but adding more stages can potentially make the problem harder to solve. Generally speaking, it is difficult to determine if the marginal benefits of adding another decision stage to the model will necessarily justify the increased computational effort. Depending on the problem instance – especially features such as the total available malaria treatments and the demand uncertainty profiles – the marginal benefits of adding another decision stage can vary.

In Figure 4.6 we observe that both the two and the three-stage models reduce the expected shortage to the same level. In other words, based on the available supply and the realized demand scenarios, adding one extra recourse action does not have a significant impact on reducing shortage. However, the extra recourse action can



(a) Expected Transportation Cost (b) Expected Shortage Penalty

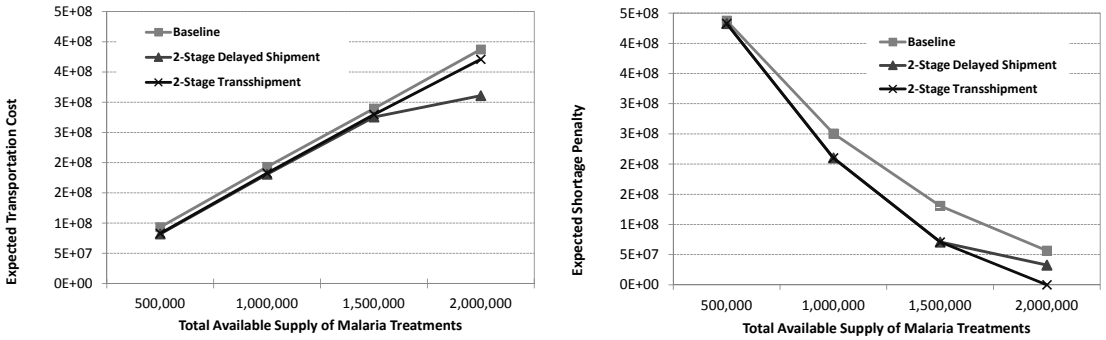
Figure 4.5: Expected cost of the stochastic models compared to the baseline.

result in a significant reduction in transportation cost.

4.6.3 Supply Availability

Supply availability is a major challenge in distributing malaria treatments in holoendemic areas. To better assess the effectiveness of our proposed stochastic models, we compare their results for a range of possible supply values, between 500,000 to 2,000,000 units. The results of this analysis are illustrated in Figure 4.6. As the number of available treatments is increased, the stochastic models' capability to reduce shortage and distribute treatments efficiently also increases. However, the transshipment model seems to be reducing the shortage more effectively. Here is an explanation: To avoid shortage, the delayed shipment model sends additional ACTs from district hospitals to local clinics after the demand is realized. The distance between district hospitals to local clinics, on average, is much higher than the distance between a local clinics and its neighboring clinics. Thus, in the instance where 2,000,000 ACTs are available, the delayed shipment model faces such a high-cost recourse action (due to high transportation cost between regional warehouses and local clinics) that it fails to address all the demand. In the transshipment model, however, the recourse actions are far less costly. Therefore the transshipment model

can effectively address all the demand with no shortage.



(a) Expected Transportation Cost (b) Expected Shortage Penalty

Figure 4.6: Available supply sensitivity analysis.

4.6.4 Clinic Clusters

One of the key insights gained from the computational experiments on the strategic level stochastic program is the appearance of what we call clinic clusters. That is, the stochastic program groups clinics together into clusters such that transshipment occurs within clusters only, and not between different clusters. In our transshipment model experiments, for example, 15% to 25% of clinics send treatments to other clinics in the recourse stage. These clinics transship their excess inventory to an average of 2 to 5 proximal receiver clinics. Based on this observation, we define the notion of *clinic clusters*. Each clinic consists of one *sender* clinic and between 2 to 5 *receiver* clinics in the vicinity of the sender clinic. Figure 4.7 illustrates five representative clinic clusters in the northern area of Malawi. Note that the actual size and structure of clusters depends on the problem instance. This idea of clusters can be used to decompose the country-level problem into tractable cluster problems that can be solved independently at the operational level. This is the key to integrating our strategic models with our operational models that we develop in Section 4.7.

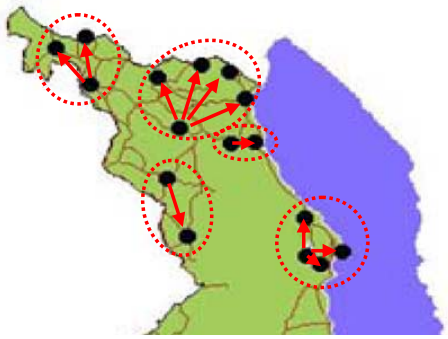


Figure 4.7: Five clinic clusters in the northern area of Malawi.

4.7 Operational Model for Transshipment in Clusters

For the purposes of strategic planning we have assumed in previous sections that demand can be completely satisfied by transshipment if enough supply exists to transship. In this section, we use the output of the strategic planning transshipment model and design an operational mechanism to manage the transshipment of treatments at a cluster level. The operational reality is that clinics will transship batches of treatments periodically, rather than transshipping demanded ACTs in a Just in Time manner that perfectly matches shipments to demand, which is the implicit modeling assumption of the stochastic program. To capture this reality, in this section we develop a periodic review model for transshipment between clinics within a cluster of clinics. These clusters are generated from the strategic-level stochastic models presented in Sections 4.4 and 4.5.

In our operational model, we consider each clinic cluster separately. We consider a bi-monthly periodic review in which clinics survey their inventory and then decide how much to transfer to other clinics. At the beginning of each period, each clinic incurs a shortage penalty proportional to the amount that demand exceeded inventory in the prior stage (indicated by a negative inventory value). Next, a decision is made regarding how much product to ship between clinics. Then demand arrives to

each clinic within the cluster according to a distribution $\mathbf{d} \sim F$ and then the state is updated for the next decision epoch. Notation is presented in table 4.4.

$\boldsymbol{\Xi}$	n-dimensional vector for the amount of inventory at each clinic for $\boldsymbol{\Xi} \in \mathbb{R}^n$
$\mathcal{U}_\Xi$	n-dimensional integer vector space where $\mathbf{u} \in \mathcal{U}$ is defined in Equation (4.52), which enforces flow conservation
Π	n-dimensional vector of shortage penalty
c	Unit cost of transshipment between clinics
$\mathbf{d}_n$	Random variable for pharmaceutical demand in period n
Φ	Set of clinics in the cluster, a subset of $\mathcal{C}$.

Table 4.4: Clinic transshipment model dynamic program notation.

$$f_n(\boldsymbol{\Xi}) = \Pi^{\mathrm{T}}(-\boldsymbol{\Xi})^+ + \min_{\mathbf{u}\in\mathcal{U}_\Xi}\{c\sum_{j\in\Phi}(u_j)^+ + \mathbb{E}\{f_{n-1}((\boldsymbol{\Xi})^+ + \mathbf{u} - \mathbf{d}_n)\}\}, \tag{4.51}$$

where the action space is given by

$$\mathcal{U}_\Xi = \{\mathbf{u} = (u_1, ..., u_s) : u_j \leq \xi_j \text{ and } \sum_{j\in\Phi} u_j = 0\}. \tag{4.52}$$

Equation (4.51) presents a finite horizon MDP formulation of the malaria treatment distribution within a cluster. Equation (4.52) presents the set of feasible actions, where an action will be taken if there is any inventory available at the clinics. Total volume of shipment out of each clinic is bounded above by its inventory.

Note that, the total medication transshipped between clinics is zero. The expected cost to go is based on the positive part of $\boldsymbol{\Xi}$, because in malaria treatment, the dynamics behave as "lost sales" not back orders.

Theorem IV.1. *$f_n(\boldsymbol{\Xi})$ is non-increasing in ξ_j for all n and j.*

Proof. We prove this theorem by induction. *Base Case:* $f_0(\boldsymbol{\Xi}) = 0$ for all $\boldsymbol{\Xi}$ and therefore is trivially non-increasing.

Induction Step:

Assume$f_{n-1}(\boldsymbol{\Xi})$ is non-increasing in $\boldsymbol{\Xi}$.

$$f_n(\boldsymbol{\Xi}) - f_n(\boldsymbol{\Xi} - e_j) = \Pi^{\mathrm{T}}(-\boldsymbol{\Xi})^+ + \min_{\mathbf{u}\in\mathcal{U}_{\boldsymbol{\Xi}}}\{c\sum_{j\in\Phi}(u_j)^+ + \mathbb{E}\{f_{n-1}((\boldsymbol{\Xi})^+ + \mathbf{u} - \mathbf{d}_n)\}\} -$$

$$\Pi^{\mathrm{T}}(-(\boldsymbol{\Xi} - e_j))^+ - \min_{\mathbf{u}\in\mathcal{U}_{\boldsymbol{\Xi}-e_j}}\{c\sum_{j\in\Phi}(u_j)^+ + E\{f_{n-1}((\boldsymbol{\Xi} - e_j)^+ + \mathbf{u} - \mathbf{d}_n)\}\}.$$

We compare the equation term by term. First, the instantaneous cost is clearly greater in the system with less inventory:

$$\Pi^{\mathrm{T}}((-\boldsymbol{\Xi})^+ - (-(\boldsymbol{\Xi} - e_j))^+) \leq 0. \tag{4.53}$$

Next we compare the minimization term. If the optimal action, $\mathbf{u}^*$, is the same in both $f_n(\boldsymbol{\Xi})$ and $f_n(\boldsymbol{\Xi} - e_j)$, it follows from the induction hypothesis that

$$E\{f_{n-1}((\boldsymbol{\Xi})^+ + \mathbf{u}^* - \mathbf{d}_n)\} - E\{f_{n-1}((\boldsymbol{\Xi} - e_j)^+ + \mathbf{u}^* - \mathbf{d}_n)\} \leq 0.$$

If, on the other hand, the optimal actions for $f_n(\boldsymbol{\Xi})$ and $f_n(\boldsymbol{\Xi} - e_j)$ are different, without loss of generality we assume that optimal action in state $(\boldsymbol{\Xi} - e_i)$ is $\mathbf{u}^0$. We then have that

$$\begin{aligned}
&\min_{\mathbf{u}\in\mathcal{U}_{\boldsymbol{\Xi}}}\{c\sum_{j\in\Phi}(u_j)^+ + \mathbb{E}\{f_{n-1}(\boldsymbol{\Xi} + \mathbf{u} - \mathbf{d}_n)\}\} - \\
&c\sum_{j\in\Phi}(u_j^0)^+ - E\{f_{n-1}((\boldsymbol{\Xi} - e_j)^+ + \mathbf{u}^0 - \mathbf{d}_n)\} \leq \\
&c\sum_{j\in\Phi}(u_j^0)^+ + E\{f_{n-1}((\boldsymbol{\Xi})^+ + \mathbf{u}^0 - \mathbf{d}_n) - \\
&c\sum_{j\in\Phi}(u_j^0)^+ - E\{f_{n-1}((\boldsymbol{\Xi} - e_j)^+ + \mathbf{u}^0 - \mathbf{d}_n)\} \leq 0.
\end{aligned} \tag{4.54}$$

Inequality (4.53) follows because the minimizing action at $\boldsymbol{\Xi}$ is clearly at least as small as action $\mathbf{u}^0$. Inequality (4.54) follows directly from the induction hypothesis. This completes the proof. □

Theorem IV.1 supports the intuition that the more inventory a particular clinic within the cluster has, the better the entire cluster will perform in terms of serving their population. In the next section we can gain additional operational insight via a cluster of two clinics.

4.7.1 Analyzing a Two-Clinic Cluster for Operational Insight

In this section we develop a model for a cluster with two clinics. From this stylized model we develop some insights into the operational management of clinic clusters that form in the transshipment model. The next theorem shows that if demand at each clinic is correlated and symmetric, the more balanced the inventory is between the two clinics, and the better the system performs.

Definition IV.1. We call a function $f : \mathbb{R}^2 \to \mathbb{R}$ *balanced* if given $\boldsymbol{\Xi}$ and $\boldsymbol{\Xi}'$, such that $\xi_1 + \xi_2 = \xi_1' + \xi_2'$, if $|\xi_1 - \xi_2| \leq |\xi_1' - \xi_2'|$ then $f(\boldsymbol{\Xi}) \leq f(\boldsymbol{\Xi}')$.

In the following lemma we show that for symmetric demand distributions, the expected cost to go function of the MDP preserves the balanced property.

Lemma IV.1. *If the demand $\mathbf{d}_n$ at the two clinics is symmetric and the function $f_n(\boldsymbol{\Xi})$ is balanced for all n, then $\mathbb{E}[f_n(\boldsymbol{\Xi} - \mathbf{d}_n)]$ is also balanced.*

Proof. Without loss of generality let $\mathbf{d}_n$ be distributed as $q_{i,j}$ for $i, j = 1, \ldots, n$ where $q_{i,j} = q_{j,i}$ is the probability of observing i units of demand in clinic 1 and j units of demand in clinic 2 and vice versa. Let $\boldsymbol{\Xi}$ and $\boldsymbol{\Xi}'$ be such that $\xi_1 + \xi_2 = \xi_1' + \xi_2'$ and $\Delta\boldsymbol{\Xi} = |\xi_1 - \xi_2| \leq |\xi_1' - \xi_2'| = \Delta\boldsymbol{\Xi}'$. We now show that $\mathbb{E}[f_{n-1}(\boldsymbol{\Xi}' - \mathbf{d}_n)] - \mathbb{E}[f_{n-1}(\boldsymbol{\Xi} -$

$\mathbf{d}_n)] \geq 0$.

$$\begin{aligned}
&\mathbb{E}[f_{n-1}(\boldsymbol{\Xi}' - \mathbf{d}_n)] - \mathbb{E}[f_{n-1}(\boldsymbol{\Xi} - \mathbf{d}_n)] = \\
&\sum_{i=1}^{n}\sum_{j=1}^{n} q_{i,j} f_{n-1}(\boldsymbol{\Xi}' - i\mathbf{e}_1 - j\mathbf{e}_2) - \sum_{i=1}^{n}\sum_{j=1}^{n} q_{i,j} f_{n-1}(\boldsymbol{\Xi} - i\mathbf{e}_1 - j\mathbf{e}_2) \\
&= \sum_{i=1}^{n}\sum_{j=i}^{n} \Big(q_{i,j}[f_{n-1}(\boldsymbol{\Xi}' - i\mathbf{e}_1 - j\mathbf{e}_2) - f_{n-1}(\boldsymbol{\Xi} - i\mathbf{e}_1 - j\mathbf{e}_2)] + \\
&\qquad q_{j,i}[f_{n-1}(\boldsymbol{\Xi}' - j\mathbf{e}_1 - i\mathbf{e}_2) - f(\boldsymbol{\Xi} - j\mathbf{e}_1 - i\mathbf{e}_2)] \Big). \qquad (4.55)
\end{aligned}$$

We now perform a term by term comparison of Equation (4.55). First note that, because of the symmetric demand assumption, $q_{i,j} = q_{j,i}$. If $j - i \leq \Delta\boldsymbol{\Xi}$ then both terms within the sum are positive. Otherwise we have that $f_{n-1}(\xi_1' - i, \xi_2' - j) - f_{n-1}(\xi_1 - i, \xi_2 - j)$ is negative and the $f_{n-1}(\xi_1 - j, \xi_2 - i) - f_{n-1}(\xi_1 - j, \xi_2 - i)$ is positive. What we show is that the magnitude of the negative portion is smaller than the magnitude of the positive portion. To do so we consider two cases:

Case 1: $\Delta\boldsymbol{\Xi} < j - i < \Delta\boldsymbol{\Xi}'$.

The amount of imbalance for each term is:

$$\Delta(\boldsymbol{\Xi}' - i\mathbf{e}_1 - j\mathbf{e}_2) = \xi_2' - j - \xi_1' + i = \Delta\boldsymbol{\Xi}' - (j - i) \qquad (4.56)$$

$$\Delta(\boldsymbol{\Xi} - i\mathbf{e}_1 - j\mathbf{e}_2) = \xi_1 - i - \xi_2 + j = -\Delta\boldsymbol{\Xi} + (j - i) \qquad (4.57)$$

$$\Delta(\boldsymbol{\Xi}' - j\mathbf{e}_1 - i\mathbf{e}_2) = \xi_2' - j - \xi_1' + i = \Delta\boldsymbol{\Xi}' + (j - i) \qquad (4.58)$$

$$\Delta(\boldsymbol{\Xi} - j\mathbf{e}_1 - i\mathbf{e}_2) = \xi_2 - j - \xi_1 + i = \Delta\boldsymbol{\Xi} + (j - i) \qquad (4.59)$$

In Equations (4.56) and (4.57), $-\Delta\boldsymbol{\Xi} + (j - i) \leq \Delta\boldsymbol{\Xi}' - (j - i)$, therefore $f_{n-1}(\xi_1' - i, \xi_2' - j) - f_{n-1}(\xi_1 - i, \xi_2 - j) \geq 0$, so that term of the sum in Equation (4.55) will be positive. If, however, the opposite is true, then the amount of imbalance for the negative term – which directly correlates with the magnitude – is given by subtracting Equation (4.56) from Equation (4.57). In this situation, the state

$(\xi_1'-i, \xi_2'-j)$ actually becomes more balanced than the state (ξ_1-i, ξ_2-j). Therefore the difference in the amount of imbalance of the negative term is given by

$$0 \leq -\Delta\boldsymbol{\Xi} + (j-i) - (\Delta\boldsymbol{\Xi}' - (j-i)) \leq \Delta\boldsymbol{\Xi}' - \Delta\boldsymbol{\Xi}. \tag{4.60}$$

The first inequality holds by the assumption that $-\Delta\Xi + (j-i) \geq (\Delta\Xi' - (j-i))$. Then second inequality holds because we have $\Delta\boldsymbol{\Xi} < j-i < \Delta\boldsymbol{\Xi}'$ and the fact that the difference in imbalance is non-decreasing in $j-i$.

Likewise we know that the difference in imbalance for the positive term, $f(\xi_1' - j, \xi_2' - i) - f(\xi_1 - j, \xi_2 - i)$, is at least as large as the difference in imbalance for the negative term by subtracting Equation (4.59) from Equation (4.58).

$$0 \leq \Delta\boldsymbol{\Xi}' + (j-i) - (\Delta\boldsymbol{\Xi} + (j-i)) = \Delta\boldsymbol{\Xi}' - \Delta\boldsymbol{\Xi}. \tag{4.61}$$

Where the inequality follows from the fact that the $\boldsymbol{\Xi}'$ term is more imbalanced than the $\boldsymbol{\Xi}$ term and the equality follows directly. Clearly the negative term has less difference in imbalance between its components than the positive term, and therefore must be smaller in magnitude.

Case 2: $\Delta\Xi' \leq j-i$.

This case is straightforward, because we now have that for the pair of terms for the negative term, $f_{n-1}(\xi_1' - i, \xi_2' - j) - f(\xi_1 - i, \xi_2 - j)$, both $\xi_2 - j < \xi_1 - i$ and $\xi_2' - j < \xi_1' - i$. Therefore the imbalance for each component is now given by

$$\Delta(\boldsymbol{\Xi}' - i\mathbf{e}_1 - j\mathbf{e}_2) = \xi_2' - j - \xi_1' + i = -\Delta\boldsymbol{\Xi}' + (j-i) \tag{4.62}$$

$$\Delta(\boldsymbol{\Xi} - i\mathbf{e}_1 - j\mathbf{e}_2) = \xi_1 - i - \xi_2 + j = -\Delta\boldsymbol{\Xi} + (j-i) \tag{4.63}$$

$$\Delta(\boldsymbol{\Xi}' - j\mathbf{e}_1 - i\mathbf{e}_2) = \xi_2' - j - \xi_1' + i = \Delta\boldsymbol{\Xi}' + (j-i) \tag{4.64}$$

$$\Delta(\boldsymbol{\Xi} - j\mathbf{e}_1 - i\mathbf{e}_2) = \xi_2 - j - \xi_1 + i = \Delta\boldsymbol{\Xi} + (j-i) \tag{4.65}$$

For the negative term the $\boldsymbol{\Xi}'$ component is still more balanced than the $\boldsymbol{\Xi}'$ component as can be seen from Equations (4.62) and (4.63). So the difference in imbalance between the two terms is given by

$$-\Delta\boldsymbol{\Xi} + (j-i) - (-\Delta\boldsymbol{\Xi}' + (j-i) = \Delta\boldsymbol{\Xi}' - \Delta\boldsymbol{\Xi}. \tag{4.66}$$

For the positive term, the difference in imbalance remains the same:

$$\Delta\boldsymbol{\Xi}' + (j-i) - (\Delta\boldsymbol{\Xi} + (j-i)) = \Delta\boldsymbol{\Xi}' - \Delta\boldsymbol{\Xi}. \tag{4.67}$$

Therefore in Case 2, the difference in imbalance between the components of the negative term and the difference in the imbalance between the components of the positive term are equal and thus the subtraction will be 0. □

Lemma IV.2. *If function $f_n(\boldsymbol{\Xi})$ is balanced for all n, the optimal action will be within the class of policies that transships medication from a clinic with higher inventory to the one with lower inventory.*

Proof. Assuming $\xi_1 \leq \xi_2$, the optimal action will be within the class of policies than transships medication from clinic 2 with the inventory level of ξ_2 to clinic 1 with the inventory level of ξ_1. To prove this, we simply show that the action of "do nothing" will result in less cost than shipping from the clinic with a lower inventory level to the one with a higher stock of medication. If the action is to do nothing, the cost function will be $J_n^0 = \Pi^{\mathrm{T}}(-\boldsymbol{\Xi})^+ + 0 + E\{f_{n-1}((\boldsymbol{\Xi})^+ - \mathbf{d}_n)\}$, otherwise the amount of $\widehat{u}$ medication is moved from clinic 1 to clinic 2 ($\widehat{\mathbf{u}} = (-\widehat{u}, \widehat{u})$), we have $J_n^{\widehat{\mathbf{u}}} = \Pi^{\mathrm{T}}(-\boldsymbol{\Xi})^+ + c\widehat{u} + E\{f_{n-1}((\boldsymbol{\Xi})^+ - \widehat{\mathbf{u}} - \mathbf{d}_n)\}$. Clearly by induction hypothesis,

we have:

$$
\begin{aligned}
&c\widehat{u} \geq 0, \\
&E\{f_{n-1}((\boldsymbol{\Xi} - \widehat{u}\mathbf{e}_1 + \widehat{a}\mathbf{e}_2)^+ - \mathbf{d}_n)\} - E\{f_{n-1}((\boldsymbol{\Xi})^+ - \mathbf{d}_n)\} \geq 0, \quad \Longrightarrow \\
&f_n^{\widehat{u}}(\boldsymbol{\Xi}) - f_n^0(\boldsymbol{\Xi}) \geq 0.
\end{aligned}
$$

This is simply because $\xi_2 - \xi_1 \leq \xi_2 + \widehat{u} - \xi_1 + \widehat{u} = \xi_2 - \xi_1 + 2\widehat{u}$. This result will help us reduce the action space significantly. As a result the optimal action u^* can only be an element of $\{0, ..., \max\{\xi_1, \xi_2\} - \min\{\xi_1, \xi_2\}\}$. u^* will be the amount of medication shipped from the clinic with higher inventory to the one with lower inventory. □

The result of this lemma will be employed to further prove the validity of Theorem IV.2. The next lemma demonstrates that a balanced inventory distribution will have the lowest cost.

Lemma IV.3. *If $f_n(\boldsymbol{\Xi})$ is a balanced function, for any total inventory level $\xi_1 + \xi_2$, the value function is minimized where $\xi_1^* = \xi_2^*$.*

Proof. This is the direct result of $f_n(\boldsymbol{\Xi})$ being a balanced function. We employ induction to show this property.

Base Case:

Since $f_0(\boldsymbol{\Xi}^*) = f_0(\boldsymbol{\Xi}) = 0$ for all $\boldsymbol{\Xi}$, the hypothesis holds.

Induction Step:

We assume that the induction hypothesis holds for stage $n-1$. Now we show that it holds for stage n. Consider $\boldsymbol{\Xi}^*$ where $\xi_1^* = \xi_2^*$ versus $\boldsymbol{\Xi}$ where $\xi_1 \neq \xi_2$.

$$
\begin{aligned}
f_n(\boldsymbol{\Xi}) &= \Pi^{\mathrm{T}}(-\boldsymbol{\Xi})^+ + \min_{u \in \mathcal{U}_{\boldsymbol{\Xi}}} \{c \sum_{j \in \Phi} (u_j)^+ + \mathbb{E}\{f_{n-1}((\boldsymbol{\Xi})^+ + \mathbf{u} - \mathbf{d}_n)\}\}, \\
f_n(\boldsymbol{\Xi}^*) &= \Pi^{\mathrm{T}}(-\boldsymbol{\Xi}^*)^+ + \min_{u \in \mathcal{U}_{\boldsymbol{\Xi}^*}} \{c \sum_{j \in \Phi} (u_j)^+ + \mathbb{E}\{f_{n-1}((\boldsymbol{\Xi}^*)^+ + \mathbf{u} - \mathbf{d}_n)\}\}.
\end{aligned}
$$

By Lemma IV.1 we know that $f_n(\boldsymbol{\Xi})$ is balanced, thus $E\{f_{n-1}((\boldsymbol{\Xi})^+ + \mathbf{u} - \mathbf{d}_n)\}$ is also balanced. Therefore the optimal action at $\boldsymbol{\Xi}^*$ is $\mathbf{u}^* = \mathbf{0}$, which achieves the lowest possible value for the expectation. Thus we have:

$$E\{f_{n-1}((\boldsymbol{\Xi}^*)^+ - \mathbf{d}_n)\} \leq \mathbb{E}\{f_{n-1}((\boldsymbol{\Xi})^+ + \mathbf{u} - \mathbf{d}_n)\}, \tag{4.68}$$

and because the optimal policy at $\boldsymbol{\Xi}^*$ has no penalty because $\mathbf{u}^* = \mathbf{0}$ it is clear that

$$\begin{aligned} &\min_{u \in \mathcal{U}_{\Xi^*}} \{c \sum_{j \in \Phi} (u_j)^+ + \mathbb{E}\{f_{n-1}((\boldsymbol{\Xi}^*)^+ + \mathbf{u} - \mathbf{d}_n)\}\} \leq \\ &\min_{u \in \mathcal{U}_{\Xi}} \{c \sum_{j \in \Phi} (u_j)^+ + \mathbb{E}\{f_{n-1}((\boldsymbol{\Xi})^+ + \mathbf{u} - \mathbf{d}_n)\}\}. \end{aligned} \tag{4.69}$$

Finally, it can quickly be verified that the instantaneous cost is lower for the balanced inventory, $\boldsymbol{\Xi}^*$:

$$\Pi^{\mathrm{T}}(-\boldsymbol{\Xi}^*)^+ \leq \Pi^{\mathrm{T}}(-\boldsymbol{\Xi})^+. \tag{4.70}$$

Thus we have shown that $f_n(\boldsymbol{\Xi}^*) \leq f_n(\boldsymbol{\Xi})$. □

Theorem IV.2. *When the demand is symmetric, the value function in Equation (4.51) is balanced.*

Proof. Without loss of generality, we assume $\xi_1 \leq \xi_2$ and $\xi_1' \leq \xi_2'$. This ordering along with our assumption that $\xi_1 + \xi_2 = \xi_1' + \xi_2'$ and $\Delta\boldsymbol{\Xi} = |\xi_1 - \xi_2| \leq |\xi_1' - \xi_2'| = \Delta\boldsymbol{\Xi}'$, implies that $\xi_1' \leq \xi_1 \leq \xi_2 \leq \xi_2'$. We proceed to prove this theorem by using induction.

Base Case:

Since $f_0(\boldsymbol{\Xi}) = 0$ for all $\boldsymbol{\Xi}$. As a result, the induction hypothesis holds.

Induction Step:

We assume that the induction hypothesis holds for stage $n-1$. In order to show

$f_n(\mathbf{\Xi})$ is less than or equal to $f_n(\mathbf{\Xi}')$, we first write the expressions for both cases:

$$f_n(\mathbf{\Xi}) = \Pi^{\mathrm{T}}(-\mathbf{\Xi})^+ + \min_{u\in\mathcal{U}_{\Xi}}\{c\sum_{j\in\Phi}(u_j)^+ + \mathbb{E}\{f_{n-1}((\mathbf{\Xi})^+ + \mathbf{u} - \mathbf{d}_n)\}\},$$

$$f_n(\mathbf{\Xi}') = \Pi^{\mathrm{T}}(-\mathbf{\Xi}')^+ + \min_{u\in\mathcal{U}_{\Xi'}}\{c\sum_{j\in\Phi}(u_j)^+ + \mathbb{E}\{f_{n-1}((\mathbf{\Xi}')^+ + \mathbf{u} - \mathbf{d}_n)\}\}.$$

Let $\bar{\xi} = \bar{\xi}' = \frac{\xi_1+\xi_2}{2} = \frac{\xi_1'+\xi_2}{2}$. At stage $n-1$, the state $[\bar{\xi},\bar{\xi}] = [\bar{\xi}',\bar{\xi}']$ achieves the minimum value for the expectation of the function $f_{n-1}(\mathbf{\Xi})$ (direct result of Lemma IV.3).

By the induction hypothesis we have $f_{n-1}(\mathbf{\Xi}') \geq f_{n-1}(\mathbf{\Xi})$. Since $\xi_1 + \xi_2 = \xi_1' + \xi_2'$, we can conjuncture that $f_{n-1}(\bar{\mathbf{\Xi}})$ is the state which reaches the minimum possible cost.

$$\begin{aligned} f_n(\mathbf{\Xi}') - f_n(\mathbf{\Xi}) =& \Pi^{\mathrm{T}}(-\mathbf{\Xi}')^+ + \min_{\mathbf{u}\in\mathcal{U}_{\Xi'}}\{c\sum_{j\in\Phi}(u_j)^+ + \mathbb{E}\{f_{n-1}((\mathbf{\Xi}')^+ + \mathbf{u} - \mathbf{d}_n)\}\} - \\ & \Pi^{\mathrm{T}}(-\mathbf{\Xi})^+ - \min_{\mathbf{u}\in\mathcal{U}_{\Xi}}\{c\sum_{j\in\Phi}(u_j)^+ + E\{f_{n-1}((\mathbf{\Xi})^+ + \mathbf{u} - \mathbf{d}_n)\}\}. \end{aligned}$$

First we compare the instant cost associated with the shortage penaltyies. As we mentioned before, without loss of generality, we consider the cases where $\xi_1 \leq \xi_2$ and $\xi_1' \leq \xi_2'$. Other cases can be investigated similarly. There are the following cases:

1. $\xi_1', \xi_2', \xi_1, \xi_2 \geq 0$: In this case both $f_n(\mathbf{\Xi})$ and $f_n(\mathbf{\Xi}')$ incur zero shortage penalties. As a result $\Pi^{\mathrm{T}}(-\mathbf{\Xi}')^+ - \Pi^{\mathrm{T}}(-\mathbf{\Xi})^+ = 0$.

2. $\xi_1' \leq 0$ and $\xi_2, \xi_1, \xi_2' \geq 0$: In this case $f_n(\mathbf{\Xi}')$ has a positive shortage cost while $f_n(\mathbf{\Xi})$ incurs zero shortage penalty. Therefore $\Pi^{\mathrm{T}}(-\mathbf{\Xi}')^+ - \Pi^{\mathrm{T}}(-\mathbf{\Xi})^+ \geq 0$.

3. $\xi_1', \xi_1 \leq 0$ and $\xi_2, \xi_2' \geq 0$: In this case $f_n(\mathbf{\Xi}')$ and $f_n(\mathbf{\Xi})$ both have positive shortage cost but since $\xi_1' \leq \xi_1$, $f_n(\mathbf{\Xi})$ has a greater shortage penalty. As a result $\Pi^{\mathrm{T}}(-\mathbf{\Xi}')^+ - \Pi^{\mathrm{T}}(-\mathbf{\Xi})^+ \geq 0$.

4. $\xi_1', \xi_1, \xi_2 \leq 0$ and $\xi_2' \geq 0$: We conclude that having $\xi_2' \geq 0$, will result in $\xi_1' \leq \xi_1 + \xi_2$ as a direct result of the assumption, since $\xi_1 + \xi_2 = \xi_1' + \xi_2'$. Therefore $\Pi^{\mathrm{T}}(-\boldsymbol{\Xi}')^+ - \Pi^{\mathrm{T}}(-\boldsymbol{\Xi})^+ \geq 0$.

5. $\xi_1', \xi_1, \xi_2, \xi_2' \leq 0$: this implies that $\Pi^{\mathrm{T}}(-\boldsymbol{\Xi}')^+ = \Pi^{\mathrm{T}}(-\boldsymbol{\Xi})^+ \geq 0$.

The next step is to investigate the possible actions and compare the cost to go terms for both $f_n(\boldsymbol{\Xi})$ and $f_n(\boldsymbol{\Xi}')$. Let assume the optimal action in state $\boldsymbol{\Xi}'$ is $u^{'*}$. Having $\xi_1' \leq \xi_2'$ and based on the result of Lemma IV.2, $u^{'*} \in= \{0, ..., u_{\bar{\xi}'}\} = \mathcal{U}_{\Xi'}$ and $\mathbf{u}^{'*} = (u^{'*}, -u^{'*})$, the optimal action either will be to ship from clinic 2 to clinic 1 or do nothing (result of Lemma IV.2). We also know that $\xi_2' - \xi_1' \geq \xi_2 - \xi_1$, as a result, the optimal action u^* of the state $\boldsymbol{\Xi}'$ will be a member of $\{0, ..., u_{\bar{\xi}}\}$. We have:

$$\mathcal{U}_{\Xi} \subset \mathcal{U}_{\Xi'}$$

There are two possible scenarios, either $\mathbf{u}^{'*} \in \mathcal{U}_{\Xi}$ or $\mathbf{u}^{'*} \in \mathcal{U}_{\Xi'} \setminus \mathcal{U}_{\Xi'}$.

1. Scenario 1: $u^{'*} \in \mathcal{U}_{\Xi}$:

$$f_n(\boldsymbol{\Xi}') - f_n(\boldsymbol{\Xi}) \geq \overbrace{\Pi^{\mathrm{T}}(-\boldsymbol{\Xi}')^+ - \Pi^{\mathrm{T}}(-\boldsymbol{\Xi})^+}^{=Q^1 \geq 0} +$$

$$cu^{'*} + \mathbb{E}\{f_{n-1}((\boldsymbol{\Xi}')^+ + \mathbf{u}^{'*} - \mathbf{d}_n)\} -$$

$$\min_{\mathbf{u} \in \mathcal{U}_{\Xi}} \{c \sum_{j \in \Phi} (u_j)^+ + \mathbb{E}\{f_{n-1}((\boldsymbol{\Xi})^+ + \mathbf{u} - \mathbf{d}_n)\}\} \geq$$

$$Q^1 + cu^{'*} + \mathbb{E}\{f_{n-1}((\boldsymbol{\Xi}')^+ + \mathbf{u}^{'*} - \mathbf{d}_n)\} - cu^{'*} +$$

$$\mathbb{E}\{f_{n-1}((\boldsymbol{\Xi})^+ + \mathbf{u}^{'*} - \mathbf{d}_n)\} = \mathbb{E}\{f_{n-1}((\boldsymbol{\Xi}')^+ + \mathbf{u}^{'*} -$$

$$\mathbf{d}_n)\} - \mathbb{E}\{f_{n-1}((\boldsymbol{\Xi})^+ + \mathbf{u}^{'*} - \mathbf{d}_n)\} \geq 0. \quad \text{by induction hypothesis}$$

2. Scenario 2: $\mathbf{u}^{'*} \in \mathcal{U}_{\Xi'} \setminus \mathcal{U}_{\Xi}$:

$$f_n(\boldsymbol{\Xi}') - f_n(\boldsymbol{\Xi}) \geq \overbrace{\Pi^{\mathrm{T}}(-\boldsymbol{\Xi}')^+ - \Pi^{\mathrm{T}}(-\boldsymbol{\Xi})^+}^{Q^2 \geq 0} + cu^{'*} +$$

$$E\{f_{n-1}((\boldsymbol{\Xi}')^+ + \mathbf{u}^{'*} - \mathbf{d}_n)\} - \min_{\mathbf{u} \in \mathcal{U}_{\Xi}} \{c \sum_{j \in \Phi} (u_j)^+ + \mathbb{E}\{f_{n-1}((\boldsymbol{\Xi})^+ + \mathbf{u} - \mathbf{d}_n)\}\} \geq$$

$$Q^2 + \overbrace{cu^{'*} - cu_{\bar{\xi}}}^{B \geq 0} + \overbrace{E\{f_{n-1}((\boldsymbol{\Xi}')^+ + \mathbf{u}^*_{\xi'} - \mathbf{d}_n)\} - \mathbb{E}\{f_{n-1}((\boldsymbol{\Xi})^+ + \mathbf{u}_{\bar{\xi}} - \mathbf{d}_n)\}}^{C \geq 0} \geq 0.$$

by induction hypothesis

We should note that $B \geq 0$ since the total units shipped from clinic 2 to clinic 1 in scenario 2 is more than $u_{\bar{\xi}}$. Also after transshipping $u_{\bar{\xi}}$, $f_{n-1}(\boldsymbol{\Xi})$ is reaching its minimum (as a direct result of Lemma IV.3), therefore $C \geq 0$. □

As a result of Lemmas IV.2, IV.3, and Theorem IV.2, the following corollaries IV.1 and IV.2 hold.

Corollary IV.1. *Under a fixed shipping cost, the optimal policy is of threshold nature, where* $\mathbf{u} = \mathbf{0}$ *if* $c > \mathbb{E}\{f_{n-1}((\boldsymbol{\Xi})^+ - \mathbf{d}_n)\}$*; otherwise* $\mathbf{u}$ *is the action that balances the inventory.*

Corollary IV.2. *Under a linear shipping cost, the optimal policy is of threshold nature with multiple stage dependent thresholds. That is, depending on the shipping cost, each threshold will move the inventory closer into balance or does nothing.*

Any analytical proof regarding the structure of an optimal policy for clusters consisting of more than two clinics can be complex. Also, relaxing the demand assumptions can increase the complexity of characterizing the optimal actions.

4.7.2 Illustrative Numerical Example for a Two-Clinic Cluster

In this section, a numerical example is used to gain insight into state-specific optimal actions. For purposes of exposition, we begin with an example of a cluster

consisting of two clinics. We also scale the demand and supply to obtain the following restricted state space:

$$\Xi = \Big\{(\xi_1, \xi_2) \in \mathbb{R}^2 : -5 \leq \xi_i \leq 9, \forall i \in \{1, 2\} \text{ and } \xi_1 + \xi_2 \leq 9\Big\}.$$

We solve the two-dimensional MDP under three different parameter settings where the ratio of shortage penalty to unit transportation cost was either: (1) low , (2) moderate, or (3) high and solve it for six stages. Figure 4.8 illustrates the optimal actions at stage 5 (i.e. $n-1$) for each state for cases (1), (2) and (3).

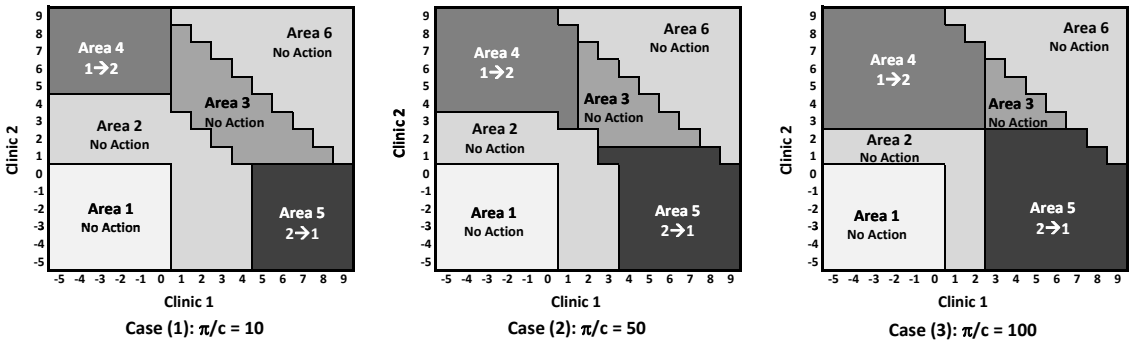


Figure 4.8: Optimal actions in period 5 for three parameter settings.

The optimal actions in Figure 4.8 are identified by six areas, 1 through 6. Details of the optimal actions in each area are described in Table 4.5.

Area	Description	Actions
1	Clinics 1 and 2 face shortage	No action is possible
2	Clinics have low inventories	No action is recommended
3	Both clinics have high surplus	No action is required
4	Clinic 1 is facing shortage while clinic 2 has surplus	Clinic 2 transships to clinic 1
5	Clinic 2 is facing shortage while clinic 1 has surplus	Clinic 1 transships to clinic 2
6	Infeasible state	No action is defined

Table 4.5: Six areas in the two-dimensional illustrative example.

As illustrated in Figure 4.8, the ratio of shortage penalty to transportation cost (π/c) plays an important role in the structure of optimal policy. As this ratio increases, there is more transshipment between clinics. As the π/c ratio decreases we observe less transshipment. Notice that areas 5 and 6 become larger as π/c increases.

At the same time, areas 2 and 3 shrink. Note that in all three cases the size of areas 1 and 6 stays constant. This is because there is no action in area 1 due to the lack of inventory. Area 6 represents infeasible states and thus no action is defined in that area.

4.7.3 Periodic Transshipment for Generalized Cluster Sizes

In Section 4.7.1 we develop an optimal transshipment policy within clusters consisting of two clinics when the demands in each period across all the clinics in the cluster are symmetric. To gain insight into the structure of the optimal transshipment policy in a generalized case with more than two clinics in a cluster, parameterized with historical demand, we solved the MDP formulation described by (4.51) and (4.52) numerically for clusters of size three. We constructed the demand data for each cluster based on the actual bi-monthly demand (disguised for reasons of confidentiality) from a five-year period between 2003 to 2008. Here we summarize the insights we gained through numerical experiments on a representative subset of clusters:

Insight IV.1. *When the demand across all the clinics in a cluster are identically distributed, the optimal action is to balance the inventory between the clinics in the cluster. The finding of Corollary IV.1 extends to clusters of size greater than two.*

Insight IV.2. *As the ratio of the shortage penalty to the transshipment cost increases, the optimal action tends to ship more units between the clinics.*

In Lemma IV.3 we have proven the validity of Insight IV.1 for the case of clusters consisting of two clusters. Generalizing Lemma IV.3 to larger clusters, however, requires further analysis. The above insights are further argued through an illustrative numerical example in Section 4.7.4.

4.7.4 Illustrative Numerical Example for a Three-Clinic Cluster

To illustrate the insights described in Section 4.7.3, we consider a cluster consisting of three clinics. Similar to the previous example, we scale the demand and supply to obtain the following restricted state space:

$$\Xi = \Big\{(\xi_1, \xi_2, \xi_3) \in \mathbb{R}^3 : -5 \leq \xi_i \leq 9, \forall i \in \{1, 2, 3\} \text{ and } \xi_1 + \xi_2 \leq 9\Big\}.$$

Note that in this case, the state of the system has three dimensions. Therefore illustrating the optimal actions for the entire state-space can be challenging. Thus we only illustrate the optimal actions for three inventory levels at clinic 3: 0, 2, and 3. For ease of comparison, we chose a moderate ratio of shortage penalty to transportation cost, i.e. $\pi/c = 10$.

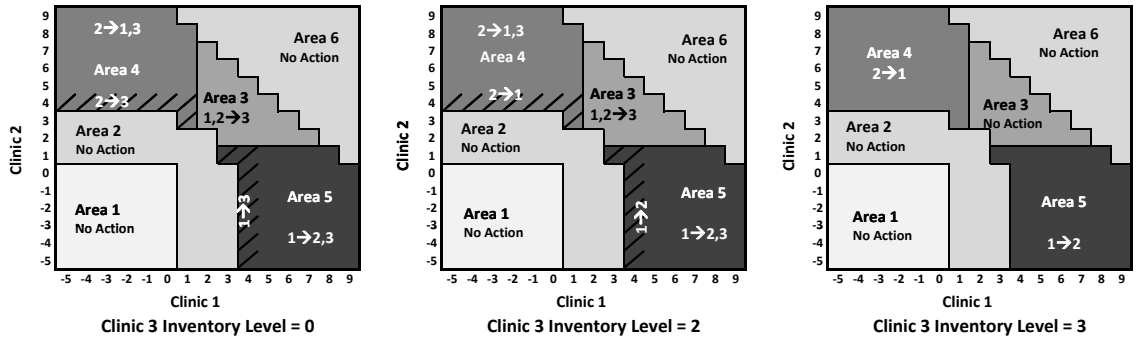


Figure 4.9: Optimal actions in period 5 for a cluster consisting of three clinics.

The optimal actions in Figure 4.9 are identified by six areas, 1 through 6. Details of the optimal actions in each area are described in Table 4.6.

Area	Description	Actions
1	Clinics 1 & 2 face shortage	No action is possible if clinic 3 does not have surplus
2	Clinics 1 & 2 have low inventories	No action is recommended if clinic 3 does not have surplus
3	Clinics 1 & 2 have high surplus	1 & 2 transship to 3 if needed
4	Clinic 2 has surplus	Clinic 2 transships to clinics 1 and/or 3 if needed
5	Clinic 1 has surplus	Clinic 1 transships to clinics 1 and/or 3 if needed
6	Infeasible state	No action is defined

Table 4.6: Six areas in the three-dimensional illustrative example.

As illustrated in Figure 4.9, in the presence of symmetric demand, the optimal policy balances the inventory between the three clinics. As the inventory level of clinic 3 increases, that clinic ships more pharmaceutical units to clinics 1 and 2. This observation is also valid for the amount of medication shipped by clinics 1 and 2.

4.8 Conclusion and Future Research

This chapter has addressed the challenging problem of distributing malaria treatments through centralized health systems in developing world. This problem is different from problems of traditional pharmaceutical in many ways: transportation infrastructure in countries such as Malawi is under-developed, demand is spatially and temporally uncertain and financial resources to address an incredible disease burden are limited and subject to donor interest and motivation. In addition, forecasting demand and finding exact parameter values (such that of shortage penalty) may not be feasible, given the non-commercial nature of health operations.

The main contribution of this chapter has been to develop an analytical approach to solving the problem of efficiently delivering malaria drugs, both at a strategic level (where the planning horizon spans through the malaria season) and an operational (periodic) level. Further, we do so in a manner that integrates the strategic with the operational planning, using the policy structure and numerical results to decompose the national problem into clinic clusters. This enabled a tractable solution to the periodic review operational MDP.

To illustrate the tractability and effectiveness of our proposed stochastic models, we performed a set of computational experiments using 5 years of data from Malawi. Our initial results show that both the two-stage and the three-stage models can effec-

tively reduce the expected shortage at least by 16%, while distributing the available malaria treatments efficiently through the supply network.

As discussed earlier, one major challenge in this domain is limited supply. To show the sensitivity of the solution to the available supply value, we illustrated the results under three different supply assumptions: low, medium, and high. Based on our preliminary results, the transshipment model often outperforms the delayed shipment model by having a lower transportation cost (only 4% less). However, due to its decentralized nature as well as the politics of asking an individual clinic to relinquish some of its inventory, transshipment would likely be more difficult to implement.

The structure of the optimal solution to the strategic planning model exhibited a clustering effect, with groups of clinics clustering together to transship. We used this structure to decompose the full operational problem into tractable sub-problems by solving the periodic review operational MDP problem for each cluster independently. We also gained key insights into the periodic review model by analyzing a stylized model. Specifically, we were able to show that the structure of the optimal policy was of threshold form where the clinic with higher inventory levels will ship to the clinic with lower inventory levels or do nothing. These insights were then extended to a model that was fully parameterized by historical demand and the true solution of the strategic level model by numerically solving the MDP for a number of representative cases.

An interesting future extension to consider is uncertainty in the supply of medications, since much of the resources for medications are donated. Another realistic extension is to consider a budget constraint. This way, the objective is to minimize shortage (perhaps with an equity measure) subject to the limited funds to be spent

on transportation.

In conclusion, the result of our integrated strategic and operational models is a workable decision support system approach that can guide government policy in driving better health outcomes at a lower cost. In a country such as Malawi, where malaria is widespread and budgets are severely constrained, such a system can have a significant impact on the society.

CHAPTER V

Priority-Based Routing with Strict Deadlines and Server Flexibility under Uncertainty

5.1 Introduction

Recently, the ability to deliver information technology (IT) services from multiple locations has given rise to an entirely new IT service delivery model. In this model, organizations outsource components of their IT infrastructure operations to one or more service providers, who in turn use a combination of onsite and offsite resources to manage the components on behalf of their clients. To provide these capabilities, service providers have pioneered a new onsite-offsite delivery model called the global delivery model (GDM), in which a service provider uses a number of delivery centers around the globe to provide services to its clients. The agents at these delivery centers perform a variety of tasks including remote monitoring and management of hardware and software, developing new applications, testing configurations, applying security patches, etc. While call centers are often the primary interface between clients and service providers, the delivery centers perform a variety of tasks at different levels of service complexity behind the scenes. Using this model, the clients of the GDM can scale their core business operations to match demand without worrying about resources and skills required to manage their IT infrastructure. At the same time, by leveraging local skills, cost structure and process standardization, service providers

can ensure the level of quality of the services performed from different locations.

The critical factor for a service provider to achieve a high service quality in the GDM is efficient utilization of the agents and resources available at its delivery centers. The process by which a service provider assigns a service re-quest to an agent at a service delivery center is known as "dispatching." The nature of the IT service centers presents a number of challenges to efficient dispatching, primarily due to variability in agent skills and complexity of service requests. It is difficult to incorporate agent skill variability and service request complexity in the dispatching procedure for several reasons. First, there have been very few studies on quantifying agent skill variability in performing IT infrastructure services. The experience level and skill sets of an agent strongly affect diagnosis and resolution of IT service requests. Therefore it is crucial that service requests are dispatched to the most suitable agents. Second, IT service requests exhibit a wide range of complexity and require varying levels of effort and coordination by the agents.

Although the well-studied call center staffing problem has some similarity with the service-dispatching problem, there are many important differences. In a service request fulfillment system, there is no abandonment, whereas in a call center environment, requests may leave the system even before their service begins. Furthermore, IT service centers typically serve requests with different levels of priority (or severity), whereas call centers have a more homogenous demand structure.

The main contributions of our study are: (1) a model to address variability in service time and skill level for each agent for each specified request type, (2) a novel dispatching policy that considers the complexity of each service request in addition to the variability in agent service times and skill levels, and (3) investigation of the performance of this proposed approach using data from an IT service center.

5.2 Literature Review

Determining the number of agents and their skill requirements is a well studied problem in the literature. As demonstrated by Bartholdi [15], even in the deterministic case where the demand and supply are fixed, finding the number of agents required for each time period can be a complex task. In real world scenarios, one faces even more complexities due to the stochastic nature of the demand. Another dimension of complexity is taking into account an agent's skill level in environments where different skills and levels of skills are necessary to serve an incoming request. Van Oyen et al. [86] address how cross training can be aligned with organizational strategies. Their model includes improving performance measures of the system but does not consider fixed deadlines for jobs in the system. Brusco and Johns [22] use integer programming to show that cross training can be a very useful approach when agent skills can be combined in a sequential setting.

Perhaps the most relevant research framework to our problem can be found in the call center literature. Similar to call centers, IT service centers require agents with a variety of skills to handle the arriving requests. However, it is almost impossible to expect that every agent is fully cross-trained for every task. Therefore it is important to route problems to the best agents considering their skills and skill levels.

Investigation of different algorithms and methodologies for routing traffic has been done with the purpose of system improvement. Many researchers considered this concept to improve the Quality of Service (QoS). Ma and Steenkiste [64] propose a model in order to decrease delay time of calls for a call center as a measure of QoS by finding a feasible path.

Feldman et al. [33] and Whitt [90] study the problem of call center staffing in

a complex structure. They investigated the effect of time-varying demand on the performance measures of the system. They also studied variability of the service time.

Finding the most appropriate agent training policy is another important factor which can affect both the stability and the performance of the system. It is important to note that training all agents for all skills is typically too expensive. A suitable algorithm can approximate the number of agents with the required set of skills. Wallace and Whitt [88] introduce an algorithm to route problems based on the skill level of the agent. Other algorithms were developed by Sisselman and Whitt [81], who introduced the concepts of "value-based routing" and "preference-based routing" to the existing skill-based routing algorithm. All of these algorithms are based on simulation and heuristic approaches but none addressed having different priority of requests and strict (or hard) deadlines, both of which are incorporated into our proposed approach.

5.3 System Description

The dispatching system consists of several agents who are assigned to different service request resolution groups. Each resolution group is capable of handling several different types of service requests. Each service request can be characterized by a request type and a priority level. We assume that each service request with a specified type and priority level has a lump sum penalty cost associated with violating its deadline according to a service level agreement (SLA) contracted with the customer. The inter-arrival time distributions are independent (but not necessarily identical) for each request type and priority level. The time required to resolve a request type can vary by agent.

The goal of an efficient dispatching policy is to minimize the total penalty cost of violating deadlines. The differences between a traditional call center problem and the IT service center problem motivate us to study this problem. Here we mention some of these differences. First, in a typical call center, the SLA is solely a function of the waiting time for the customer before the service starts (i.e. waiting time in the queue), whereas in an IT environment, the SLA is concerned with the total time a request spends in the system until the request is resolved. This, in turn, adds another source of uncertainty to the dispatching problem. Second, in a call center, a customer may leave the system before the service starts. However, in a typical IT application, a customer would never leave the system before the request is resolved. Third, the nature of operations in IT environments requires a more sophisticated set of skills for agents, whereas agents in a call center generally have a more limited set of skills. In fact, requests that cannot be resolved at the call center are often passed on to the IT service centers. Fourth, IT service requests may be handled by multiple agents simultaneously depending on their complexity, whereas in a call center, each agent typically handles a single call in its entirety or possibly hands off the call in a sequential fashion.

In this research, we study agent-level variability and complexity of the service request to build a dispatching model for IT infrastructure service requests in a single-stage service delivery center. Figure 1 presents an example of dispatching systems commonly found in many service delivery centers. We discuss factors critical to developing an efficient dispatching policy. These factors include an agent's skill level, service time variability, and the penalty cost associated with violating the deadline.

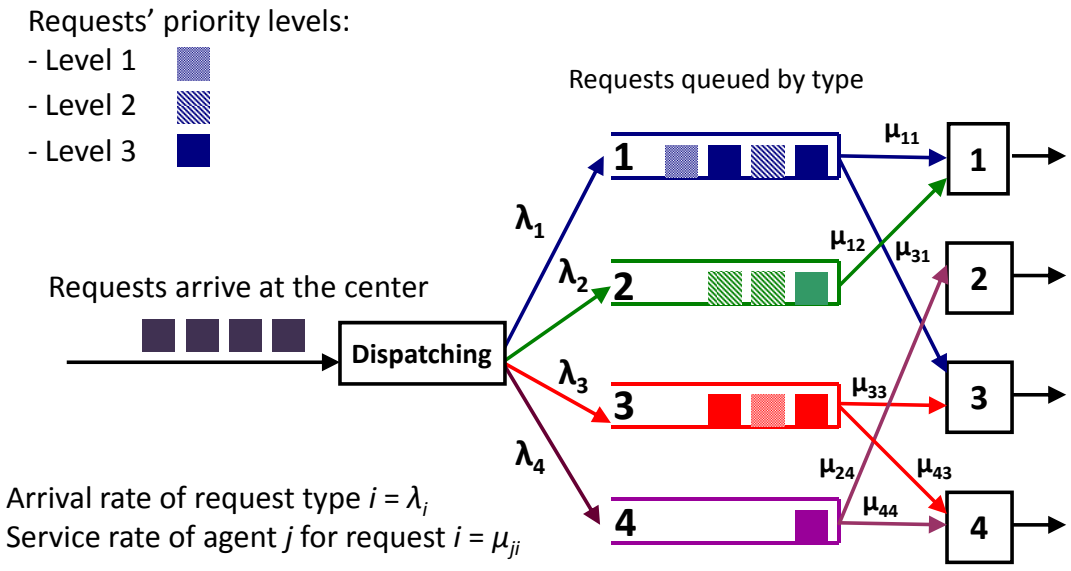


Figure 5.1: The service request dispatching and resolution process.

5.4 Input model

In a typical IT service delivery environment, there is considerable variability among the agents in resolving similar requests. This variability stems from experience level (e.g. number of years), certifications, subject-matter expertise, specialized training, and other factors. As a result, some agents are able to diagnose the root causes of a problem faster and more accurately than others, resulting in a shorter mean service restoration time and a smaller standard deviation. There is also a temporal variation in performance of an agent in performing the same task when monitored over time. This can be attributed to an inconsistent performance level and is generally complex to model. In recent years, manufacturing processes have been automated to the point that only random sources of variation remain, resulting in mostly normal distributions of the process outcomes. However, many IT service management processes such as maintaining servers, patch management, and installation have a high degree of manual involvement. This results in a high agent-induced

variability in the process.

Variability has many sources. However, our input analysis shows that the two most important factors that contribute to variability in service resolution time are the complexity level of service requests and their priority level. The complexity level of service requests is illustrated in Figure 5.2.

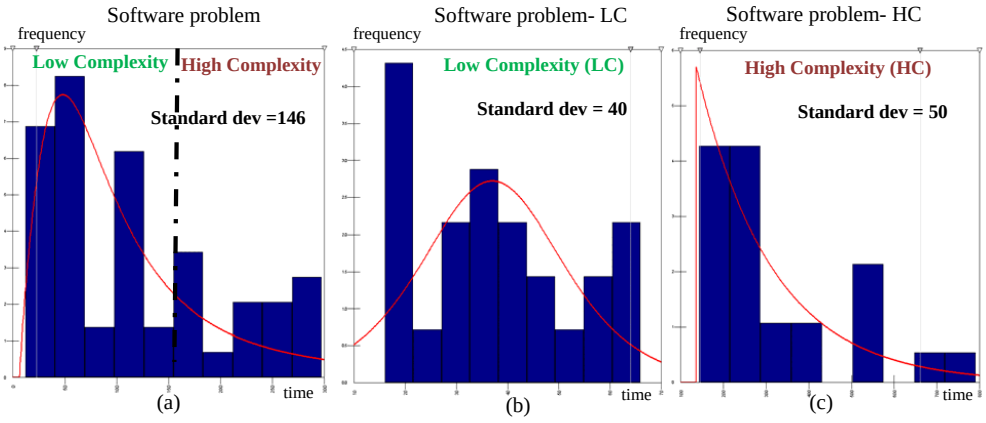


Figure 5.2: Task complexity in defining a request type can decrease variability

Figure 5.2(a) shows a histogram of service time associated with a sampled agent, serving requests categorized as "software problems." Based on this high-level categorization, the service time has a standard deviation of 146. We then refine the categorization further by introducing a complexity level. This way, the "software problem" is broken into two categories: "low-complexity software problems (LC)" and "high-complexity software problems (HC)." Examples of such problems are restarting an application (LC) versus installing a new application middleware on a server (HC). As shown in Figure 5.2(b) and Figure 5.2(c), adding a dimension of complexity dramatically reduces the standard deviation of service time. As a result of this analysis, we redefine the request categorization across all request types in the system to account for the complexity level of service requests. We fit a probability distribution to the service time for each request type, at each complexity level, and for each agent. We

note that mapping an incoming service request to its complexity level can be done either manually by a human dispatcher or automatically by a classification agent who analyzes the text description of the request and other structured attributes before assigning a complexity class to the request. We describe the key aspects of our simulation model next.

5.5 Simulation Model

5.5.1 Priority-Basel Allocation

In order to heuristically address the dispatching model described above, we develop a policy to assign each service request to the appropriate agent, based on an allocation index. The priority-based allocation index associated with service request k with type i and priority s, if allocating agent j, where the remaining time to deadline is t_d, is defined in 5.1:

$$I^k_{jis}(\theta, t_d) = \frac{m_{ji}(\theta, t_d)\pi_{is}}{t^{\alpha}_{ji}} \tag{5.1}$$

where:

π_{is}	Penalty cost of violating the deadline associated with service request i with priority s
t^{α}_{ji}	Time required by agent j to serve service request type i with a confidence level of $1-\alpha$
t_d	Remaining time to deadline
$m_{ji}(\theta, t_d)$	Step function associated with agent j and request type i to be defined below
θ	Positive coefficient used to tune the algorithm to the application

Table 5.1: Notation.

As soon as an agent finishes serving a request (i.e. the system is non-preemptive), we update all the indices for all the requests and the request with the highest index will be assigned to the idle agent. In order to develop this index, we calculate the average service rate at which agent j can solve the type associated with the service request i (μ_{ji}) from historical data. However, it is important to note that the average service times do not capture the inherent variability of the service time. In the case of

IT applications, this is even more important due to non-normal distributions that are a consequence of manually-performed steps, as explained by Pyzde [73]. Therefore, we need to include another measure that better represents the service time variability. An appropriate measure of variability can be defined based on the confidence level at which an agent can serve a request before its deadline. Let t_{ji}^{α} denote the time it takes for agent j to serve request type i with probability $1-\alpha$ (see Equation (5.2)). Based on this equation, the values of t_{ji}^{α} are calculated a priori.

$$P_{ji}(T \leq t_{ji}^{\alpha}) = 1-\alpha \Longrightarrow t_{ji}^{\alpha} = F_{ji}^{-1}(1-\alpha) \tag{5.2}$$

We also define a step function ($m_{ji}(\theta, t_d)$) which assigns weight to each request based on the remaining time to deadline (t_d). This function takes the value of one when the time to deadline is relatively large. As a request gets close to its deadline, this function takes the value of M, where M is a sufficiently large number. This gives a high priority to the service requests that are critically close to their deadline. Specifically, for a service request that is reaching its deadline in less than $\frac{1}{\mu_{ji}}$, this function returns a value of M. Finally, for requests that already missed their deadlines, the value of this function will be zero (see Equation (5.3)).

$$x_p = \begin{cases} 1 & \text{if } t_d > \frac{\theta}{\mu_{ji}} \\ M & \text{if } 0 < t_d \leq \frac{\theta}{\mu_{ji}} \\ 0 & \text{if } t_d = 0 \end{cases} \tag{5.3}$$

The priority-based allocation index is calculated by multiplying function $m_{ji}(\theta, t_d)$ by the penalty cost (π_{is}) of violating the deadline and dividing it by t_{ji}^{α}. According to this index, service requests with a high penalty cost and an imminent deadline receive the highest priority while requests with a passed deadline are given the lowest priority. We note that there may be service requests for which the penalty function may be

duration-based rather than deadline-based, e.g. how long a web server remains down after an outage. This type of penalty may be associated with the most critical or revenue-generating components of IT infrastructure (e.g., a web server that processes payments). The present simulation model, however, does not address these types of penalty cost functions.

5.5.2 Simulation Results

In our initial testing, we consider a system consisting of four agents, four problem (service request) types as a way to incorporate their complexity, and three levels of priority. We assume that agents have different skill levels. Table 5.2 illustrates the mean service time of each agent for each service request type. The penalty of missing the deadline for priorities 1, 2, and 3 are 100, 80 and 30 respectively. The deadlines for priorities 1, 2 and 3 are 40, 60 and 85 minutes respectively. Our performance criterion is to maximize the long run average SLA violation penalty per unit time.

	Service request type (i			
Agent (j)	1	2	3	4
1	10	10	∞	∞
2	∞	∞	∞	3
3	10	∞	15	∞
4	∞	∞	5	8

Table 5.2: Mean service time ($1/\mu_{ji}$) for each agent and each service request type. Infinity represents that the agent is not skilled in handling the type of request.

In order to demonstrate the performance of the allocation indices, we simulate this system under two settings. In the first setting, service requests are assigned to agents based on a first-come-first-serve (FCFS) policy. In the alternative setting, we use the priority-based allocation index defined in Equation 5.1. We use common random numbers for both systems in order to reduce the variance of the performance difference. We then analyze the warm up period using Welch's method, presented by Law and Kelton [61]. After examining the output we conclude that 100 service

requests provides a sufficient warm-up period. After deleting the data from the warm-up period, we calculate the average difference between the cost of our proposed algorithm and that based on FCFS policy and construct 90% confidence intervals and used a paired t-test to compare the results.

Our initial computations show that the proposed dispatching algorithm dramatically reduces SLA violation penalties compared to a FCFS policy. However, in spite of reducing the penalty cost, the proposed algorithm increases the average queue length and average delay in the queue. This is primarily due to the fact that the service requests which passed their deadline receive the lowest priority and therefore have to wait longer. We summarize the results (SLA violation penalty) of 40 replications in Table 5.3.

	FCFS policy	Proposed policy	Difference
Mean performance	37.33	7.28	30.04
Confidence interval (90%)	(35.42,39.23)	(6.76,7.81)	(28.04,32.05)

Table 5.3: Comparing the long run average SLA violation penalty per unit time of the proposed dispatching policy with FCFS.

5.5.3 Sensitivity Analysis

In this section we investigate the effects of:

- increasing the aggregate arrival rate,
- improving the skill levels of the agents, and
- decreasing the deadlines on the SLA violation penalty.

To investigate the sensitivity of the proposed algorithm to any possible change in arrival rate, we analyze the system where the aggregate arrival rate is increased from 0.5 to 4 arrivals per minute (see Figure 5.3). We observe that the penalty cost increases for both algorithms. However, the rate of increase in our proposed

algorithm is much slower compared to FCFS as the aggregate arrival rate increases. In other words, prioritizing the service requests is more critical as the system becomes more crowded.

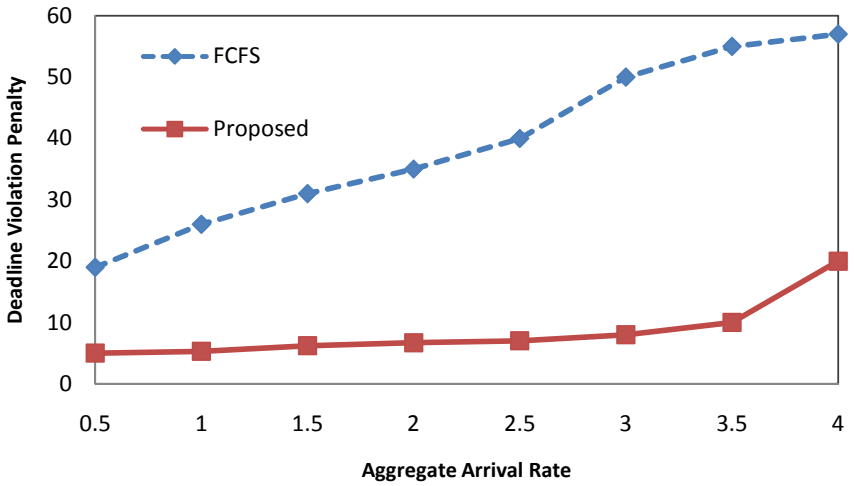


Figure 5.3: SLA violation penalty as a function of arrival rate

The second important analysis here is to show how the SLA violation cost changes as the agent's skill levels improve marginally. In this case we decrease the mean service time of one of an agent (Agent 1) for service request type two ($1/\mu_{12}$) from 10 to 8 minutes. There are many potential ways to achieve this objective in a real-life delivery environment, e.g. via implementing customized training programs, streamlining the problem diagnosis process, or automating some of the tasks performed by the agent. The results (see Figure 5.4) show that the proposed algorithm takes advantage of this improvement; however, the improvement under FCFS is not significant. This result suggests that the overall performance of the system in terms of cost is highly sensitive to the efficiency of the dispatching system. Moreover, improving the skill levels of agents does not necessarily decrease the penalty cost associated with deadline violation, because the policy employed may not be effective.

We examine how our proposed algorithm responds to modifications of the terms of the service contract, particularly shortening the deadlines. In order to test this scenario, we reduce the deadline for service requests of priority level 1, from 40 to 20 minutes. Our results (see Figure 5.4) show that both the proposed and FCFS policies are very sensitive to this reduction. In both settings, the SLA violation penalty is increased drastically. However the increase in the total penalty cost is more pronounced under the FCFS policy. This result is consistent with our hypothesis that a sound dispatching policy can mitigate the negative impacts of environmental factors, in this case, the terms of the contract.

Finally we test the effectiveness of our proposed heuristic on improving average agents' utilization. As it is demonstrated in Figure 5.5, the proposed heuristic improves the overall agents utilization by 7%.

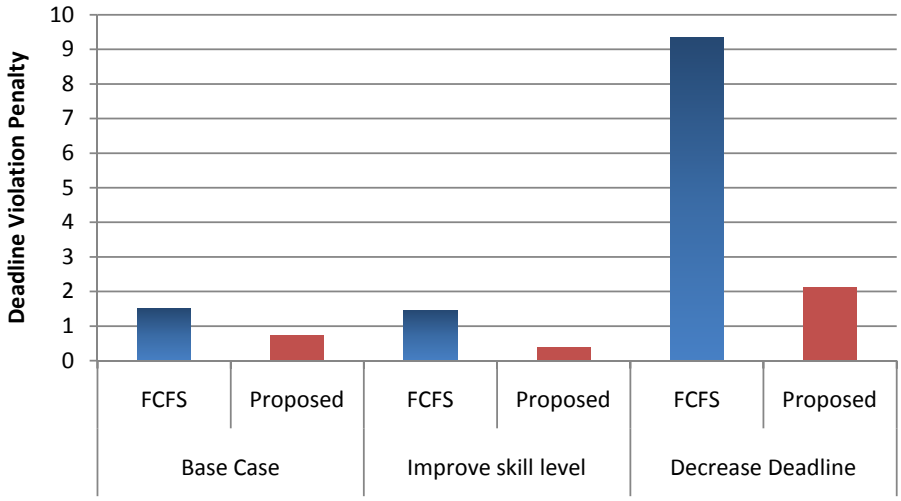


Figure 5.4: Effects of improving agents' skill levels and decreasing the deadlines on the overall cost of the system

5.6 Conclusion and Future Research

In this research, we propose a new priority-based dispatching policy that incorporates the inherent complexity of the IT service delivery environment. In the proposed

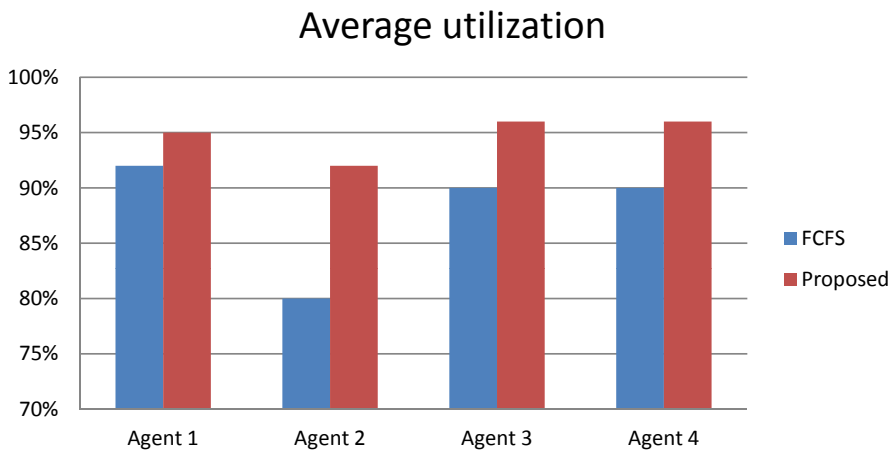


Figure 5.5: Comparison of average utilization of agents

approach, we first introduce a more accurate service complexity categorization by refining the granularity level of our input model. We also develop a new allocation index that captures uncertainty in agents' service time, which is a significant source of variability commonly observed in such environments. This index considers important factors such as deadlines and the variability in service time. It also incorporates the nature of probability distributions to address a more accurate measure of variability. Our proposed dispatching algorithm assigns a priority-based allocation index to each service request in the queue. This index is dynamically updated upon each service termination in the system (i.e., we assume non-preemptive service).

Our initial results show that the proposed dispatching policy can have a significant impact on reducing the penalty cost of violating deadlines. Sensitivity analysis shows that the proposed dispatching policy is even more significant in reducing penalty cost when the aggregate arrival rate increases or deadlines are shortened (both cases represent a more congested system in some sense). Further benchmarking is warranted.

In this research, we restricted attention to non-idling policies. However, there is no guarantee that our proposed policy can always perform well in other settings such as the case where idling is allowed. In this particular problem setting where

agents are cross-trained, one may argue that keeping some skilled agents idle for short periods of time may produce better results. Consider this simplified example: Agent 1 is very efficient at resolving service requests of type 1. At a given time, Agent 1 becomes idle but there is no request of type 1 in the queue. Here we have two choices: we can either assign another service request (e.g. type 2 at which Agent 1 is less skilled) or wait for a certain period of time expecting that another service request of type 1 arrives. In the current framework, we only restrict our approach to the first option. Future investigations are required to address this type of policy.

BIBLIOGRAPHY

[1] H.S. Ahn, I. Duenyas, and R.Q. Zhang. Optimal stochastic scheduling of a two-stage tandem queue with parallel servers. *Advances in Applied Probability*, 31(4):1095–1117, 1999.

[2] H.S. Ahn, I. Duenyas, and R.Q. Zhang. Optimal control of a flexible server. *Advances in Applied Probability*, 36(4):139–170, 2004.

[3] N. Altay and W.G. Green. OR/MS research in disaster operations management. *European Journal of Operational Research*, 175(1):475–493, 2006.

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

[5] S. Andradóttir, H. Ayhan, and D.G. Down. Server assignment policies for maximizing the steady-state throughput of finite queueing systems. *Management Science*, 47(10):1421–1439, 2001.

[6] S. Andradottir, H. Ayhan, and D.G. Down. Server assignment policies for maximizing the steady-state throughput of finite queueing systems. *Management Science*, 47(10):1421–1439, 2001.

[7] S. Andradottir, H. Ayhan, and D.G. Down. Dynamic assignment of dedicated and flexible servers in tandem lines. *Probability in the Engineering and Informational Sciences*, 21(4):497–538, 2007.

[8] S. Andradóttir, H. Ayhan, and D.G. Down. Maximizing the throughput of tandem lines with flexible failure-prone servers and finite buffers. *Probability in the Engineering and Informational Sciences*, 22(2):191–211, 2008.

[9] S. Andradottir, H. Ayhan, and D.G. Down. Design principles for flexible systems. working paper, 2010.

[10] R.G. Askin and J. Chen. Dynamic task assignment for throughput maximization with worksharing. *European Journal of Operational Research*, 168:853–869, 2006.

[11] F. Baccelli, P. Boyer, and G. Hebuterne. Single-server queues with impatient customers. *Advances in Applied Probability*, 16(4):887–905, 1984.

[12] J. Bae, S. Kim, and E.Y. Lee. Single-server queues with impatient customers. *Queueing Systems*, 38(4):485–494, 2001.

[13] G. Barbarosoğlu and Y. Arda. A two-stage stochastic programming framework for transportation planning in disaster response. *Journal of the Operational Research Society*, 55(1):43–53, 2004.

[14] D.Y. Barrer. Queuing with impatient customers and ordered service. *Queueing Systems*, 5(5):650–656, 1957.

[15] J.J. Bartholdi. A guaranteed-accuracy round-off algorithm for cyclic scheduling and set covering. *Operations Research*, 29(3):501–510, 1981.

[16] J.J. Bartholdi and D.D. Eisenstein. A production line that balances itself. *Operations Research*, 44(1):21–34, 1996.

[17] J.J. Bartholdi, D.D. Eisenstein, and R.D. Foley. Performance of bucket brigade when work is stochastic. *Operations Research*, 49(5):710–719, 2001.

[18] S.L. Bell and R.J. Williams. Dynamic scheduling of a system with two parallel servers: asymptotic policy in heavy traffic. *Annals of Applied Probability*, 11:608–649, 2001.

[19] S.L. Bell and R.J. Williams. Dynamic scheduling of a parallel server system in heavy traffic with complete resource pooling: Asymptotic optimality of a threshold policy. *Electronic Journal of Probablity*, 10:1044–1115, 2005.

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

[21] N.K. Boots and H. Tijms. Queuing with impatient customers and ordered service. *Management Science*, 45(3):444–448, 1999.

[22] M.J. Brusco and T.R. Johns. Staffing a multiskilled workforce with varying levels of productivity: an analysis of cross-training policies. *Decision Sciences*, 29(2):499–515, 1998.

[23] M.J. Brusco, T.R. Johns, and J.H. Reed. Cross-utilization of a two-skilled workforce. *International Journal of Operations and Production Management*, 18(6):555–564, 1998.

[24] C. Buyukkoc, P. Varaiya, and J. Walrand. The cμ rule revisited. *Advances in Applied Probability*, 17(1):237–238, 1985.

[25] J.A. Buzacott and J.G. Shanthikumar. *Stochastic Models of Manufacturing Systems*. Prentice Hall, Englewood Cliffs, NJ, 1993.

[26] A. Conticini. *Macroeconomics and Health in Malawi: What Way Forward*. World Health Organization, 2004.

[27] L.E. de la Torre, I.S. Dolinskaya, and K.R. Smilowitz. Disaster relief routing: Integrating research and practice. *Socio-Economic Planning Sciences*, 2011.

[28] N.B. Dimitrov, S. Goll, L.A. Meyers, B. Pourbohloul, and N. Hupert. Optimizing tactics for use of the us antiviral strategic national stockpile for pandemic (H1N1) influenza. *Public Library of Science*, 6(1), 2009.

[29] N.B. Dimitrov and D.P. Morton. Combinatorial design of a stochastic markov decision process. *Operations Research and Cyber Infrastructure*, pages 167–193, 2009.

[30] D.G. Down, G. Koole, and M.E. Lewis. Dynamic control of a single-server system with abandonments. *Queueing Systems*, 67:63–90, 2011.

[31] D.G. Down and M.E. Lewis. The n-network model with upgrades. working paper, 2009.

[32] F. Dzinjalamala. Epidemiology of malaria in malawi. Technical report, 2004.

[33] Z. Feldman, A. Mandelbaum, W.A. Massey, and W. Whitt. Staffing of time-varying queues to achieve time-stable performance. *Management Science*, 54(2):324–338, 2008.

[34] P.D. Finch. Deterministic customer impatience in the queueing system gi/m/1. *Biometrika*, 47(1):45–52, 1960.

[35] S. Foster. Supply and Use of Essential Drugs in Sub-Saharan Africa: Some Issues and Possible Solutions. *Social Science & Medicine*, 32(11):1201–1218, 1991.

[36] P. Garner and P.M. Graves. The benefits of artemisinin combination therapy for malaria extend beyond the individual patient. *PLoS Medicine*, 2(4):e105, 2005.

[37] O. Garnet, A. Mandelbaum, and M. Reiman. Designing a call center with impatient customers. *Manufacturing and Service Operations Management*, 4(3):208–227, 2002.

[38] B. Gavish and P.J. Schweitzer. The markovian queue with bounded waiting time. *Management Science*, 23(12):1349–1357, 1977.

[39] J. Gayon, S. Benjaafar, and F. Vricourt. Using imperfect advance demand information in production-inventory systems with multiple customer classes. *Manufacturing and Service Operations Managemnet*, 11(1):128–143, 2009.

[40] E.G. Gel, W.J. Hopp, and M.P. Van Oyen. Hierarchical cross-training in work-in-process-constrained systems. *IIE Transactions*, 39:125–143, 2007.

[41] S. Ghamami and A.R. Ward. Dynamic scheduling of a two-server parallel server system with complete resource pooling and reneging in heavy traffic: Asymptotic optimality of a two-threshold policy. working paper, 2010.

[42] J. Gittins, K. Glazebrook, and R. Weber. *Multi-armed Bandit Allocation Indices*. John Wiley and Sons, New York, NY, second edition, 2011.

[43] S. Gurumurthi and S. Benjaafar. Modeling and analysis of flexible queueing systems. *Naval Research and Logistics*, 51(5):755–782, 2004.

[44] M.J. Harrison. Heavy traffic analysis of a system with parallel servers: Asymptotic optimality of discrete-review policies. *The Annals of Applied Probability*, 8(3):822–848, 1998.

[45] F.S. Hillier and K.C. So. Throughput maximization for tandem lines with two stations and flexible servers. *Queueing Systems*, 21(3):245–266, 2005.

[46] K. Holmberg. Efficient decomposition and linearization methods for the stochastic transportation problem. *Computational Optimization and Applications*, 4(4):293–316, 1995.

[47] W.J. Hopp and M. Spearman. *Factory Physics*. McGraw-Hill, New York, NY, 2 edition, 2000.

[48] W.J. Hopp, E. Tekin, and M.P. Van Oyen. Benefits of skill chaining in serial production lines with cross-trained workers. *Management Science*, 50(1):83–98, 2004.

[49] W.J. Hopp and M.P. Van Oyen. Agile workforce evaluation: a framework for cross-training and coordination. *IIE Transactions*, 36:919–940, 2004.

[50] D.W. Hutton, M.L. Brandeau, and S.K. So. Doing Good with Good OR: Supporting Cost-Effective Hepatitis B Interventions. *Interfaces*, 41(3), 2011.

[51] S.M. Iravani, B. Kolfal, and M.P. Van Oyen. Call-center labor cross-training: It's a small world after all. *Management Science*, 53(7):1102–1112, 2007.

[52] S.M. Iravani, M.P. Van Oyen, and K.T. Sims. Structural flexibility: A new perspective on the design of manufacturing and service operations. *Management Science*, 51(2):151–166, February 2005.

[53] O. Jennings and J.E. Reed. overloaded multiclass fifo queue with abandonments. 2010.

[54] W.J. Jordan and S.C. Graves. Principles on the benefits of manufacturing process flexibility. *Management Science*, 41(4):577–594, 1995.

[55] E. Kim and M.P. Van Oyen. Dynamic scheduling to minimize lost sales subject to set-up costs. *Queueing Systems*, 29(2):193–229, October 1998.

[56] J.H. Kim, H.S. Ahn, and R. Righter. Optimal production policies with multistage stochastic demand lead times. *Probability in Engineering and Informational Sciences*, 23(3):515–543, 2009.

[57] G. Koole and R. Righter. Optimal control of tandem reentrant queues. *Queueing Systems*, 28(4):337–347, 1998.

[58] S.V. Lall, H. Wang, and T. Munthali. Explaining High Transport Costs within MalawiBad Roads or Lack of Trucking Competition? Technical report, The World Bank, 2009.

[59] A. Lasry, S.L. Sansom, K.A. Hicks, and V. Uzunangelov. A model for allocating cdc's hiv prevention resources in the united states. *Health Care Management Science*, pages 1–10, 2011.

[60] A. Lasry, G.S. Zaric, and M.W. Carter. Multi-level resource allocation for hiv prevention: A model for developing countries. *European Journal of Operational Research*, 180(2):786–799, 2007.

[61] A. Law and W.D. Kelton. *Simulation Modeling and Analysis*. McGraw Hill, New York, NY, 2000.

[62] E.K. Lee, C.H. Chen, F. Pietz, and B. Benecke. Modeling and optimizing the public health infrastructure for emergency response. *Interfaces*, 39(5):476–490, 2009.

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

[64] Q. Ma and P. Steenkiste. Routing traffic with quality-of-service guarantees in integrated services networks. In *In The 8th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 98)*, 1998.

[65] P. Malaney, A. Spielman, and J. Sachs. The malaria gap. *The American Journal of Tropical Medicine and Hygiene*, 71(2):141–146, 2004.

[66] J.O. McClain, K.L. Schultz, and J.L. Thomas. Management of worksharing systems. *Manufacturing and Service Operations Management*, 2(1):49–67, 2000.

[67] H.O. Mete and Z.B. Zabinsky. Stochastic optimization of medical supply location and distribution in disaster management. *International Journal of Production Economics*, 126(1):76–84, 2010.

[68] G. Miller, S. Randolph, and D. Gower. Simulating the response of a rural acute health-care delivery system to a bioterrorist attack. *International Journal of Disaster Medicine*, 2(1):24–32, 2004.

[69] G. Miller, S. Randolph, and J.E. Patterson. Responding to bioterrorist smallpox in San Antonio. *Interfaces*, 36(6):580–590, 2006.

[70] S.S. Panwar, D. Towsley, and J.K. Wolf. Optimal scheduling policies for a class of queues with customer deadlines to the beginning of service. *Journal of the Association for Computing Machinery*, 35(4):832–844, 1988.

[71] H. Parvin, M.P. Van Oyen, D.G. Pandelis, D.P. Williams, and J. Lee. Fixed task zone chaining: Worker coordination and zone design for inexpensive cross-training in serial conwip lines. working paper, 2011.

[72] H.G. Perros. Queueing networks with blocking: a bibliography. *ACM SIGMETRICS Performance Evaluation Review*, 12(2):8–12, 1984.

[73] T. Pyzde. Why normal distributions aren't [all that normal]. *Quality Engineering*, 7(4):769–777, 1995.

[74] Republic of Malawi Ministry of Health. National malaria control program supervision report for monitoring act and malaria control activities. Technical report, 2008.

[75] H.E. Romeijn and F.Z. Sargutb. The stochastic transportation problem with single sourcing. *European Journal of Operational Research*, 214(2):262–272, 2011.

[76] S. Ross. *Introduction to Stochastic Dynamic Programming*. Academic Press, San Diego, CA, second edition, 1995.

[77] J. Salmerón and A. Apte. Stochastic optimization for natural disaster asset prepositioning. *Production and Operations Management*, 2010.

[78] L.I. Sennott. *Stochastic Dynamic Programming and the Control of Queueing Systems*. John Wiley and Sons, New York, NY, 1999.

[79] L.I. Sennott, M.P. Van Oyen, and S.M. Iravani. Optimal dynamic assignment of a flexible worker on an open production line with specialists. *European Journal of Operational Research*, 170(2):541–566, April 2006.

[80] N.C. Simpson and P.G. Hancock. Fifty years of operational research and emergency response. *Journal of the Operational Research Society*, 60(Supplement 1):S126–S139, 2009.

[81] M.E. Sisselman and W. Whitt. Value-based routing and preference-based routing in customer contact centers. In *Production and Operations Management*, page 16, 2004.

[82] R.E. Stanford. Optimal scheduling policies for a class of queues with customer deadlines to the beginning of service. *Mathematics of Operations Research*, 4(2):162–178, 1979.

[83] M. Treleven. A review of the dual source constrained system research. *IIE Transactions*, 21(3):279–287, 1989.

[84] United Nations. Human development report. Technical report, 2011.

[85] J.A. Van Mieghem. Dynamic scheduling with convex delay costs: The generalized cμ rule. *The Annals of Applied Probability*, 5(3):809–833, 1995.

[86] M.P. Van Oyen, E.S. Gel, and W.J. Hopp. Performance opportunity for workforce agility in collaborative and noncollaborative work systems. *IIE Transactions*, 33(9):761–777, 2001.

[87] M.H. Veatch. A cμ rule for parallel servers with two-tiered cμ preferences. working paper, July 2009.

[88] R.B. Wallace and W. . A staffing algorithm for call centers with skill-based routing. *Manufacturing and Service Operations Management*, 7:276–294, 2005.

[89] J. Walrand. *An Introduction to Queueing Networks*. Prentice-Hall, New York, NY, 1988.

[90] W. Whitt. Staffing a call center with uncertain arrival rate and absenteeism. *Production and Operations Management*, 15(1):88–102, 2006.

[91] P. Whittle. Restless bandits: Activity allocation in a changing world. *Journal of Applied Probability*, 25:287—298, 1980.

[92] A.C. Williams. A stochastic transportation problem. *Operations Research*, 11(5):759–770, 1963.

[93] D.P. Williams. *Investigations into Flexible Operational Paradigms to Mitigate Variability*. PhD thesis, University of Michigan, Industrial and Operations Engineering, 2009.

[94] World Health Organization. World malaria report. 2010.

[95] G.S. Zaric and M.L. Brandeau. Resource allocation for epidemic control over short time horizons. *Mathematical Biosciences*, 171(1):33–58, 2001.

[96] G.S. Zaric and M.L. Brandeau. Dynamic resource allocation for epidemic control in multiple populations. *Mathematical Medicine and Biology*, 19(4):235, 2002.

[97] G.S. Zaric and M.L. Brandeau. A little planning goes a long way: Multilevel allocation of hiv prevention resources. *Medical Decision Making*, 27(1):71, 2007.

[98] J. Zhu, J. Huang, D. Liu, and J. Han. Resources allocation problem for local reserve depots in disaster management based on scenario analysis. In *The 7th International Symposium on Operations Research and its Applications. Lijiang, China*, pages 395–407, 2008.

[99] E. Zohar, A. Mandelbaum, and N. Shimkin. Adaptive behavior of impatient customers in tele-queues: Theory and empirical support. *Management Science*, 48(4):566–583, 2002.