Iowa Research Online
The University of Iowa's Institutional Repository

University of Iowa
Iowa Research Online

Theses and Dissertations

Summer 2016

Workforce planning in manufacturing and healthcare systems

Huan Jin
University of Iowa

Copyright © 2016 Huan Jin

This dissertation is available at Iowa Research Online: http://ir.uiowa.edu/etd/5784

Recommended Citation
Jin, Huan. "Workforce planning in manufacturing and healthcare systems." PhD (Doctor of Philosophy) thesis, University of Iowa, 2016.
http://ir.uiowa.edu/etd/5784.

Follow this and additional works at: http://ir.uiowa.edu/etd


Part of the Business Administration, Management, and Operations Commons

WORKFORCE PLANNING IN MANUFACTURING AND HEALTHCARE SYSTEMS

by

Huan Jin

A thesis submitted in partial fulfillment of the
requirements for the Doctor of Philosophy
degree in Business Administration (Management Sciences)
in the Graduate College of
The University of Iowa

August 2016

Thesis Supervisor: Associate Professor, Barrett Thomas

Copyright by
HUAN JIN
2016
All Rights Reserved

Graduate College
The University of Iowa
Iowa City, Iowa

CERTIFICATE OF APPROVAL

PH.D. THESIS

This is to certify that the Ph.D. thesis of

Huan Jin

has been approved by the Examining Committee for the thesis requirement for the Doctor of Philosophy degree in Business Administration (Management Sciences) at the August 2016 graduation.

Thesis committee: _____
Barrett Thomas, Thesis Supervisor

_____
Samuel Burer

_____
Renato de Matta

_____
Michael Hewitt

_____
Qihang Lin

ACKNOWLEDGEMENTS

I would like to thank many people who helped with this research for their participation.

Foremost, I would like to express my sincere gratitude to my advisor Prof. Barrett Thomas for the continuous support of my Ph.D. study and research.His guidance and encourage helped me in all the time of my research and writing of this thesis. I am greatly fortunate to have an advisor who gives me the freedom to explore new areas of research on my own and gives me direction when my steps get stuck. His enthusiasm helps me overcome many difficulties and get me through finishing my Ph.D. study. I am deeply grateful to him for the long discussion every week that helps me figure out the direction and technical details of my work. I am also thankful to him for showing me good academic writing skills and the use of correct grammar and for carefully reading and commenting on countless revisions of my manuscripts.

I also would like to thank Prof. Michael Hewitt from Loyola University, for his substantial advice on grasping the key issue of my research papers, designing computational experiments, and the techniques of professionally present computational results. Prof. Hewitt's help is significant in publishing my research.

I would like to thank the rest of my thesis committee: Prof. Samuel Burer, Prof. Renato de Matta, and Dr. Qihang Lin, for their encouragement, insightful comments, and hard questions. In particular, I am grateful to Dr. Qihang Lin for enlightening me the first glance of some of my research problems.

I am also indebted to the Wendy Voigt, the manager of the Nursing Clinical Education Center at University of Iowa Hospitals and Clinics (UIHC). She showed me a tour at the medical laboratory and introduced the detailed working process at UIHC. More importantly, she provided me a one-month range dataset which included the detail information of the phlebotomists and sample draws collected by the medical laboratory. Without Wendy's help, the last part of my research could not be so smooth.

I thank many of my friends in Management Sciences Department, Computer Sciences Department, and Statistics Department, including but not limit to Farley Lai, Sider Rezwanul Huq, Amin Vahedian, and Hanqin Cai for giving me insights on programming and help me efficiently implement algorithms in all kinds of programming languages.

Many friends have helped me through these difficult years, especially my local family friends, Donna Hollinger, Rhonda Fruhling, and Jackie Heinle. Their support and care helped me adjust to the life in a new country and overcome setbacks and stay focused on my graduate study. I greatly value their friendship and I deeply appreciate their belief in me.

Last but not the least, I would like to thank my family: my parents and my brother, for supporting me spiritually throughout my life.

ABSTRACT

This dissertation explores workforce planning in manufacturing and healthcare systems. In manufacturing systems, the existing workforce planning models often lack fidelity with respect to the mechanism of learning. Learning refers to that employees' productivity increases as they gain more experience. Workforce scheduling in the short term has a longer term impact on organizations' capacity. The mathematical representations of learning are usually nonlinear. This nonlinearity complicates the planning models and provides opportunities to develop solution methodologies for realistically-sized instances. This research formulates the workforce planning problem as a mixed integer nonlinear program (MINLP) and overcomes the limitations of current solution methods. Specifically, this research develops a reformulation technique that converts the MINLP to a mixed integer linear program (MILP) and proposes several techniques to speed up the solution time of solving the MILP.

In organizations that use group work, workers learn not only by individual learning but also from knowledge transferred from team members. Managers face the decision of how to pair or team workers such that organizations benefit from this transfer of learning. Using a mathematical representation that incorporates both individual learning and knowledge transfer between workers, this research considers the problem of grouping workers to teams and assigning teams to sets of jobs based on workers' learning and knowledge transfer characteristics. This study builds a Mixed-integer nonlinear programs (MINP) for parallel systems with the objective of maxi-

mizing the system throughput and propose exact and heuristic solution approaches for solving the MINLP.

In healthcare systems, we focus on managing medical technicians in medical laboratories, in particular, the phlebotomists. Phlebotomists draw specimens from patients based on doctors' orders, which arrive randomly in a day. According to the literature, optimizing scheduling and routing in hospital laboratories has not been regarded as a necessity for laboratory management. This study is motivated by a real case at University of Iowa Hospital and Clinics, where there is a team of phlebotomists that cannot fulfill doctors requests in the morning shift. The goal of this research is routing these phlebotomists to patient units such that as many orders as possible are fulfilled during the shift. The problem is a team orienteering problem with stochastic rewards and service times. This research develops an a priori approach which applies a variable neighborhood search heuristic algorithm that improves the daily performance compared to the hospital practice.

PUBLIC ABSTRACT

This dissertation explores workforce planning in manufacturing and healthcare systems. The existing workforce planning models often lack fidelity of human learning. Learning refers to that employees' productivity increases as they gain more experience. The nonlinearity of learning function complicates the planning models and provides opportunities to develop methodologies for realistically-sized instances. This dissertation formulates the problem as a nonlinear model and overcomes the limitations of current solution methods.

In organizations that use group work, workers learn not only by individual learning but also from knowledge transferred from teammates. Managers face the decision of how to team workers such that organizations benefit from this transfer of learning. Using a mathematical representation that incorporates both individual learning and knowledge transfer between workers, this dissertation considers the problem of grouping workers to teams and assigning teams to jobs based on workers' learning characteristics. This dissertation builds a nonlinear model for parallel systems that maximizes the system throughput and propose an exact approach for solving the model.

In healthcare systems, phlebotomists draw specimens from patients based on doctors' orders, which arrive randomly in a day. Efficient management of phlebotomists has not been regarded as a necessity in laboratory management. This dissertation analyzes a real case at University of Iowa Hospital and Clinics, where

there is a team of phlebotomists that cannot fulfill doctors requests in the morning shift. This dissertation builds a mathematical model that routes phlebotomists to patient units such that as many orders as possible are fulfilled during the shift.

TABLE OF CONTENTS

LIST OF TABLES

Table

LIST OF FIGURES

Figure

LIST OF ALGORITHMS

Algorithm

LIST OF LEMMAS AND THEOREMS

Lemma\Theorem

CHAPTER 1
INTRODUCTION AND RELATED WORK

1.1 Introduction and Motivation

Human workers are viewed as important resources in organizations. Workers' skills and aptitudes provide the necessary level of flexibility. Workforce planning is the process by which companies deploy their workforce to complete the tasks of the organization. Workforce planning involves designing workers assignment, scheduling, and routing that meet the organizations' strategic goals. In making these planning, it is important to recognize that the schedules or assignments made in the short term have a longer term impact on the organization's capacity. In particular, workers learn and become more productive the more experience they gain. The failure to consider workers' learning leads to underestimation of the workforce capacity. Thus, workforce planning should account for both the current and future needs of the organization. It is also critical to model learning so that the company can more effectively match its human resource capacity to the demand. The mathematical representations of learning are called learning curves, which are usually nonlinear. This nonlinearity of the learning curve complicates the planning models and provides opportunities to develop solution methodologies for realistically-sized instances.

The current research on workforce planning often uses aggregate planning models. Wirojanagud et al. (2007) consider workforce decisions, such as hiring, cross-training, and assigning, in terms of several levels of workers. Fowler et al. (2008)

develop a mixed integer programming model that determines staffing decisions in the aggregate level. From a strategic planning perspective, treating individuals in the aggregate can lead to underestimation of the workforce capacity. For example, Shafer et al. (2001) mention that "modeling only central tendency and not also variation among workers can result in substantially underestimating overall system productivity." From an operational perspective, aggregate models are insufficient to accurately represent workers learning and are hard to effectively implement. Different workers have different productivities and learn at different paces. Accounting for the heterogeneity of the workforce adds complexity to workforce planning.

Similarly, most existing research on workforce planning prescribes the skill mix of a workforce to perform a set of jobs without considering how to develop the workforce (Agnihothri et al. (2003), Hopp et al. (2004), Iravani et al. (2007), Colen and Lambrecht (2010)). Chapter 2 focuses on determining workforce plans in an operational level given the tasks that need to be done over the planning horizon. The problem requires task assignment decisions of matching workers to jobs. We build mathematical models in a discrete planning horizon that answer the question of "How to create daily workforce plans that incorporate human learning and meet some strategic goals." The model makes decisions of assigning a set of heterogeneous workers to jobs accounting for their learning abilities. We assume that each individual worker has a known learning curve that determines that workers' productivity with respect to the accumulated experience. The objective is to minimize the time required to complete all of the jobs (known as minimizing the makespan). The key challenge

in solving these problems is that the learning curves are usually nonlinear. Chapter 2 studies the optimization techniques for makespan minimizing workforce assignment problems wherein human learning is explicitly modeled. We present a set of techniques that enable the solution of much larger instances of the workforce assignment problems than seen in the literature to date. The first technique is an exact linear reformulation for the general makespan minimizing workforce assignment models with learning. We then introduce a computationally efficient means for generating an initial feasible solution. Finally, we present methods for strengthening the formulation with cover inequalities and a lower bound on the objective function value of the optimal solution. With an extensive computational study, we demonstrate the value of these techniques and that large instances can be solved much faster than have previously been solved in the literature.

In a subsequent study, we consider not only individual learning-by-doing but also learning by knowledge transfer between team workers in organizations. For an individual, knowledge transfer occurs when working closely with team workers who are doing the same or related jobs. In manufacturing systems, short product life cycles and intense competition require workers to collaborate in groups. Workers are often grouped together and perform related jobs. For example, workers on the same production line are a team with a shared sense of each individual's expertise. This is critical for defining roles and responsibilities inside the team and for assigning the most competent person to each role. Additionally, this shared sense of others' expertise enhances the interaction and exchange of knowledge between people performing

distinct roles (Nembhard and Bentefouet; 2015). For example, in Audi's production system, thousands of small jobs need to be done before a car rolls off the assembly line, including fitting dashboards, screwing on the underfloor, and installing the steering wheel. In the Audi production system, this is done by means of group work. In almost all areas, employees are divided into small teams. Members in a team make suggestions about how processes can be improved and work can be organized more ergonomically.

A new hire that works in a collaborative environment with other experienced workers is another example of knowledge transfer. In health care systems, a new technician in a medical laboratory is usually paired with an experienced technician for training and learning purposes. The learning process can vary from a couple of weeks to several months. The new technician will eventually obtain all the skills necessary for independent performance.

For both situations, managers face the decisions of how to team or pair workers considering individual learning and knowledge transfer within groups, and how these decisions affect the productivity of the organizations.

Chapter 3 considers the workforce allocation problem in which workers learn by experience and through knowledge transferred from co-located employees. This chapter models the allocation of workers to tasks, including grouping workers to teams based on individual characteristics and assigning teams to different sets of jobs. The study in Chapter 3 is essential to workforce management in organizations that group workers into teams and match workers to the right set of jobs. Chapter 3

makes the following contributions: First, we build a mixed integer nonlinear program (MINLP) for parallel production systems. Second, by exploiting the structure and characteristics of the optimal solutions, we develop an exact approach that linearizes the MINLP to a mixed integer linear program (MILP). This reduction in complexity allows us to solve workforce allocation problems with different problem settings. We show the computational efficiency of the proposed exact approach by comparing to the heuristic policies in the literature. Third, with the extensive computational results, we provide a solution structure analysis and give managerial insights for managers when making grouping and assignment decisions.

Chapter 4 studies workforce planning in health care systems. We focus on technicians in medical laboratories. Laboratories are health care facilities where pathologists provide testing of patient samples. Laboratory services account for more than 10% of hospital billing. The laboratory workforce includes phlebotomy technicians (referred to as phlebotomists in the rest of the dissertation) who draw samples from patients and other technicians (medical laboratory technicians and medical technologists) who perform tests on these samples. Phlebotomists primarily draw blood, urine, and other samples from patients and are often the patients' only contact with medical laboratories. Medical laboratory technicians perform general tests in all laboratory areas, such as hematology and immunology. Medical technologists perform complex laboratory tests using high-level troubleshooting skill sets. Laboratory services in health care play an important role in inpatient care. Studies have shown that laboratory performance affects approximately 65% of the most critical decisions on

admission, discharge, and medication of the inpatients (Leaven and Qu; 2014).

The shortage of hospital laboratory personnel, especially phlebotomists, continues to be of concern for many laboratories. This shortage often causes tardiness of fulfilling some sample draws, which affects decisions on admission, discharge, and medication of inpatients. With the projected need for more hospital laboratory personnel in the coming years coupled with an aging workforce, effective management of the clinical laboratory workforce is essential to meet the demand for staffing in the near future and beyond (Hamilton and Sm; 2014). The study in Chapter 4 focuses on phlebotomist routing that aims to improving the laboratory performance in terms of fulfilling more sample draws in a shift. The phlebotomist routing in Chapter 4 focuses on designing routes of visiting the patient rooms for a team of phlebotomists. The goal of these routes is to facilitate in a timely manner as many sample draws in as many patient units as possible. We call this the phlebotomist intra-hospital routing problem.

The study in Chapter 4 is motivated by the Department of Pathology at the University of Iowa Hospitals and Clinics (UIHC). At UIHC, there is a team of phlebotomists that works in the morning and another team works later in the day. We focus on the morning shift as workload often cannot be completed on time in the morning shift.Typically, they work from 05:30 to 09:30 and serve 27 patient units located in different buildings during their shift. This team of phlebotomists performs approximately 40% of the daily draws at UIHC. These draws are requested by doctors randomly during the day. We assume each order corresponds to a sample draw from

a patient. The arrival orders at each unit can be thought of as similar to arrivals to a queueing process. This random process makes the quantity of orders and service time of a unit uncertain.

The question is how to schedule these phlebotomists to a subset of the units and in what order these units should be served, with the objective of serving as many orders as possible during the shift. The random quantity of orders in the units and the uncertain service time make the routing of phlebotomists difficult. The problem can be considered a variant of the team orienteering problem on a network of queues, where the phlebotomist visits locations to serve orders and collect rewards. After arrival at a location, the phlebotomist start serving a queue of orders until all orders including pre-orders and add-ons are cleared.

Chapter 4 makes the following contributions to the literature. First, this paper is the first that formulate and solve the team orienteering problem with stochastic rewards and service times. Second, we develop a priori approach that gives a fixed route for each phlebotomist and derives an analytical procedure to evaluate the expected reward of a priori tour. Finally, our computational results demonstrate the value of our a priori tour in terms of serving more pre-orders than the hospital practice. Future work of this problem is suggested in Chapter 5.

1.2 Literature Review

In this section, we present a review of literature that is related to each chapter.

1.2.1 Literature on Human Learning

The impact of experience on service or production times is often called "learning" in the literature. The study of how humans learn has a long history. There exists an extensive body of literature that develops mathematical representations for the improvement in service and production times as experience increases. These representations are often called learning curves. With the introduction of his power model, Wright (1936) is often credited with introducing the first mathematical description of the relationship between experience and productivity. Since that time, many authors have made additions to the literature. Dar-El (2000), Jaber and Sikström (2004), Jaber (2006), and Anzanello and Fogliatto (2011) provide broad and thorough surveys of the subject. Dar-El (2000) provides a comprehensive review of both learning and forgetting models as well as parameter estimation for learning models developed before 2000. Hewitt et al. (2014) provide a review of learning curves in optimization models. Among the various learning curves, the best knowns are log-linear model, exponential model, and hyperbolic model, which are discussed in the following sections.

1.2.1.1 Log-linear Models

Wright (1936)'s model is the first learning model and is referred to as the "Log-linear Model" with the following form:

$$y = C_1 x^b \tag{1.1}$$

where y is the average time per unit demanded to produce x units, and C_1 is the time to produce the first unit. Parameter b $(-1 < b < 0)$ is the slope of the LC, which describes the workers learning rate. Values of b close to -1 indicate high learning rate and fast adaptation to task performance (Dar-El; 2000).

Modifications in the log linear model were initially proposed to adapt the equation to specific applications, and then recognized as alternative models. One such model is the Stanford-B presented in Equation (1.2), which incorporates workers prior experience.

$$y = C_1(x + B)^b \tag{1.2}$$

where B is an 'experience factor' which corresponds to the units of prior experience available at the start of a manufacturing program. It shifts the learning curve downwards with respect to the time/unit axis (Dar-El; 2000).

DeJong's model in Equation (1.3) incorporates both manual and machine control elements in the learning process. The manual parts are compressible with respect to experience, the machine part is not (Yelle; 1979). DeJong defines $M(0 \leq M \leq 1)$ represents the incompressible factor that demonstrates the fraction of the task executed by machines, so his model is:

$$y = C_1 \left[M + (1 - M)x^b\right]. \tag{1.3}$$

1.2.1.2 Exponential Models

There are three exponential learning curve models in the literature. They are: the 3-parameter exponential, the 2-parameter exponential, and the constant time model. The 3-parameter exponential model is:

$$y = K\left[1 - e^{-(x+p)/r}\right], \tag{1.4}$$

where y is worker's performance in terms of number of units produced after x units of time. Parameter K defines the asymptote production rate and delineates a prediction of a worker's performance after an infinite amount of practicing time. Parameter p corresponds to a worker's prior experience evaluated in units of time, while r is the time needed to reach 63% of the asymptote's production rate. The parameter r is called the learning rate. The larger the value of r, the slower the rate of improvement in performance (see Figure 1.1).

In the 2-parameter exponential model, parameter p is not present, offering poorer fit to performance data compared to the 3-parameter exponential model (Mazur and Hastie; 1978).

The constant time model is similar to the 3-parameter exponential model. It was proposed by F.W. Bevis (1970).

$$y = y_c + y_f(1 - e^{-t/\tau}) \tag{1.5}$$

where y_c is the starting production rate, y_f is the steady state production rate during the transient part of the curve. τ is the time constant, which is the time needed to reach 63% of the steady state production rate.

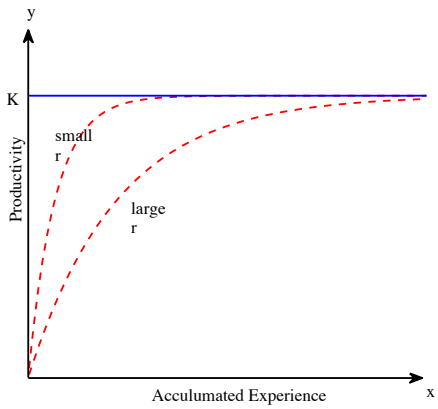

Figure 1.1: Three-parameter exponential learning curve by learning rate

1.2.1.3 Hyperbolic Models

Mazur and Hastie (1978) proposed a 3-parameter hyperbolic learning model that relates the number of conforming units to the total number of units produced.

$$y = k\left(\frac{x+p}{x+p+r}\right) \tag{1.6}$$

Parameters in Equation (1.6) are defined as those in the 3-parameter exponential model in Equation (1.4).

In the 2-parameter hyperbolic model, parameter p is not include and has the form of:

$$y = k\left(\frac{x}{x+r}\right) \tag{1.7}$$

In production processes, there is often some interruptions due to modifications

in product or some unexpected events. One of the consequences of frequent interruption is forgetting when they resume activities (Jaber; 2006). The forgetting can lead to reduction in production rate after an inactive period.

Some of the learning curves enable assessing both workers' learning and forgetting process. Table 1.1 summarizes the discussed learning models in terms of their mathematical representations, the number of parameters, and if the model enable to assess workers' forgetting. As shown in Table 1.1 , all of the commonly employed learning curves (Log linear, exponential, hyperbolic) are nonlinear. Dar-El (2000), Jaber and Sikström (2004), Jaber (2006), and Anzanello and Fogliatto (2011) provide broad and thorough surveys of the subject. Hewitt et al. (2014) provide a review of learning curves in optimization models.

Model	Mathematical Representation	Number of Parameters	Enable Forgetting?
Wright's Log Linear	$y = C_1 x^b$	2	N
Stanford-B	$y = C_1(x + B)^b$	3	N
DeJong's	$y = C_1\left(M + (1 - M)x^b\right)$	3	Y
3-parameter exponential	$y = K\left(1 - e^{-(x+p)/r}\right)$	3	Y
Constant time	$y = y_c + y_f(1 - e^{-t/\tau})$	3	N
2-parameter hyperbolic	$y = kx/(x + r)$	2	Y
3-parameter hyperbolic	$y = k(x + p)/(x + p + r)$	3	Y

Table 1.1: Comparative analysis of learning curves

1.2.2 Literature on Makespan Minimizing Problem with Human Learning

Chapter 2 focuses on scheduling jobs to workers with the goal of minimizing the makespan. Our models account for the fact that workers learn and thus increase productivity as they gain experience.

The most closely related work to Chapter 2 is Corominas et al. (2010), who introduce a piecewise linearization of a learning function for a task assignment model. The largest problem solved has five tasks and four workers and they do so with a two-segment piecewise linear approximation to linearize a concave learning funtion. Olivella et al. (2013) extend Corominas et al. (2010) to consider due dates and cross-training goals and use a similar solution approach to Corominas et al. (2010). The chapter solves problems with up to four workers and eight tasks. Heimerl and Kolisch (2010) addresses a problem similar to Olivella et al. (2013) while also including forgetting and company skill levels. Using a nonlinear programming solver that cannot guarantee optimal solutions, Heimerl and Kolisch (2010) solve problems with six workers and 20 tasks. In contrast, in the work presented in this chapter, we solve exactly problems with up to 20 workers and 30 tasks. A comprehensive review of other work that incorporates learning into workforce planning models can be found in Hewitt et al. (2014).

Using a model for assembly-line production first presented in Nembhard and Norman (2007), Hewitt et al. (2014) introduces a technique for deriving exact linear reformulations of nonlinear learning functions. The reformulation technique presented in Hewitt et al. (2014) models nonlinear functions with discrete domains and ranges

as sets of binary and linear variables and constraints. Hewitt et al. (2014) can solve problems up to 20 workers and 40 tasks in less than an hour. However, Hewitt et al. (2014) considers a different non-linear model of human learning and a different setting than we consider in this chapter. Thus, while this chapter adapts the reformulation technique presented in Hewitt et al. (2014), it differs in three important ways: (1) We consider a different class of scheduling problems (makespan minimizing workforce assignment problems), (2) We consider a different quantitative model of how experience translates to proficiency, and, (3) We provide techniques that significantly reduce the solve time of the reformulated model.

Table 1.2 summarizes the literature most closely related to that in Chapter 2. The first column cites the paper being summarized and the second notes from what paper the summarized paper's model is derived. The third and fourth columns identify whether or not the model includes cross-training goals and whether or not the model is nonlinear, respectively. The fifth and sixth columns (found in the second layer of the table) highlight the solution method and whether or not the method is a heuristic method, respectively. The final three columns indicated the largest problem solved in each paper by indicating the number of workers, tasks and time periods, respectively, of the largest problem solved.

1.2.3 Literature on Workforce Grouping and Knowledge Transfer

Compared to individual learning studies, studies of learning in a team are very rare in the literature. When working in a team, individuals benefit from not only their

Publication	Related Models	Cross-Training Goals	Nonlinear Model
Nembhard and Norman (2007)			
Corominas et al. (2010)	-		✓
Heimerl and Kolisch (2010)	-	✓	✓
Olivella et al. (2013)	Corominas et al. (2010)	✓	✓
Hewitt et al. (2014)	Nembhard and Norman (2007)		
Chapter 2	Corominas et al. (2010)		

Publication	Solution Approach	Heuristic	Workers	Tasks	Periods
Nembhard and Norman (2007)	Nonlinear Programming		2	4	10
Corominas et al. (2010)	Piecewise Linearization	✓	4	5	20
Heimerl and Kolisch (2010)	Primal-Dual Interior Point	✓	6	4	5
Olivella et al. (2013)	Approximate Convex Piecewise Linearization	✓	4	8	40
Hewitt et al. (2014)	Reformulated Mixed Integer Program		20	40	40
Chapter 2	Reformulated Mixed Integer Program		20	30	NA

Table 1.2: Summary of related literature to Chapter 2

own learning but also knowledge transferred from their colleagues. There have been investigations into how and why working in a team environment can improve individual performance. Faraj (2000) and Lewis (2003) suggest that teams composed of individuals who have experience working together are more accurate and have a collective sense of each individual's expertise. Uzzi and Lancaster (2003) discuss how trust can promote the exchange of "private" knowledge and information. Norman et al. (2002) develop a team based model for worker scheduling in manufacturing cells that considers both technical and human skills. With this model, they show that when skills are considered in both worker training plans and assignment strategies, the performance of the cell will improve significantly. However, there has been little research on how to quantify the transfer of knowledge to a worker from his/her colleagues. Recently, Nembhard and Bentefouet (2015) propose a mathematical description of knowledge transfer by extending a well-known 3-parameter hyperbolic learning curve. Nembhard and Bentefouet (2015) develop several heuristic approaches for selecting a workforce, grouping the workforce into teams, and assigning teams to jobs. They identify the top

policies that yield the maximum throughput for each decision based on their computational studies. Motivated by Nembhard and Bentefouet (2015)'s work, Chapter 3 consider the workforce allocation problem, including grouping workers into teams based on individual learning characteristics, and assigning teams to different sets of jobs. Unlike Nembhard and Bentefouet (2015), we develop an exact approach that reformulate the nonlinear mixed integer program to a linear mixed integer program and solve large-sized problems in reasonable time limit.

Publication	Team Decision	Knowledge Transfer	Nonlinear Model	Solution Approach	Problem Size Workers	Tasks
Faraj (2000)	✓	✓		Regression	-	-
Norman et al. (2002)	✓			MIP	6	6
Jin et al. (2016)			✓	Reformulated MIP	20	30
Nembhard and Bentefouet (2015)	✓		✓	Heuristic	9	9
Chapter 3	✓	✓	✓	Reformulated MIP	60	60

Table 1.3: Summary of related literature to Chapter 3

Besides the literature of human learning in Section 1.2.1 and Section 1.2.2, we summarize the literature related to learning by knowledge transfer and workforce grouping decision in Table 1.3. The first column cites the paper being summarized. The second and third columns identify whether or not the model includes team decisions and knowledge transfer in the team. The fourth and fifth columns indicate whether or not the model is nonlinear and what solution approach is used to solve the model, respectively. The sixth and seven columns indicate the largest problem solved in each paper by indicating the number of workers and tasks of the largest problem

solved.

1.2.4 Literature on Phlebotomist Routing

The proposed study of phlebotomist intra-hospital routing in Chapter 4 is in the scope of workforce planning in healthcare systems. The problem is considered the team orienteering problem in a polling system with a network of queuing. This section gives a literature review of workforce planning in health care systems, team orienteering problem, routing on a network of queues, and polling systems.

1.2.4.1 Literature on Workforce Planning in Health Care Systems

The majority of research on workforce planning in health care systems focuses on nurse and physician scheduling and rostering. Azaiez and Al Sharif (2005), Beliën and Demeulemeester (2008), Maenhout and Vanhoucke (2009) typically include decisions of determining the number of nurses to be hired, assigning nurses to a set of shifts so that feasibility requirements are met, and assigning patients to nurses at the beginning of shifts. The extensive reviews of related research can be referred to Cheang et al. (2003), Landeghem (2004), Denton (2013).

However, phlebotomist optimization is different from nurse and physician optimization. For example, patients are assigned to the same nurse during the shift, while there is no such restriction for phlebotomist assignments. This makes phlebotomist scheduling more flexible and more complicated. Research on phlebotomist optimization is very limited in literature. Leaven and Qu (2014) state the importance of phlebotomist scheduling in laboratory accounting for the uncertainty associated with

the number of blood draws ordered in each shift. They propose a scenario reduction model and return the scenarios with the largest likelihood of occurrence in each shift. However, they do not address the issue of how to schedule the phlebotomists in each shift and how to assign the blood draws to each phlebotomist.

Other work that addresses intra-hospital routing focuses on intra-hospital routing focuses on patient routing between different units in the hospital. Schmid et al. (2014) consider the operating room scheduling and intra-hospital routing of patients. Patients typically undergo several examinations before their actual surgery. This paper considers the problem of scheduling patients to examination rooms, such that no more than one patient is scheduled to be in a room at any point in time, and the precedence requirements of appointments associated with the same patients are met.

1.2.4.2 Literature on Orienteering Problem

The phlebotomist intra-hospital routing problem is modeled as a team orienteering problem considering stochastic demand and service time. The stochastic variant of the orienteering problem is limited. Tang and Miller-Hooks (2005) propose an orienteering problem with stochastic service time, travel time and travel costs, which is formulated as a chance-constrained stochastic integer program with a objective of maximizing the total profits collected from a priori tour while restricting the probability that the tour length exceeds a certain threshold to a given value. Campbell et al. (2011) address an orienteering problem with stochastic travel and service times where a reward is collected from a visited customer, and a penalty is

incurred for a customer that is not reached before a known deadline. They give heuristics for general problem instances and computational results for a variety of parameter settings. Papapanagiotou et al. (2013) investigate an orienteering problem with stochastic travel and service times and a given deadline. They focus on developing a Monte Carlo sampling procedure to approximate the objective function. They demonstrate the effectiveness of the objective approximation comparing to a exact objective evaluation. Evers et al. (2014) study an orienteering problem with a stochastic weight on each arc and a hard constraint on the total weight of a tour. They apply a sample average approximation to solve a two-stage stochastic model. A heuristic approach is developed to improve the computational efficiency. These papers consider stochastic travel time and service time that are independent among customers. However, our problem considers stochastic demand at each customer that affects the arrival time and service time in the other customers due to the queueing process at each customer.

1.2.4.3 Literature on Routing on a Network of Queues

The arrival of orders at a patient unit is a queueing process. We use Poission process to model the queueing at each patient unit. The phlebotomist routing problem is on a network of queuing. Literature of routing an individual through the queues focuses on the decision rules when the individual is in the queue. Yechiali (1969),Yechiali (1972), and Burnetas (2013) investigate the balking and reneging decision rules. Honnappa (2014) studies the arrival process in a network of queues, where

travelers choose when to arrive at a parallel queueing network and which queue to join upon arrival. Zhang et al. (2014, 2015) consider the team orienteering problem on a queue network on customers and consider the uncertain arrival time induced by the waiting time in the queues at previous customers. Our problem considers a network of queues but does not involve decisions of balking or joining rules. Instead, the orders arrive to a patient unit and are simply added to the queue. Furthermore, our problem routes the server to visit a sequence of queues, the queue length is changing while the server is serving the queue.

1.2.4.4 Literature on Polling Systems

The problem is considered in a polling system with a network of queues, where each phlebotomist is a server that visits a sequence of units and serves the queue of orders at the unit in a "first come first served" policy until the queue is empty. A polling model system consists of a number of queues, attended by a single server who visits the queues in some order to render service to the customers waiting at the queues. Customers arrive at the queues according to mutually independent homogeneous single Poisson arrival process. Levy et al. (1990), Vishnevskii and Semenova (2006), and Boon et al. (2011) conduct a survey of applications, modeling, and optimization methods of polling systems. Our problem routes the server to queues in the network and follow the exhaustive discipline. A phlebotomist is a server that fulfills the orders at a patient unit until the queue becomes completely empty. The quantity of orders at a patient unit depends on the evolution of the queue length during the

server visit. Literature regarding server routing considers the order the server visits the queues focuses on either static routing mechanisms or dynamic mechanisms.

1.2.4.5 Other Applications

While the focus of Chapter 4 is on the routing of phlebotomists at UIHC, this problem is applicable to several operations, such as scheduling/routing security staff in airport or railway systems. Yan and Gao (2010) consider the problem of routing of additional backup screening teams who help check in passengers besides stationing a regular security team at each departure gate, where each gate has a queue of passengers with random loads and arrival rates. The similarity is that the problem in Yan and Gao (2010) is on a network of queuing-based customers with stochastic demands and service times, and the difference is that Yan and Gao (2010) focuses on the problem of deploying the back up teams to the gates within a certain stretch of time to meet the potential demand with the smallest number of back up teams. In our problem, there is no regular servers at the patient units, and we consider the routing of serves/phlebotomists with the objective of maximize the number of customers served in a shift. Thorlacius et al. (2010) enable the construction of schedules for tickets inspectors in local urban trains in Copenhagen, Denmark, so that the income from penalty fares claimed from passengers without a valid ticket is maximized. The decision in Thorlacius et al. (2010) is to find out at which time and where to perform spot check ticket inspections in order to maximize the net revenue gained from the penalty fares. This problem can be also considered as a network

of queuing-based customers, however, it is a problem that schedules the inspectors regarding temporal, the spatial with the objective of maximizing the revenue, while our problem is a routing problem that routes technicians in a shift that maximizes the rewards of served customers.

CHAPTER 2
MAKESPAN MINIMIZING WORKFORCE ASSIGNMENT MODELS THAT RECOGNIZE HUMAN LEARNING

2.1 Introduction

This chapter focuses on workforce assignment problem that consider human learning. We assume a set of heterogeneous workers and a set of jobs that must be completed over the course of a given horizon. As workers work on particular type of job, they become more experienced and thus more productive. We assume that each individual worker has a known learning function that determines the worker's productivity for each unit of experience. Importantly, We allow the work on jobs to be split among workers and over time. The objective is to minimize the makespan to complete all of the jobs. We call this problem the makespan problem with learning.

The key challenge in solving the makespan problem with learning is that the learning functions are usually nonlinear. As a result, most work in the literature is limited to solving small-sized problems. This chapter presents a set of techniques that enable the solution of much larger instances of such problems.

In particular, this chapter presents three techniques that are contributions to the literature on solving makespan problems that explicitly model learning. First, we present an exact linear reformulation for the general makespan model with learning. While the reformulation technique is adapted from the literature, this chapter is the first to apply it to the learning function considered in this chapter and in the context of a makespan problem. Then, we introduce a computationally efficient means for

generating an initial feasible solution (which our computational experiments indicate is often near-optimal). We also present methods for strengthening the formulation with cover inequalities and a lower bound on the objective function value of the optimal solution. With an extensive computational study, we demonstrate the value of these techniques. To focus the chapter on the techniques, we solve a makespan problem that has few complicating constraints. However, the techniques can be adapted to speed up the solution of most any makespan problem. Further, the presented techniques do not depend on the particular learning and forgetting function.

This chapter is organized as follows. Section 2.2 first presents a nonlinear formulation of the makespan minimizing workforce assignment problem with learning. The section then introduces the linear reformulation of the problem, a way to generate initial feasible solutions, as well as the inequalities and bound that strengthen the formulation. Section 2.3 presents our datasets and computational experiments demonstrating the value of the reformulation, the initial solution, cover inequalities, and lower bound. Section 2.4 offers the conclusions and discusses future avenues of research.

2.2 Problem Formulation

In this section, we introduce a model for the makespan minimizing workforce assignment problem with learning. We first present the straightforward nonlinear formulation and then present the exact linear reformulations. We also introduce the procedure for generating initial feasible solution, the cover inequality, and the lower

bound to strengthen the reformulation.

We first introduce the notation that is used throughout the chapter, followed by a formal problem description and mathematical formulation.

Data

$\mathcal{W}$ Set of workers, $i = 1, \ldots, |\mathcal{W}|$.

$\mathcal{J}$ Set of jobs, $j = 1, \ldots, |\mathcal{J}|$.

$\mathcal{T}$ Set of periods in the horizon, $t = 1, \ldots, |\mathcal{T}|$.

v_j Volume of work required by job j, measured in units of standard work time, typically hours (standard work time units) of work when the work is carried out by an experienced worker, $j \in \mathcal{J}$

p_{ij} Initial experience of worker i on task j, in the same measurement of the job volume, $i \in \mathcal{W}, j \in \mathcal{J}$.

r_{ij} Individual learning rate of worker i on task j, $i \in \mathcal{W}, j \in \mathcal{J}$.

K_{ij} The maximum worker i's performance on task j when the learning process is concluded, given in the number of units produced per time period, $i \in \mathcal{W}, j \in \mathcal{J}$.

Variables

T_{max} Completion time of the last task to be finished.

x_{ijt} Binary variable that indicates whether task j is done by worker i in a period of time t, $\quad i \in \mathcal{W}, j \in \mathcal{J}, t \in \mathcal{T}$.

c_{ijt} The accumulated experience done by worker i for task j at the end of period t, measured in units of time periods performed by i on j by time t, $\quad i \in \mathcal{W}, j \in \mathcal{J}, t \in \mathcal{T}$.

φ_{ijt} Productivity of job j that worker i is expected to do in period t, $\quad i \in \mathcal{W}, j \in \mathcal{J}, t \in \mathcal{T}$.

We assume that a set $\mathcal{W}$ of workers works the jobs in $\mathcal{J}$ in a finite planning horizon $\mathcal{T}$. The completion of each job j in $\mathcal{J}$ requires a volume of work v_j. As in Corominas et al. (2010), we assume that the volume of work is measured in units of standard work time, typically hours (standard work time units) of work when the work is carried out by an experienced worker.

Each worker i in $\mathcal{W}$ has a learning function on each job j in $\mathcal{J}$. We let c_{ijt} be the accumulated experience of worker i on job j before period t. It is defined as the sum of the work time units done by worker i for job j by time t (see Equation (2.6)). The initial accumulated experience c_{ij1} is 0. Then, the amount of work completed in a unit of time by worker i on job j during time t will be a function of c_{ijt}:

$$\varphi_{ijt} = K_{ij}\left[1 - e^{-(c_{ijt}+p_{ij})/r_{ij}}\right], \tag{2.1}$$

where K_{ij} is an asymptotic parameter specifying the maximum productivity that can

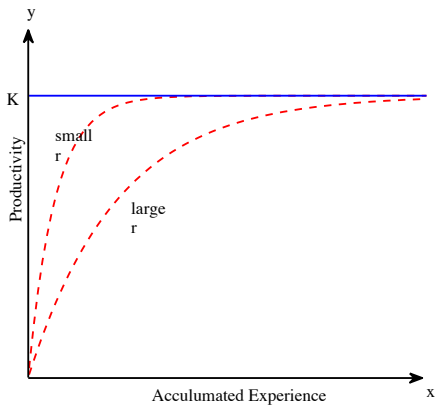

Figure 2.1: Three-parameter exponential learning curve by learning rate

be achieved by worker i on job j, learning rate r_{ij} is the time needed to reach 63% of the asymptote's production rate. The larger the value of r is, the slower is the rate of improvement in performance. (see Figure 2.1). Parameter p_{ij} corresponds to a worker's prior experience evaluated in units of standard work time units-typically hours of standard work time. The objective is to minimize the "makespan", or the completion of the last job, denoted as T_{max}.

2.2.1 Nonlinear Formulation

Given the above assumptions and definitions, we present the following Makespan Minimizing Workforce Assignment with Learning (MwL) model. A solution to this model prescribes an assignment for each worker i in $\mathcal{W}$ to a job j in $\mathcal{J}$ in period t in $\mathcal{T}$. We denote these assignments by the binary variables x_{ijt}. The MwL is:

$$Min \quad T_{max} \tag{2.2}$$

$$S.t \quad tx_{ijt} \leq T_{max} \quad i \in W, j \in J, t \in \mathcal{T} \tag{2.3}$$

$$\sum_{j \in J} x_{ijt} \leq 1, \quad i \in W, t \in \mathcal{T} \tag{2.4}$$

$$\sum_{i \in W} x_{ijt} \leq 1, \quad j \in J, t \in \mathcal{T} \tag{2.5}$$

MwL

$$c_{ijt} = \sum_{k=1}^{t-1} x_{ijk}, \quad i \in W, j \in J, t = 2, \ldots T \tag{2.6}$$

$$\varphi_{ijt} = K_{ij} \left(1 - e^{-(c_{ijt}+p_{ij})/r_{ij}}\right), i \in W, j \in J, t \in \mathcal{T} \tag{2.7}$$

$$\sum_{i \in W} \sum_{t \in \mathcal{T}} x_{ijt} \varphi_{ijt} \geq v_j, \quad j \in J \tag{2.8}$$

$$x_{ijt} \in \{0, 1\} \quad i \in W, j \in J, t \in \mathcal{T} \tag{2.9}$$

Equation (2.2) is the objective minimizing the makespan T_{max}. Constraints (2.3) define the makespan T_{max} to be the completion time of the last job. Constraints (2.4) enforce that a worker cannot work more than one job in a period. Constraints (2.5) limit only one worker can work on a job in any period. Constraints (2.6) define c_{ijt} as the accumulated experience of worker i for job j before t. Then, Constraints (2.7) uses this accumulated experience value to compute productivity during period t. Constraints (2.8) enforce that the volume of work associated with job j, v_j, be performed by the end of the horizon. Constraints (2.9) define the domain of the variables x_{ijt}.

We focus on a makespan minimizing workforce assignment model with few complicating constraints in this chapter to highlight the general applicability of the solution techniques we present next. The MwL can be modified to handle many

constraints such as adding ready times, due dates, and precedence constraints to jobs. However, the solution techniques we discuss next can still be used for each of these modified models.

2.2.2 Exact Linear Reformulation of MwL

In a preliminary work, we ran experiments in which we sought of to solve the nonlinear model MwL using the nonlinear solver Couenne. Couenne reached its node limit and could not return an integer solution even for a problem with two workers and two jobs with a time limit of 1800 seconds. As a result, we turn to other means of solving the instances of the proposed problem.

This section presents a reformulation that linearizes MwL, leaving a model that is easier to solve. We call this reformulation the Makespan Minimizing Workforce Assignment with Reformulated Learning (MwRL) model. Hewitt et al. (2014) introduce an analogous technique for a serial production line problem with human learning and prove that the reformulation is exact for the problem studied in their chapter. We introduce a proof for the reformulation in the context of makespan minimizing workforce assignment. The reformulation of MwL relies on the fact that job assignments, and thus the accumulated experience, are measured in discrete units. As a result, the nonlinear learning function can be linearized by enumerating its possible range values and associating a binary variable with each one. The result is that an optimal solution to the MwRL model is also an optimal solution to the MwL model.

We let $\hat{\varphi}_{ijt}^{l}$ be the productivity when worker i has l accumulated experience

doing job j up to time t. Note that, given i, j, t, and l, we can compute $\hat{\varphi}^l_{ijt}$ *a priori* as $\hat{\varphi}^l_{ijt} = K_{ij}\left(1 - e^{-(l+p_{ij})/r_{ij}}\right)$ (see Equation (2.7)) and thus treat it as a parameter to an optimization model. Recalling the definition of c_{ijt} in Constraints (2.6), a worker can gain at most $t-1$ units of accumulated experience on a job by t. Thus, there are at most $t-1$ possible values for each worker and each job for period t. We define binary variables z^l_{ijt} associated with each parameter $\hat{\varphi}^l_{ijt}$, where l is the accumulated experience of worker i up to time t on job j. With these new parameters and variables, the nonlinear Constraints (2.7) can be replaced by(2.10)-(2.12). Constraints (2.10) assign the productivity φ_{ijt} the appropriate pre-defined parameter value $\hat{\varphi}^l_{ijt}$ when $z^l_{ijt} = 1$, while Constraints (2.11) ensure that the l in the assigned pre-defined $\hat{\varphi}^l_{ijt}$ is equal to the accumulated experience c_{ijt}. Constraints (2.12) ensure that each individual is assigned only one productivity in each period. Given the redefinition of φ_{ijt} in Constraints (2.10), nonlinear Constraints (2.8) can be replaced by linear Constraints (2.13)-(2.14), where M_{ijt} is a big number. Specifically, we use $M_{ijt} = \hat{\varphi}^{t-1}_{ijt} = \max\{\hat{\varphi}^l_{ijt}, l = 0, 1, \ldots, t-1\}$; the maximum production rate worker i can achieve on job j in period t. We choose this value for M_{ijt} as it does not render any feasible solutions infeasible.

$$
\begin{aligned}
& Min \quad T_{max} \\
& S.t \quad (2.3)-(2.6),(2.9) \\
\mathrm{M}wRL \qquad & \varphi_{ijt} = \sum_{l=0}^{t-1} \hat{\varphi}_{ijt}^{l} z_{ijt}^{l}, \quad \forall i \in W, j \in J, t \in \mathcal{T}; && (2.10) \\
& \sum_{l=0}^{t-1} l z_{ijt}^{l} \leq c_{ijt}, \quad \forall i \in W, j \in J, \quad t \in \mathcal{T}, && (2.11) \\
& \sum_{l=0}^{t} z_{ijt}^{l} \leq 1, \quad \forall i \in W, j \in J, \quad t \in \mathcal{T}, && (2.12) \\
& \varphi_{ijt} \leq M_{ijt} x_{ijt}, \quad \forall i \in W, \quad j \in J, t \in \mathcal{T} && (2.13) \\
& \sum_{i=1}^{W} \sum_{t=1}^{T} \varphi_{ijt} \geq v_j, \quad \forall j \in J, && (2.14) \\
& z_{ijt}^{l} \in \{0,1\} \quad i \in W, j \in J, t \in \mathcal{T}, l = 0, \ldots, t-1. && (2.15)
\end{aligned}
$$

Theorem 2.1. *MwRL is an exact linear reformulation of MwL.*

Proof. First, note that the MwRL and MwL share the same objective function and Constraints (2.3)-(2.6) and (2.9). Further, Constraints (2.8) are equivalent to Constraints (2.13)-(2.14) that guarantee the volume of all jobs are finished. Second, Constraints (2.7) in MwL are equivalent to Constraints (2.10)-(2.12) in MwRL. To see this, note that the variable φ_{ijt} is a function of c_{ijt}. The variable c_{ijt} has at most $t-1$ possible values given the definition of c_{ijt} in Constraints (2.6). Thus, we can precompute $\hat{\varphi}_{ijt}^{l}, l = 0, \ldots, t-1$ for accumulated experience values of $0, \ldots, t-1$. For each i and j, Constraints (2.10) associate the appropriate $\hat{\varphi}_{ijt}^{l}$ with the accumulated experience c_{ijt} (by Constraints (2.11)). Constraints (2.12) enforce that only one productivity is assigned to a worker in one period.

Thus, because the MwRL has the same objective function as the MwL and either shares the same constraints or replaces constraints in the MwL with equivalent linearized constraints, the MwRL is an exact linear reformulation of MwL.

By Theorem (2.1), the MwRL is an exact reformulation of the MwL, but lacks the nonlinearities. While this linearization does make it possible to solve larger instances than are possible with the nonlinear formulation, the reformulation does require additional variables and constraints. When reforming MwL to MwRL, we add $|\mathcal{W}||\mathcal{J}|\sum_{t=1}^{T} t$ extra binary variables z_{ijt}^{l}, and $|\mathcal{W}||\mathcal{J}|\sum_{t=1}^{T} t$ extra parameters $\hat{\varphi}_{ijt}^{t}$. We have also added $3|\mathcal{W}||\mathcal{J}||\mathcal{T}|$ extra linear constraints. These extra variables and constraints mean that large instances of the MwRL are still challenging for commercial branch and bound solvers. In the remainder of this section, we discuss ways in which the computational burden of the MwRL can be reduced. Section 2.2.3 demonstrates how we generate initial feasible solutions to MwRL, while Section 2.2.4 introduces valid inequalities and a lower bound on the objective function value of MwRL.

2.2.3 Initial Feasible Solution

We note that MwRL requires definition of the set $\mathcal{T}$, and the instantiation of $\mathcal{T}$ requires knowing an upper bound on T_{max}. We generate this bound through a high quality initial feasible solution (IFS). We also use this IFS to improve our branch-and-bound approach. Knowledge of the objective function value of a high quality IFS to an integer program at the start of the solution process can reduce solution time in multiple ways. First, this upper bound on the optimal value of the optimization

problem can be used during prepossessing routines to further reduce the domains of variables. Second, during branch and bound, this upper bound can be used to fathom nodes for which the lower bound is too great. Finally, with an IFS, the solver does not need to spend time finding its own initial feasible solution.

To generate the IFS, we solve a makespan minimizing workforce assignment problem that recognizes human learning (like the MwRL) but does not allow jobs to be split. Specifically, a job can only be assigned to one worker, and once a worker starts working a job, he or she will complete it. We call the resulting problem the Makespan Minimizing Workforce Assignment Problem with Learning and No Splitting (MwLNS). Note that the MwLNS is a restriction of the MwRL; all the assignments that are feasible for the MwLNS are feasible for the MwRL, but the converse is not true. As such, solutions to the MwLNS can be mapped to solutions to the MwRL.

The MwLNS includes the binary decision variables $x_{ij}^{'}$ that indicate whether job j is assigned to worker i. Given the restriction that an individual complete a job once they begin it, we can calculate *a priori* the minimum number of time units the worker would need to complete the job, which we denote by pt_{ij}. Specifically, we calculate pt_{ij} by solving the optimization problem given in equation (2.16). We can solve this optimization problem by iterations, increasing the accumulated experience c_{ij} (starting at 0) until the productivity reaches or exceeds the volume of job j. Note that the fact that an individual learns is embedded in this optimization problem.

$$pt_{ij} = \min_{l}\{\sum_{c_{ij}=0}^{l} K_{ij}\left(1 - e^{-(c_{ij}+p_{ij})/r_{ij}}\right) \geq v_j, l \in Z^{+}\} \tag{2.16}$$

Using these pt_{ij} values, we formulate and solve the MwLNS to find an initial solution to the MwRL and a value for T_{max}.:

$$
\begin{aligned}
& Min \quad T_{max}^{MwLNS} \\
& S.t \quad \sum_{i \in W} x_{ij}^{'} = 1, \quad j \in J && (2.17) \\
(MwLNS) \qquad & \sum_{j \in J} x_{ij}^{'} pt_{ij} \leq T_{max}, \quad i \in W && (2.18) \\
& x_{ij}^{'} \in \{0,1\} \quad i \in W, j \in J, && (2.19)
\end{aligned}
$$

Commercial solvers can often solve instances of the MwLNS model almost instantaneously. This solution can then be transformed into a solution to the MwRL with a procedure we next describe with an example. Consider a case in which there are two workers and four jobs. Table 2.1 presents the processing time pt_{ij} for each worker i on each job i, accounting for the learning that takes place as worker i performs job j. Table 2.1 also present the assignments from an optimal solution of this instance of the MwLNS.

The solution shows that worker 1 is assigned to job 2 with a processing time of 2 periods and job 3 with a processing time of 3 periods. Worker 2 performs job 1, which takes 1 period and job 4, which takes 4 periods. The objective value (makespan) of this example is 5 periods.

We create a solution to the MwRL by ordering the jobs performed by worker i in ascending order of processing time pt_{ij}(see Table 2.2). This solution also yields an objective value of 5 periods and does not violate any constraints of the MwRL.

pt_{ij}	Job 1	Job 2	Job 3	Job 4
Worker 1	2	2	3	5
Worker 2	1	3	4	4
$x_{ij}^{'}$	Job 1	Job 2	Job 3	Job 4
Worker 1	0	1	1	0
Worker 2	1	0	0	1

Table 2.1: Parameters and solutions of MwLNS for an example of two workers and four jobs

	x_{ijt}	Period 1	Period 2	Period 3	Period 4	Period 5
Worker 1	Job 2	1	1	0	0	0
	Job 3	0	0	1	1	1
Worker 2	Job 1	1	0	0	0	0
	Job 4	0	1	1	1	1

Table 2.2: Transform into solutions x_{ijt} of MwRL

We can then feed this transformed solution to an optimization solver as the starting point for its search for an optimal solution to the MwRL.

2.2.4 Strengthening the Formulation

While overcoming the nonlinearities of the learning curve improves the tractability of the model, we can also strengthen the formulation using valid inequalities and a lower bound on the objective function value of the optimal solution. By strengthening the formulation, a mixed integer programming solver is likely to require much less time to solve instances of the problem.

2.2.4.1 Valid Inequality

First, we present a cover inequality that is added to MwRL to strengthen the bound produced by its linear relaxation. Consider Constraint (2.8) and derive valid inequality for MwRL. We can weaken the constraint $\sum_{i\in W}\sum_{t\in\mathcal{T}} x_{ijt}\varphi_{ijt} \geq v_j$ by replacing the variable φ_{ijt} with the value $\bar{\varphi}_j = \max_i \hat{\varphi}_{ijT}^{T-1}$, where $\hat{\varphi}_{ijT}^{T-1} = \max\{\hat{\varphi}_{ijT}^{l}, l = 0, 1, \ldots, T-1\}$ is the maximum production rate of a worker i on a job j over the entire time horizon. The resulting constraint is:

$$\sum_{i\in W}\sum_{t\in\mathcal{T}} \bar{\varphi}_j x_{ijt} \geq v_j. \tag{2.20}$$

Dividing both sides by $\bar{\varphi}_j$ and round the right hand side to the ceiling integer since the sum of the binary variables is integral. As such, we have the cover inequality:

$$\sum_{i\in W}\sum_{t\in\mathcal{T}} x_{ijt} \geq \left\lceil \frac{v_j}{\bar{\varphi}_j} \right\rceil, \quad \forall j \in J. \tag{2.21}$$

2.2.4.2 Lower Bound

One can also strengthen a formulation by adding a lower bound on the optimal objective value. To derive such a bound, we solve an integer program that is similar in form to MwL, but much simpler to solve. Assume all the workers have the maximum possible productivity on all jobs at all time periods $\hat{\varphi}_{ijt}^{t-1}$. We remove all the nonlinearities in MwL and call the following program the Makespan Minimizing Workforce Assignment with Maximum Learning (MwML) model.

$$
\begin{aligned}
& Min \quad T_{max}^{MwML} \\
& S.t \quad (2.3) - (2.5), (2.9) \\
(MwML) \qquad & \sum_{i \in W} \sum_{t \in \mathcal{T}} x_{ijt} \hat{\varphi}_{ijt}^{t-1} \geq v_j, \quad \forall j \in J
\end{aligned} \tag{2.22}
$$

The MwML is a relaxation of MwL (some constraints are removed, some are weakened by replacing variables with their maximum values). We can solve the MwML prior to solving the MwRL to provide a global bound on the optimal objective function value. We add the constraint:

$$T_{max}^{MwML} \leq T_{max} \tag{2.23}$$

to the MwRL. We can generate additional bounds during the execution of the branch and bound by solving the MwML at different nodes in the branch-and-bound tree with some variables in the MwML fixed to values determined by branching decisions.

2.3 Computational Experiments and Results

In this section, we present the results of computational experiments that we performed to test the effectiveness of our formulations and techniques.

2.3.1 Experimental Setting

We generate 11 classes of instances, each class differentiated by the number of workers and tasks. Our goal in generating the sets is to test to limits of the proposed solution methodology rather than to recreate instances reflecting any particular environment. The classes range from 5 workers, 10 tasks up to 20 workers, 30 tasks. To

give perspective on the sizes of these instances, recall that Corominas et al. (2010) solve a close variant of the MwRL in which instances of only five tasks and four workers were solved via an approximation (a piecewise linearization) scheme. We detail the instance sizes, with respect to the number of workers, $|\mathcal{W}|$, and the number of jobs, $|\mathcal{J}|$ in Table 2.3.

$\lvert\mathcal{W}\rvert$	$\lvert\mathcal{J}\rvert$
5	10, 15
10	10, 15, 20
15	15, 20, 25
20	20, 25, 30

Table 2.3: Instance sizes

For each class, we generate nine cases for a total of 99 instances. Each case is defined by the learning traits of the workers (asymptote parameter K, prior experience p, and learning rate r) and the job volumes. We sample the learning traits data based on the empirical data found in Mazur and Hastie (1978). Mazur and Hastie (1978) collect performance-related results from various experimental settings, including students writing letters and factory workers making cigars. In our dataset, the asymptote parameter is generated from the interval $[8, 10]$, and the prior experience is assumed to be 0.5, which is the minimum experience for a worker to start on a job. The learning rate is classified as "fast", "medium", or "slow," with the sampling intervals for each classification given in Table 2.4. Figure 2.2 illustrates these curves at the midpoint of their respective range. The job volumes also define the problem

	Notation	Settings		
Asymptote	K	[8, 10]		
Prior experience	p	0.5		
		Fast	Medium	Slow
Learning rate	r	$[0.5, 1]$	$[2, 4]$	$[5, 8]$
		Short	Medium	Long
Volume of work	v	$\left[\frac{\lvert\mathcal{W}\rvert}{\lvert\mathcal{J}\rvert}, 5\frac{\lvert\mathcal{W}\rvert}{\lvert\mathcal{J}\rvert}\right]$	$\left[6\frac{\lvert\mathcal{W}\rvert}{\lvert\mathcal{J}\rvert}, 10\frac{\lvert\mathcal{W}\rvert}{\lvert\mathcal{J}\rvert}\right]$	$\left[11\frac{\lvert\mathcal{W}\rvert}{\lvert\mathcal{J}\rvert}, 15\frac{\lvert\mathcal{W}\rvert}{\lvert\mathcal{J}\rvert}\right]$

Table 2.4: Instance design

and a job volume is classified as "short", "medium", or "long" job with the integer values in intervals in Table 2.4.

For each combination of $|\mathcal{W}|, |\mathcal{J}|$, nine cases are generated, with the cases varying according to worker learning rates and job volumes in Table 2.5. For example, all workers in case 1 are fast learners whose learning rates are randomly sampled from $[0.5, 1]$, while the job volumes are randomly drawn from the integers in $[|\mathcal{W}|/|\mathcal{J}|, 5|\mathcal{W}|/|\mathcal{J}|]$. Other cases are generated similarly. Overall, these datasets are chosen to test the computational effectiveness of the reformulations and other model enhancements and not to necessarily reflect the settings found in any particular practical setting. The datasets are available from http://ir.uiowa.edu/tippie$_p$*ubs*/66/.

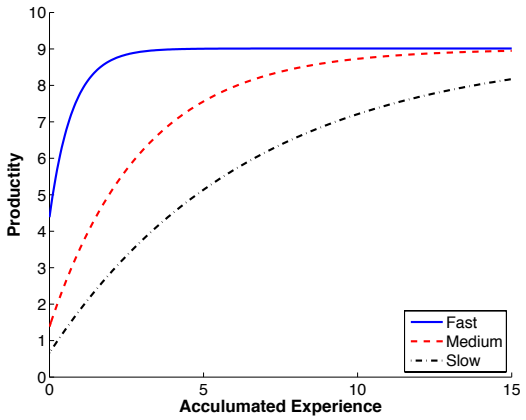

Figure 2.2: Learning curves with different learning rates

Case	Learning rate	Job volume
1	Fast	Short
2	Fast	Medium
3	Fast	Long
4	Medium	Short
5	Medium	Medium
6	Medium	Long
7	Slow	Short
8	Slow	Medium
9	Slow	Long

Table 2.5: Instance cases

In the experiments, we seek to understand the computational performance of the integer programs and the proposed speed-up techniques. First, we seek to understand to what degree the solvability of MwRL scales with respect to the number of workers and jobs. Next, we seek to understand whether the solution time for an instance of the MwRL is related to the case used to create it. Then, we seek to understand whether the speed-up techniques have an impact.

All the experiments presented in the next section were performed on a computer cluster where each node has 32-64 cores with AMD Opteron 2.2 GHz or AMD Interlagos 2.6 bulldozer processors and 128-256 GB of memory. All instances were solved using Gurobi version 5.2, and all Gurobi parameters were left at their default values, other than the time limit, which was set to 600 seconds.

	Case								
$\lvert\mathcal{W}\rvert - \lvert\mathcal{J}\rvert$	1	2	3	4	5	6	7	8	9
5-10	0.00	0.00	1.00	0.00	2.00	2.00	0.00	2.00	5.00
5-15	0.00	0.00	0.00	0.00	4.00	3.00	1.00	5.00	0.00
10-10	0.00	0.00	0.00	0.00	1.00	1.00	0.00	1.00	2.00
10-15	0.00	0.00	0.00	0.00	1.00	3.00	0.00	2.00	0.00
10-20	0.00	0.00	1.00	0.00	3.00	5.00	1.00	7.00	0.00
15-15	0.00	0.00	0.00	0.00	1.00	5.00	0.00	4.00	0.00
15-20	0.00	0.00	0.00	0.00	1.00	3.00	1.00	4.00	0.00
15-25	0.00	0.00	2.00	1.00	5.00	7.00	1.00	5.00	0.00
20-20	0.00	0.00	0.00	0.00	2.00	0.00	1.00	7.00	0.00
20-25	0.00	1.00	1.00	0.00	1.00	6.00	1.00	5.00	0.00
20-30	0.00	0.00	1.00	0.00	2.00	6.00	1.00	0.00	0.00
Average	0.00	0.09	0.55	0.09	2.09	3.73	0.64	3.82	0.64

Table 2.6: Abs. gaps when solving MwRL

2.3.2 Computational Results

We first focus on the computational tractability of the MwRL. All times reported are in seconds. Because the makespan values for these instances are often relatively small (less than 20), we report absolute gaps as opposed to relative gaps. As such, we report in Table 2.6 the average gap at termination for each instance of the MwRL that Gurobi tries to solve. An entry of "0.00" indicates that Gurobi was able to solve the instance. Similarly, we report in Table 2.7 the time Gurobi requires to reach termination, either because it solved the instance or it reached the 600 seconds time limit.

First, we observe that Gurobi solves 41.41% of the instances (41 out of 99). We next observe that it is not the size of the instance that impacts Gurobi's ability to solve it, but the case used to generate it. This can also be seen in the following

	Case								
$\lvert\mathcal{W}\rvert - \lvert\mathcal{J}\rvert$	1	2	3	4	5	6	7	8	9
5-10	0.29	8.11	600.00	4.12	600.00	600.00	8.45	600.00	600.00
5-15	6.30	6.76	7.72	0.97	600.00	600.00	600.00	600.00	600.00
10-10	0.38	16.35	28.61	18.70	600.00	600.00	274.72	600.00	600.00
10-15	9.01	23.06	111.25	15.62	600.00	600.00	463.55	600.00	600.00
10-20	14.01	18.67	600.00	17.38	600.00	600.00	600.00	600.00	600.00
15-15	1.97	20.60	80.16	11.00	600.00	600.00	331.51	600.00	600.00
15-20	6.73	14.23	56.78	9.81	600.00	600.00	600.00	600.00	600.00
15-25	11.75	29.55	600.00	600.00	600.00	600.00	600.00	600.00	600.00
20-20	4.65	40.01	129.53	39.26	600.00	600.00	600.00	600.00	600.00
20-25	7.14	600.00	600.00	21.97	600.00	600.00	600.00	600.00	600.00
20-30	11.91	63.32	600.00	27.17	600.00	600.00	600.00	600.00	600.00
Average	6.74	76.42	310.37	69.64	600.00	600.00	479.84	600.00	600.00

Table 2.7: Times when solving MwRL

figures. Figure 2.3 presents the average time Gurobi requires for all instances of a given size and Figure 2.4 the average time for all instances of a given case. We see that runtimes are fairly constant across instance sizes, but differ greatly by case. Gurobi can (relatively) easily solve instances from Cases 1, 2, 3, 4, and 7. Examining the results further, we see that Gurobi can quickly solve instances that either have workers that possess a fast learning rate or jobs that have a short volumes.

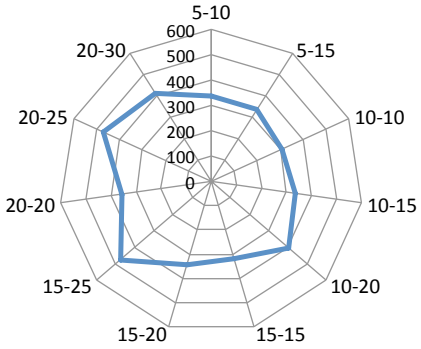

Figure 2.3: Average run-time for instances by $|\mathcal{W}| - |\mathcal{J}|$

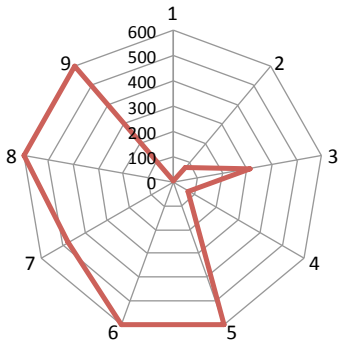

Figure 2.4: Average run-time for instances of a given case

We next turn our attention to the impact of strengthening the formulation (cover inequalities and lower bound) and the IFS proposed earlier. Table 2.8 indicates for each method name which techniques were used. We first display in Figure 2.5 the average runtimes when the cover inequalities are and are not used, by case. We see that using the cover inequalities is quite effective as they decrease the runtime in every case. On average, the runtime with the cover inequalities is 74.90% of the runtime than without. Similarly, while 41.41% of the instances were solved within the 600 second time limit when the cover inequalities are not used, 56.57% of instances are solved when it is. On the other hand, Figure 2.6 indicates that using the lower bound as well as the cover inequalities does not reduce run-times.

We next consider whether there is an improvement when an IFS is used along with the cover inequalities. Figure 2.7 compares solve times when an IFS is and is not used along with the cover inequalities. We see that by providing an IFS to Gurobi, runtime is reduced significantly. To be precise, averaged over all instances,

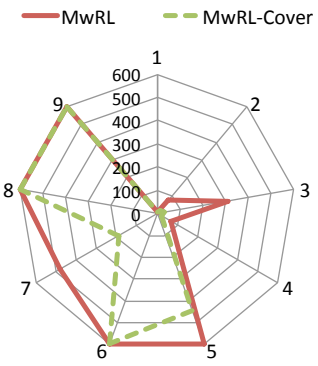

Figure 2.5: Average run-time for instances of a given case when using Cover-inequality

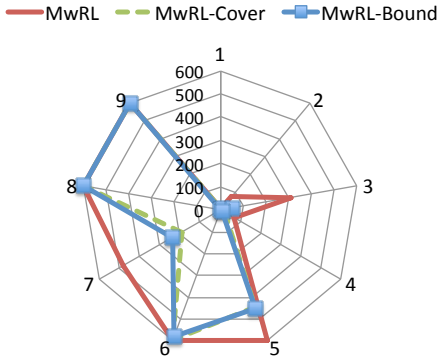

Figure 2.6: Average run-time for instances of a given case when using Cover-ineq7ality and Lower bound

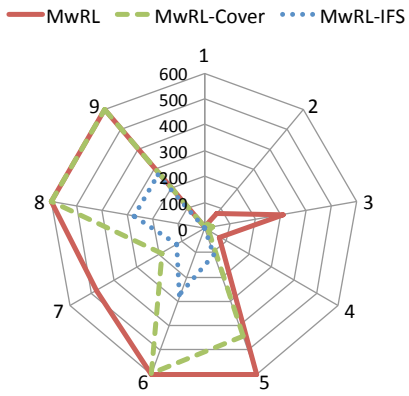

Figure 2.7: Average run-time for instances of a given case when using7Cover-inequality and IFS

Method	Cover ineq.	Lower bound	IFS
MwRL-Cover	Y	N	N
MwRL-Bound	Y	Y	N
MwRL-IFS	Y	N	Y
MwRL-Bound-IFS	Y	Y	Y

Table 2.8: Method configuration

the run time when an IFS and the cover inequalities is used is 42.09% of the run time when only the cover inequalities are used. Similarly, while using the cover inequalities increases the percentage of instances solved to 56.57%, using an IFS in addition to the cover inequalities increases that percentage to 80.81%. We also note that the pattern in the average runtime by case is the same when an IFS is used as when it is not. Both approaches still struggle the most with instances from Cases 6, 8, and 9. Not surprisingly those are the cases where the fewest instances are solved for both MwRL-Cover and MwRL-IFS.

We next return to considering the use of a lower bound, and report by method and case the average of the absolute gap reported at termination. We note that using the lower bound in conjunction with the IFS has little to no impact on runtime. However, comparing the lines MwRL-IFS and MwRL-Bound-IFS we see that using both the lower bound and the IFS leads to the (provably) highest-quality solution at termination.

We next seek to understand why using an IFS is so effective. To that effect, Table 2.10 reports, averaged over all instances for a given case, the proportion of the number of binary variables that are in the MwRL-Bound-IFS formulation after

	Case									
Method	1	2	3	4	5	6	7	8	9	Avg.
MwRL	0.00	0.09	0.55	0.09	2.09	4.10	0.64	4.20	3.50	1.45
MwRL-Cover	0.00	0.00	0.00	0.00	1.18	2.73	0.27	2.57	2.75	0.85
MwRL-Bound	0.00	0.00	0.00	0.00	0.91	2.27	0.27	2.70	3.25	0.86
MwRL-IFS	0.00	0.00	0.00	0.00	0.18	0.73	0.18	0.55	1.45	0.34
MwRL-Bound-IFS	0.00	0.00	0.00	0.00	0.18	0.64	0.18	0.45	0.82	0.25

Table 2.9: Average absolute gap by method and case

Gurobi performs preprocessing to the number of binary variables that are in the MwRL formulation after preprocessing. With the IFS, Gurobi is able to preprocess (fix to 0 or 1) many of the binary variables that are in the original MwRL formulation. Not surprisingly, the cases (6, 8, and 9) that we have observed to be the hardest are the ones where Gurobi is able to preprocess the fewest numbers of variables.

	Case								
	1	2	3	4	5	6	7	8	9
% binary var	15.04	34.38	57.95	31.84	58.47	75.74	55.94	76.70	87.96

Table 2.10: Fraction of MwRL variables in MwRL-Bound-IFS formulation after preprocessing

Similarly, Table 2.11 presents the absolute gap between the objective function value of the IFS and the lower bound reported by Gurobi at termination. We then average these gaps over all instances created with the same case. Not surprisingly, we see a fairly clear relationship between the quality of the IFS and Gurobi's ability to preprocess variables; the better the quality of the IFS, the more variables Gurobi

is able to preprocess.

	Case								
	1	2	3	4	5	6	7	8	9
Abs. gap	0.00	0.00	0.36	0.00	0.18	0.64	0.64	0.45	0.82

Table 2.11: Average absolute gap between value of initial solution and final lower bound

The quality of the IFS is also related to the case used to generate the instance. In particular, for instances where the learning rate is Fast and the job volume is Short or Medium, the IFS is optimal. Similarly, when the learning rate is Medium and the job volume is Short, the IFS is optimal. In a sense, this is not surprising. The IFS is generated by solving a restriction of the MwL in which each job is assigned to exactly one worker. This may be less of a restriction when workers learn very quickly or the job has a Short volume. In both cases, the job can be completed quickly enough that there is no reason for a worker not to finish it.

Finally, recalling that the largest instances of models of this sort that have been solved in the literature (and that was with a piecewise-linear approximation scheme) consisted of four workers and five jobs, we repeat Tables 2.6 and 2.7 only for when MwRL-Bound-IFS is solved (Tables 2.12 and 2.13). Table 2.13 shows that all instances up to 10 workers and 10 jobs are solved instantaneously, and it is not until instances with 15 workers that the time limit of 600 seconds is consistently hit. However, again it depends on the case. For the cases in which workers learn at a fast rate, instances of all sizes are solved in seconds. Yet, from Table 2.12, we see that,

$\lvert\mathcal{W}\rvert - \lvert\mathcal{J}\rvert$	Case								
	1	2	3	4	5	6	7	8	9
5-10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
5-15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
10-10	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
10-15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	2.00
10-20	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
15-15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
15-20	0.00	0.00	0.00	0.00	1.00	3.00	1.00	1.00	2.00
15-25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	1.00
20-20	0.00	0.00	0.00	0.00	1.00	2.00	0.00	0.00	0.00
20-25	0.00	0.00	0.00	0.00	0.00	1.00	1.00	1.00	2.00
20-30	0.00	0.00	0.00	0.00	1.00	3.00	1.00	1.00	2.00
Average	0.00	0.00	0.00	0.00	0.27	1.00	0.27	0.45	0.82

Table 2.12: Abs. gaps when solving MwRL-Bound-IFS

even when the time limit is hit, the solutions found at that point are typically of high quality as they are only two units away from the best known bound.

2.4 Conclusions and future work

In this paper, we study optimization techniques for makespan minimizing workforce assignment problems wherein human learning is explicitly modeled. In contrast to the commonly-studied problem in which a job is only performed by one individual and hence learning can be encoded in the parameters of the problem, we instead study problems wherein learning must be explicitly represented within the optimization model. To date, the challenge with such models has come from the fact that quantitative models of how humans learn are nonlinear, leaving optimization problems that are computationally intensive to solve for all but the smallest instances.

	Case								
$\lvert\mathcal{W}\rvert - \lvert\mathcal{J}\rvert$	1	2	3	4	5	6	7	8	9
5-10	0.06	1.15	0.13	0.51	0.10	0.13	1.87	0.28	0.31
5-15	0.56	0.52	1.21	2.05	0.34	0.45	0.18	0.23	0.29
10-10	1.27	0.11	0.19	1.12	0.18	0.18	0.20	0.19	0.22
10-15	0.23	0.17	6.10	0.29	0.40	600.00	5.96	600.00	600.00
10-20	1.00	1.01	0.25	0.92	0.41	0.69	4.71	1.23	0.61
15-15	0.26	0.55	0.68	0.24	0.26	0.32	0.37	0.52	0.29
15-20	0.12	0.27	6.15	0.85	600.00	600.00	5.20	600.00	600.00
15-25	0.34	0.35	1.47	0.45	0.86	600.00	600.00	600.00	600.00
20-20	0.26	0.34	0.46	0.60	0.69	0.84	1.15	0.36	0.52
20-25	0.25	0.55	6.71	0.63	1.76	600.00	600.00	600.00	600.00
20-30	0.74	0.78	31.61	0.70	600.00	600.00	88.25	600.00	600.00
Average	0.46	0.53	5.00	0.76	109.55	272.96	118.90	272.98	272.93

Table 2.13: Times when solving MwRL-Bound-IFS

To overcome the nonlinearity, we use an exact reformulation technique which replaces the nonlinear function used to model learning with binary variables and linear constraints. As such we transform the nonlinear optimization problem to a mixed integer program (MIP), a type of optimization problem that commercial solvers are capable of solving for many instance sizes. We then develop speed-up techniques for this MIP. Namely, we derive a procedure for creating an initial feasible solution for the MIP, a valid inequality for the formulation, and a procedure for producing a lower bound on the optimal value of the MIP.

With an extensive computational study, we show that our techniques can solve larger instances in a much faster speed than has been seen in the literature. We also illustrate the effectiveness of the proposed speed-up techniques as well as analyze why they are effective. Finally, we present computational results that show that our new

formulation can often be solved in mere seconds. Ultimately, we believe this paper sets the standard, computationally-speaking, for how one solves makespan minimizing workforce assignment problems that explicitly recognize human learning.

There are many human learning activities that are not captured in the problems we study. For example, psychologists have claimed that, by training, workers not only gain experience but can also fundamentally change their learning rates. Also, our problem assumes that jobs are distinct from a learning perspective. In other words, performing one job does not improve the productivity of a worker on another. However, oftentimes there is similarity between jobs (they may share common subjobs). We intend to explore both of these factors in future work. Finally, we intend to incorporate an explicit representation of human learning into more complicated scheduling problems, such as those that have releasing time, due dates, precedence relationship, or have a more complicated structure, such as flow shops.

Funding

This material is based upon work supported by the National Science Foundation under Grant No. CMMI-1266010.

CHAPTER 3
WORKER GROUPING AND ASSIGNMENT WITH LEARNING-BY-DOING AND KNOWLEDGE TRANSFER

3.1 Introduction

Recall that Chapter 2 considers individual learning. When working in a team environment, not only does an individual learn from direct first hand experience, but also, potentially, from knowledge transferred from others in the team. We now consider the case in which workers working in proximity can transfer knowledge. Knowledge transfer occurs when individuals work in close proximity with team members who are doing the same or related jobs.

This work is motivated by Nembhard and Bentefouet (2015), who first propose the concept of knowledge transfer among workers in the same team and is also the first to propose a mathematical representation of it. The mathematical representation in Nembhard and Bentefouet (2015) is an extension of the 3-parameter hyperbolic learning curve by letting the cumulative experience be a function of both direct and indirect experience. Direct experience, D, is measured as the number of repeated time a person performs a job. Indirect experience, S, is measured as the output of others working on jobs that are in close enough proximity for an individual to learn from their work. However, not all indirect experience is transferred. Instead, S is scaled by a parameter θ that represents an individual's ability to learn via transfer. Thus parameter, θ, (a percentage) represents what fraction of indirect experience is transferred from workers to the individual. We use Nembhard and Bentefouet (2015)'s

learning model that incorporates knowledge transfer as in Equation (3.1):

$$\varphi = K(\frac{\theta \times S + D + p}{\theta \times S + D + p + r}). \tag{3.1}$$

Equation (3.1) describes the production output φ as a function of cumulative experience $\theta \times S + D$ in units of time or trials. The three parameters of the function are: (1) K, which defines the asymptote limit of the production output φ, (2) p, corresponds to a worker's prior experience evaluated in the same units as x, and, (3) r, the learning rate.

Nembhard and Bentefouet (2015) propose several heuristic approaches for selecting a workforce from a pool, grouping the workforce into teams, and assigning teams to jobs. Based on their computational study, they identify the top policies that yield the maximum throughput for each decision.

This chapter considers the workforce allocation problem, including grouping workers into teams based on individual learning characteristics, and assigning teams to different sets of jobs. Unlike Nembhard and Bentefouet (2015), we do not include the selection decision because workforces are fixed in general. We assume a set of workers and a set of jobs in a given time horizon for parallel systems. Each individual worker has a known learning function that determines the worker's productivity for each unit of experience. We also account for workers knowledge gain transferred from their team members. The objective is to maximize the system throughput over the horizon.

The challenges of this problem lie in the nonlinearity of the learning curve that

incorporates knowledge transfer, and the large number of variables and constraints. There are three contributions in this chapter: First, we build a mixed integer nonlinear program (MINLP) for parallel production systems. Second, by exploiting the structure and characteristics of the optimal solutions, we develop an exact approach that linearizes the MINLP to a mixed integer linear program (MILP). This reduction in complexity allows us to solve workforce allocation problems with different problem settings. Third, with the extensive computational results, we provide a solution structure analysis and give managerial insights for managers when making grouping and assignment decisions.

The remainder of this chapter is organized as follows. Section 3.2 presents the MINLP for parallel systems. Section 3.3 reformulates this MINLP to a MILP. Section 3.4 presents the experiment design and summarizes the results of the experiments as well as insights gained from solutions. Section 3.5 offers the conclusions of this chapter.

3.2 Problem Description and Mathematical Model

We first present a description of the problem followed by a mathematical model of the problem.

In this chapter, we consider an organization that seeks to develop a workforce plan for a parallel system over a finite planning horizon. In this parallel system, jobs have already been partitioned into different types, and jobs of the same type are in close enough proximity in the production facility to facilitate the transfer

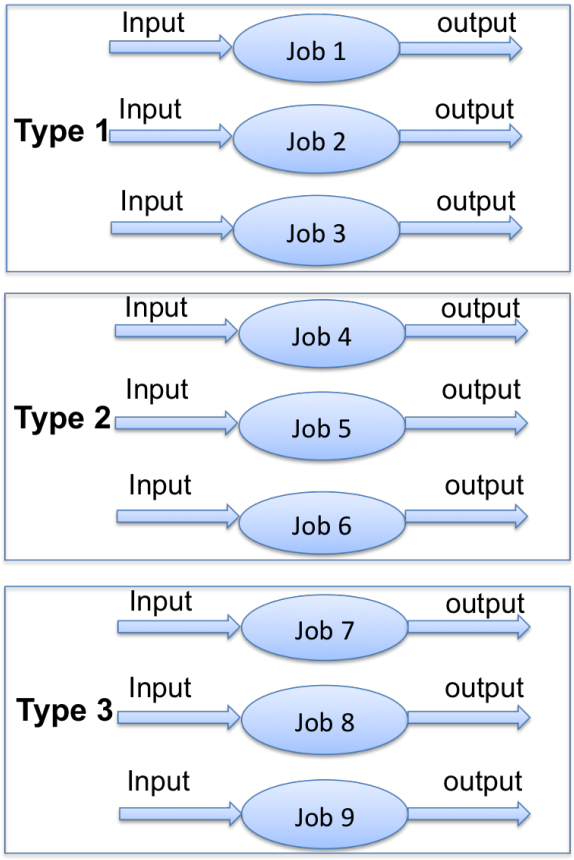

Figure 3.1: An example of parallel system

of knowledge. We illustrate a parallel system in Figure 3.1, wherein nine jobs are grouped into three types. The goal of the organization is to team workers to groups, and then groups to jobs, in order to maximize overall productivity. We consider a setting wherein each job can be performed by at most one worker, and each worker can perform at most one job.

To model this problem, we assume a horizon of T periods, and let the set $\mathcal{T} = \{1, 2, \ldots, T\}$ index the set of periods in the horizon. We define $\mathcal{I} = \{1, 2, \ldots, I\}$ to be a set of workers that perform jobs in set $\mathcal{J} = \{1, 2, \ldots, J\}$. With this discrete

planning horizon, we model the assignment of individuals to jobs with the binary variable x_{ij}^t, which indicates whether worker i performs job j in period t. Associated with each pair of a worker and a job is the learning parameters that determine the worker's ability to learn the job. Referring to equation (3.1), we let K_{ij}, p_{ij}, and r_{ij} denote the asymptote parameter, prior experience, and learning rate of worker i on job j. We note that with these parameters we are explicitly modeling that the workforce is heterogeneous with respect to productivity and learning.

Jobs are partitioned into G different types, and $\mathcal{G} = \{1, 2, \ldots, G\}$ is the set of these job types. Jobs in type $g \in \mathcal{G}$ are a subset of jobs in $\mathcal{J}$. Jobs of the same type are grouped together so that workers working them are close enough to transfer knowledge. We assume that the ability of a worker to absorb knowledge from her/his colleagues' accumulated experience does not depend on the jobs that colleagues are performing. As such, associated with each worker is the transfer parameter $\theta_i \in [0, 1]$, which represents the fraction of knowledge transferred to worker i from her/his coworkers that are working on jobs in the same job type.

For each period t in the horizon, we let c_{ij}^t denote the accumulated work experience of worker i for task j up to t. As discussed previously, the accumulated work experience c_{ij}^t is a function of direct and indirect experience. The direct experience is the number of periods in which worker i perform j by time t, i.e. $D_{ij}^{t'} = \sum_{t'<t} x_{ij}^{t'}$.

Regarding indirect experience, for worker i in period t, we denote the experience of colleagues that impacts i's productivity on job j as S_{ij}^t. We calculate this as the total production output of all colleagues on jobs that are in the same group as

j. As such, we define the binary indicator f_{jk} as taking on the value 1 when jobs j, k are the same type and the value 0 otherwise. Thus, we have the following calculation of S_{ij}^t :

$$S_{ij}^t = \sum_{i' \in \mathcal{I}: i' \neq i} \sum_{k \in J: k \neq j} \sum_{t' < t} f_{jk} \varphi_{i'k}(c_{i'k}^{t'}), \quad i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T} \tag{3.2}$$

Thus, we calculate the accumulated experience of worker i on job j in period t with the following equation:

$$c_{ij}^t = \theta_i S_{ij}^t + D_{ij}^t, \quad i \in \mathcal{I}, j \in \mathcal{J}, g \in \mathcal{G}, t \in \mathcal{T} \tag{3.3}$$

We calculate the productivity of worker i on job j in period t as:

$$\varphi_{ij}(c_{ij}^t) = K_{ij} \frac{c_{ij}^t + p_{ij}}{c_{ij}^t + p_{ij} + r_{ij}}, \quad i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T} \tag{3.4}$$

Recall that we presume that jobs have already been partitioned into types such that knowledge can be transferred between individuals working on jobs of the same type. As such, we model the choice of putting workers into groups (or teams), with each group of workers assigned to a single job type. Note we also presume that a worker can work only on jobs within his group's assigned job type. Thus, there is a one-to-one mapping between groups/teams of workers and job types.

To model these choices, we define the binary variable y_{ig} to indicate whether worker i is assigned to group/job type $g \in \mathcal{G}$. We use the following constraints to enforce that a worker is assigned to only one group (Constraints 3.5), and only works

on jobs within their assigned type during the planning horizon (Constraints 3.6):

$$\sum_{g \in \mathcal{G}} y_{ig} = 1, \quad i \in \mathcal{I} \tag{3.5}$$

$$\sum_{j \in g} x_{ij}^{t} \leq y_{ig}, \quad i \in \mathcal{I}, g \in \mathcal{G}. \tag{3.6}$$

We presume a job can be performed by only one worker in a period, which we enforce with the following constraint:

$$\sum_{i \in \mathcal{I}} x_{ij}^{t} \leq 1, \quad j \in \mathcal{J}, t \in \mathcal{T} \tag{3.7}$$

Finally, the objective is to maximize the system production at the end of the horizon (which we often refer to as system throughput). We let the continuous variable o_{ij}^{t} represent worker i's output on job j in period t. However, their output on a job in a period is limited both by their productivity and whether they performed that job in that period. Both restrictions are modeled in the following constraint:

$$o_{ij}^{t} \leq x_{ij}^{t} \varphi_{ij}^{t}(c_{ij}^{t}), \quad i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T}. \tag{3.8}$$

Ultimately, with these variable and constraint definitions, we define the following non-linear program for assigning workers to groups and groups to job types in a parallel system. We refer to this model as "PM" :

$$
\begin{aligned}
Max \quad & \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{t\in\mathcal{T}} o_{ij}^{t} \\
S.t \quad & \sum_{i\in\mathcal{I}} x_{ij}^{t} \leq 1, \quad j\in\mathcal{J}, t\in\mathcal{T} && (3.9)\\
& \sum_{g\in\mathcal{G}} y_{ig} = 1, \quad i\in\mathcal{I} && (3.10)\\
(PM) \quad & \sum_{j\in g} x_{ij}^{t} \leq y_{ig}, \quad i\in\mathcal{I}, g\in\mathcal{G} && (3.11)\\
& o_{ij}^{t} \leq x_{ij}^{t}\varphi_{ij}(c_{ij}^{t}), \quad i\in\mathcal{I}, j\in\mathcal{J}, t\in\mathcal{T} && (3.12)\\
& \varphi_{ij}(c_{ij}^{t}) = K_{ij}\frac{c_{ij}^{t}+p_{ij}}{c_{ij}^{t}+p_{ij}+r_{ij}}, \quad i\in\mathcal{I}, j\in\mathcal{J}, t\in\mathcal{T} && (3.13)\\
& c_{ij}^{t} = \theta_i S_{ij}^{t} + D_{ij}^{t}, \quad i\in\mathcal{I}, j\in\mathcal{J}, t\in\mathcal{T} && (3.14)\\
& S_{ij}^{t} = \sum_{i'\in\mathcal{I}:i'\neq i}\sum_{k\in J:k\neq j}\sum_{t'<t} f_{jk}\varphi_{i'k}(c_{i'k}^{t'}), \quad i\in\mathcal{I}, j\in\mathcal{J}, t\in\mathcal{T} && (3.15)\\
& D_{ij}^{t} = \sum_{t'<t} x_{ij}^{t'}, \quad i\in\mathcal{I}, j\in J, t\in\mathcal{T} && (3.16)\\
& o_{ij}^{t} \geq 0, \quad i\in\mathcal{I}, j\in\mathcal{J}, t\in\mathcal{T}. && (3.17)\\
& x_{ij}^{t}, y_{ig} \in \{0,1\}, \quad i\in\mathcal{I}, j\in\mathcal{J}, t\in\mathcal{T}, g\in\mathcal{G} && (3.18)
\end{aligned}
$$

3.3 Reformulation of the MINLP

Given the state of today's off-the-shelf optimization solvers, the nonlinear nature of "PM" renders it computationally challenging to solve for instances of moderate-to-large size. However, by analyzing the structure present in optimal solutions to "PM" we can reformulate it to a mixed integer program. Specifically, we extend the analysis of Nembhard and Bentefouet (2012), who show that in a parallel system and under some conditions, the optimal schedule of a worker task assignment problem with individual learning is a schedule with specialized workers. In other words, workers

perform the same job during the planning horizon. However, their result does not address a situation wherein knowledge transfer occurs. We show (Proof in Appendix A) that the result also extends to situations wherein there is knowledge transfer, or, precisely, that:

Theorem 3.1. *In parallel systems, when the number of workers is equal to the number of jobs with a monotonically non-decreasing learning function, the optimal schedule is a schedule with specialized workers. In other words, no workers switch jobs within the horizon and workers are assigned to a single task.*

Proof in Appendix A.

The conclusion of Theorem 3.1 enables us to simplify "PM" in two ways: (1) we can ignore the time component, and, (2) we can remove the nonlinearity through an enumeration of all possible assignments of workers to groups (and hence job types). Regarding the time component, as a worker will perform the same job during the planning horizon, we can remove the index t from the variables $x_{ij}^t, S_{ij}^t, D_{ij}^t, c_{ij}^t$, and φ_{ij}^t.

Regarding the enumeration, we enumerate the possible assignment of workers to jobs in each job type. For a job-type g consisting of n jobs, we let W_g denote the set of all possible permutations of n workers, e.g. $w = \{i_1, i_2, \ldots, i_n\}, w \in W_g$. Note that we derive from the ordering of the individuals in w the assignment of a worker in w to a job in g. In other words, with $w = \{i_1, i_2, \ldots, i_n\}$ and $g = \{j_1, j_2, \ldots, j_n\}$ we assign i_1 to perform j_1, i_2 to perform j_2, etc. We then calculate the throughput of this group of workers, w, when assigned to group g, which we denote as O_w^g, as

follows:

$$
\begin{aligned}
O_w^g &= \sum_{l=1}^{n} o_{i_l j_l} \\
&= \sum_{l=1}^{n} \sum_{t \in \mathcal{T}} K_{i_l j_l} \frac{c_{i_l j_l}^t + p_{i_l j_l}}{c_{i_l j_l}^t + p_{i_l j_l} + r_{i_l j_l}} \\
&= \sum_{l=1}^{n} \sum_{t \in \mathcal{T}} K_{i_l j_l} \frac{t + p_{i_l j_l} + \sum\limits_{q=1:q\neq l}^{n} \sum\limits_{r=1:r\neq l}^{n} \sum\limits_{t'<t} \varphi_{i_q j_r}(c_{i_q j_r}^{t'})}{t + p_{i_l j_l} + \sum\limits_{q=1:q\neq l}^{n} \sum\limits_{r=1:r\neq l}^{n} \sum\limits_{t'<t} \varphi_{i_q j_r}(c_{i_q j_r}^{t'}) + r_{i_l j_l}}.
\end{aligned} \tag{3.19}
$$

Then, to reformulate "PM" to a mixed integer program we introduce the binary variable z_w^g to indicate whether the group of workers $w \in W_g$ is assigned to the job type g. The resulting reformulation, which can be seen as a set packing problem, is as follows:

$$
(RPM) \qquad
\begin{aligned}
Max \quad & \sum_{g \in \mathcal{G}} \sum_{w \in W_g} O_w^g z_w^g && (3.20)\\
& \sum_{w \in W_g} z_w^g \leq 1, \quad g \in \mathcal{G} && (3.21)\\
& \sum_{g \in \mathcal{G}} \sum_{w \in W_g : i \in w} z_w^g \leq 1, \quad i \in I && (3.22)\\
& z_w^g \in \{0,1\}, \quad w \in W_g, g \in \mathcal{G}. && (3.23)
\end{aligned}
$$

The objective (3.20) maximizes the throughput of the system. Constraints (3.21) ensure that at most one group is assigned to each job type. Similarly, constraints (3.22) ensure that each worker is assigned to at most one group. Finally, constraints (3.23) define the domain of the variables. Note that whereas Nembhard and Bentefouet (2015) assume a homogeneous workforce with respect to learning pa-

rameters (K, p, r), we are able to model a heterogeneous workforce without increasing the complexity of RPM.

3.4 Experimental Design and Computational results

In this section, we describe the computational study we performed to address the following issues/questions. First, we evalute the benefits an organization can see from recognizing that knowledge transfer occurs when developing their workforce plans. Second, we assess the capabilities of solving the reformulated MIP (RPM) as a planning process. To do so, we first benchmark the quality of the solutions it produces against those produced by the heuristic policies presented in Nembhard and Bentefouet (2015). Then, we perform a computational study to determine how large an instance of the RPM we can solve within a given time limit. Third, we study how the (optimal) solutions to the RPM differ, structurally, from those prescribed by the heuristics proposed in Nembhard and Bentefouet (2015). We next describe the settings that goven how our computational experiments were performed.

3.4.1 Experimental Setting

All results presented in this section were derived from experiments that were performed on a computer cluster wherein each node has 16 2.6 GHz cores and 4 GB of memory. All instances were solved using Gurobi version 6.5, and all Gurobi parameters were left at their default values, other than the time limit, which was set to 3600 seconds.

As the datasets used vary according to the issue we are studying, we defer

$$\mu = \begin{bmatrix} 1.448 \\ 1.963 \\ 2.0446 \end{bmatrix} \begin{matrix} log(K) \\ log(p) \\ log(r) \end{matrix} \qquad \Sigma = \begin{matrix} \\ log(K) \\ log(p) \\ log(r) \end{matrix} \begin{matrix} \begin{matrix} log(K) & log(p) & log(r) \end{matrix} \\ \begin{bmatrix} 0.0045 & 0.0167 & 0.0191 \\ 0.0167 & 0.4621 & 0.2443 \\ 0.0191 & 0.2443 & 0.1905 \end{bmatrix} \end{matrix}$$

Table 3.1: Mean and variance-covariance matrix for K, p, and r

description of them until the description and analysis of each issue. However, each instance includes a workforce, and a worker in our model is defined by his/her learning traits (asymptote productivity rate K, learning rate r, prior experience p, and knowledge transfer parameter θ). In all instances, the first three traits are determined for each individual in a workforce by drawing from normal univariate distributions with means and variance-covariances given in Table 3.1. These distribution parameters have been used in Nembhard and Bentefouet (2015), and are originally from Shafer et al. (2001). The transfer parameter θ is assumed to be normally distributed, with mean 0.664 and variance 0.409. These are the same distribution and parameters as used in Nembhard and Bentefouet (2015).

3.4.2 The Impact of Knowledge Transfer on Throughput

We first study the extent by which recognizing that knowledge transfer occurs when determining workforce groups and plans impacts throughput. We study this on a set of 100 instances, each with nine workers and nine jobs. For each instance we generate two solutions (groups of workers, and assignments of groups to job types)

and compare the throughput yielded by each solution. The first solution is derived by solving RPM, thus this solution is derived when explicitly recognizing that knowledge transfer occurs. We refer to the throughput associated with such a solution as obj_{kt}.

The second solution is derived in a manner that mimics when knowledge transfer is not considered when deriving a workforce plan. To derive such a solution, we again solve RPM but do so after setting each individual's knowledge transfer parameter (θ_i) to zero. Setting $\theta_i = 0, \forall i \in \mathcal{I}$ essentially turns off recognition of knowledge transfer when deriving the solution. However, knowledge transfer does occur and can impact throughput even if it is not recognized when deriving the plan. Thus, to evaluate the throughput, we evaluate this solution in RPM, albeit with the knowledge parameter for each individual set to with the parameter value. We refer to this throughput as obj_{no-kt}.

Figure (3.2) shows a clear difference in the objective values of models with and without knowledge transfer on the 100 nine-worker datasets. We see that over the 100 instances the average gap in througput (measured as $(obj_{kt} - obj_{no-kt})/obj_{kt}$) is 27.18%. We illustrate the distribution of these gaps in Figure 3.3. Both clearly indicate that recognizing that knowledge transfer occurs when determining a grouping of workers and assignment of groups to job types can have a significant impact on throughput. In other words, managers should take into account the impact of the knowledge transfer between team workers when making grouping and assignment decisions. Failing to consider knowledge transfer leads to underestimation of the workforce capacity.

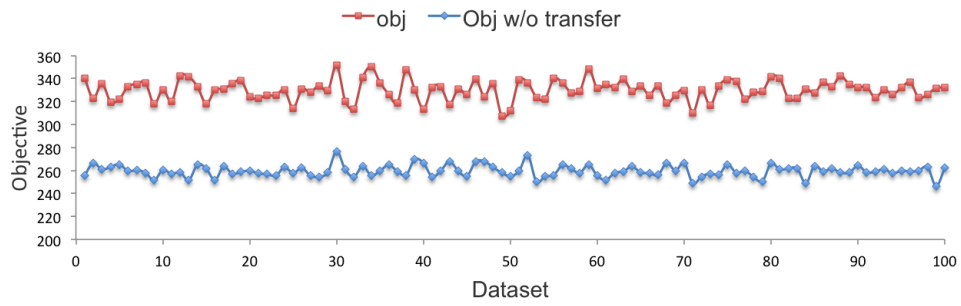

Figure 3.2: System Throughput on the 100 Datasets

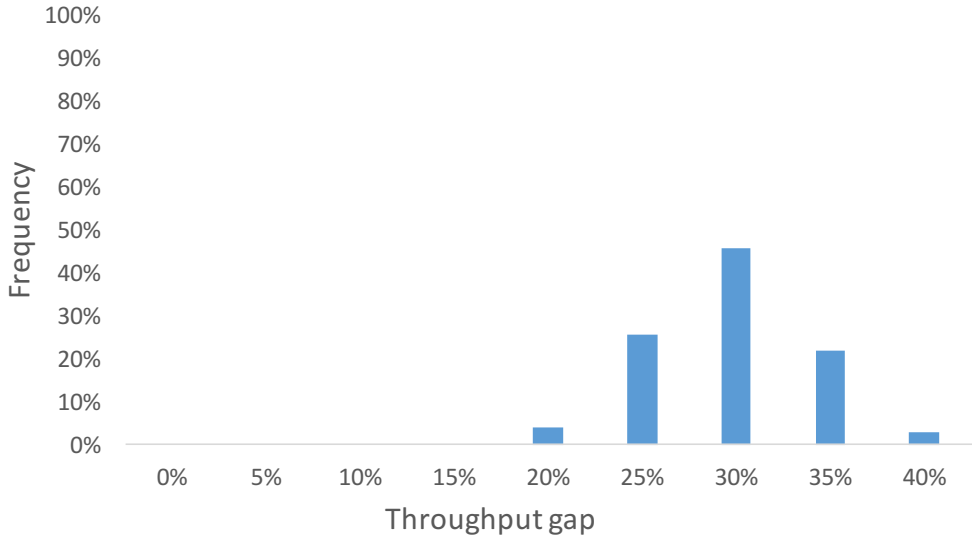

Figure 3.3: Distribution of throughput gaps when recognizing knowledge transfer and not.

Having validated the merit in recognizing knowledge transfer, we next turn to assessing the capabilities of solving the RPM to develop workforce plans while considering that knowledge transfer occurs.

3.4.3 Comparing RPM to Nembhard and Bentefouet (2015)

In this section, we assess the solutions of RPM by comparing the solution quality and solution structure to the solutions from Nembhard and Bentefouet (2015)'s heuristic policies.

3.4.3.1 Solution Quality

Nembhard and Bentefouet (2015) propose a class of heuristics for three levels of decision-making: selection (select workers from a pool), grouping workers, and assigning groups of workers to job-types. These heuristics differ from each other in terms of which instance/worker attributes are used by the heuristic to inform decisions regarding grouping and selection. Out of this class of heuristics they propose/identify what they conclude to be the 10 best (for reference, we list these in Table B.1 in Appendix B).

As our proposed reformulation only prescribes the grouping and assignment decisions, we benchmark the throughput produced by the plans proposed by solving RPM against these 10 heuristics, but we assume the workforce has already been selected and thus "skip" the selection step in their heuristic. We then benchmark the throughput yieled by solving the RPM against the best (largest) of the ten througputs yielded by these 10 heuristics over the same 100 instances used for the previous com-

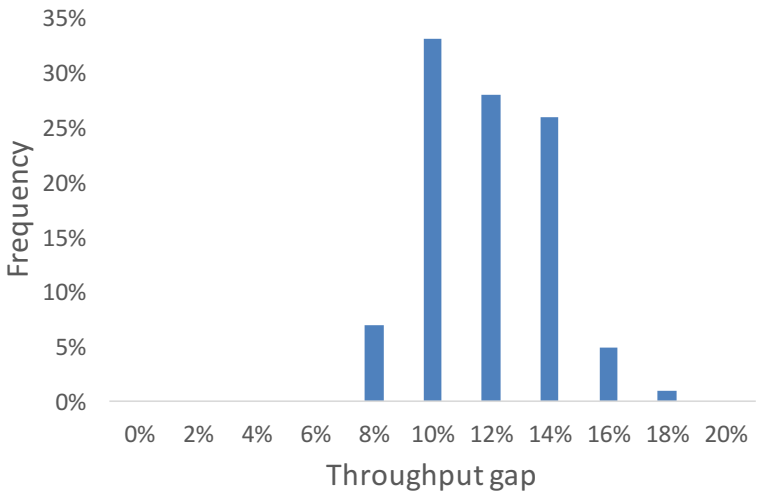

Figure 3.4: Distribution of throughput gaps between solving RPM and Nembhard's best heuristic

parison. Formally, with obj_N^k representing the througput produced by the k^{th} heuristic proposed by Nembhard and Bentefouet (2015) and obj_{RPM} the througput produced by solving the RPM, we calculate $gap = (obj_{RPM} - \max_{k=1}^{10} obj_N^k)/obj_{RPM}$. The average of these gaps is 12.85%. We also display the distribution of these gaps in Figure 3.4. We conclude that solving the RPM produces significantly higher througput than the heuristics proposed by Nembhard and Bentefouet (2015).

Having observed that plans prescribed by the RPM produce significantly higher throughput, we next seek to understand the robustness of a planning approach that includes solving the RPM. Specifically, we seek to understand how the time needed to solve an instance of the RPM changes as the number of workers, groups, and/or group sizes changes. We first focus on situations where workers are to be paired (i.e. groups consist of two workers). In Figure 3.5, we illustrate the

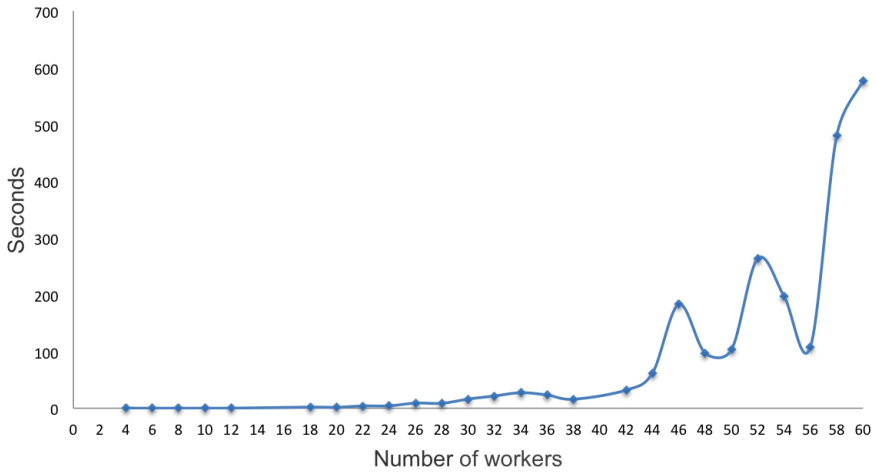

Figure 3.5: Time to solve RPM by number of workers when pairing workers

time required to solve the RPM for different number of workers. For each number of workers, we consider 10 instances and illustrate the average solve time over those ten instances. We see that the solve time grows in the number of workers, but even when putting 60 workers into pairs, instances can be solved in less than 10 minutes.

We next study how easily the RPM can be solved when there are larger groups (e.g. group sizes ranging from three to six). For each combination of number of workers and group size, we consider 10 instances. In Figure 3.6, we report the average time required to solve instances of the RPM when workers are put into groups of size three and size four as the number of workers changes. We again see that for a fixed group size, the solve time increases as the number of workers increases. However, the solve time remains relatively short (under four minutes) for all settings. However, when considering groups of five workers, we note that we were only able to solve instances with 15 workers in 3,600 seconds. When considering groups of six workers,

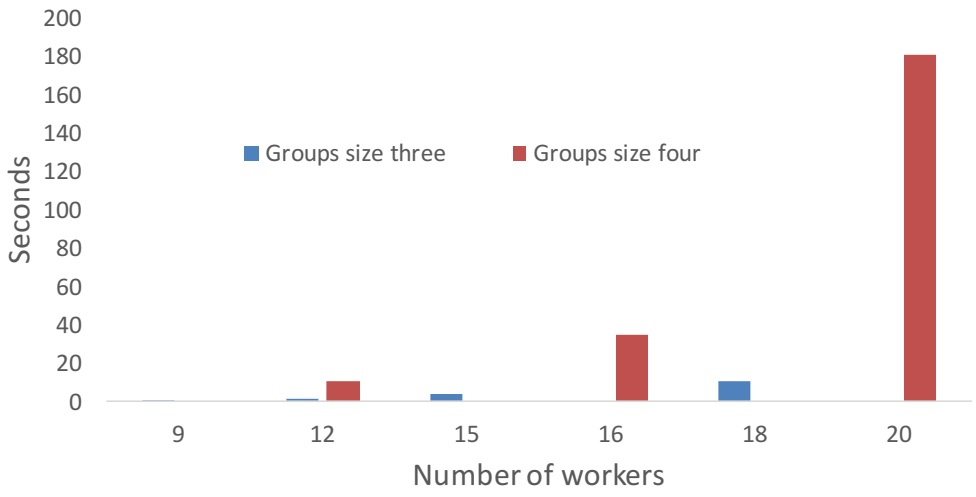

Figure 3.6: Time to solve RPM by number of workers for larger groups

we were unable to solve any instances in the time limit.

3.4.3.2 Structure of Solutions

We next turn our attention to how the solutions prescribed by solving the RPM differ from those prescribed by the heuristics of Nembhard and Bentefouet (2015). Specifically, we first focus on how the groupings of workers are different. To group workers, the heuristics proposed by Nembhard and Bentefouet (2015) seek to group workers that have similar values for one of the learning attributes $(K, p, 1/r, \theta)$. Thus, we next study whether or not the groupings chosen by solving the RPM have such a structure. We let σ_w^α denote the standard deviation associated with the different values of the learning attributes α in the group of workers w. We then calculate $\sigma_{max}^\alpha = \max_{w \in \mathcal{W}} \sigma_w^\alpha$ and $\sigma_{min}^\alpha = \min_{gw \in \mathcal{W}} \sigma_w^\alpha$. In other words, for an attribute α, σ_{max}^α

represents the largest standard deviation over all possible groupings of workers and σ^{α}_{min} the smallest.

We use these values to determine the degree to which the individuals in group w are dissimilar with respect to learning attribute α. Specifically, for group w and attribute α we calculate $dissim^{\alpha}_{w} = (\sigma^{\alpha}_{max} - \sigma^{\alpha}_{w})/(\sigma^{\alpha}_{max} - \sigma^{\alpha}_{min})$. $dissim_w$ takes on a value between 0 and 1, with a value of 1 implying that w is a grouping of very similar individuals with respect to learning attribute α (or at least as similar as possible given the workforce) and 0 implying that w is a grouping of very dissimilar individuals. Then, given a set of groups, $\mathcal{W}_{RPM}$ prescribed by a solution to the RPM, we calculate a dissimilarity index with respect to α, $dissim^{\alpha}_{RPM}$, as $\sum_{w \in \mathcal{W}_{RPM}} dissim_w / |\mathcal{W}_{RPM}|$. We report averages of $dissim^{\alpha}_{RPM}$ over the 100 instances tested and the four learning attributes considered in Table 3.2. What we see in this table is that, in fact, it is better to group individuals who are dissimilar (as the indices are greater than 50%) than similar, which is in stark contrast to how the heuristics proposed by Nembhard and Bentefouet (2015) operate.

	K	p	$1/r$	θ
Avg. $dissim^{\alpha}_{RPM}$	70.12%	74.80%	79.17%	68.83%

Table 3.2: Average dissimilarlity index

We next turn our attention to the assignment phase. Recall that in this phase two decisions must be made: (1) which group of workers should be assigned

to each job type, and, (2) given the workers $w = (w_1, w_2, \ldots, w_n)$ in the group assigned to the job type with jobs $(j_1, j_2, \ldots, j_n)$, which worker should perform each job. We focus on analyzing the first decision. Specifically, consider workers $w = (w_1, w_2, \ldots, w_n)$ and job-type $g = (j_1, j_2, \ldots, j_n)$. Let $\alpha_{w_i}^{j_{i'}}$ represent worker w_i's value for attribute α on job $j_{i'}$. We let $\alpha_{max}^{wg} = \max_{i=1}^{n} \max_{i'=1}^{n} \alpha_{w_i}^{j_{i'}}$. Then, with $\mathcal{W}_{RPM}$ representing the groups selected in the solution to RPM and $\mathcal{G}$ the job-types in the instance we calculate $\alpha_{max} = \max_{w \in \mathcal{W}_{RPM}} \max_{g \in \mathcal{G}} \alpha_{max}^{wg}$. Similarly, we calculate $\alpha_{min} = \min_{w \in \mathcal{W}_{RPM}} \min_{g \in \mathcal{G}} \alpha_{max}^{wg}$. Finally, with w_g representing the group of workers assigned to job-type g in a solution to the RPM, we calculate $\alpha_{RPM} = \max_{g \in \mathcal{G}} \alpha_{max}^{w_g g}$. A greedy assignment policy would first assign the group of workers w to the job-type g that maximized α_{wg} (or the group w_g such that $\alpha_{max}^{w_g g} = \alpha_{max}$). To measure whether solutions to the RPM exhibit the same property, we calculate a greedy index, $greedy_{RPM}^{\alpha} = (\alpha_{RPM} - \alpha_{min})/(\alpha_{max} - \alpha_{min})$. $greedy_{RPM}^{\alpha}$ will take on a value between 0 and 1, with the closer the value is to 1, the more greedily the assignment stage was performed. We report averages of $greedy_{RPM}^{\alpha}$ over the 100 instances tested and three learning attributes in Table 3.3.

	K	p	$1/r$
Avg. $greedy_{RPM}^{\alpha}$	85.87%	52.63%	31.18%

Table 3.3: Average greedy index

We conclude from these results that whether the assignment of groups to job-

types should be done in a greedy fashion depends on the attribute that is being considered. When focusing on steady-state productivity (K), groups should be assigned to job-types in a greedy fashion. However, when focusing on learning rate ($1/r$), a greedy approach is less effective.

The structure analysis of the grouping stage gives a sense of the common learning and knowledge transfer characteristics shared by workers who are grouped in the same team. The structure analysis of the assignment stage provides a greedy approach of assigning groups to job types. The grouping sense and the greedy assignment approach can help managers to make good grouping and assignment decisions without the expensive computational time of solving the model optimally.

3.5 Conclusions and Future Work

This chapter studies the operational decision making process of grouping and assignment of workers to jobs, while accounting for individual learning and knowledge transfer within groups. We build a mixed-integer nonlinear program (MINLP) for parallel systems. Nembhard and Bentefouet (2015) develop some heuristic policies, while this chapter provides several solution methods for the MINLP. First, we provide an exact solution method that solves the models optimally. Specifically, we present a reformulated model that is exact to the original MINLP and solve the reformulated model optimally. Second, we propose some heuristic policies that perform better than Nembhard and Bentefouet (2015) based on the computational results.

Our computational experiments explore the impact of knowledge transfer on

the system throughput. The results demonstrate that the system benefits greatly from knowledge transfer between team workers. This indicates that managers should take into account the impact of the knowledge transfer between team workers when making grouping and assignment decisions and failing to consider knowledge transfer leads to underestimation of the workforce capacity. We also test the solvability of the reformulation on both pairing problems and larger problems. Pairing problems can be solved optimally within 110 seconds for problems up to 52 workers and 600 seconds for problems with 60 workers. Larger problem settings up to 18 workers when there are five or fewer workers in a team can be solved within 2000 seconds.

Next, we compare the solution quality and solving speed between our reformulation and Nembhard's heuristic policy. Our computational experiments have shown that the reformulated model achieve 12.85% higher system throughput than the top policies of Nembhard and Bentefouet (2015). Additionally, we compare the solution structure between the solutions from reformulation and the solutions by the heuristics of Nembhard and Bentefouet (2015) with respect to the learning and knowledge transfer parameters. In the grouping stage, the comparison shows that workers should be grouped with workers that are the most different from them which maximizes the within-team standard deviation of the parameters. The result conflicts the grouping policies in Nembhard and Bentefouet (2015). In the assignment stage, the results show that a set of jobs should be performed by a team of workers that has the highest parameter values on this set of jobs. The result is consistent with Nembhard and Bentefouet (2015)'s assignment policies. These greedy heuristics can be used when

the exact approach takes too long to solve a problem.

In future work, we plan to build mathematical models for serial systems and design exact approaches to solve the problem, since serial system is the base for many systems. Another direction of future work is exploring the solution structure statistically with respect to the common characteristics shared by workers who are grouped in the same team. We hope that the statistical results can help managers make better grouping and assignment decisions without the expensive computational time of solving the model optimally. Lastly, the current reformulation can only solve relatively large problems. Integer techniques for larger size problems need to be studied.

CHAPTER 4
INPATIENT PHLEBOTOMIST ROUTING PROBLEM

4.1 Introduction

Laboratories are health care facilities where pathologists provide testing of patient samples. Laboratory services account for more than 10% of hospital billing. The laboratory workforce includes phlebotomy technicians (phlebotomists) who draw samples from patients, and technicians who perform tests on these samples.

The shortage of hospital laboratory personnel continues to be of concern for many laboratories. This shortage often causes tardiness of fulfilling some sample draws, which affects decisions on admission, discharge, and medication of inpatients. With the projected need for more hospital laboratory personnel in the coming years coupled with an aging workforce, effective management of the clinical laboratory workforce is essential to meet the demand for staffing in the near future and beyond (Hamilton and Sm; 2014). This study focuses on phlebotomist routing that aims to improve the laboratory performance in terms of fulfilling more sample draws in a shift. The phlebotomist routing in this chapter focuses on designing routes of visiting the patient rooms for a team of phlebotomists. The goal of these routes is to facilitate in a timely manner as many sample draws as possible. We call this the phlebotomist intra-hospital routing problem.

This study is motivated by the Department of Pathology at the University of Iowa Hospitals and Clinics (UIHC). At UIHC, there is a team of phlebotomists that

works in the morning and another team works later in the day. We focus on the morning shift as workload often cannot be completed on time in the morning shift. Typically, they work from 05:30 to 09:30 and serve 27 patient units located in different buildings during their shift. This team of phlebotomists performs approximately 40% of the daily draws at UIHC. We assume an order corresponds to a sample draw from a patient. These draws are ordered by doctors randomly during the day.

When phlebotomists arrive at work at 05:30, they see the current outstanding orders. These orders are certain and called "pre-orders." Orders that arrive randomly between 05:30 and 09:30 are "add-ons." There are approximately 180 to 250 orders requested from the 27 units in a morning shift. Less than 5% of the orders are add-ons in a typical morning. An example of orders on Oct 10th, 2015 is illustrated in Figure 4.1.

Phlebotomists typically perform two types of draws at UIHC: simple venipuncture and blood culture. The service time of a draw is uncertain, though a rough estimate of an order's service time can be determined based on the service type. For example, a simple venipuncture typically takes six to eight minutes, while a blood culture can take 17 to 25 minutes. Only 5% of these 180 to 250 orders are blood cultures. These 180 to 250 orders approximately correspond to 50 to 70 hours of work per day. There are 8 to 16 phlebotomists available for a morning shift. The orders are usually not entirely fulfilled by the end of the shift at 09:30. This research uses the average as the service time of an order type. In other words, it takes 7 minutes to serve a simple venipuncture and 21 minutes to fulfill a blood culture.

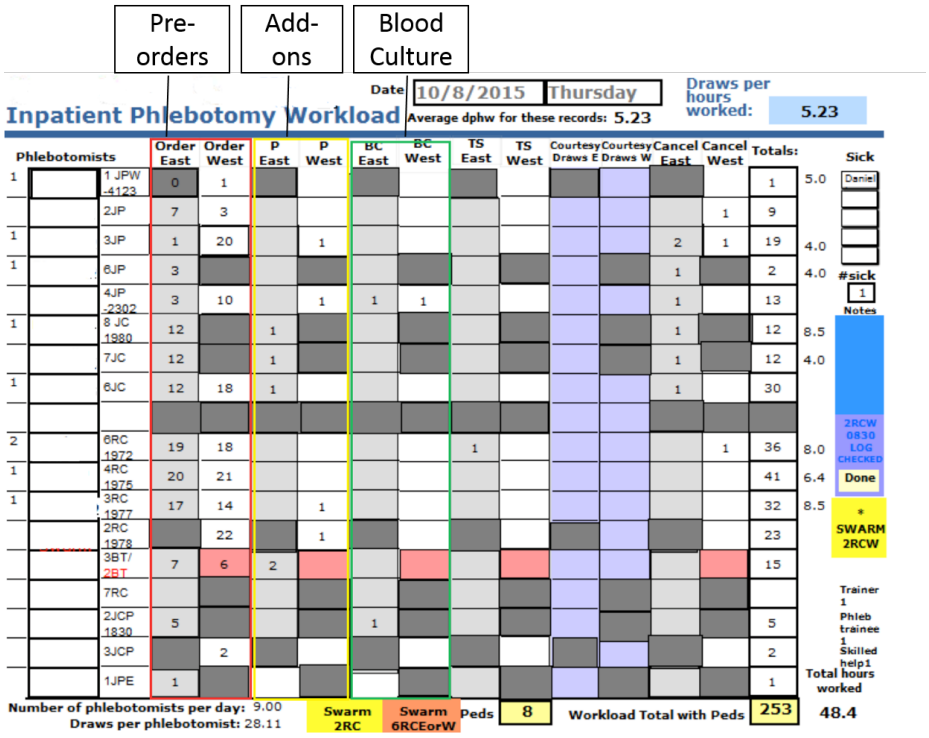

Figure 4.1: One day example of orders

Figure 4.2: The map of UIHC

Given the time required to travel between units, the units are divided based on the three "arms" of UIHC: Roy Carver Pavilion, John Colloton Pavilion, and John Pappajohn Pavilion. Figure 4.2 shows a map of UIHC with these three arms [1]. Table 4.1 lists the 27 patient units. In general, it takes about three and a half minutes to walk between any two nearby arms, while traveling up or down floors generally takes about one minute per floor using the elevator. Travel time between units that are on the same floor of the same building are less than half a minute. These travel times are essential when routing each phlebotomist to a sequence of patient units.

NO.	Unit	NO.	Unit
0	laboratory	14	IPCU-5 Carver
1	1 Pappajohn East	15	4 Pappajohn East
2	1 Pappajohn West	16	4 Pappajohn West
3	2 Boyd Tower	17	4 Carver East
4	2 Colloton P	18	4 Carver West
5	2 Pappajohn East	19	6 Colloton East
6	2 Pappajohn West	20	6 Colloton West
7	2 Carver	21	6 Pappajohn
8	3 Boyd Tower	22	6 Carver East
9	3 Colloton East	23	6 Carver West
10	3 Colloton West	24	7 Colloton East
11	3 Pappajohn West	25	7 Carver West
12	3 Carver East	26	8 Colloton
13	3 Carver West	27	4 Colloton

Table 4.1: 27 patient units at UIHC

At 05:30 on a given day, the available phlebotomists are known and the pre-

[1] https://www.uihealthcare.org/a-z-directory/

orders of each unit are observed. The add-ons of each unit arrive randomly between 05:30 and 09:30. The arrival of add-ons can be thought of as the arrivals to a queueing process. Note that add ons are occurring even when a phlebotomist is in a unit serving patients. The later the phlebotomist arrives and the longer that it takes to serve the orders of a unit, the more add ons that occur. When a phlebotomist arrives at a unit, s/he first serves pre-orders followed by add-ons. The service time of a unit is uncertain and depends on the arrival time of the phlebotomist and the quantity of orders in the unit.

Multiple phlebotomists are allowed to serve a unit, and these phlebotomists can enter or leave the unit at different times. When a group of phlebotomists is scheduled to serve a unit in the shift, this unit is said to be "swarmed." The advantage of swarming is that it allows the phlebotomists to serve orders at a unit as fast as possible, avoiding receiving and serving too many add-ons when phlebotomists are present. Without swarming, some units would receive so many add-ons that it would be impossible to serve the orders of other units. For example, Unit 22 has the highest number of patients and pre-orders, and it is also the most likely to get add-ons while the phlebotomists are present. Therefore, in current practice, this unit is often swarmed.

It is possible that the phlebotomists do not have time to visit or finish some units and postpone them to the next shift. The objective is to fulfill as many orders as possible in the morning shift. The question is how to schedule these phlebotomists and in what order the units should be served. The random demand and uncertain

service time of each unit make the routing of phlebotomists difficult. The problem is considered a variant of the team orienteering problem with stochastic rewards and service times (TOPSRS).

4.2 Problem Description

This section mathematically describes the phlebotomist routing problem that explicitly accounts for the stochastic rewards and service times resulting from the queuing process at the patient units. To simplify the problem, this research only consider simple venipuncture.

The problem is defined on a complete graph $G = (\mathcal{N}, \mathcal{E})$. The set $\mathcal{N} = \{0, 1, \ldots, N\}$ is a set of $N + 1$ nodes, and the set $\mathcal{E} = \{(n, n') : n, n' \in \mathcal{N}\}$ is the set of edges connecting the nodes. Node 0 represents the laboratory from where phlebotomists depart at the start of the day and nodes $1, \ldots, N$ represent the locations of the patient units. The travel time of $d(n, n')$ associated with each edge (n, n') is known. Let $\mathcal{M} = \{1, \ldots, M\}$ be a set of M phlebotomists initially located at the lab.

The rewards of each unit n contains the pre-orders that are known and the random add-ons. Let p_n be the number of pre-orders of unit n. Phlebotomists serve orders in the unit until there are no orders in the unit. We assume that the distributions of the add-ons received in each unit before phlebotomists leave are independent, but follow identical distributions differing in their parameters. We let Z_n be a random variable representing the number of add-ons in unit n and z_n be the realization of Z_n. The distribution of Z_n is a function of the pre-orders p_n, denoted by $f(z_n|p_n)$. The

reward can be collected at unit n, denoted by R_n, is a random variable that is the sum of pre-orders and random add-ons. The expected reward at unit n is:

$$E[R_n] = p_n + E[Z_n]. \tag{4.1}$$

The reward is collected by visiting and serving the unit n before the deadline, which is the end of the shift 09:30. When arriving at a unit, the orders are served to the maximum subject to the available time left before the deadline. Let S_n be a random variable representing the service time of fulfilling all orders in unit n. Assume each order takes a constant time μ to fulfill, the distribution of S_n is a function of p_n and z_n. Therefore, the expected service time of serving all orders at a unit is $S_n = \mu E[R_n]$.

4.3 A Priori Approach

A common approach to handle uncertainty in an orienteering problem is to restrict attention to a priori policies. A priori policies are characterized by a priori routes, or predetermined sequences of locations. A review of challenges and advances of using a priori routing is given in Campbell and Thomas (2008).

In this chapter, an a priori policy requires phlebotomists to visit patient units in the order specified by a set of pre-defined routes. As an example, consider a case in which there are two phlebotomists and 5 patient units. Let the a priori route of phlebotomist m_1 be (2, 3) and phlebotomist m_2 be (4, 1, 5). This route requires that phlebotomist m_1 visits unit 2 and fulfills all orders before s/he visits unit 3, while

phlebotomist m_2 visits unit 4 followed by units 1 and 5.

We denote an a priori route for phlebotomist m by a sequence of units $v^m = (v_0^m, \ldots v_i^m, \ldots v_{Im}^m)$. We denote by $(v^m)_{m \in M}$ a set containing an a priori route for each phlebotimist m in M. In this approach, each unit appears exactly once on exactly one route. We assume that phlebotomists visit the patient units in the order they appear in the a priori routes.

Our goal is to find an a priori route that maximizes the expected rewards by the end of the shift. For a priori route v, let $R_{v_i^m}$ be the random variable representing the reward collected from location v_i^m, and $E[R^m]$ the expected rewards of all locations in route v^m. Then the expected rewards for a priori route v is $\Omega(v) = \sum_{m \in \mathcal{M}} E[R^m]$. The objective of this problem is to find an a priori route v^* such that $\Omega(v^*) \geq \Omega(v)$ for every v.

4.3.1 Evaluation of a Fixed Route

To compute the expected rewards of an a priori route, we must compute the expected reward of each unit in the route. We start with analyzing the potential reward of a unit. The poential reward is the reward that a unit can get assuming there is enough time to fulfill all orders in the unit, including the existing number of orders when the phlebotomist arrives and add-ons that arrive while the phlebotomist is in the unit. The potential reward of a unit n consists of the realized pre-orders p_n and the realized add-ons z_n. For a unit, the realized add-ons include two parts: the realized add-ons between 05:30 and phlebotomist's arrival time (denoted by y_n), and

the realized add-ons between the arrival time and the leaving time (denoted by y_n^+). The potential reward of a unit r_n^{max} is computed as in (4.2):

$$\begin{aligned} r_n^{max} &= p_n + z_n \\ &= p_n + y_n + y_n^+. \end{aligned} \tag{4.2}$$

Time Range	Type	Notation	Arrival Rate	Fixed?
(00:00, 05:30)	Pre-order	p_n	λ_{1n}	Yes
(05:30, Arrival time]	Add-on	y_n	λ_{2n}	Yes
(Arrival time, Leave time]	Add-on	y_n^+	λ_{2n}	No

Table 4.2: Order categories of each patient unit

We assume that the arrival of orders to a patient room is a Poisson process. A Poisson process with arrival rate λ implies that the probability that the number of orders received by time t, denoted by $N(t)$, is equal to k is given by: $P(N(t) = k) = e^{-\lambda t}\frac{(\lambda t)^k}{k!}$. In TOPSRS, the arrival rate of a patient unit is λ_{1n} for pre-orders and λ_{2n} for add-ons. Table 4.2 shows the detailed information for the three categories of orders, including the arrival time range of each category, the type of orders based on the arrival time range, notation, arrival rate, and whether the quantity of orders is fixed and known when a phlebotomist arrives at a unit according to his/her route.

We assume a phlebotomist m arrives at a patient room n with $l = p_n + y_n$ orders. The phlebotomist first serves these existing orders followed by additional add-

ons y_n^+ that arrive until the order quantity in the unit hits zero. We call this a renewal process with one jump time, where the jump time is the time that the quantity of orders hits zero and is also the time when the phlebotomist leaves the unit. The time between the arrival time and the jump time is considered the renewal interval. The core analysis of this process is to find the distribution of the renewal interval, which requires determining the distribution on the number of add-ons received during the renewal interval, y_n^+, conditioning on the number of existing l orders when the phlebotomist arrives, denoted by $P(y_n^+|l)$.

4.3.1.1 Assessing $P(y_n^+|l)$

We assume that there are k add-ons arrive in the renewal interval, $y_n^+ = k$, where k is a non-negative integer. Because of the memory-less property of Poisson process, without loss of generality, we consider the arrival time of the phlebotomist as time 0, and the jump time as $(l+k)\mu$. Figure 4.3 shows a time axis of the renewal interval with an increment of μ.

Let X_0 be the random variable representing the number of orders received when serving the l existing orders, X_1 be the random variable representing orders received when serving the first additional add-on in the interval of $(l\mu, (l+1)\mu]$, X_2 is the random variable for interval of $((l+1)\mu, (l+2)\mu]$, and so forth.

The probability $P(y_n^+ = k|l)$ indicates that k add-ons arrive in the renewal interval (recall that the renewal interval refers to the interval between the phlebotomist's arrival time and his/her leaving time) and the quantity of orders does not hit zero in

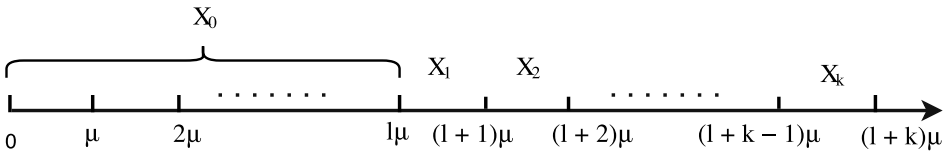

Figure 4.3: Renewal process at a unit

this interval, requiring the following three events:

1. At least one add-on arrives when serving the l existing orders, $X_0 \geq 1$;
2. k add-ons in total arrive in the renewal interval and the quantity of orders does not hit zero in any of the time intervals in $[\mu, (k-1)\mu]$. $X_0 + X_1 + \cdots + X_k = k$;
3. No order arrives when serving the k_{th} add-on, $X_k = 0$.

Let $F_i = X_0 + X_1 + \cdots + X_i$ be the random variable of accumulated orders received before time $i\mu$, events 1 and 2 can be referred to as $P(F_0 \geq 1)$ and $P(F_k = k)$ respectively.

Lemma 4.1. *The necessary condition that the quantity of orders does not hit 0 in the k periods is that:* $\forall i \in \{0, 1, \ldots, k\}, F_i \geq i + 1$.

Proof. (Proof by Contradiction.) Assume to the contrary that $F_i < i + 1$ for some i, there must exist a period $j, j \leq i$, that there is no order to serve in period j and the quantity of orders in the unit hits 0. This contradicts the precondition of the lemma.

We then return to the analysis of the three events. Events 1 and 2 can be

analyzed by a backward induction. In Equation (4.3), we consider a base condition of $F_0 = j$ where $j \geq 1$:

$$P(F_0 = j) = P(X_0 = j). \tag{4.3}$$

This base condition ensures that event 1 is satisfied. The joint probability of event 2 and 3 $P(F_k = k) \times P(X_k = 0)$ is equivalent to $P(F_{k-1} = k)$. Lemma 4.1 indicates $F_{k-2} \geq k - 1$, so we have :

$$X_{k-1} = F_{k-1} - F_{k-2} \leq k - (k-1) = 1. \tag{4.4}$$

Equation (4.4) demonstrates that there are no more than one order in the $(k-1)^{th}$ period. Another explanation is that if two or more orders arrive in $(k-1)^{th}$ period, these two orders cannot be finished by the k^{th} period. This contradicts the prerequisite.

$P(F_{k-1} = k)$ can be conducted in the recursive Equation (4.5):

$$P(F_{k-1} = k) = \begin{pmatrix} P(F_{k-2} = k) \\ P(F_{k-2} = k-1) \end{pmatrix}^T \quad \begin{pmatrix} P(X_{k-1} = 0) \\ P(X_{k-1} = 1) \end{pmatrix}, \tag{4.5}$$

Equation (4.5) specifies the only possible two cases in which k orders arrive in the $k-1$ periods:

1. All k orders arrive during the first $k-2$ periods while no orders in the $(k-1)^{th}$ period,
2. $k-1$ orders arrive in the first $k-2$ periods while one order arrives in the $(k-1)^{th}$ period.

In Equation (4.5), $P(F_{k-2} = k-1)$ is a recursive call when k decreases by 1, while $P(F_{k-2} = k)$ can be computed in a different recursion in Equation (4.6) that for any $1 < i < j < k$,

$$P(F_i = j) = \begin{pmatrix} P(F_{i-1} = j) \\ P(F_{i-1} = j-1) \\ P(F_{i-1} = j-2) \\ \dots \\ P(F_{i-1} = i) \end{pmatrix}^T \quad \begin{pmatrix} p(X_i = 0) \\ p(X_i = 1) \\ p(X_i = 2) \\ \dots \\ p(X_i = i-j) \end{pmatrix}. \tag{4.6}$$

Then, the distribution of y_n^+ conditioning on the number of existing orders is a joint probability of events 1, 2, and 3, which is:

$$P(y_n^+ = k|l) = P(F_{k-1} = k) \times P(X_k = 0) \tag{4.7}$$

The probability matrix for $P(y_n^+ = k|l)$ is listed in Appendix C.

4.3.1.2 Fixed Route Evaluation

Evaluating a fixed route v is accomplished by calculating the expected reward at each unit on the route. We develop a method to calculate the expected reward at a single unit. Applying this method to each unit in a fixed route and summing the results allows for the evaluation of a fixed-route. We start with evaluating a fixed route for a phlebotomist m, denoted by $v^m = (v_0^m, \dots v_i^m, \dots v_{Im}^m)$. Denote that $R_{v_i^m}$ the random orders served at unit v_i^m and by $E[R_{v_i^m}]$ the expected orders served at unit v_i^m. The quantity of $R_{v_i^m}$ depends on the arrival time at unit v_i^m (denoted by $A_{v_i^m}$). The quantity of $A_{v_i^m}$ depends on the arrival time to unit v_{i-1}^m, the orders served at unit v_{i-1}^m (denoted by $r_{v_{i-1}^m}$), and the travel time between location $i-1$ and i :

$$A_{v_i^m} = A_{v_{i-1}^m} + \mu r_{v_{i-1}^m} + d(v_{i-1}^m, v_i^m). \tag{4.8}$$

Then, $R_{v_i^m}$ can be calculated by three cases:

$$R_{v_i^m} = \begin{cases} r_{v_i^m}^{max}, & if\ A_{v_i^m} < D\ ,\ A_{v_i^m} + \mu r_{v_i^m}^{max} \leq D, \\ \left\lfloor \frac{D - A_{v_i^m}}{\mu} \right\rfloor, & if\ A_{v_i^m} < D\ ,\ A_{v_i^m} + \mu r_{v_i^m}^{max} > D, \\ 0, & if\ A_{v_i^m} > D. \end{cases}$$

In the first case, there is enough time to serve the potential rewards at unit v_i^m. In the second case, the phlebotomist arrives at the unit before the deadline but does not have enough time to serve all the orders. In this case, $R_{v_i^m}$ is the number of orders that can be served by the deadline. In the third case, the arrival time of unit v_i^m is later than the deadline, the reward is 0.

The expected reward at a unit v_i^m is:

$$E[R_{v_i^m}] = \sum_t \sum_y R_{v_i^m}(t, r_{v_i^m}^{max}) \times P(A_{v_i^m} = t, r_{v_i^m}^{max} = l_{v_i^m} + y_{v_i^m}^{+}) \times P(y_{v_i^m}^{+} | l_{v_i^m}). \tag{4.9}$$

The expected rewards of a solution tour v is the sum of the expected reward at each unit:

$$E[v] = \sum_{m=1}^{M} \sum_{i=0}^{Im} E[R_{v_i^m}] \tag{4.10}$$

Algorithm (4.1) explains the procedure of calculating the expected objective value of a fixed tour v. Lines 5 and 6 set the arrival time and reward of the medical

laboratory to 0. For a unit v_i^m in the tour of phelbotomist m, the arrival time can be computed by the previous location's arrival time and service time as in Line 8. Line 9 uses Monte Carlo simulation based on the distribution of $P(y^+_{v_i^m}|l_{v_i^m})$, line 10 computes the expected rewards of the unit by Equation (4.9). Line 11 updates the expected reward of a tour as the sum of the expected rewards in all units. For a given tour, we repeat this procedure 200 times ($TrialNum = 200$) to estimate the expected rewards of the tour.

4.4 Solution Approach

We apply a variable neighborhood search (VNS) heuristic to solve the problem. Algorithm 4.2 outlines the VNS implementation. This is similar to the algorithm in Zhang et al. (2014) and Campbell et al. (2011) where the algorithm was shown to be successful for stochastic orienteering problems. In our problem, the feasible solution contains M tours. Each tour is a permutation of the locations and starts at the medical laboratory $n = 0$. We assume any tour is feasible but the locations whose arrival time are later than the deadline D have a reward 0.

The solution is represented by a vector in which each phlebotomist's tour is separated by the medical laboratory "0". The initialization in Line 4 gives a random solution of TOPSRS. The initial solution v is generated in a way such that no two laboratories are adjacent in the tour. In other words, each tour starts with 0 and followed by at least one patient unit represented by non zero numbers. Line 6 states the stopping criteria, which includes an iteration limit and a level limit.

Algorithm 4.1 Calculation of the expected objective of a tour v

Initialization:

$E[v] = 0$

for $Trial = 1, \ldots, TrialNum$ **do**

  for $m = 1, \ldots, M$ **do**

    $A_{v_0^m} = 0,$

    $r_{v_0^m} = 0$

    for $i = 2, \ldots, I^m$ **do**

      $A_{v_i^m} = A_{v_{i-1}^m} + \mu r_{v_{i-1}^m} + d(v_{i-1}^m, v_{i-1}^m)$

      Monte Carlo Simulation to sample $y^+_{v_i^m}$ with the probability mass function of $P(y^+_{v_i^m} | l_{v_i^m})$

      Update $E[R_{v_i^m}]$ `by Equation` (4.9)

      $E[v] \leftarrow E[v] + E[R_{v_i^m}]$

    end for

  end for

end for

$E[v] \leftarrow E[v]/TrialNum$

Algorithm 4.2 Variable neighborhood search(VNS)

Input: Data for TOPSRS instance, the number of different neighborhoods N

Output: A TOPSRS solution v

Initialization:

  Randomly generate a TOPSRS solution v and evaluate it with Algorithm 4.1

  $i \leftarrow 0$, $k \leftarrow 1$, level $\leftarrow 0$

while $i < iterationMax$ or $level < levelMax$ **do**

  $v' \leftarrow$ shake(v, k)

  $v'' \leftarrow VND(v')$

  Excute Algorithm 4.1 to evaluate tour v''

  if $f(v'') > f(v)$ **then**

    $v \leftarrow v'', level \leftarrow 0$

  else if $k \leq N$ **then**

    $k \leftarrow 1$, level $\leftarrow$ level+1

  else

    $k \leftarrow k + 1$, level $\leftarrow$ level+1

  end if

  $i \leftarrow i + 1$

end while

With an improved solution, the *level* variable tracks the number of iterations that has no improvement based on the current improved solution. It ensures that there are at least $levelMax$ iterations of no improvement. We set $iterationMax = 40$ and $levelMax = 1$ to get robust solutions in our experiments. The $shake(v, k)$ in Line 7 returns a solution v', which is a random neighbor solution from the neighborhood $k \in \{1, \ldots, N\}$ of v. Line 8 then applies a variable neighborhood descent (VND) procedure to solution v' to obtain a local optimal solution v''. Line 9 and 10 evaluate this local optimal tour v'' by Algorithm 4.1 and compare it to the incumbent solution. Lines 10-16 show the updates of the current solution, the active neighborhood, the iteration counters, and the level counters.

In our implementation, the *shake* procedure considers the neighborhood of 1-shift, which shifts one node to a position after another node in the solution including within the same phlebotomist's tour and among different phletomists' tours.

Algorithm 4.3 describes the VND procedure in Line 8 of Algorithm 4.2. The VND finds a local optimal solution with respect to the 1-shift neighborhoods N'. The $BestNeighbor$ procedure in Line 6 returns the best solution in the neighborhood of the incumbent solution. Line 7 uses Algorithm 4.1 to evaluate a tour. Line 8-14 manage the updates of local optimal solution, active neighborhood, and the neighborhood counters.

Algorithm 4.3 Variable neighborhood descent (VND)

1: **Input:** A TOPSRS solution v, the number of different neighborhoods N'

2: **Output:** A local optimal solution v

3: **Initialization:**

4: $j \leftarrow 1$, $count \leftarrow 1$, improving $\leftarrow true$

5: **while** *improving* **do**

6: $v' \leftarrow BestNeighbor(v, j)$

7: **if** $g(v^{'}) > g(v)$ **then**

8: $v \leftarrow v^{'}, count \leftarrow 1$

9: **else if** $j = N'$ and $count < n'$ **then**

10: $j \leftarrow 1, count \leftarrow count + 1$

11: **else if** $j < N'$ and $count < n'$ **then**

12: $j \leftarrow j + 1, count \leftarrow count + 1$

13: **else**

14: $improving \leftarrow false$

15: **end if**

16: **end while**

4.5 Experimental Design

In this section, we discuss the design of experiments used to test the quality and efficiency of the proposed a *priori* approach.

4.5.1 Data Collection and Parameter Fitting

We collect historical data from 9/1/2015 to 9/30/2015 at UIHC. The data includes the number of available phlebotomists and infomation of each order, including the arrival time, patient unit the order is in, and order type. Table (4.3) shows the number of available phlebotomists each day. We note that orders' arrival rates before and after 05:30 are different. Therefore, we use the orders before 05:30 to fit the pre-orders Poisson parameter λ_1, and orders after 05:30 to fit the add-on Poisson parameter λ_2. Table (4.4) shows the arrival rates of each unit. Based on the empirical data and estimation of an expert in the hospital, the average time to draw a simple venipuncture is 7 minutes. Therefore, we set $\mu = 7/60$ in all experiments.

Day	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
# of PBTs	11	13	15	12	11	10	12	13	15	16	13	14	10	14	13
Day	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
# of PBTs	13	10	10	9	10	11	13	9	10	11	11	8	16	12	11

Table 4.3: Number of phlebotomists each day

Unit	0	1	2	3	4	5	6
λ_1	0	0.109	0.364	0.515	0.661	0.339	0.182
λ_2	0	0.001	0.003	0.004	0.006	0.003	0.002
Unit	7	8	9	10	11	12	13
λ_1	3.212	1.018	0.006	0.424	1.897	2.455	1.491
λ_2	0.027	0.008	0.000	0.004	0.016	0.020	0.012
Unit	14	15	16	17	18	19	20
λ_1	0.012	0.194	2.127	2.115	2.770	1.806	1.764
λ_2	0.000	0.002	0.018	0.018	0.023	0.015	0.015
Unit	21	22	23	24	25	26	27
λ_1	0.503	2.691	2.467	1.473	0.115	0.909	0.182
λ_2	0.004	0.022	0.021	0.012	0.001	0.008	0.002

Table 4.4: Arrival rate per hour at each unit

4.5.2 Experimental Design

To test the quality of the solutions from our approach, we design the following four experiments on the 30-day dataset:

- Experiment (1): Use the actual pre-orders and the available phlebotomists, and simulate the add-ons at each patient unit based on the fixed pre-order and arrival time of a unit.
- Experiment (2): Use the actual available phlebotomists of each day, and simulate both pre-orders and add-ons at a patient unit. The add-ons of a patient unit is generated based on the simulated pre-order and the arrival time of the location.
- Experiment (3): Based on Experiment (1), we consider swarming that multiple phlebotomists share the same routes such that they enter and leave a unit at the same time. The productivity of the team of phlebotomists is doubled for a

team of two phlebotomists and tripled for a team of three. In the experiments, we set the service time of an order in unit n as $\mu/|\mathcal{M}_n|$, where $|\mathcal{M}_n|$ is the cardinality of the set of phlebotomists at unit n.

- Experiment (4): Swarming in the same fashion above is allowed based on Experiment (2).

The four experiments are summarized in Table (4.5).

Experiment	Pre-order	Add-on	Swarm?
1	Actual	Simulate	No
2	Simulate	Simulate	No
3	Actual	Simulate	Yes
4	Simulate	Simulate	Yes

Table 4.5: Experimental design

4.6 Computational Results

This section presents the computational results of the experiments. We first focus on the the served orders from our experiments compared to the hospital practice. In each experiment, on a given day i, we compute the percentage difference between the results of the experiment and the hospital practice as in Equation (4.11):

$$\% \text{ difference} = \frac{\text{Served orders on day i in experiment} - \text{Served orders on day i at hospital}}{\text{Served orders on day i at hospital}}. \tag{4.11}$$

Figure 4.4 depicts the percentage difference of served pre-orders in Experiment

(1) in blue bar and Experiment (2) in red bar over the 30 days. Both experiments perform more pre-orders than the hospital practice in the majority of days except for Day 10 and 28. That is because these two days have the highest quantity of phlebotomists, which are more than enough to finish all the pre-orders. The real practice allows multiple phlebotomists to serve a unit at the same or different time and the productivity of the unit is increased. Our approach in Experiment (1) only allows one phlebotomist in a route and pre-orders in some units, especially the ones in the end of a route, are not fully served by the deadline. Figure 4.5 shows the percentage difference of served add-ons in Experiment (1) and (2) over the 30 days. In most days, there are more add-ons served in the hospital practice than the experiments. The reason is that our solutions spend most of time serving more pre-orders and leave a patient unit before it receives too many add-ons.

The blue and red horizontal lines in these figures are the average percentage difference from Experiment (1) and (2) with respect to pre-orders and add-ons respectively. It is shown that solutions from Experiment (1) fulfill more pre-orders and fewer add-ons on average than Experiment (2). Since the pre-orders are more important and the solution approach is designed to solve more pre-orders, we claim that the solutions in Experiment (1) are better than Experiment (2). With the exact known pre-orders, the a priori approach is able to offer better solution.

Next, we present the results of Experiments (3) and (4). We show the percent difference of served pre-orders in Figure (4.6) and served add-ons in Figure (4.7). There are more pre-orders and fewer add-ons served in both experiments compared to

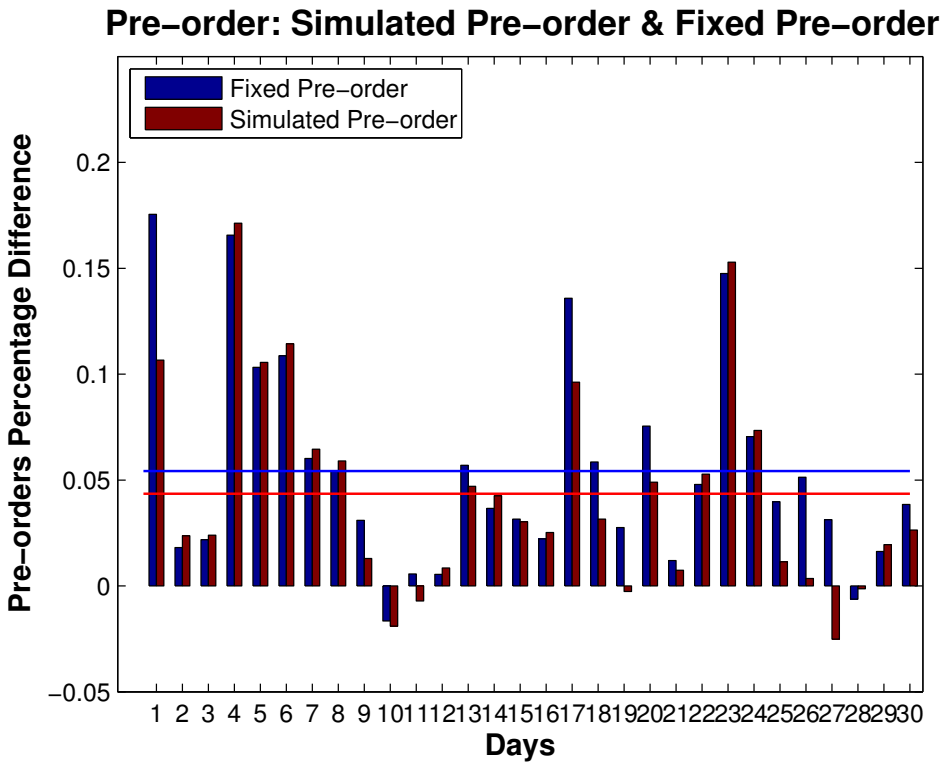

Figure 4.4: Comparison of served pre-orders

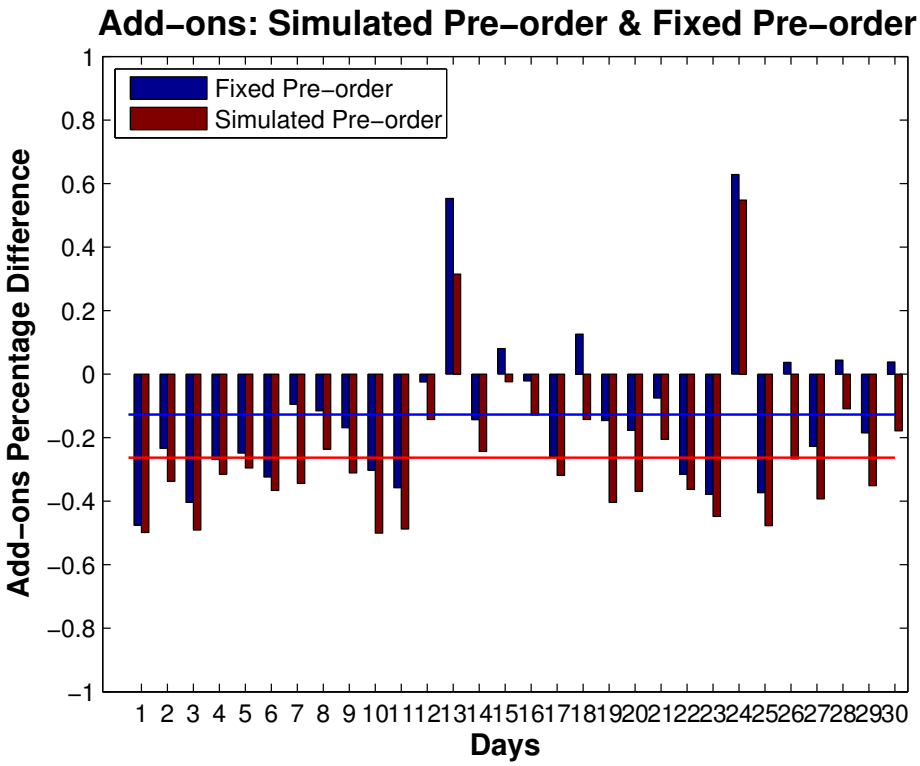

Figure 4.5: Comparison of served add-ons

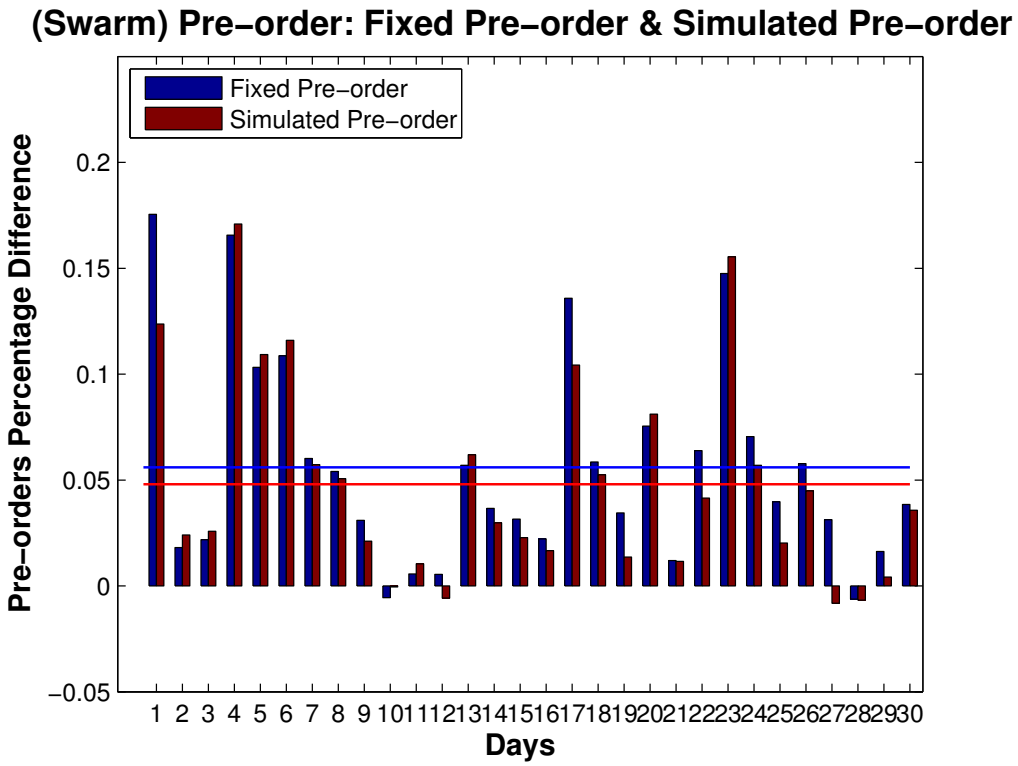

Figure 4.6: Comparison of served pre-orders with swarm

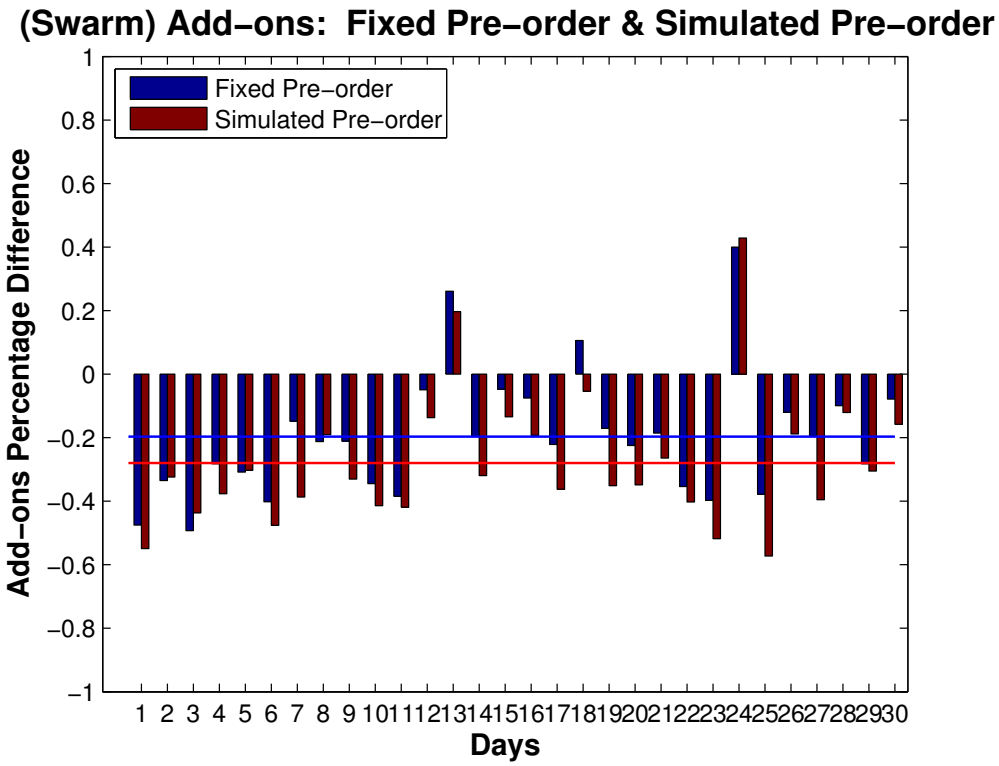

Figure 4.7: Comparison of served add-ons with swarm

hospital practice. Specifically, solutions from Experiment (3) fulfill more pre-orders and fewer add-ons on average than Experiment (4) as shown in the blue and red horizontal lines. Therefore, we claim that solutions from Experiment (3) are better than Experiment (4).

Experiment	Avg served pre-order	Avg served add-on
1	5.4%	-12.7%
2	4.3%	-26.3%
3	5.6%	-19.7%
4	4.8%	-28.0%

Table 4.6: Percentage difference compared to the hospital served orders

Table (4.6) shows the average percentage difference of the served pre-orders and add-ons over 30 days in the four experiments. When comparing Experiment (1) and (3), we see that Experiment (3) serves slightly more pre-orders and fewer add-ons than Experiment (1). This is because our solution without swarm serve most of the pre-orders and there is not much space to improve when considering swarming.

Experiment	Avg Solution time (sec)
1	769.6
2	2582.2
3	783.0
4	7923.9

Table 4.7: Solution time of the experiments

Table (4.7) shows average solution time over the 30 instances in the four experiments. Instances in Experiment (1) and (3) can be solved in less then 15 minutes. Experiment (2) uses simulated pre-orders and it takes around 45 minutes on average. Experiment (4) takes more than two hours when using swarm and simulated pre-orders. Considering both the solution quality and solution time, experiment (1) and (3) are good enough for practice in real hospital. Solutions with good quality can be achieved within 15 minutes.

Experiment	Avg unserved pre-order	Avg unserved add-on
1	0.6%	28.2%
2	1.6%	39.6%
3	0.6%	28.2%
4	1.2%	40.9%
Hospital Practice	5.6%	17.0%

Table 4.8: Percentage of unserved orders

Furthermore, we would like to see the quantity of unserved orders in the four experiments. For each day, the percentage of unserved orders is given by Equation 4.12:

$$\text{Unserved } \% = \frac{\text{Ordered order on day i at hospital} - \text{Served order on day i in experiment}}{\text{Ordered order on day i at hospital}}. \tag{4.12}$$

Table (4.8) shows the average percentage of the unserved pre-orders and add-ons over the 30 days. It is shown that around 99% of pre-orders are served in the

four experiments, while around 27%-40% add-ons are not served. This suggests that there is still room for improvement.

4.7 Conclusions

Motivated by the phlebotomist intra-hospital routing problem at UIHC, we introduce a team orienteering problem with stochastic rewards and queueing-based service times. Phlebotomists serve orders in patient units based on doctors' orders. Doctors request orders randomly throughout the day. The arrival of orders in a unit is a Poisson process. A phlebotomist arrives at a unit fulfill existing orders as well as additional orders received while s/he is in the unit. The rewards of a unit are stochastic depending on the served orders given the arrival time and the existing orders when s/he arrives. The service time of a unit is also stochastic depending on the quantity of orders in the unit. This chapter develops a priori approach that gives a fixed route for each phlebotomist and derives an analytical procedure to evaluate the expected rewards of a priori tour. We apply the a priori approach to a 30-day dataset collected from UIHC. The computational results demonstrate that the proposed a priori approach gives solutions that serve more pre-orders but fewer add-ons than the hospital practice.

CHAPTER 5
FUTURE WORK

Future work of this dissertation focuses on the phlebotomist routing problem. In this chapter, we formulate the problem as a Markov decision process (MDP) which dynamically assigns a phlebotomist to patient units according to orders' status in each patient unit.

5.1 Markov Decision Process

We formulate the phletomist intra-hosptial routing problem as a Markov decision process. The state of the system represents the sufficient information for the phlebotomist to execute a decision of going to another unit. The state consists of information on phlebotomists' status as well as the status of all the patient units. We define the status of the phlebotomists $m \in \mathcal{M}$ in a pair (l_m, t_m), where l_m is the last node visited of phlebotomist m (or the current location if the phlebotomist has not left), and t_m represent the arrival time of the last node visited.

Let $(l, t)_{m \in \mathcal{M}}$ denote the vector of the phlebotomist attributes. We then characterize the state at a unit n by $(d_n, V, \hat{n})$, where d_n is the known pending reward at unit n in $\mathcal{N} \backslash \{0\}$ in $\{\{?\} \cup [0, \infty)\}$. The initial pending reward at unit n is the number of pre-orders p_n. The set V represents the nodes visited prior to and including l_m for all $m \in \mathcal{M}$. Among the last node visited for all phlebotomists $\{\cup_{m \in \mathcal{M}} l_m\}$, node $\hat{n}$ is the node whose pending reward has just become zero. A decision epoch k is triggered when this happens. At each decision epoch k, the

complete state of the system is a tuple $((l,t)_{m\in\mathcal{M}},(d_{n\in\mathcal{N}},V,\hat{n}))$, which is in the state space $S=\mathcal{N}^M\times[0,D]^M\times\{\{?\}\cup[0,\infty)\}^N\times\mathcal{N}\times 1$. The initial state is $s_0=((0,0)^M,((p_n)^N,\{0\},0))$.

Let $\mathcal{M}_{\hat{n}}$ be the set of phlebotomists that are in unit $\hat{n}$ at decision epoch k. An action at decision epoch k is an assignment of the phlebotomists in $\mathcal{M}$ to the units in $\mathcal{N}$. The action a is a M dimensional vector where the m_{th} element, a_m, is the action directing phlebotomist m to a unit in $\mathcal{N}$. The set of actions at decision epoch k is:

$$\begin{aligned}\mathcal{A}(s_k)= \quad & \{a\in\mathcal{N}^M: \\ & a_m\in\mathcal{N}\backslash V,\ \forall m\in\mathcal{M}_{\hat{n}} && (5.1)\\ & a_m=l_m,\ \forall m\in\mathcal{M}\backslash\mathcal{M}_{\hat{n}}\}. && (5.2)\end{aligned}$$

Condition (5.1) enforces that phlebotomists at unit $\hat{n}$ can be directed to an unvisited unit in $\mathcal{N}\backslash V$, while Condition (5.2) requires that phlebotomists en route continue to their destination or stay at their current unit if the pending reward has not finished at epoch k.

The transition from state s_k to a pre-decision state s_{k+1} with action a is:

$$s_{k+1}=S^M(s_k,a,\omega_{k+1}), \tag{5.3}$$

where $S^M(\cdot)$ is the state transition function and ω_{k+1} is the exogenous information observed during $k+1$. The exogenous information includes that a unit's pending reward reaches zero and $\hat{n}$ is updated. We assume that the reward of a unit is

collected when the phlebotomists leave this unit. A transition from state s_k to state s_{k+1} with action a results in a reward at unit $\hat{n}$:

$$R_k(s_k, a) = r_{\hat{n}}. \tag{5.4}$$

Assume Π is the set of the all Markovian deterministic policies for the problem. A policy π is a sequence of decision rules: $\pi = (\delta_0^\pi, \delta_1^\pi, \ldots, \delta_K^\pi)$, where each decision rule $\delta_k^\pi : s_k \to \mathcal{A}(s_k)$ is a function that specifies the selected action at state s_k when following policy π. We seek a policy π in Π that maximizes the total expected reward, conditional on initial state s_0: $E[\sum_{k=0}^{K} R_k(s_k, \delta^\pi(s_k))|s_0]$. Let $V(s_k)$ denote the expected reward-to-go from state s_k in epoch k through finish decision epoch K. An optimal policy can be obtained by solving the optimality equation $V_{s_k} = \max_{a \in \mathcal{A}(s_k)}\{R_k(s_k, a) + E[V(s_{k+1})|s_k, a]\}$ for each epoch k and state s_k in state space S.

In the MDP, we allow multiple phlebotomists to serve a unit at the same/different time. In other words, swarming is allowed in the above MDP.

5.2 Future Work

In the future work of this dissertation, we will develop rollout policies to the MDP in Section 5.1 that make dynamic routing decisions to the phlebotomist intra-hospital routing problem.

Furthermore, there are a number of directions to model this problem considering more realistic and intricate settings. First, the current model in Section 5.1

allow each unit to be visited only once. However, since the add-ons arrive randomly between 05:30 and 09:30, a unit might receive additional add-ons after phlebotomists leave. In the practice at UIHC, the phlebotomists can return to a unit to do more add-ons after other units are finished. In the future work, we will consider a model that dynamically assign phlebotomist to serve the additional add-ons based on the order status in all patient units. Second, the current model considers only simple venipunctures while blood cultures are excluded. Future study should include both orders to offer better routing plans for phlebotomists. Third, the MDP model in Section 5.1 considers routing decisions only in the morning shift. The real practice at UIHC shows that unfulfilled orders in the morning shift will be postponed to the next shift. Therefore, expanding the decision time horizon from one shift to multiple shifts and considering the impacts of the postponed orders would be an interesting extension.

Another direction of the future work is to incorporate human learning and knowledge transfer in the phlebotomist routing problem. Phlebotomy requires the same set of skills, however, phlebotomists have heterogeneous levels of experience. New phlebotomists are always paired with experienced phlebotomists for practice and training purposes. In the long term, new phlebotomists gain experience as they practice. Their improved performance affects the decision of scheduling and intra-hospital routing. When pairing/teaming new phlebotomists with experienced phlebotomists for learning and training purposes, it is essential to consider learning by experience as well as learning by knowledge transferred from team workers. In these cases, models

of Chapter 2 and 3 are applicable to this problem.

APPENDIX A
PROOF OF THEOREM 3.1

Proof. By contradiction.

We first consider the case of two workers and two tasks with T periods. We assume a specialized scheduling s is the best scheduling from all specialized that yields the maximum system output over the horizon. We prove that this specialized scheduling is the optimal solution for the system.

In a parallel system with two workers and two jobs, we assume the best specialized scheduling is worker 1 is assigned to job 1 while worker 2 is assigned to job 2. The output of the specialized scheduling s^* is:

$$O_s = \sum_{k \in \mathcal{T}} \varphi_{11}(c_{11}^k) + \sum_{k \in \mathcal{T}} \varphi_{22}(c_{22}^k) \tag{A.1}$$

Since s^* is the best specialized scheduling, the other specialized scheduling, such as worker 1 perform job 2 while worker 2 work on job 1, yields smaller system output. we have:

$$\sum_{k \in \mathcal{T}} \varphi_{11}(c_{11}^k) + \sum_{k \in \mathcal{T}} \varphi_{22}(c_{22}^k) \geq \sum_{k \in \mathcal{T}} \varphi_{12}(c_{12}^k) + \sum_{k \in \mathcal{T}} \varphi_{21}(c_{21}^k). \tag{A.2}$$

If there exists a non-specialized scheduling ns yields a higher system output than s. We have:

$$O_s < O_{ns} \tag{A.3}$$

There is at least one switch time point in the scheduling, and we say the switch point is $t, t > 1$. We have the units of time periods of assigning each worker to each job in Table A.1.

	Job 1	Job 2
Worker 1	t	$T-t$
Worker 2	$T-t$	t

Table A.1: Assignment of a two-job flow line

Therefore, the total system throughput for the non-specialized scheduling ns is:

$$O_{ns} = \sum_{k \leq t} \varphi_{11}(c_{11}^k) + \sum_{k > t} \varphi_{12}(c_{12}^k) + \sum_{k > t} \varphi_{21}(c_{21}^k) + \sum_{k \leq t} \varphi_{22}(c_{22}^k). \tag{A.4}$$

To analyze the knowledge transferred from coworkers in the same team, we assume that the knowledge/experience gained from coworkers in a given period can never exceed the experience gained if the worker was assigned to the task j itself during the same period. We add the constraints that:

$$S_{ij}^t - S_{ij}^{t-1} = \sum_{p \in \mathcal{I}}^{p \neq i} \sum_{q \in g} \varphi_{pq}(c_{pq}^t) < 1 \tag{A.5}$$

Constraints (A.5) enforce that knowledge transferred from team member's experience on other jobs could not result in a learning for job j superior to the

learning the worker would have reached if the worker is assigned to j during the same time period.

$$\begin{aligned} O_{ns} &= \sum_{k\le t}\varphi_{11}(c_{11}^k)+\sum_{k>t}\varphi_{12}(c_{12}^k)+\sum_{k>t}\varphi_{21}(c_{21}^k)+\sum_{k\le t}\varphi_{22}(c_{22}^k) \\ &= \sum_{k\le t}\varphi_{11}(k-1+\theta_1\varphi_{22}(c_{22}^k))+\sum_{k>t}\varphi_{12}(k-t-1+\theta_1\varphi_{21}(c_{21}^k)) \\ &\quad +\sum_{k\le t}\varphi_{22}(k-1+\theta_2\varphi_{11}(c_{11}^k))+\sum_{k>t}\varphi_{21}(k-t-1+\theta_2\varphi_{12}(c_{12}^k)) \quad \text{(A.6)} \end{aligned}$$

We expand Equation A.4,

$$\begin{aligned} O_s &= \sum_{k\le t}\varphi_{11}(c_{11}^k)+\sum_{k>t}\varphi_{11}(c_{11}^k)+\sum_{k>t}\varphi_{22}(c_{22}^k)+\sum_{k\le t}\varphi_{22}(c_{22}^k) \\ &= \sum_{k\le t}\varphi_{11}(k-1+\theta_1\varphi_{22}(c_{22}^k))+\sum_{k>t}\varphi_{11}(k-1+\theta_1\varphi_{22}(c_{22}^k)) \\ &\quad +\sum_{k>t}\varphi_{22}(k-1+\varphi_{11}(c_{11}^k))+\sum_{k\le t}\varphi_{22}(k-1+\varphi_{11}(c_{11}^k)) \end{aligned}$$

By (A.2):

$$\begin{aligned} &> \sum_{k\le t}\varphi_{11}(k-1+\theta_1\varphi_{22}(c_{22}^k))+\sum_{k>t}\varphi_{12}(k-1+\theta_1\varphi_{22}(c_{22}^k)) \\ &\quad +\sum_{k>t}\varphi_{22}(k-1+\varphi_{11}(c_{11}^k))+\sum_{k\le t}\varphi_{21}(k-1+\varphi_{11}(c_{11}^k)) \end{aligned}$$

By (A.5) and $t > 1$:

$$\begin{aligned} &> \sum_{k\le t}\varphi_{11}(k-1+\theta_1\varphi_{22}(c_{22}^k))+\sum_{k>t}\varphi_{12}(k-t-1+\theta_1\varphi_{21}(c_{21}^k)) \\ &\quad +\sum_{k\le t}\varphi_{22}(k-1+\theta_2\varphi_{11}(c_{11}^k))+\sum_{k>t}\varphi_{21}(k-t-1+\theta_2\varphi_{12}(c_{12}^k)) \\ &= O_{ns} \end{aligned} \quad \text{(A.7)}$$

This contradicts to the assumption: (A.3). Therefore, we prove that the best specialized scheduling s^* yields the highest system throughput in the system, and therefore is the optimal scheduling.

We can extending the above proof to the general case of $|\mathcal{I}|$ workers and $|\mathcal{J}|$ tasks. The results follow in the same fashion.

APPENDIX B
TOP TEN POLICIES IN NEMBHARD AND BENTEFOUET (2015)

The top 10 policies for parallel and serial systems considering grouping and assignment stages are listed in Table B.1

	Parallel System		Serial System	
Rank	Grouping	Assignment	Grouping	Assignment
1	Min variance K	Maximax K	Min variance $1/r$	Maximax O
2	Min variance p	Maximax θ	Min variance K	Maximax θ
3	Min variance K	Maximax θ	Min variance K	Maximax K
4	Min variance $1/r$	Maximax K	Min variance O	Maximax O
5	Min variance $1/r$	Maximax θ	Max variance p	Maximax K
6	Min variance p	Maximax K	Min variance K	Maximax θ
7	Min variance θ	Maximax θ	Max variance p	Maximax θ
8	Min variance θ	Maximax K	Min variance p	Maximax θ
9	Min variance O	Maximax p	Min variance K	Maximax K
10	Min variance O	Maximax p	Min variance O	Maximax $1/r$

Table B.1: Top 10 policies in Nembhard and Bentefouet (2015)

Nembhard and Bentefouet (2015) did not explain how each policy is calculated. We contacted the author and got the explicit form for the policies. For example, Min variance K follows the procedure below. Nembhard and Bentefouet (2015) considered parallel and serial systems with three types of job machines each with three identical jobs. Workers have common learning and transfer parameters for each machine type. Assume the three machine types are a, b, and c. Each worker i has three values for the parameters, let's say K has values of K_{ia}, K_{ib}, K_{ic}. The average K value of worker

i is $K_i = (K_{ia} + K_{ib} + K_{ic})/3, i = 1, \ldots 9$. The grouping policy of Min variance K is a way of grouping the nine workers to three groups such that the total variance of the three groups is the smallest. The variance of a group with workers i_1, i_2, i_3 is: $((k_{i_1} - \mu)^2 + (k_{i_2} - \mu)^2 + (k_{i_3} - \mu)^2)/3$, where μ is the average of the three workers k values. The assignment policy assigns each team to a job type. An policy of Maximax K assigns the job type to the team whose maximal K value on this job is the highest over the three teams. The other grouping and assignment policies follow the same fashion.

APPENDIX C
PROBABILITY MATRIX OF $P(Y_N^+ = K|L)$

The matrix is listed in Table C.1. The columns are the number of existing orders when a phlebotomist arrives to a unit, while the rows represent the phlebotomist's leave time.

	n						
Leave time	0	1	2	3	...	l	...
0	0	0	0	0	...	0	...
μ	0	$e^{-\lambda\mu}$	0	0	...	0	
2μ	0	$e^{-2\lambda\mu}(\lambda\mu)$	$e^{-2\lambda\mu}$	0	...	0	...
3μ	0	$\frac{3}{2}e^{-3\lambda\mu}(\lambda\mu)^2$	$e^{-3\lambda\mu}(2\lambda\mu)$	$e^{-3\lambda\mu}$	...	0	...
4μ	0	$\frac{8}{3}e^{-4\lambda\mu}(\lambda\mu)^3$	$4e^{-4\lambda\mu}(\lambda\mu)^2$	$e^{-4\lambda\mu}(3\lambda\mu)$	...	0	...
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
$(k+l)\mu$	0	.	.	.	.	. $P(F_{k-1} = k)$ $\times P(X_k = 0)$	
.	.	.	.	.	.	.	.

Table C.1: Probability matrix

REFERENCES

Agnihothri, S. R., Mishra, a. K. and Simmons, D. E. (2003). Workforce cross-training decisions in field service systems with two job types, *Journal of the Operational Research Society* **54**(4): 410–418.
URL: *http://www.palgrave-journals.com/doifinder/10.1057/palgrave.jors.2601535*

Anzanello, M. J. and Fogliatto, F. S. (2011). Learning curve models and applications: Literature review and research directions, *International Journal of Industrial Ergonomics* **41**(5): 573–583.
URL: *http://linkinghub.elsevier.com/retrieve/pii/S016981411100062X*

Azaiez, M. and Al Sharif, S. (2005). A 0-1 goal programming model for nurse scheduling, *Computers & Operations Research* **32**(3): 491–507.
URL: *http://linkinghub.elsevier.com/retrieve/pii/S0305054803002491*

Beliën, J. and Demeulemeester, E. (2008). A branch-and-price approach for integrating nurse and surgery scheduling, *European Journal of Operational Research* **189**(3): 652–668.
URL: *http://linkinghub.elsevier.com/retrieve/pii/S0377221706011751*

Boon, M. A. A., Mei, R. D. V. D. and Winands, E. M. M. (2011). Surveys in Operations Research and Management Science Applications of polling systems, *Surveys in Operations Research and Management Science* **16**(2): 67–82.
URL: *http://dx.doi.org/10.1016/j.sorms.2011.01.001*

Burnetas, A. N. (2013). Customer equilibrium and optimal strategies in Markovian queues in series, pp. 515–529.

Campbell, A. M., Gendreau, M. and Thomas, B. W. (2011). The orienteering problem with stochastic travel and service times, *Annals of Operations Research* **186**: 61–81.

Campbell, A. M. and Thomas, B. W. (2008). Challenges and Advances in A Priori Routing.

Cheang, B., Li, H., Lim, a. and Rodrigues, B. (2003). Nurse rostering problemsa bibliographic survey, *European Journal of Operational Research* **151**(3): 447–460.
URL: *http://linkinghub.elsevier.com/retrieve/pii/S0377221703000213*

Colen, P. and Lambrecht, M. (2010). Cross Training Policies in a Maintenance Field Service Organization, *Proceedings of APMS 2010 International Conference Advances in Production Management Systems*, Como, Italy, pp. 1–7.
URL: *https://lirias.kuleuven.be/handle/123456789/280316*

Corominas, A., Olivella, J. and Pastor, R. (2010). A model for the assignment of a set of tasks when work performance depends on experience of all tasks involved, *International Journal of Production Economics* **126**(2): 335–340.
URL: *http://linkinghub.elsevier.com/retrieve/pii/S0925527310001337*

Dar-El, E. M. (2000). *Human Learning: From Learning Curves to Learning Organizations*, Vol. 29 of *International Series in Operations Research & Management Science*, Kluwer Academic Publishers, Boston.

Denton, B. T. (2013). *Handbook of Healthcare Operations Management*.

Evers, L., Glorie, K., Ster, S. V. D., Isabel, A. and Monsuur, H. (2014). Computers & Operations Research A two-stage approach to the orienteering problem with stochastic weights, *Computers and Operation Research* **43**: 248–260.
URL: *http://dx.doi.org/10.1016/j.cor.2013.09.011*

Faraj, S. (2000). Coordinating Expertise in Software Development Teams, *management Sciences* **46**(October 2015): 1554–1568.

Fowler, J., Wirojanagud, P. and Gel, E. (2008). Heuristics for workforce planning with worker differences, *European Journal of Operational Research* **190**(3): 724–740.
URL: *http://linkinghub.elsevier.com/retrieve/pii/S0377221707006017*

F.W. Bevis, C. Finniear, D. T. (1970). Prediction of operator performance during learning of repetitive tasks, *International Journal of Production Research* **8**(4): 293–305.

Hamilton, L. T. and Sm, M. T. A. (2014). Managing the Laboratory Technical Workforce, **27**(2007): 807–821.

Heimerl, C. and Kolisch, R. (2010). Work assignment to and qualification of multi-skilled human resources under knowledge depreciation and company skill level targets, *International Journal of Production Research* **48**(13): 3759–3781.
URL: *http://www.tandfonline.com/doi/abs/10.1080/00207540902852785*

Hewitt, M., Chacosky, A., Grasman, S. E. and Thomas, B. W. (2014). Integer Programming Techniques for Solving Non-linear Workforce Planning Models with Learning, *European Journal Of Operational Research* . Available from http://myweb.uiowa.edu/bthoa/iowa/Research.html.

Honnappa, H. (2014). Strategic Arrivals into Queueing Networks : The Network Concert Queueing Game, **00**(0).

Hopp, W. J., Tekin, E. and Van Oyen, M. P. (2004). Benefits of Skill Chaining in Serial Production Lines with Cross-Trained Workers, *Management Science* **50**(1): 83–98.
URL: *http://mansci.journal.informs.org/cgi/doi/10.1287/mnsc.1030.0166*

Iravani, S. M. R., Kolfal, B. and Van Oyen, M. P. (2007). Call-Center Labor Cross-Training: It's a Small World After All, *Management Science* **53**(7): 1102–1112.
URL: *http://mansci.journal.informs.org/cgi/doi/10.1287/mnsc.1060.0621*

Jaber, M. Y. (2006). Learning and Forgetting Models and Their Applications, *in* A. B. Badiru (ed.), *HandBOOK of Industrial and Systems Engineering*, Industrial Innovation Series, CRC Press, Boca Raton, FL, USA, chapter 30, pp. 30–1–30–27.

Jaber, M. Y. and Sikström, S. (2004). A numerical comparison of three potential learning and forgetting models, *International Journal of Production Economics* **92**(3): 281–294.

Jin, H., Thomas, B. W. and Hewit, M. (2016). Integer Programming Techniques for Makespan Minimizing Workforce Assignment Models that Recognize Human Learning, *Computers & Industrial Engineering* **97**: 202–211.
URL: *http://linkinghub.elsevier.com/retrieve/pii/S0360835216301000*

Landeghem, H. V. A. N. (2004). THE STATE OF THE ART OF NURSE ROSTERING, pp. 441–499.

Leaven, L. and Qu, X. (2014). Applying Scenario Reduction Heuristics in Stochastic Programming for Phlebotomist Scheduling, **8**(3): 1–4.

Levy, H., Sidi, M., Member, S. and Paper, I. (1990). Polling Systems : Applications , Modeling , and Optimization, **38**(10): 1750–1760.

Lewis, K. (2003). Measuring transactive memory systems in the field: Scale development and validation., *Journal of Applied Psychology* **88**(4): 587–604.
URL: *http://doi.apa.org/getdoi.cfm?doi=10.1037/0021-9010.88.4.587*

Maenhout, B. and Vanhoucke, M. (2009). Branching strategies in a branch-and-price approach for a multiple objective nurse scheduling problem, *Journal of Scheduling* **13**(1): 77–93.
URL: *http://link.springer.com/10.1007/s10951-009-0108-x*

Mazur, J. E. and Hastie, R. (1978). Learning as accumulation: a reexamination of the learning curve., *Psychological bulletin* **85**(6): 1256–74.
URL: *http://www.ncbi.nlm.nih.gov/pubmed/734012*

Nembhard, D. a. and Bentefouet, F. (2012). Parallel system scheduling with general worker learning and forgetting, *International Journal of Production Economics* **139**(2): 533–542.
URL: *http://linkinghub.elsevier.com/retrieve/pii/S0925527312002162*

Nembhard, D. a. and Bentefouet, F. (2015). Selection, grouping, and assignment policies with learning-by-doing and knowledge transfer, *Computers & Industrial Engineering* **79**: 175–187.
URL: *http://linkinghub.elsevier.com/retrieve/pii/S0360835214003672*

Nembhard, D. A. and Norman, B. A. (2007). Cross Training in Production Systems with Human Learning and Forgetting, *in* D. A. Nembhard (ed.), *Workforce Cross Training*, CRC Press, Boca Raton, FL, USA, chapter 4, pp. 111–129.

Norman, B. a., Tharmmaphornphilas, W., Needy, K. L., Bidanda, B. and Warner, R. C. (2002). Worker assignment in cellular manufacturing considering technical and human skills, *International Journal of Production Research* **40**(6): 1479–1492.

Olivella, J., Corominas, A. and Pastor, R. (2013). Task assignment considering cross-training goals and due dates, *International Journal of Production Research* **51**(3): 37–41.

Papapanagiotou, V., Weyland, D., Montemanni, R. and Gambardella, L. M. (2013). A sampling-based approximation of the objective function of the orienteering problem with stochastic travel and service times, *Lecture Notes in Management Science* **5**: 143–152.
URL: *http://www.tadbir.ca/lnms/archive/v5/lnmsv5p143.pdf*

Schmid, V., Doerner, K. F., Schmid, V. and Doerner, K. F. (2014). Examination and Operating Room Scheduling Including Optimization of Intrahospital Routing Examination and Operating Room Scheduling Including Optimization of Intrahospital Routing, (September 2015).

Shafer, S. M., Nembhard, D. a. and Uzumeri, M. V. (2001). The Effects of Worker Learning, Forgetting, and Heterogeneity on Assembly Line Productivity, *Management Science* **47**(12): 1639–1653.
URL: *http://mansci.journal.informs.org/cgi/doi/10.1287/mnsc.47.12.1639.10236*

Tang, H. . and Miller-Hooks, E. . (2005). Algorithms for a Stochastic Selective Travelling Salesperson Problem, *Palgrave Macmillan Journals on behalf of the Operational Research Society Stable* **56**(4): 439–452.

Thorlacius, P., Clausen, J. and Brygge, K. (2010). Scheduling of inspectors for ticket spot checking in urban rail transportation, pp. 1–14.

Uzzi, B. and Lancaster, R. (2003). Relational Embeddedness and Learning : The Case of Bank Loan Managers and Their Clients, **49**(4): 383–399.

Vishnevskii, V. M. and Semenova, O. V. (2006). Mathematical Methods to Study the Polling Systems, **67**(2): 173–220.

Wirojanagud, P., Gel, E. S., Fowler, J. W. and Cardy, R. (2007). Modelling inherent worker differences for workforce planning, *International Journal of Production Research* **45**(3): 525–553.
URL: *http://www.tandfonline.com/doi/abs/10.1080/00207540600792242*

Wright, T. (1936)). Factors affecting the cost of airplanes, *Journal of the Aeronautical Sciences* **3**(4): 122–128.

Yan, Z. and Gao, S. Y. (2010). On the Design of Sparse but Efficient Structures in Operations.

Yechiali, U. (1969). ON OPTIMAL BALKING RULES AND TOLL CHARGES.

Yechiali, U. R. I. (1972). CUSTOMERS ' OPTIMAL JOINING RULES FOR THF Gl / M / s QUEUE *, **18**(7).

Yelle, L. E. (1979). The learning curve: historical review and comprehensive survey, *Decision Sciences* **10**: 302 – 328.

Zhang, S., Ohlmann, J. W. and Thomas, B. W. (2014). A priori orienteering with time windows and stochastic wait times at customers, *European Journal of Operational Research* **239**(1): 70–79.
URL: *http://dx.doi.org/10.1016/j.ejor.2014.04.040*

Zhang, S., Ohlmann, J. W. and Thomas, B. W. (2015). Dynamic Orienteering on a Network of Queues, (2010): 1–39.