Simulation-Optimization, Markov Chain and Graph Coloring Approaches to Military Manpower Modeling and Deployment Sourcing

by

Mark M. Zais

B.S., United States Military Academy, 1997

M.S., Georgia Institute of Technology, 2008

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Leeds School of Business

2014

This thesis entitled:
Simulation-Optimization, Markov Chain and Graph Coloring Approaches to Military Manpower Modeling and Deployment Sourcing
written by Mark M. Zais
has been approved for the Leeds School of Business

_____

Manuel Laguna

_____

Dan Zhang

Date _____

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Zais, Mark M. (Ph.D., Operations Management)

Simulation-Optimization, Markov Chain and Graph Coloring Approaches to Military Manpower Modeling and Deployment Sourcing

Thesis directed by Prof. Manuel Laguna

The Army manpower system is a integration of numerous elements that can be independently modeled. Identifying and closing gaps in modeling research can reduce workforce inefficiencies and costs. Military manpower models are predominantly focused on forecasting behavior and inventory within given demand requirements. Moreover, research directed towards predicting behavior is almost entirely disaggregated by pecuniary and non-pecuniary goals with disproportionate effort devoted to modeling the external factors that effect such behavior. This thesis proposes modeling approaches to improve the management capabilities of the Army's manpower system.

First, we consider a simulation-optimization approach to estimating workforce requirements examines the capabilities and limitations of Monte Carlo simulation and optimization methods within the context of workforce demand forecasting, modeling and planning. Specifically, we focus on these methods as a viable improvement for aligning strategic goals with workforce requirements. A general model is presented for estimating workforce requirements given uncertain demand. Using a real-world data example, we assess the benefits of this methodology to determine an optimal mix of workforce skills while providing the flexibility and robustness to incorporate uncertainty, assess risk and improve effectiveness of the workforce planning process.

Second, we address the critical stay-or-leave decision associated with military retention. Personnel retention is one of the most significant challenges faced by the U.S. Army. Central to the problem is understanding the incentives of the stay-or-leave decision for military personnel. Using three years of data from the U.S. Department of Defense, we construct and estimate a Markov chain model of military personnel. Unlike traditional classification approaches, such as logistical regression models, the Markov chain model allows us to describe military personnel dynamics

over time and answer a number of managerially relevant questions. Building on the Markov chain model, we construct a finite horizon stochastic dynamic programming model to study the monetary incentives of stay-or-leave decisions. The dynamic programming model computes the expected payoff of staying versus leaving at different stages of the career of military personnel, depending on employment opportunities in the civilian sector. We show that the stay-or-leave decisions from the dynamic programming model possess surprisingly strong predictive power, without requiring personal characteristics that are typically employed in classification approaches. Furthermore, the results of the dynamic programming model can be used as input in classification methods and lead to more accurate predictions. Overall, our work presents an interesting alternative to classification methods and paves the way for further investigations on personnel retention incentives.

Finally, a graph coloring approach to deployment sourcing addresses one of the external factors of personnel inventory behavior, deployments. The configuration of persistent unit deployments has the ability to affect everything from individual perceptions of service palatability to operational effectiveness. There is little evidence to suggest any analytical underpinnings to U.S. Army deployment scheduling and unit assignment patterns. This paper shows that the deployment scheduling and unit assignment (DSUA) problem can be formulated as an interval graph such that modifications to traditional graph coloring algorithms provide an efficient mechanism for dealing with multiple objectives.

Disclaimer

The views expressed in this dissertation are those of the author and do not reflect the official policy or position of the United States Army, Department of Defense, or the U.S. Government.

Acknowledgements

Thank you to my professors and my committee who have helped me during this long process, the students of the Leeds School of Business, and the Operations Division. Thanks to Jim Kelly, Jay April, Fred Glover, Marco Better and many others at OptTek Systems, Inc. for providing a vast amount of professional and academic knowledge pertaining to a variety of operations research and operations management subject areas, specifically in the area of simulation-optimization. A special thanks to Jose Ramirez who was alongside me during this journey from the first day. Having another military officer and friend in the program with me was critical to my success, especially as we navigated the same classes together and challenged each other's research ideas and methods.

I would also like to offer a special thanks to Professor Manuel Laguna for his leadership and guidance. Most of all, I want to give the biggest thanks to Christina Zais for her continual love and support during this entire academic process that actually began in 2006 when I started my first graduate school program.

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

The U.S. Army personnel system is constrained by a budget that often belies its own mission requirements. As a result, the force is stretched thin with a minimal surplus in capacity. At the same time, as the Army constantly realigns its personnel in a manner that fulfills as many requirements as possible, the dynamic nature of external effects and strategic policies cause these requirements to change. The perpetual motion of meeting this change with a personnel inventory that evolves at a slower rate produces the necessity to either anticipate change or maintain a level of surplus and flexibility that can adapt. Moreover, decisions can be incorporated at different points of the manpower system that maximize the utility (and minimize the cost) of an optimal surplus level, reduce uncertainty and influence the behavior of individuals within the system.

The U.S. Army historically maintains a workforce capable of only meeting defined static requirements. Expectations in the field are that all authorizations will be filled by skill and not lower than one grade below the authorization. If the Army is resourced for one face in every space then personnel friction prevents Army units from having the right number and types of Soldiers to meet mission requirements. In order to meet authorizations by grade and skill, workforce personnel must exceed requirements. If the expectation is to fill by at least grade and skill, the Army must have enough slack or flexibility in the manning system to compensate for uncertainty in the manpower demand (e.g., changes in force structure, deployment requirements, etc.) and the manpower supply (e.g., changes in policy, Soldier availability, etc.). It is fairly intuitive to show a method of measuring the level of personnel friction that exists; however, it is much harder to define

the flexibility required to compensate the impact of skill and grade imbalances as well as policy constraints on manning the force by grade and skill.

When the U.S. Army meets its overall strength goals, why are commanders still saying they dont have all the people they need to accomplish the mission? If there are enough faces to fill the spaces at a macro level (Army), why do they fail to match at the micro level (units)? Without careful analysis, the quick reaction may be to place sole blame on the inefficiencies of the personnel system. However, this reaction causes us to ignore a level of unavoidable misalignment that is inevitable in any organization with the size and personnel movement of the U.S. Army. Given a system comparable to just-in-time delivery with no inventory on the shelves, one could reasonably argue that the Army personnel system produces results better than should be expected.

Acceptance of the status quo encourages us to overlook policy changes that could reduce inefficiencies. Previous studies have referred to the concept of friction as the cost of doing business for organizations that have numerous specialties, ranks and locations such as the Army (Jehle[60]). A certain level of misalignment has always been viewed as a byproduct of a dynamic system with many moving parts. Even the staunchest critics of the Army personnel system would acknowledge that the distribution process (meeting the specific needs of both units and individuals) is an unenviable challenge. Unlike object-oriented supply chain systems, personnel systems do not grant the luxury of extra inventory stock to meet just-in-time needs. We are operating in a personnel system that is not designed for rapid adjustments or just-in-time delivery. However, as persistent conflict increases the stress on the force and the scale tilts towards demand over supply, there exists a greater need and urgency to maximize effectiveness and readiness. Years of persistent conflict have improved some measures of effectiveness by creating a better trained and more experienced force. Unfortunately, persistent conflict appears to compound personnel friction rather than reduce it.

Suppose we achieve a perfect level of personnel alignment to fill workforce requirements; what is the value if the requirements themselves are inaccurate or obsolete? Large organizations cannot even begin to address personnel inefficiencies, forecasting improvements or policy changes if demand driven requirements are not assessed with the same analytical rigor. In effect, accurately forecasting

to suboptimal workforce requirements is the same as hitting the bullseye of the wrong target. In this paper, we initially focus on an analytical approach to estimating workforce requirements while accounting for the uncertainty of demand. It is arguable that workforce requirements, while assumed as a given, are the critical foundation to all manpower models that effect the personnel system as a whole. With this in mind, it surprising that so much effort in research is devoted to personnel scheduling and so little focus is given to incorporating the uncertainty of demand when inputing requirements.

In addition to identifying and developing points of efficiency in workforce (structure) requirements, it is possible to shape the personnel inventory such that it aligns with requirements. Intuitively, the most common lever for shaping inventory is through accessions. However, influencing retention behavior is a means of shaping personnel inventory with reduced lag time in the event of sudden changes in demand. It is important to understand the tradeoff between accessions and retention. The former is a lever that the Army can use to compensate for the latter consequence of human behavior. Although retention can be influenced with incentives, its power as a point of leverage is restricted by budget constraints and incentive options. Therefore, the ability to accurately predict the stay-or-leave decision of individuals and attribute causal effects on behavior makes it possible to apply optimal policies to retention incentives, thereby maximizing retention effects and minimizing cost. A common misconception is to assume that if retention goals are not met, the Army can just react by adjusting its recruiting mission.

Elements of Markov chain estimation and dynamic programming can be applied to analyze the cost of decision policies that enhance the ability to predict the stay or leave decision. Ultimately, an ability to evaluate the value of incentives that may change a decision enables policy makers to influence retention and shape the force by skills and attributes. Underpinning any effective incentive policy is the need for a framework to identify an individual's propensity for continued service such that policies can be constructed without using discriminatory attributes for categorization.

It is certainly arguable that deployments are a significant external factor affecting personnel system, particularly retention behavior. Although deployments cannot be avoided or altered in

the context of the personnel management system, it is reasonable to suggest that there are certain deployment scheduling objectives that can influence behavior and predictive modeling. Moreover, modeling techniques can assist in optimizing a unit deployment sourcing plan to meet desired objectives that affect individual experiences. The personal system is large-scale and complex; therefore, modeling techniques which address all aspects of an individual's career, to include external factors, may residually influence career satisfaction and retention behavior.

In this research, each of these components is analyzed within the domain and context of Army personnel; however, the implications of this research can extend to any type or scale of workforce planning. The complexity and scale of the Army suggests that methods presented in this paper may be applicable to many types of organizations when tailored to strategic goals, requirements and constraints.

1.1 Current Manpower Modeling

The primary tool used by the Army to forecast enlisted strength is incorporated within the Active Army Strength Forecasting (A2SF) model. This software uses a complex combination of optimization and time series techniques to forecast monthly enlisted Army strength by Grade, Service Term Category (First-Term or Reenlistee), Months of Service (MSV) and gender. The A2SF model can incorporate 8 years of historical data to develop monthly loss forecasts by category over an eight year horizon.

A2SF itself is composed of three sub-models that are connected to each other. Output from the Individual Accounts (IA) model is an input for the Enlisted Grade (EG) model, which in turn produces input for the Enlisted Specialty (ES) model. Each model is composed of a complex combination of variables, constraints, and underlying modeling techniques. In simple terms, the IA model forecasts the composition of the Army that is not part of the Operating Strength; therefore, those Soldiers who are in one of the following states: training, transition (between units), holdees (patients and prisoners), or students. Incorporating IA output and monthly officer corps projection inputs from an external model, the EG model forecasts monthly enlisted strength by grade and

gender. The ES model then uses the EG output as an input for its forecast for enlisted strength by Military Occupation Specialty (MOS), also referred to as skill.

A2SF uses historical rates and time series techniques to forecast monthly loss rates by each major voluntary and involuntary loss category: Expiration of Term of Service (ETS), Retirement, Adverse Action, Administrative Action. Gains (accessions) are treated as model inputs that can be in the monthly form of fixed values or floating ranges that meet prescribed goals. The optimization portion of A2SF uses accessions seasonality and counts as levers for an objective function that attempts to minimize Operating Strength Deviation (OSD). Time series techniques are applied to every combination of loss category, skill, grade and MSV in order to produce loss rates that better capture the behavior of similar groups. Viewed as a large-scale network of states (grade, MSV, and gender), each node is treated as a variable in the optimized system. Several top level user-defined constraints are applied to the problem to comply with numerous policy and budget decisions such accessions seasonality, monthly strength and end strength requirements, and average annual strength goals. Overall, A2SF is an extremely large-scale optimization problem with over 2 million variables and 100,000 constraints.

1.1.1 Modeling Limitations

Despite the complexity and relative sophistication of A2SF, it is not without its limitations. The most obvious limitation is also the impetus of this analysis and is centered around the fact that loss rates are created entirely through the use of time series techniques. As a result, without manual manipulation, history is the only predictor of the future and time is the only variable that is ultimately used in generating the loss rates. Moreover, an entire database filled with potential predictive variables is discarded in favor of incorporating time as the only significant predictor. When time is the sole predictive variable, there are only three viable mechanisms for responding to known external effects:

(1) Allow enough history to accumulate to influence the future

(2) Manually suppress historical data

(3) Manually manipulate the forecast by applying multipliers to the time series rates based on the judgment of the user/analyst

The time series and optimization techniques used in A2SF possess strengths that are mostly seen in the ability to aggregately shape a force and meet the important objective function of minimizing aggregate Operating Strength Deviation. However, some limitations, as mentioned above, are masked within the aggregation of forecasted manpower in the post-procession phase. When losses are aggregated by type, grade, skill, and MSV, compensating errors often produce an acceptably accurate enlisted strength projection. Unfortunately, the underlying errors often limit or completely negate reliable analysis disaggregated categories and more importantly, precludes the ability to forecast individual specific behavior. Moreover, since time series rates are generated by aggregate versus individual specific data, conclusions cannot be made about predictive loss behavior in specific units or locations.

Ultimately current information provided as an output of A2SF is greatly limited in its ability to forecast uncertainty below an aggregate level and does little to prescribe a force mix or personnel surplus level that is most efficient and cost effective towards maximizing requirements that are met at sub-aggregate (location, unit, MOS) level. A motivating factor of this research is to provide a better understanding of how different segments of the Army personnel system or any workforce planning system for that matter can be leveraged with analytical concepts that are being researched and applied in other areas of industry. These concepts can provide the tools to give an organization targeted points within the system where a workforce can be efficiently shaped and structured in order to reduce cost, maximize flexibility and influence behavior.

1.1.2 Complexities of the Army Personnel System

The Army personnel system is one of the most large and complex organizations when viewed in comparison with other organizations throughout all industry. Moreover, it possess certain at-

tributes and unique policies that make it more difficult to model and manage than other organizations. In this section, several of these differences are outlined, which have lead many analysts to describe the Army personnel system as afflicted by the "curse of dimensionality". Specific aspects regarding the progression of individuals throughout the system, such as promotions, entitlements and separations are addressed in detail in latter sections when applicable to modeling objectives.

Undoubtedly, an organization as large as the Army (greater than $550,000$ in 2011) is difficult to model by reason of scale alone. Making the system more complex is the fact that there are over 180 different military occupation specialties (i.e. skill categories) and nine pay grades. These specific authorizations create over 800 skill/grade combinations. Considering the diverse number of assignment locations throughout the world, there are over 17,000 skill/grade/location combinations. Individuals are constantly entering the system, leaving the system, and being promoted. Moreover, the nature of Army service requires Soldiers to move from station to station at fairly regular intervals. Adding to the complexity, approximately 15% of enlisted Soldiers may leave the Army during any fiscal year, and more than half of those are generally unplanned due to adverse or administrative actions. Moreover, 30-40% of enlisted Soldiers historically fail to complete their first term of service. Because of necessary policy requirements that induce a certain level of predictability, and fairness, only 14-18% of individuals may be eligible for a change in assignment location at any given time (Zais[90]).

There are other aspects of the Army personnel system that make it unique in comparison to other organizations. Most obvious is the fact that the job requirements and the potential for harm during conflict can shape the behavior of individuals, especially pertaining to retention. Experiences for individuals with the same skills and attributes may vary depending on the frequency and duration of deployments, not to mention operational circumstances. Much research has been devoted to quantifying the effects of a wartime environment; the interpretations are varying, especially in regard to temporary versus persistent conflict.

In one aspect, the Army personnel system can be compared to a business that uses just-in-time delivery operations with no inventory on the shelves. The Army is resources with one"face"

for every "space"; therefore, it doesn't have the luxury to carry excess personnel inventory, referred to as a "bench" in some organizations. At times, the Army does not have enough slack or flexibility in the manning system to compensate for uncertainty in the manpower demand (e.g., changes in force structure, deployment requirements, etc.) and the manpower supply (e.g., changes in policy, Soldier availability, etc.) (Zais[90]). Adding to the issues of flexibility is the relatively nonexistent option for lateral entry. Imagine a civilian company, which must fill all of it's middle-management and senior level positions from within. Unlike almost any other organization, enlisted personnel must enter from the lowest level (Skill Level 1).

1.1.3 Tradeoff Between Accessions and Retention

As mentioned earlier, there exits a common misconception that unmet retention goals can be compensated with accessions requirements. Trading retention for accessions can produce residual ramifications that extend beyond the obvious negative impact of trading experience for new recruits. The Army, like most organizations desires a certain variance and composition of youth and experience. Shaping a force to meet these goals, ensures that there is enough leadership and experience to accomplish mission requirements and a junior level strength that can produces future leader. A tradeoff between retention and accessions can impede the Armys goal of shaping the force by skill and grade. Moreover, since enlisted Soldiers are promoted to meet requirements, a shortage in more senior enlisted categories merely produces Soldiers who are promoted too early and with too little experience.

The unmet precision of reenlistment goals can also produce problems when the accessions and retention balance swings opposite. An abundance of seniority and shortage of junior shortages can produce a immediate cohort bubble that has long-term leadership ramifications as the cohort progresses through the system. Additionally, over incentivising retention leads to two intuitive negative effects. Either the Army retains a number of Soldiers beyond its annual goal and experiences the above mentioned effects or it restricts further retention upon reaching retention goals. The latter may seem harmless, but ultimately results in higher budget costs than are necessary

and begs the question of how much money could the Army have saved and still met its goals. Lastly, a compounding budget effect of excess seniority is in direct contrast to the positive effects of experience and leadership. In a purely budget oriented context, despite the fact that a more junior force incurs higher training cost, maintaining a more senior force can be more expensive after incorporating pay and benefits. All of these factors add to the importance of developing models that can leverage predictive variables and allow the Army to forecast and influence behavior with as much precision as possible.

1.2 Army Manpower and Deployments

The third element of this paper is an essay related to deployment scheduling, specifically pertaining to the sourcing of units to meet demand from a designated sourcing pool. There are several significant external factors that have been proven to influence the behavior of individuals in Army personnel system. One such factor is overseas deployments. There is some evidence to suggest that the parameters and experiences of deployment influence the retention decision of personnel; however, some predictive models have yielded conflicting results. An analysis of historical deployment patterns also shows that units regularly deploy to different locations in subsequent deployments. There is a reasonable argument that the predictability and consistency of deployment locations for each unit affects their performance and readiness.

There is a significant modeling gap in the domain of deployment scheduling and sourcing. In periods of persistent conflict, the Army does not employ an analytical method to establish deployment schedules that optimize a single objective or multiple objectives. Past deployment schedules have been built around policies and constraints that lack strong analytical underpinnings. Many of the policies that dictate deployment schedules are either arbitrary or without an analytical foundation. In 2003, the Army established a policy for standard deployment lengths of 12 months; in 2011, deployments lengths were reduced to 9 months.

Many reports suggest that there are numerous negative ramifications ranging from stress on the force to medical issues that can be attributed to the frequency and duration of deployments. The

standard policy for time between deployments has historically been 24 months, or more accurately, a 1 to 2 ratio of time deployed to time at home-station. Like the length of deployments, policies pertaining to deployment frequency lack reliable analytical support in terms of the optimality of any criteria. More importantly, there is little evidence to suggest that unit sourcing mechanisms attempt to optimize on any criteria.

1.2.1 Army Force Generation

In 2009, Dabkowski et al. [24] published an analysis of unit and individual rotational stresses on the force due frequent deployments. They describe what the Army refers to the time an individual or unit spends deployed overseas in a combat environment, Boots on the Ground time or "BOG". Conversely, the time a Soldier or BCT spends between deployments is known as "dwell". Dabkowski et al. noted that the BOG:Dwell ratio at the time for Brigade Combat Teams was between 1:0.85 and 1:1.

In response to these ratios, which were viewed as unsustainable, the Army initiated programs and implemented policies aimed at improving these numbers. Most notably, the self-explanatory "Grow the Army" program (in conjunction with modularization) was implemented to increase the number of deployable units. Additionally, the Department of the Army (DA) issued several policies designed to protect minimum dwell time. Those policies stated BOG:Dwell ratio goals as 1:3 for steady-state demand and 1:2 for surge rotations.

As described by Dabkowski et al., concurrent with these programs and policies, the Army implemented a management support tool designed to fairly and equitably rotate units while providing the combatant commanders with the force necessary to meet our national security requirements. Army Force Generation (ARFORGEN) is a "structured progression of increased unit readiness over time resulting in recurring periods of availability of trained, ready, and cohesive units" (Department of the Army, [26]). However, due to the impact of new budget constraints in 2011, the Army began to downsize as directed by the National Defense Authorization Act (NDAA). While downsizing has corresponded with a decrease in operational demand, there little analytical evidence that the

management support tool efficiently optimizes any desired personnel and unit objectives.

1.2.2 End Strength

As noted above, the size of the Army affects the rotational capabilities of units in meeting operational demand. The size of the Army is constrained by a maximum end strength authorization dictated by annual NDAAs. End strength is specifically defined as the actual strength at the end of the fiscal year. Changes to the end strength constraint not only impact deployment planning but also have a significant impact on determining optimal workforce requirements and shaping the force with accessions and retention.

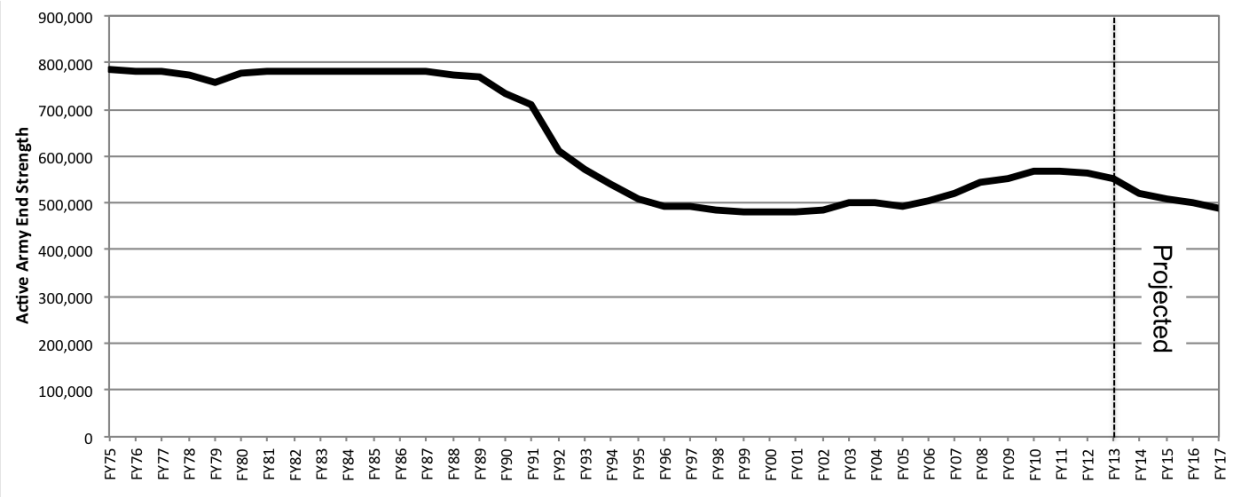


Figure 1.1: Active Army End Strength by Fiscal Year (1975-2017)

Figure 1.1 shows the Active Army end strength for fiscal years 1975-2013. Additionally, it shows projected end strengths as the Active Army downsizes to meet a required end strength of 490,000 by FY 2017 (Department of Defense [25]). The significance of historical end strength is in viewing that the size, composition and structure of the Army is dynamic and rarely in steady-state, ever responding to strategic requirements and sometimes conflicting constraints. Military manpower modeling within this domain is complex and involves many components working together systematically.

1.3 Research Overview

This research looks at three different aspects of the Army personnel system and proposes methods to improve the efficiency, timeliness and cost of personnel management. Viewed holistically, the Army manpower system is a integration of numerous elements that can be independently modeled. Identifying and closing gaps in modeling research can reduce workforce inefficiencies and costs. A predominance of military manpower models are focused on forecasting behavior and inventory within given demand requirements. Moreover, research directed towards predicting behavior is almost entirely disaggregated by pecuniary and non-pecuniary goals with disproportionate effort devoted to modeling the external factors that effect such behavior. This thesis is presented as three separate papers that propose modeling approaches to improve the management capabilities of the Army's manpower system.

A simulation-optimization approach to estimating workforce requirements examines the capabilities and limitations of Monte Carlo simulation and optimization methods within the context of workforce demand forecasting, modeling and planning. Specifically, we focus on these methods as a viable improvement for aligning strategic goals with workforce requirements. A general model is presented for estimating workforce requirements given uncertain demand. Using a real-world data example, we assess the benefits of this methodology to determine an optimal mix of workforce skills while providing the flexibility and robustness to incorporate uncertainty, assess risk and improve effectiveness of the holistic workforce planning process.

Using data from the US Department of Defense, we construct and estimate a Markov chain model of military personnel. Unlike traditional classification approaches, such as logistical regression models, the Markov chain model allows us to describe military personnel dynamics over time and answer a number of managerially relevant questions. Building on the Markov chain model, we construct a finite horizon stochastic dynamic programming model to study the monetary incentives of stay-or-leave decisions. The dynamic programming model computes the expected payoff of staying versus leaving at different stages of the career of military personnel, depending on employment

opportunities in the civilian sector. We show that the stay-or-leave decisions from the dynamic programming model possess surprisingly strong predictive power, without requiring personal characteristics that are typically employed in classification approaches. Furthermore, the results of the dynamic programming model can be used as input in classification methods and lead to more accurate predictions. Overall, our work presents an interesting alternative to classification methods and paves the way for further investigations on personnel retention incentives.

A dynamic programming approach to managing retention rates with incentives is presented as a response to a limited number of levers available for influencing retention behavior. This approach is logical compliment to statistical and data mining methods of predicting reenlistment behavior and specifically the stay-or-leave decision of enlisted individuals in the U.S. Army. It integrates the value function with non-pecuniary parameters identified by regression and classification techniques to maximize retention incentives within budget constraints. In isolation, modeling the Army career path as a finite Markov Decision Process is useful in determining an optimal decision policy for maximizing expected long-term earnings compensation. Moreover, discrete choice and logit models are capable of analyzing a quantifiable propensity to serve or taste-for-service based on determinants such as career history, demographics and family attributes. The integration of these methods provides a framework to understand and influence retention behavior in a holistic manner and enhance the targeting of incentive amounts to specific individuals based on a predicted propensity for continued service.

Finally, a graph coloring approach to deployment sourcing addresses one of the external factors of personnel inventory behavior, deployments. The configuration of persistent unit deployments have the ability to affect everything from individual perceptions of service palatability to operational effectiveness. There is little evidence to suggest any analytical underpinnings to U.S. Army deployment scheduling patterns. This paper shows that the deployment scheduling problem can be formulated as an interval graph such that modifications to traditional graph coloring algorithms provide an efficient mechanism for dealing with multiple objectives.

Chapter 2

A Simulation-Optimization Approach to Estimate Workforce Requirements

This simulation-optimization research is conducted in collaboration with OptTek Systems, Inc., which proposes to develop and market a software system for workforce demand forecasting and planning. The intent of OptTek's product development is to provide a tool that can help organizations optimize resources and align their workforce requirements with future demand for products and/or services while meeting or achieving strategic goals.

We first discuss the current gaps that exist in workforce planning, highlighting common forecasting methods used to obtain workforce requirements and previous research in the field. We then introduce simulation-optimization and a generalized model for estimating workforce requirements. Finally, we present a real-world application of this methodology to determine workforce requirements for a world-wide network of Army Warrior Transition Units (WTUs), medical units designed to provide personal support to wounded Soldiers who require at least six months of rehabilitative care and complex medical attention.

2.1 Workforce Modeling

A vast number of organizations use forecasting tools to predict their workforce needs. Most commonly, manpower forecasting models project future needs as a result of employee transactions and maturations such as reassignments, promotions, gains and losses. Losses, for example, may consist of disaggregate forecasts of multiple categories such as voluntary losses, involuntary losses, and retirements. Manpower forecasting allows organizations to proactively plan and meet requirements

while minimizing the costs associated with overstaffing or unmet requirements.

Forecasting methods are proven to be of vital importance to various organizations, small and large. The cost ramifications of either not meeting demand or maintaining an oversized workforce are relative to the size, mission and budget of each organization. For example, a small company that can operate with 90 employees but has 100 on staff wastes nearly 10 percent of personnel costs that could be used elsewhere. While this may only be thousands of dollars to a smaller company, costs escalate to millions of dollars for larger companies. Moreover, demanding that too few personnel perform too many tasks can lead to poor performance, inefficiency and unnecessary stress. An organization such as the U.S. Army uses mathematical modeling to forecast a dynamic active duty force that often exceeds half a million Soldiers in strength. Each 1% difference between required an actual enlisted strength can have budget implications in excess of $1 billion. Conversely, failing to meet requirements could result in readiness deficiencies or stress on the force.

The biggest deficiency, or rather gap, in most manpower forecasting processes is hidden in the baseline workforce requirements or workforce demand. While striving to align personnel inventory with workforce requirements, most organizations take workforce requirements as a given. Often, requirements are based on historical precedence or management directives, void of significant analysis. How does a company know the optimal size and disposition of its workforce in a given planning horizon, especially when tasks can be overlapped by skills? This question may seem manageable for a small organization that can easily observe and track the performance of daily tasks and has a way to extrapolate. However, it quickly becomes less practical and accurate when the organization grows larger and more complex.

Returning to the U.S. Army example, its requirements are extreme in terms of size and dimension. Afflicted by the curse of dimensionality, Army manning requirements are defined by over 1,000 possible skill and grade (rank) combinations. When allocated by specific location, these requirements expand to over 20,000 combinations. Determining these requirements is very complex, but the task is currently accomplished by a simple process of aggregating bottom-up requested needs within an overall budget constraint.

For a simpler context, it may be useful to envision a mid-size company with several thousand employees that produces a product based on demand. Rather than relying on the opinion of subject matter experts or trial and error, it is possible to use known tasks and job requirements to optimize a workforce mix that most efficiently balances demand uncertainty with risk. At times, some employees may be idle and capable of accomplishing other tasks; other times, the uncertainty of demand could lead to inflating workforce requirements in order to mitigate risk. By defining specified constraints for which tasks can be accomplished by each skill and the distribution of tasks within each skill, it is possible to apply techniques such as simulation and optimization (when applicable) to forecast optimal workforce requirements. Workforce requirements can be tailored by overall objectives such as reducing cost or meeting strategic goals.

2.1.1 Traditional Methods

Traditional workforce planning methods focus primarily on establishing the human resource (HR) policies that will result in a personnel inventory that meets a set of given requirements. This is done by trying to predict the behavior of individuals within the system. The goal is to generate policy decisions and constraints that influence behavior in such a way that requirements are met. These tools generally start by analyzing the current inventory. Characteristics of the current inventory are combined with historical data and trends to create rate distributions or time series techniques for transaction categories. Predictions are compared with workforce requirements to determine information about shortages or excesses.

Unfortunately, the commercialization efforts of many workforce planning software packages do little to align workforce requirements with strategic goals. Workforce requirements are usually embedded or fed as a known data input. However, it makes little sense to embark on the task of finding optimal HR policies to meet workforce requirements that are not accurate.

For example, consider a company that has determined that it requires to maintain a workforce of 1,000 employees for the next 5 years. The company may use tools to configure a set of hiring, retention and promotion policies to meet the needs for different skills in multiple locations.

Regardless of the sophistication of the methods used to tailor a skilled workforce, this company may be aiming at the wrong target if the requirements are not correct. While management strives to continuously maintain its 1,000 positions, the uncertainty of product/service demand could result in changing workforce requirements. Perhaps, a framework for aligning strategic goals with workforce requirements in conditions of uncertainty may result in a more efficient distribution of skills for a workforce of 900 employees. This reality becomes more apparent as the global demand for certain products and services grows along with a scarcity for qualified personnel with the skills to deliver those products and services.

2.1.2 Previous Research

There seems to be little evidence of any published research that has been translated into commercial software for predicting workforce requirements, especially when trying to link workforce needs to strategic goals. While an abundance of technology exists to forecast the behavior and maturation of a workforce, organizations have few tools to predict workforce requirements that account for the introduction of new products, new technology or changing strategies. Effectively allocating resources and anticipating workforce needs requires data-driven approaches.

Van den Bergh et al. [86] provide a comprehensive survey of the vast amount of research devoted to personnel scheduling problems, which generally focus on the more detailed problems of staffing and/or scheduling of workers given fixed inputs. They advocate for more research devoted to integrating demand forecasting and multiple locations, noting that most research features deterministic approaches that do not address the uncertainty of real-world personnel scheduling problems. Van den Bergh et al. state that in "situations where uncertainty has a strong effect on the personnel schedule, such as volatile demand or last-minute changes, it could prove beneficial to incorporate this uncertainty in the decision-making process" and evaluate the robustness of solutions by simulating the stochastic behavior of demand.

There are some previous implementations of simulation with personnel scheduling problems, most of which favor discrete event simulations (DES) over Monte Carlo simulation (Van den Bergh

et al. [86]). For example, Qi and Bard [77] demonstrate an approach to coupling simulation with optimization in an iterative fashion. Their application is very specific to the mail handling industry and uses simulation to validate schedules and estimate labor requirements based on a given schedule; however, simulation is not used to estimate demand. Harper et al. [51] analyze the optimal size and skill mix of hospital nursing teams to meet demand predictions from simulation while trying to minimize costs. Their approach of combining discrete event simulation and stochastic programming is novel to the nurse scheduling problem, but differs from the simulation-optimization method that we present in that Harper et al. do not not optimize based on numerous simulated scenarios. Rather, they adopt a stochastic programming approach and evaluate requirements from demand over each year.

The research of Yang et al. [89] shares our focus of including cross-training as part of the scheduling problem. They use Monte Carlo simulation to generate demand for jobs. However, their remains fixated on the more specific hourly scheduling problem and comparison to flexible workdays instead of generating long-term workforce requirements.

Brucker et al. [13] address the complexity of personnel scheduling models and particularly respond to the claim of Ersnt et al. [38] that "new models need to be formulated that provide more flexibility to accommodate individual workplace practices." Brucker et al. note that personnel scheduling problems can be formulated as integer linear programs, which can be solved with LP-solvers or heuristics that combine local search and network flow techniques, depending on complexity. They also state that when task changes are allowed, the problem of assigning shifts to employees and employees to tasks to meet demand can be efficiently solved by network flow algorithms.

While most existing research in workforce requirement planning is specific to a narrow domain of professions, Thompson [84] focused on methods to translate forecasted demand into employee requirements for service organizations in general. His research describes and compares two methods of determining the required number of employees for an organization given the common objective of finding the smallest number of staff that will deliver a threshold level of customer service for each

planning period. Thompson presents evidence to support a hypothesis that an M/M/s/K queuing model with stationary and steady-state assumptions is inferior to one that explicitly accounts for service duration and customer waiting time during the process of setting employee requirements (Thompson). The latter model incorporates an awareness that demand for a particular period may affect requirements in future periods, and accounts for a reduction in task completion when demand arrival time nears the end of an arrival period. Although Thompson's method addresses the uncertainty of demand, it does not offer a framework for optimally searching among a vast set of skill mix and size combinations to meet this demand uncertainty over a planing horizon.

In the infancy of strategic workforce planning, analytical tools for talent management and demand forecasting are not nearly as common or sophisticated as those used for other areas such as logistics. Huang et al. [58] developed a simulation model called SimMan, which is one of the early attempts to use simulation to predict the performance of workforce capacity planning solutions. They classify workforce capacity planning into three categories:

(1) Operational Scheduling

(2) Tactical Planning

(3) Strategic Planning

Since most workforce capacity planning research is based on mathematical models that use aggregate demand over certain time periods, Huang et al. propose that simulation is needed to address uncertainty and predict the performance of the solutions derived from workforce planing models. They develop a discrete event simulator to evaluate the effectiveness and robustness of different planning alternatives and assignment rules (Huang). A shortcoming of SimMan is that it does not provide a built-in capability for optimizing skill requirements to meet simulated demand. This makes the comparison of policies and alternatives less efficient than the methods proposed in following sections.

SimMan uses the C^{++} language and a modular structure to model service companies, such as consulting, with project demands. The five modules in SimMan (simulator, workforce planning,

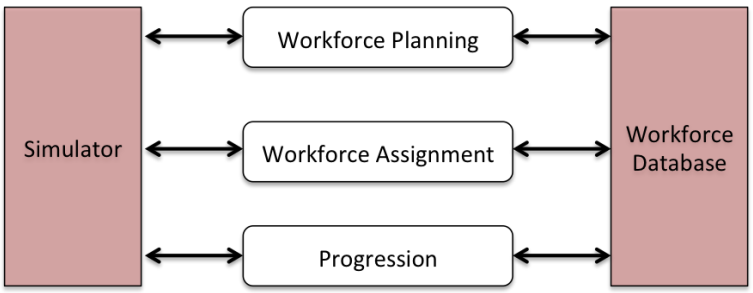


Figure 2.1: Framework of SimMan

workforce assignment, progression and workforce database) are shown in Figure 2.1. The expected values and variances of demand are estimated using Monte Carlo sampling. Within the modular structure of SimMan, rules are coded independently and used to compare existing and different planning alternatives and assignment rules (Huang, 2008).

Other research in the field of manpower planning under requirement uncertainty tends to be limited to scheduling optimizations (Bazargan-Lari et al. [19]), optimal accessions policies (Grinold [49]) or unvalidated readiness requirements, which do not adequately evaluate the cost-benefit of a large number of scenarios (Moore et al. [71]).

2.2 Simulation-Optimization

The remainder of this paper describes the methodology and real world data used to test the design framework for predicting workforce requirements. The test model is populated with real organizational data and uses a combination of Monte Carlo simulation and optimization. Briefly, we discuss how both Monte Carlo simulations and optimization methods can be used for strategic workforce planning.

The combination of simulation and optimization provides an opportunity to account for various forms of uncertainty while searching for optimal solutions. Some optimization software companies, such as OptTek Systems Inc., have recognized the value of employing simulation with optimization. OptTek states the following justification for merging simulation and optimization

(www.opttek.com):

> A principal reason underlying the importance of simulation optimization is that many real-world problems in optimization are too complex to be given tractable mathematical formulations. Multiple nonlinearities, combinatorial relationships and uncertainties often render challenging practical problems inaccessible to modeling except by resorting to simulation - an outcome that poses grave difficulties for classical optimization methods. In such situations, recourse is commonly made to itemizing a series of scenarios in the hope that at least one will give an acceptable solution.

Some researchers distinguish between "Simulation for Optimization" and "Optimization for Simulation" when describing related literature. In the context of discrete-event simulation, optimization tends to perform a subservient role, in which the optimization routine is an add-on to the underlying simulation engine. This case would be viewed as optimization for simulation. In contrast, developments in stochastic programming are more in line with simulation for optimization, where "Monte Carlo simulation is the add-on used to generate scenarios for math programming formulations from a relatively small underlying set of possible realizations" (Fu [39]).

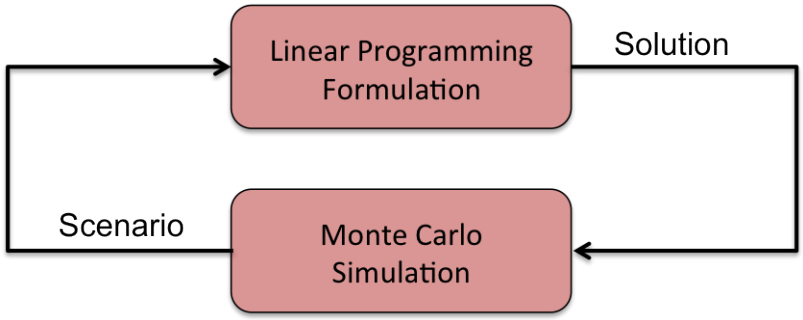


Figure 2.2: Simulation for Optimization (Fu, 2002)

2.3 General Model

There are numerous internal and external factors that affect how a particular workforce mix performs when measured against the goals of an organization. Although the exact affect to the performance of a workforce mix is not known with certainty, it may be estimated with probabilistic

models. The basis of the general model is to represent the relationship between a workforce mix and a set of external and internal factors to estimate the degree by which an organization's goals are satisfied.

The general model consists of optimization and simulation components embedded in a process as depicted in Figure 2.3. Monte Carlo simulation is an attractive method to model the uncertainty of key data; the optimization process measures the degree of satisfaction of organizational goals while incorporating a tradeoff component such as return of investment.

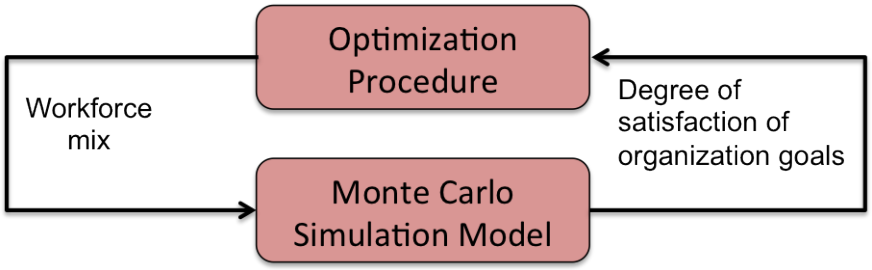


Figure 2.3: Simulation-Optimization Process

The remainder of this section details the progressed construction of a general model as an expansion of the closed-form base model which can incorporate uncertainty and remains adaptable to the data measurements of an organization. Specifically, we construct a task-oriented general model and an aggregate general model, each possessing capabilities and attributes that may be practical and amenable to specific organizational data and strategic goals.

2.3.1 Base Model

The base model is built by transforming assumptions on future demand for products or services into labor needs. The model is constructed with the following known or estimated parameters:

d_{ik}**:** demand in time period i for product k

t_{ijk}**:** time required in time period i of task j to produce a unit of product k

a_{il}**:** time availability in time period i for workers of skill l

p_{jl}**:** percentage of task j performed by a worker with skill level l ($\sum_l p_{jl} = 1$)

The workforce requirement in period i for workers with skill level l (denoted by r_{il}) is calculated as follows:

$$r_{il} = \frac{\sum_j p_{jl} \sum_k d_{ik} t_{ijk}}{a_{il}} \tag{2.1}$$

Due to the structure used for associating products/services to tasks and tasks to skill levels, this model does not require optimization to determine the mix of workers at each skill level. More specifically, the mix of skill levels is determined by p and t while the number of workers is driven by d and a.

2.3.2 Base Model with Uncertainty

The demand component is the main source of uncertainty in the base model, especially when products or services are new. To address the uncertainty of demand data, Monte Carlo simulation may be superimposed on the base model. Demand assumptions are treated as random variables rather than point estimates, and each random variable follows a specified probability distribution. Because demand is usually not independent, a correlation matrix indicating the dependencies among demands is included in the model.

2.3.3 Task-Oriented General Model

Extending the base model, an optimal workforce mix model assumes that tasks may be performed by more than one combination of workers with different skill levels. Hence, there is no longer a unique distribution of work per skill level. Instead, p_{jl} is reinterpreted to be the maximum percentage of task j that a worker with skill level l is capable of accomplishing. The problem is now described as finding the total workforce per period and the optimal mix of resources, where

c_{il}**:** cost in period i of a worker with skill level l

x_{il}**:** number of workers in period i with skill level l

y_{ijl}**:** time in period i that a worker with skill level l devotes to task j

Initially, if demands are assumed to be known or estimated with an expected value, the deterministic version of the problem is as follows:

Minimize

$$\sum_i \sum_l c_{il} x_{il} \tag{2.2}$$

subject to

$$\sum_{l:j\in J_l} y_{ijl} \geq \sum_k d_{ik} t_{ijk} \qquad \forall\, i,j \tag{2.3}$$

$$\sum_{j\in J_l} y_{ijl} \leq a_{il} x_{il} \qquad \forall\, i,l \tag{2.4}$$

$$y_{ijl} \leq p_{jl} \sum_k d_{ik} t_{ijk} \qquad \forall\, i,j,l : j \in J_l \tag{2.5}$$

$$x_{il}, y_{ijl} \geq 0 \qquad \forall\, i,j,l$$

where J_l is the set of tasks that workers with skill level l are able to perform.

The objective of (2.2) is to minimize the total cost of the workforce over the projected time horizon. Within the constraints, (2.3) represents the need to have enough workers to meet the skill demands for each product or service. Constraint (2.4) ensures the amount of time devoted to task j from skill l does not exceed the amount of time available from workers of skill l. Finally, (2.5) limits the amount of time a worker of skill l is able to devote to task j as a defined percentage of demand for all tasks within the worker's skill set.

2.3.3.1 Controlling Skill Flexibility

The model as it exists above possesses some unrealistic elements. Most noticeably, the flexibility of the model allows for drastic changes in requirements between time periods. Significant changes in requirements from year to year may not be implementable or desirable. In fact, changes

in the workforce from one period to another incur in costs associated with hiring/training and firing.

An extension of the base model introduces transaction costs for each change in requirements and enables a certain percentage of skill requirements to be replaced by other capable skills through cross-training. The extended model employs the following parameters:

b_{il}**:** transaction cost in period i of each requirement change for skill l

δ_l**:** penalty factor associated with training new personnel to skill level l

The problem remains a minimization of cost in which the objective function is expanded and represented as

$$\sum_{l=1}^{r}\left(\sum_{i=1}^{n}\left(c_{il}x_{il}\right)+\sum_{i=1}^{n-1}\left(b_{il}|x_{i+1,l}-x_{il}|\right)+\delta_l\sum_{i=1}^{n-1}\left(x_{i+1,l}-x_{il}\right)^{+}\right) \tag{2.6}$$

where n is the number of time periods in the model and r is the number of unique workforce skills.

The first term of the equation above, $\sum_{i=1}^{n}\left(c_{il}x_{il}\right)$, represents the total salary cost of a single skill during the planning horizon. The next two terms, $\sum_{i=1}^{n-1}\left(b_{il}|x_{i+1,l}-x_{i,l}|\right)$ and $\delta_l\sum_{i=1}^{n-1}\left(x_{i+1,l}-x_{i,l}\right)^{+}$, represent the total cost of all personnel transactions for a single skill and the total penalty cost of training a new individual respectively, where $|x|$ is the absolute value of x and x^{+} is equal to x if $x>0$ and equal to 0 if $x\leq 0$.

We introduce two sets of non-negative variables, $v^{+}_{i,i+1,l}$ and $v^{-}_{i,i+1,l}$, in order to linearize the objective function and turn the problem into a linear program. The adjusted task-oriented model places a stronger emphasis on reducing drastic changes and can be represented as follows:

Minimize

$$\sum_{l=1}^{r}\left(\sum_{i=1}^{n}\left(c_{il}x_{il}\right)+\sum_{i=1}^{n-1}b_{il}\left(v^{-}_{il}+v^{+}_{il}\right)+\delta_l\sum_{i=1}^{n-1}v^{+}_{il}\right) \tag{2.7}$$

subject to

$$\sum_{l:j\in J_l} y_{ijl} \geq \sum_k d_{ik}t_{ijk} \qquad \forall\, i,j \tag{2.8}$$

$$\sum_{j\in J_l} y_{ijl} \leq a_{il}x_{il} \qquad \forall\, i,l \tag{2.9}$$

$$y_{ijl} \leq p_{jl}\sum_k d_{ik}t_{ijk} \qquad \forall\, i,j,l : j \in J_l \tag{2.10}$$

$$v_{il}^{-} \geq x_{il} - x_{i+1,l} \qquad \forall\, i,l \tag{2.11}$$

$$v_{il}^{+} \geq x_{i+1,l} - x_{il} \qquad \forall\, i,l \tag{2.12}$$

$$v_{il}^{+}, v_{il}^{-}, x_{il}, y_{ijl} \geq 0 \qquad \forall\, i,j,l$$

2.3.3.2 Network Interpretation

The application of simulation-optimization is chosen in the latter sections of this paper to estimate an organization's ability to meet workforce mix goals under the uncertainty of external and internal factors. However, for conceptional awareness, it is useful to note that this model can be represented and solved as a network model given an engine to replicate uncertainty.

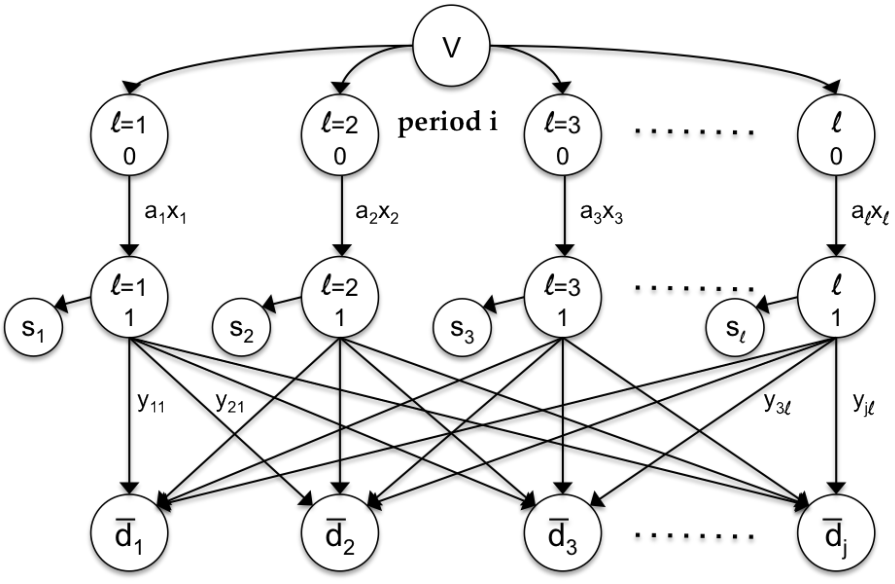


Figure 2.4: General Workforce Network (single period)

Figure 2.4 depicts a single time period of the task-oriented general model from Section 2.3.3 where V represents a source node and each flow of y_{jl} has an upper bound (arc capacity) given by the right-hand side of the inequality of equation (2.10). Workforce flow not carried on the y_{jl} arcs is deposited in the appropriate surplus node, s_l. Additionally, in Figure 2.4, we represent the demand for each skill from the right-hand side of equation (2.8) with the simplified notation $\bar{d}_j$.

2.3.4 Aggregate General Model

Many organizations are either unable or unwilling to break down products or services by tasks. In such cases, the task-oriented model presented in Section 2.3.3 is not practical, but the general model can be modified to address a more aggregate set of demand requirements.

The previous task-oriented model is modified by removing allocations that disaggregate the time requirements and the availability of tasks and workers respectively. In the absence of these task and time parameters, the aggregate model incorporates new constraints which ensure task fulfillment and limit cross-skill coverage in accordance with an organization's policies or strategic goals. The model is constructed with the following additional variables and parameters:

w_{ikl}**:** number of workers with skill l in time period i filling demand for skill k

D_{il}**:** demand in time period i for skill l

M_{ikl}**:** maximum number of workers with skill l to be used in time period i to meet demand for skill k (for $l \neq k$)

While demand remains a source of uncertainty, products and services are mapped to specific skill requirements. Moreover, the model remains flexible enough to accommodate a certain percentage of skill overlap. Some skill pairings are identified as capable of cross-training and may be replaceable in a limited capacity to address temporary demand fluctuations.

Replacing three constraints from the model in Section 2.3.3, the aggregate model takes the form

Minimize

$$\sum_{l=1}^{r}\left(\sum_{i=1}^{n}(c_{il}x_{il})+\sum_{i=1}^{n-1}b_{il}\left(v_{il}^{-}+v_{il}^{+}\right)+\delta_l\sum_{i=1}^{n-1}v_{il}^{+}\right) \tag{2.13}$$

subject to

$$\sum_{k=1}^{r}w_{ilk} \geq D_{il} \qquad \forall\, i,l \tag{2.14}$$

$$w_{ikl} \leq M_{ikl} \qquad \forall\, i,k,l \tag{2.15}$$

$$\sum_{k=1}^{r}w_{ikl} \leq x_{il} \qquad \forall\, i,l \tag{2.16}$$

$$v_{il}^{-} \geq x_{il}-x_{i+1,l} \qquad \forall\, i,l \tag{2.17}$$

$$v_{il}^{+} \geq x_{i+1,l}-x_{il} \qquad \forall\, i,l \tag{2.18}$$

$$v_{il}^{+}, v_{il}^{-}, x_{il}, w_{ikl} \geq 0 \qquad \forall\, i,j,l$$

where (2.14) enforces strategic goals and policy standards (e.g. specific skill-demand requirements), (2.15) limits the amount of cross-skill coverage and (2.16) ensures that the number of skill l fulfilling all tasks does not exceed the number of skill l available. As before, (2.17)-(2.18) enforce negative and positive transactions, which account for adding or removing requirements at each skill and location.

The structure and interpretation of each term in the objective function remains the same as before. Moreover, this version of the general model assumes that demand is location specific and cannot be redistributed between locations. For simplicity and practicality, this is a reasonable assumption for many organizations. As a result, the model decomposes by location; however, if demand can be transferred between locations, decomposition no longer remains valid.

There may be situations where resources or demand are centralized. In such case, a non-decomposable model must be solved simultaneously as a total system in which q represents each specific location and where there are a total of N locations. We do not present a non-decomposable model in this paper because it does not apply to the real-world scenario in Section 2.4, but it would require adding a cost component to the objective function that penalizes transferring demand or

workers among locations.

The aggregate model presented in (2.13)-(2.18) is a general and high-level interpretation of the simulation-optimization approach to estimating workforce requirements. Because the model decomposes by location, we can evaluate the system holistically or focus analysis on a specific location. As a proof of concept, we use this version of the model in the next section to demonstrate a real-world application where we analyze the benefits of being able to estimate workforce requirements and evaluate risk under conditions of uncertainty.

2.4 WTU/CBWTU Modeling

The purpose of applying the model in this section is to determine the optimal workforce requirements needed to manage a network of Army Warrior Transition Units (WTUs) and Community Based Warrior Transition Units (CBWTUs). WTUs and CBWTUs are medical units designed to provide personal support to wounded Soldiers who require at least six months of rehabilitative care and complex medical attention. The population, number of patients (Soldiers) assigned to each unit, varies and fluctuates over time. The size and skill disposition of cadre[1] needed to manage each unit is dependent on the size of each WTU population.

Conceptually, a variety of forecasting models may be used to project workforce (cadre) needs and behavior over a time horizon; however, these workforce requirements are not subject to the same sources of uncertainty as other manning requirements throughout the Army. While the manning requirements of other Army operational and support units are subject to complex mission goals influenced by global deployment and contingency factors, WTU requirements are predicated by patient demand. As a result, different analytical methods are needed to ensure that cadre workforce requirements themselves are accurate. In this section, we apply a version of the aggregate model (Section 2.3.4) to forecast workforce requirements; subsequently, existing methods can be applied separately to model and forecast a desired workforce inventory that aligns with strategic goals and

[1] **Cadre** refers to all personnel required manage medical, logistical and administrative operations of a WTU/CBWTU facility

risk tolerance.

2.4.1 WTU Data

The data for this model is obtained from the Medical Operational Data System (MODS). From this database, one is able to collect records from all current and historical WTU patients. As of June 2012, the MODS database contained over 47,000 records of past and current WTU patients since October 2007. Most important to the modeling process are the fields associated with WTU location, arrival date and departure date, when applicable. Databases are constructed to determine the monthly historical populations of each WTU and the respective monthly patient arrival counts, departure counts and time-in-system.

For the WTU model, we rely principally on the historical population data and use time series methods to forecast monthly population by location over a 24 month horizon. Each of the monthly forecasts is fixed with a probability distribution, which will be discussed later.

Other important data used for the model development are interpreted from cadre authorization guidelines obtained from the Warrior Transition Command (WTC), which serves as the lead proponent for the Army's Warrior Care and Transition Program. These guidelines, while not necessarily documented in policy, serve as minimum standards for cadre manning by skill position.

A WTU can consist of cadre from 47 unique skill categories. The minimum requirements for each skill are dictated by the size of the patient population and may be either a fixed number or a proportional ratio. These standards are complicated by the fact that they also deviate based on the structure category of the specific WTU.

For example, a WTU with a small population may only require the cadre standards to support a single company structure. However, a larger WTU that requires multiple companies will not simply be proportional. Instead, it will requires additional cadre structure to support a battalion headquarters that manages these companies. In total, there are eight possible unique structures.

2.4.2 Defining the Problem

As noted earlier, the intent of this model is to assist personnel structure decision makers in determining the optimal WTU cadre, i.e., workforce requirements to meet specific strategic goals. These strategic goals encompass meeting a desired level of patient care while minimizing total personnel cost. The standard level of patient care is defined by constraints associated with minimum cadre demand policies. Initially, the network of WTU cadre requirements can be modeled as a deterministic problem with the assumption that patient demand is known. Using formulated minimum skill constraints based on known demand, optimization may be applied to determine the optimal cadre mix for each WTU location that will satisfy an objective to minimize personnel cost. While an extension of the general workforce model, the WTU cadre model is like most workforce models in that it possesses unique characteristics and parameters.

Assuming demand at each of N locations is independent and non-transferable between locations, the problem consists of N connected models ($N = 40$ for the WTU/CBWTU network) which can be solved independently.

2.4.2.1 Personnel Costs

The objective of the WTU Cadre model as it is currently constructed is to minimize total cost. In this context, total cost is a function of the pay scale amount for each individual by skill as well as the transaction costs caused by gaining or losing cadre authorizations. We can refer to the transactions as structure transaction since the total cost function does not include other personnel transactions, such as reassignments and attrition that occur normally in a dynamic military. The objective function represents the total personnel cost associated with basic pay and structural transactions for all WTU locations during the planning horizon. The salary costs from the general model, c_{il}, incorporate base military compensation; in this application, b_{il}, represents cost associated with transaction, which can include permanent change of station (PCS) entitlements and transportation costs.

2.4.2.2 Cadre Policies

The population of wounded Soldiers determines the minimum cadre requirements (m_{il}) by the application of the WTC guidelines. Moreover, the policies allow for some skills to fill the role of other skills, but overlap is limited by M_{ikl}, which is set through policy.

The translation of demand into requirements is more difficult and potentially more contestable when using the aggregate model. While the task-oriented model provides a direct linkage of tasks to skills by mapping time and availability, the aggregate model is generally dictated by external guidelines or empirical analysis. In a perfect world, external guidelines are driven by timely empirical analysis; as products and services evolve, organizations should continuously revaluate policies that define these parameters.

In the case of Army WTUs/CBWTUS, the WTCO defines fairly rigid guidelines regarding the minimum structure requirements for a given level of demand. Patient demand has a direct correlation with the number and composition of units, which can create a somewhat piecewise linear mapping between demand and requirements. Moreover, the command structure is also subject to change, which can cause variations in the number and types of administrative personnel, or an absence of a headquarters staff altogether.

2.4.2.3 Population Uncertainty

Because the population of the patients is not certain, a stochastic element must be introduced to define the parameters used to model policy constraints. Capturing the uncertainty of each WTU location is important because minimum skill level standards are applied to cadre strength based on population size. To accomplish this with a degree of statistical confidence, time series techniques are applied to historical data in order to forecast individual WTU population sizes for the next n periods in the planning horizon. Each monthly data point is subsequently fitted with a probability distribution to replicate a confidence interval for the time series forecast. The general process is depicted in Figure 2.5 and is replicated for each WTU location.

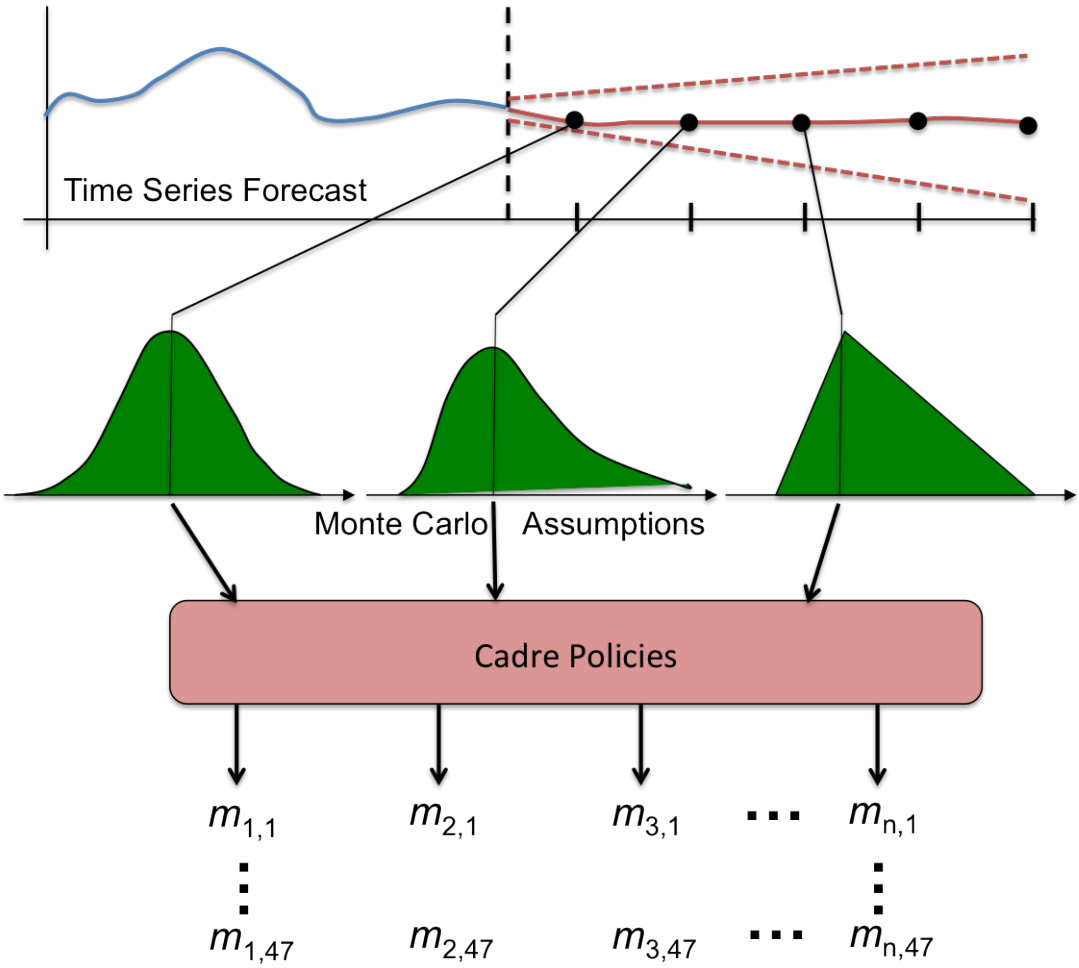


Figure 2.5: Modeling Population Uncertainty

In this practical application, time series methods are chosen to forecast demand rather than using discrete-event simulation based on parameters for arrival rates, departure rates and time in system. Time series is preferred for this specific application due to concerns about the applicability of historical arrival and time-in-system rates as a result of known policy changes designed to influence these rates. In many situations an alternative forecasting method may be be more practical, especially when history is not a reliable indicator for future behavior. A reasonable argument can be made that in the absence of persistent conflict, a time-series forecasting method is not preferable.

The historical monthly populations of each WTU and CBWTU location are analyzed from October 2007 to June 2012 and forecast using the best available seasonal, non-seasonal or ARIMA time series technique. Root Mean Square Error (RMSE) is used as an error measure for comparison. Generating a 95% confidence interval for a two year forecast horizon, Table 2.1 displays the population forecasting techniques chosen for each location.

Table 2.1: Time Series Techniques

Location	Technique	RMSE	BIC	Location	Technique	RMSE	BIC
1	ARIMA(0,2,1)	3.1964	2.398	21	ARIMA(1,1,0)	31.5753	6.978
2	Double Exponential Smoothing	7.8290	-	22	ARIMA(0,1,1)	3.7196	2.700
3	ARIMA(0,2,1)	7.9754	4.227	23	ARIMA(2,1,1)	5.2992	3.554
4	ARIMA(1,2,0)	6.4180	3.792	24	ARIMA(0,2,1)	5.1015	3.333
5	ARIMA(1,1,0)	11.0148	4.871	25	ARIMA(1,1,0)	11.0995	4.887
6	ARIMA(1,1,1)	9.0904	4.560	26	ARIMA(2,0,1)	5.7825	3.797
7	ARIMA(0,2,1)	9.5952	4.596	27	ARIMA(2,1,1)	20.2516	6.235
8	ARIMA(1,1,1)	3.4320	2.612	28	ARIMA(0,2,1)	6.4115	3.790
9	ARIMA(1,1,0)	6.0605	3.676	29	ARIMA(2,0,1)	9.6914	4.830
10	ARIMA(1,2,1)	8.3014	4.381	30	ARIMA(0,1,0)	6.6044	3.775
11	ARIMA(1,1,1)	6.0157	3.734	31	ARIMA(2,1,1)	15.0123	5.636
12	ARIMA(1,1,0)	16.0470	5.624	32	ARIMA(0,2,1)	19.5055	6.015
13	ARIMA(1,1,1)	11.5557	5.040	33	ARIMA(1,1,0)	6.9778	3.958
14	ARIMA(2,2,1)	15.9666	5.763	34	ARIMA(2,1,2)	11.5307	5.181
15	ARIMA(2,0,1)	22.9972	6.558	35	ARIMA(1,1,2)	5.1010	3.477
16	ARIMA(2,0,1)	20.1602	6.295	36	ARIMA(2,1,1)	6.1675	3.857
17	SARIMA(1,0,0)(0,0,1)	7.3367	4.201	37	ARIMA(1,2,1)	12.4304	5.188
18	ARIMA(0,2,1)	17.2883	5.774	38	ARIMA(1,1,0)	6.9541	3.952
19	ARIMA(2,0,1)	5.8034	3.804	39	ARIMA(0,2,1)	13.7366	5.314
20	ARIMA(1,1,0)	14.5462	5.428	40	ARIMA(1,1,0)	5.1712	3.359

In order to incorporate the demand uncertainty from our time series population forecasts, the

data points are imbedded as distributions with respect to the confidence interval. A Monte Carlo simulation is used to produce demand values based on these Triangular distributions are used in lieu of normal distributions in order to prevent potential extreme values that would be unrealistic for the patient populations being modeled. This method of simulating time series forecasts within an associated confidence interval is depicted in Figure 2.6.

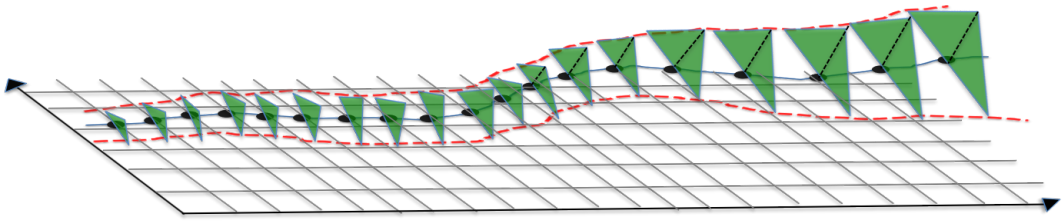


Figure 2.6: Simulation of Time Series Forecast Uncertainty

The simulation and optimization components of a single WTU location are comprehensive, amassing over 7,800 variables, 1,600 constraints and 14,000 bounds. Because each WTU location can be modeled independently, the simulation-optimization operations are conducted in separate sequential modeling steps. Each WTU location is forecasted for optimal cadre requirements, then repopulated with data from the next WTU location.

The combination of Monte Carlo simulation and cadre policies produces estimates for $m_{i,l,q}$. Given a particular instantiation of these values, optimization allows us to determine a cadre structure that will minimize the personnel cost for a WTU over a specified time horizon. Numerous simulation-optimization trials allow us to incorporate uncertainty and provide statistical context to an optimal solution. The process is repeated for all 40 WTU locations and results, discussed in Section 2.5, can be analyzed separately by location, skill or an aggregation each.

2.5 Computational Results

The results in this section represent a portion of the statistical and quantitative analysis that can be obtained from executing a large number of trials with this model. We obtain the cadre

structure of each WTU location by performing 500 trails of simulation-optimization. From these trials we are able to represent the statistical likelihood of each cadre estimate by location, skill or combination of both. Although each location is modeled independently, the same number of trials and the same sequence of random numbers are used in order to facilitate the aggregation of WTU statistics.[2]

The computational results in this section are generated using simulation-optimization with Microsoft Excel as the base platform for data inputs and data management. Computation time and intensity may dictate the use of more powerful software and programming languages for larger-scale problems; however, this specific scenario demonstrates how this methodology can be applied with commonly used software packages, especially when components of systems can be decomposed for analysis (e.g. decomposition by location). Oracle Crystal Ball serves as the Monte Carlo simulation engine for the sources of uncertainty in the WTU/CBWTU model.[3] Additionally, we use Visual Basic programming functions to conduct optimization procedures and solve the linear programs within each simulation trial using Frontline Premium Solver Pro software.

The use of simulation-optimization as a tool for estimating workforce requirements provides an organization with a multi-dimensional perspective for making informed strategic decisions that balance uncertainty and risk. Despite many perspectives of parameter and variable analysis, the goal of this specific problem is to minimize cost.

Visually, the effects of uncertainty are depicted in Figure 2.7, a three-dimensional plot of cost with respect to simulation trials and optimal cadre requirements for one quarter.[4] Figure 2.7 is not particular useful for analysis purposes; however, this section demonstrates an array of questions that can be answered based on an analysis of simulation-optimization trials.

Initially, interest may be focused on the confidence of aligning strategic budget goals with optimal operational requirements. Figure 2.8 shows the normality of the total cost solution space,

[2] Using the same sequence numbers ensures that the formula applied to generate a series of random numbers for the input probability distributions consistently uses the same initial seed value to generate these numbers and generates consistent output statistics among independent simulations.

[3] The selection of simulation software is also predicated on the type and source of uncertainty; for example, discrete-event simulation would require an alternative software package.

[4] Calculated costs are derived from military pay tables and are intentionally not displayed.

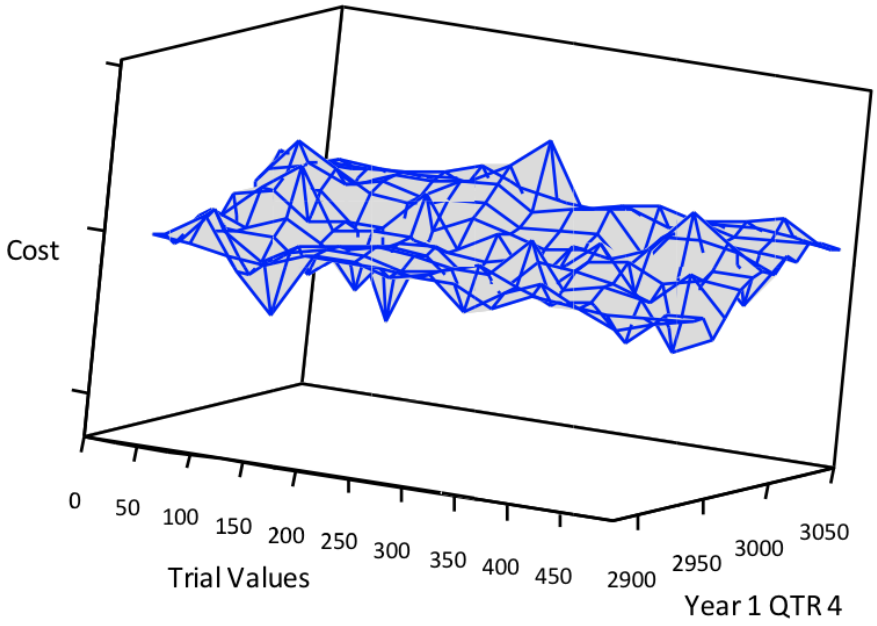


Figure 2.7: Simulation-Optimization of Total Cost

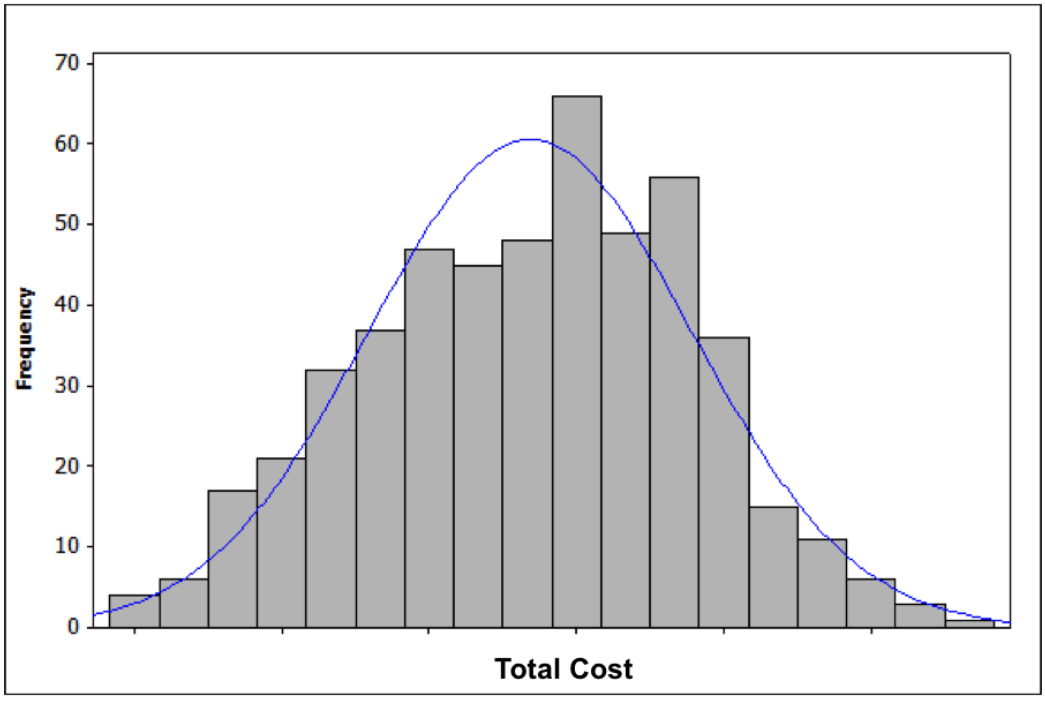


Figure 2.8: Simulated Total Cost (500 Trials)

which can be used for statistical analysis. As with the y-axis in the previous figure, the total cost values on the x-axis are intentionally not displayed.

Suppose we address the most obvious question related to determining the optimal total cadre size needed during each quarter of the two year horizon. Although demand uncertainty adds a layer of complexity, numerous iterations of the simulation model produce a distribution of the aggregated results.

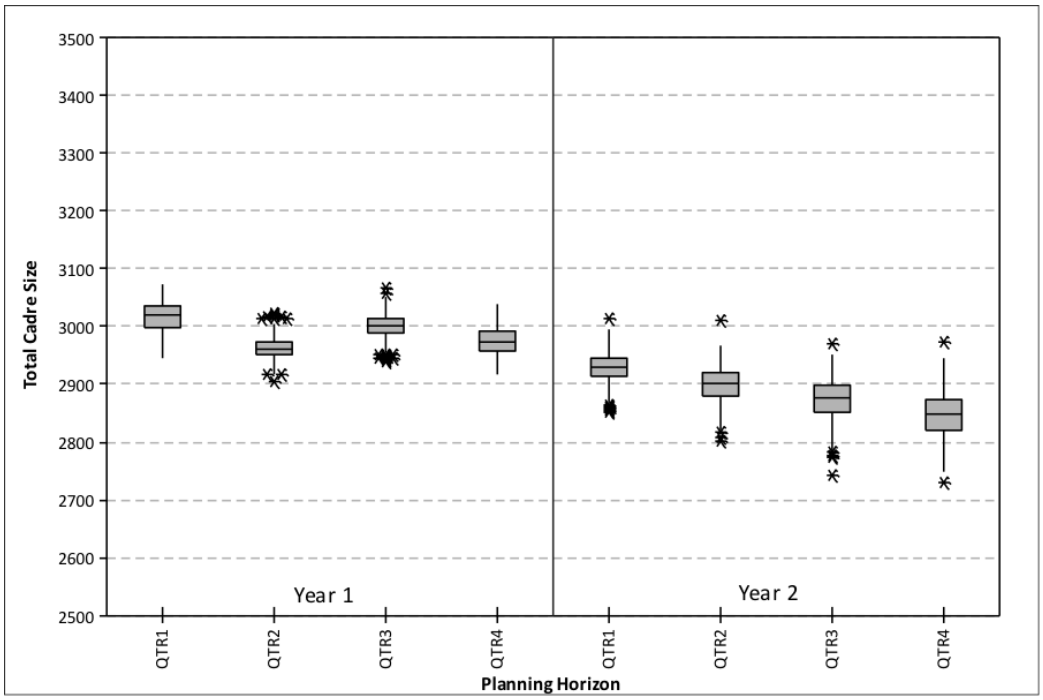


Figure 2.9: Simulation of total cadre size at all locations by fiscal quarter

Figure 2.9 depicts a boxplot diagram of the aggregated results. We can conclude that although the 25% quartile of projected cadre size exceeds 3,300 in the next immediate time period (defined as a quarter of a fiscal year), the 75% quartile never exceeds 3,000 after period three. Moreover, with a high degree of certainty, we can conclude that cadre requirements will be below 3,000 throughout the second year.

Suppose workforce requirements are administratively evaluated on an annual basis by location. Pooling the quarterly results, what is the relationship between the optimal cadre size in year 2 with respect to year 1? Figure 2.10 provides a visual representation of the distribution shifts for

the WTU locations (excluding CBWTUs). The analysis of each location may yield unique detailed results, but even at this more holistic view, location 32 clearly exhibits increased size and significant variability. Likewise, other locations, such as 14, 21, 29 and 39 demonstrate dissimilar shifts in their median and variance. Naturally, it is easy to initially conclude that anticipating these trends could be more cost effective than blindly hedging against uncertainty.

Similarly, we can isolate our analysis to a single skill. The measure used to determine optimal requirements is dependent on both the strategic goals and risk tolerance of an organization. Figure 2.11 shows a graphical depiction of all simulated solutions for Skill 40, separated by year 1 and year 2. Using the median as one measure of analysis, it is clear that there is an observable shift downward in the second year.

Obviously, more detailed conclusions can be made by separating output by location. The following set of results explores a small sample of questions that may be answered about a particular WTU location. Similar to the first results plot, Figures 2.12 & 2.13 allow us to answer the question: What are the forecasted cadre requirements at locations 21 and 32 during each quarter of the planning horizon?

It is easy to see that after the first quarter, location 21 can expect its optimal cadre requirement size to always be at or below 185 with almost no risk due to demand uncertainty.

However, location 32 shows a consistent increase in cadre requirements with noticeably more variability than location 21.

Suppose we choose to analyze the sources of uncertainty during the last quarter of the forecast horizon at location 32. Specifically, identifying the greatest sources of variability points to Skill 36 and Skill 40 (Figure 2.14). This information allows the organization to focus on a particular skill(s) and make informed risk based decisions about specific skill requirements or shape policies to influence the variability of a specific skill (e.g. cross-skill training or improving task proficiency).

Looking more closely at Skill 40, Figure 2.15 and Figure 2.16 depict the size and variability of quarterly forecasted requirements at both locations during the planning horizon. During an initial

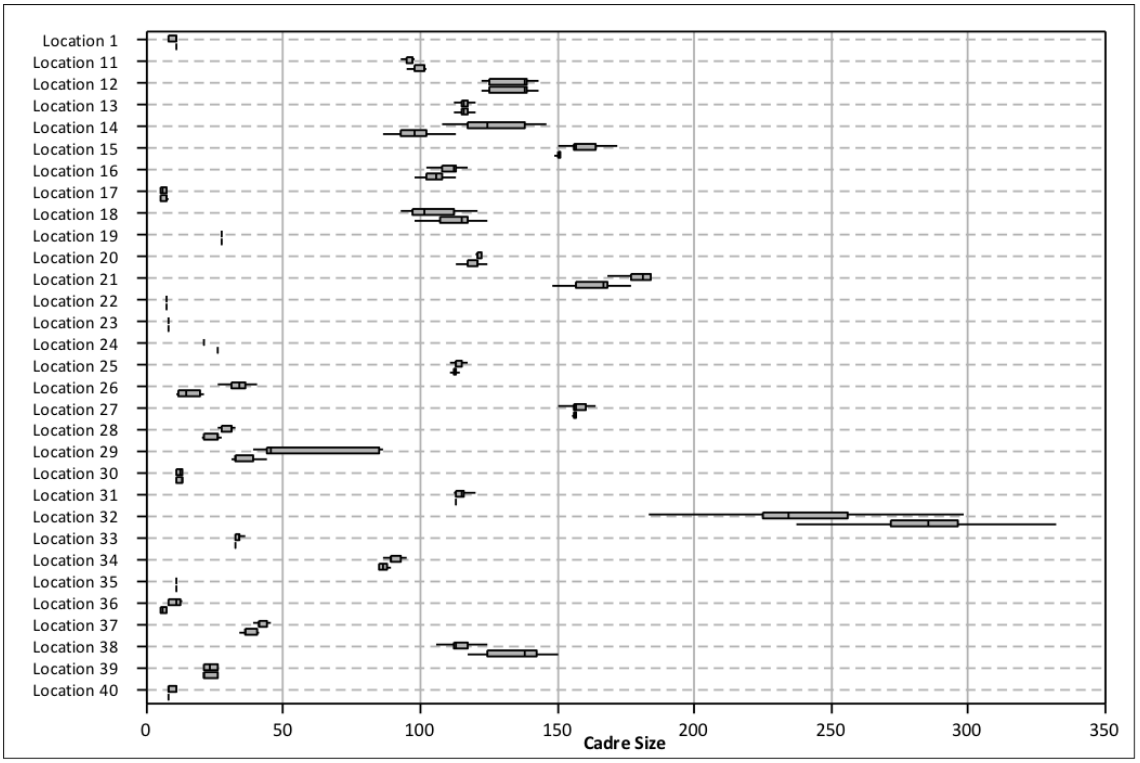


Figure 2.10: Simulation of cadre size by location (Year 1,Year 2)

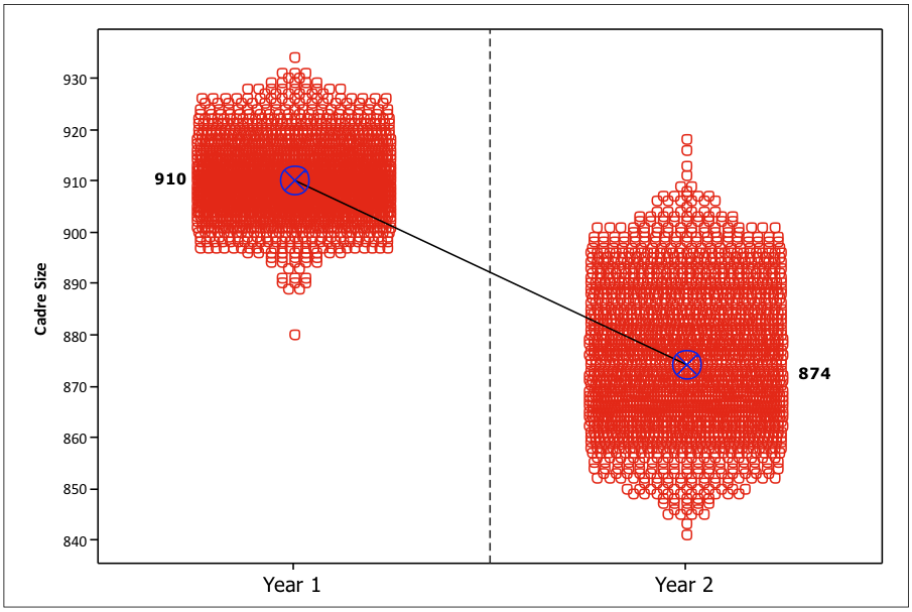


Figure 2.11: Simulations of quarterly Skill #40 forecasts

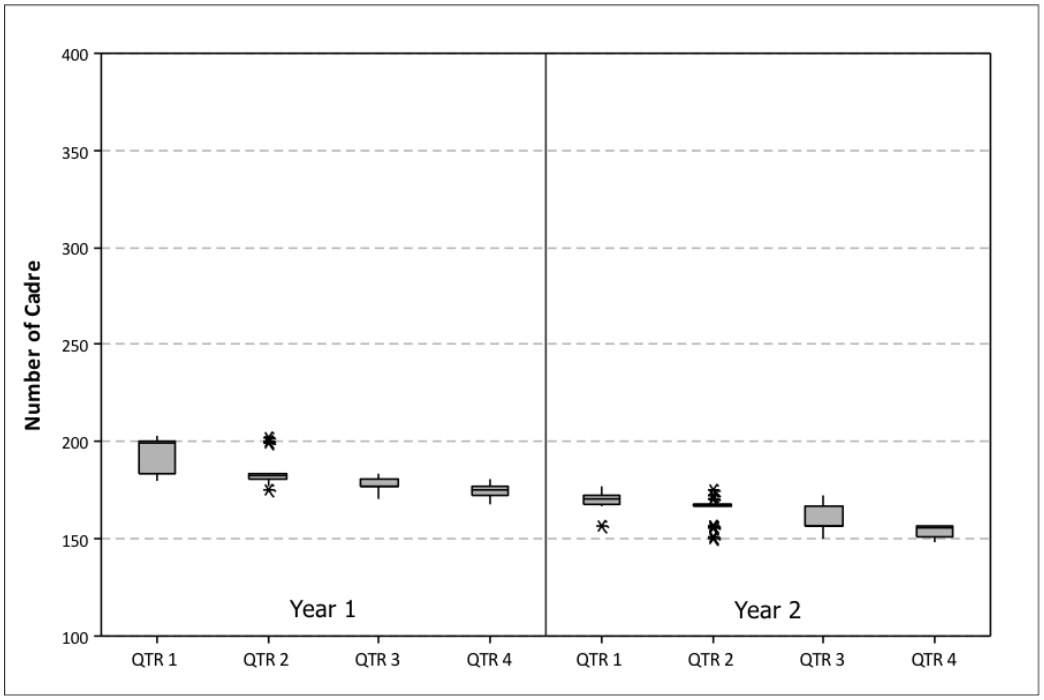


Figure 2.12: Forecast of Location 21 Aggregate Cadre Requirements by Quarter

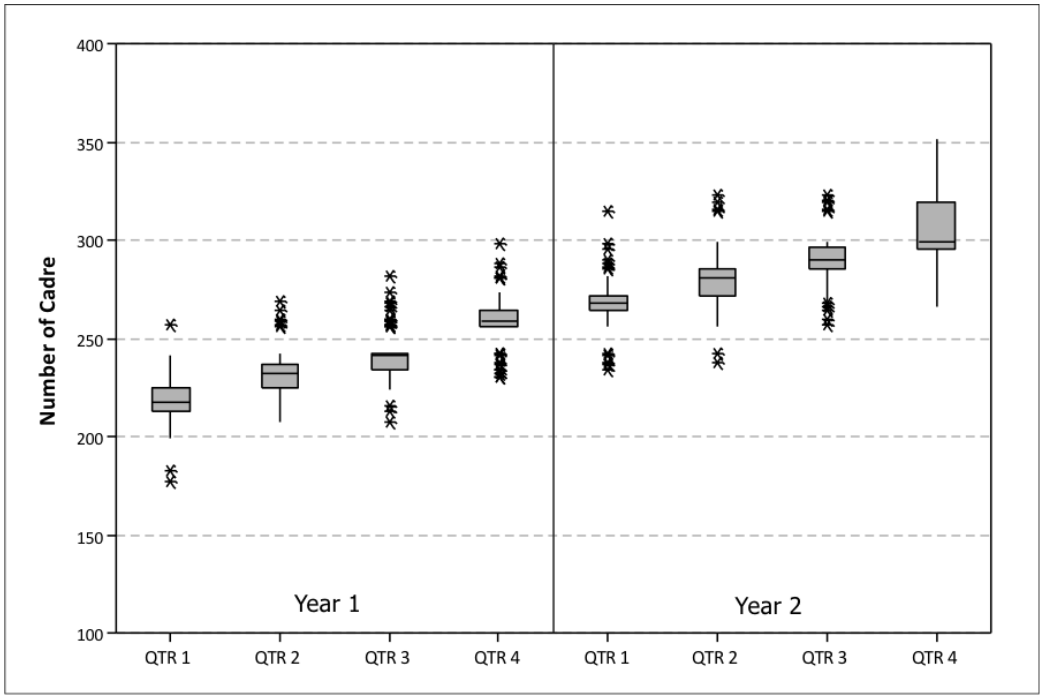


Figure 2.13: Forecast of Location 32 Aggregate Cadre Requirements by Quarter

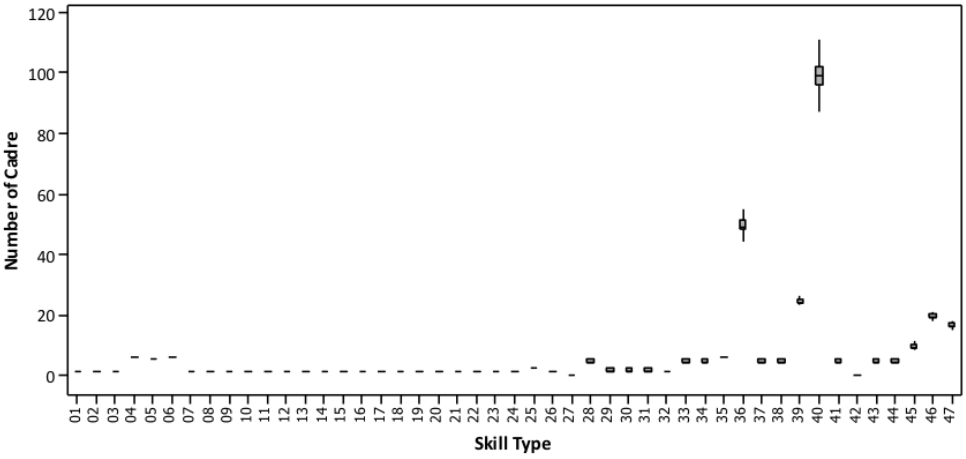


Figure 2.14: Location 32, Single Quarter Requirements by Skill

observation, without looking at detailed descriptive statistics, it is obvious that opposing trends exist.

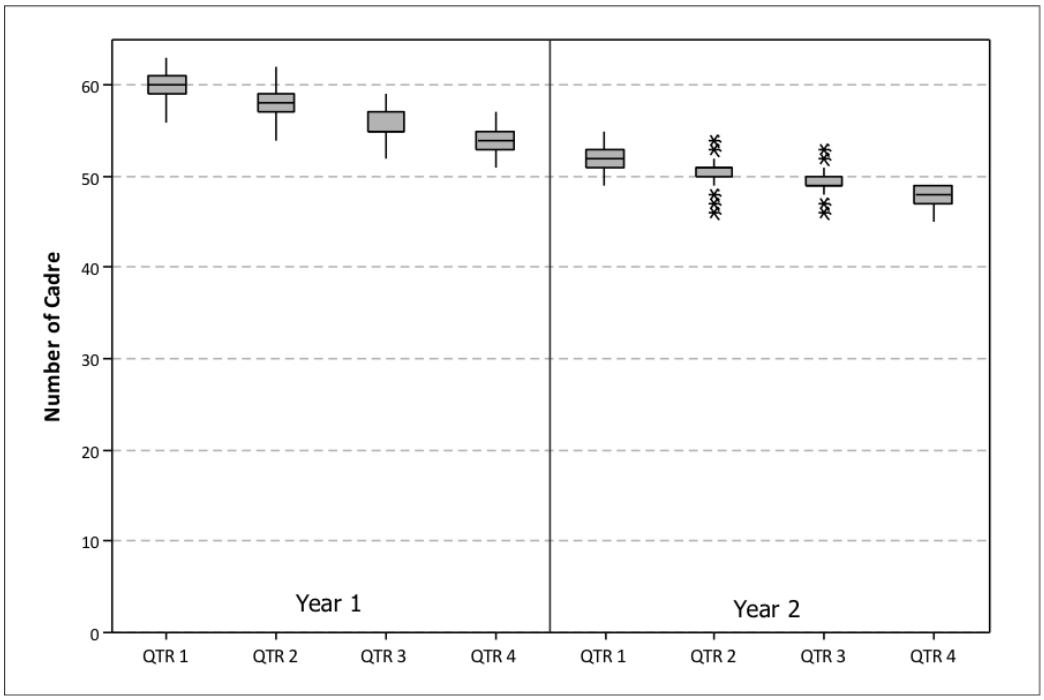


Figure 2.15: Location 21, Skill 40 Requirements

Moreover, at location 21, the 75% quartile of each fiscal quarter in year 2 is below the 25% quartile of if the first two quarters in year 1. More plainly, using current requirements (or projections from the first two periods) in year 2 would result in excess costs that are outside the bounds of reasonable risk management.

Conversely, at location 32, there is more variability and a statistical confidence that the Skill 40 requirements will exceed 80 in year 2. The disposition and appearance of Figures 2.15 & 2.16 are very similar to Figures 2.12 & 2.13 due to significant influence of Skill 40 at both locations.

2.6 Constraint Allocation Problem

The simulation-optimization approach presented and demonstrated in the previous sections lends itself to integration with a top-level optimization procedure that focuses the optimality search on external constraints. Optimizing on an external constraints allows us to incorporate uncertainty

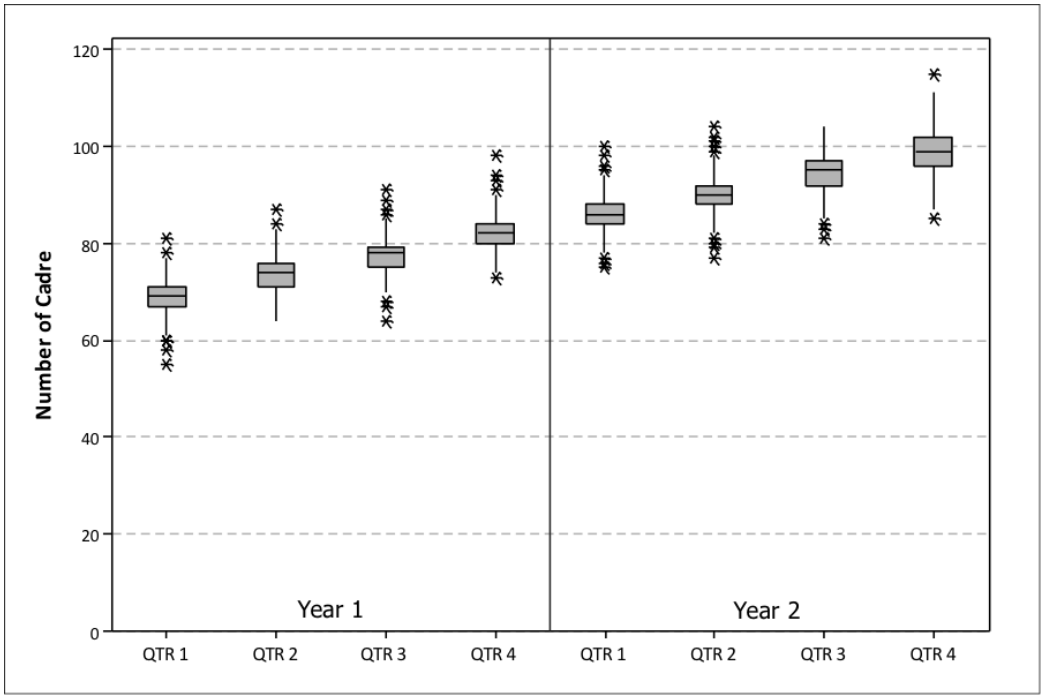


Figure 2.16: Location 32, Skill 40 Requirements

and provide a closed-loop algorithm for improving the solution beyond the risk-based decision approach of a distribution of solutions.

Consider a multiple location problem as presented in Section 2.4 with an additional centralized budget constraint. Previously the bottom-up simulation-optimization approach build a budget based on each solution. The budget constraint adds another layer of optimization, as shown in Figure 2.17, which enforces a top-down constraint approach and turns the computation into a budget allocation problem.

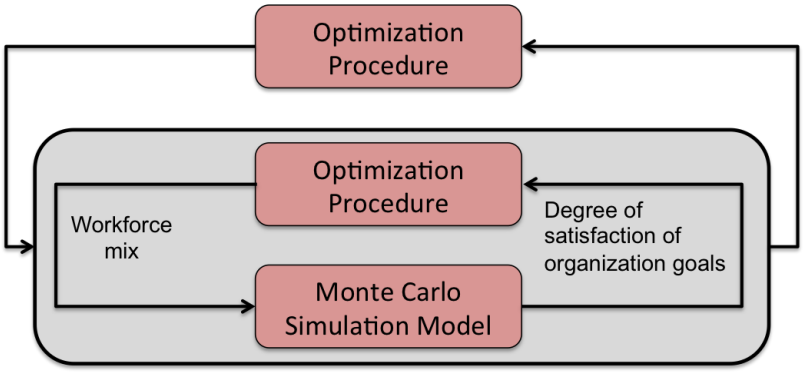


Figure 2.17: Top Level Optimization

Clearly, the budget allocation problem is significant when the budget constrains a solution that we would otherwise produce without the extra layer of optimization. Incorporating uncertainty into the optimization procedure as a large-scale stochastic program is not easily trivialized modifying the simulation-optimization solutions.

For example, when the WTU/CBWTU model is viewed as a budget allocation problem, we cannot uniformly decrease the workforce by location to meet an aggregate budget constraint. Like many real-world problems, uncertainty is not the same in each location, as depicted in Figure 2.10. Moreover, there may be different levels of priority in terms of fulfilling policy requirements by location. While we do not demonstrate the change in results from the previous section due to an absence of an aggregated budget constraint, it is easy to conceptualize how another layer of optimization would affect estimated requirements by location.

2.7 Conclusion

This research presents a simulation-optimization methodology to determine an optimal mix of workforce skills while providing the flexibility and robustness to incorporate uncertainty, specifically in demand forecasting. We address the uncertainty of real-world workforce planning problems and integrate a broader focus than most personnel scheduling problems by including multiple locations.

We present two versions of the general model to estimate workforce planning requirements. The task-oriented model assumes that tasks may be performed by more than one combination of workers with different skill levels. The aggregate general model addresses the fact than many organizations cannot deconstruct products or services by tasks and uses policy constraints to a demand to skill types.

The results in the WTU/CBWTU analysis are an example of the information that can shape the alignment of strategic goals and workforce requirements, especially in the context of an unconstrained budget. Risk based estimates can improve the effectiveness of inventory forecasting tools and human resource policies are imbedded with data driven strategic requirements rather than given inputs.

In the WTU/CBWTU analysis, we use data for an aggregate general model. Although we do not have data to replicate the same analysis in a similar task-oriented model, we believe the analysis will be similar. Most importantly, we are able to demonstrate the kind of analysis that we can perform with either approach. LPs may be different due to expanded data size with a task-oriented approach, but the resulting analysis and types of decisions are the same.

The approach and modeling of this specific problem can be restructured under different assumptions. For example, consider a realistic environment where budget is the limiting constraint. The optimization portion of this model can be completely reconstructed in a manner that essentially transposes the objective function and constraints. Likewise, when considering the perspective of a parent organization that allocates requirements among locations, another layer of optimization can be added to the existing model.

Chapter 3

A Markov Chain Model of Military Personnel Dynamics

3.1 Introduction

The size of the U.S. Army is often a function of competing budget constraints and mission requirements. Moreover, the United States experiences fluctuating periods of economic growth and decline combined with varying sentiments towards a propensity to serve. All of these affect how our nation assesses new recruits into the armed forces. A similar scenario is evident when the Army takes action to influence the retention (reenlistment) of enlisted soldiers. The Army uses forecasting methods to predict the number of reenlistment eligible soldiers it must retain each fiscal year. When projections without incentives fall below requirements, natural questions persist. Can we target incentive programs and policies to specific categories of individuals? How can we identify those most likely to leave?

The U.S. Army cannot adequately answer the more precise question related to incentive size unless it uses the most relevant data and techniques to predict behavior. Current U.S. Army incentive programs frequently target specific skills or military occupation specialties with arbitrary monetary incentives. Such policies are not backed by rigorous analysis that determines propensities to serve. There is also a serious lack of understanding of how individuals make stay-or-leave decisions. There exists a research gap for separating individual retention propensity and for explaining individual retention decisions.

We model the career progression of enlisted personnel as a discrete-time homogeneous Markov chain and estimate state transition probabilities using three years of reenlistment data from the

US Department of Defense. The estimated transition probabilities offer at least two benefits: (1) it provides an easily constructible method for evaluating the probabilistic progression of enlisted personnel through different career states, and (2) it can be used in a subsequent stochastic dynamic programming model to understand reenlistment behavior.

The Markov chain model allows us to answer meaningful questions from policy makers. For example, it allows easy computation of various statistics at both individual and aggregate levels. At the individual level, it can be used to describe the probabilistic progression for military personnel at a given career stage. At the aggregate level, it can be used to derive information on overall continuation rates and separation behavior, which are critical inputs in developing retention programs.

Our estimation methodology only requires a data set with a limited time range rather than the complete career time horizon of personnel cohorts. This should be viewed as an advantage, as the military is constantly fluctuating in size, structure and policies. Our Markov chain estimation approach is not only more adaptable and easier to implement than tracking cohort data, but captures transition probabilities while omitting obsolete data.

Building on our Markov chain estimation result, we construct a finite horizon stochastic dynamic programming model to understand the stay-or-leave decision of a military personnel at different career stages. Our model assumes that an individual is forward-looking and can rationally evaluate his expected payoff of staying in the military, taking into account future career progressions and retirement benefits. The military individual leaves the army if the expected payoff in the civilian sector dominates the expected payoff of staying in the army. Therefore, our model takes into account data on both military pay and civilian pay. Taken together, the dynamic programming model allows us to calculate the reenlistment propensity of military personnel at different career stages and therefore can be a useful tool for policy makers.

The dynamic programming model is a viable alternative to more traditional approaches based on statistical analysis. A popular approach to predict the stay-or-leave decision is to use a binary logistic regression model. We show that the dynamic programming approach can predict retention

behavior with similar accuracy as a well-constructed logistic regression model. However, unlike the logistic regression model, the dynamic programming model requires far fewer predictive variables. Our dynamic programming model uses only grade and time-in-grade variables, while the logistic regression model uses many variables on personal attributes. Using personal attributes for retention incentives often leads to perceived discrimination and therefore can be problematic. The strength of the dynamic programming models is that it takes into account expected future compensation, which is not incorporated in the logistic regression model. We also consider a hybrid logistic regression model that incorporates the output from the dynamic model and shows that the predictive power can be dramatically improved.

As a by-product of the dynamic programming model, we obtain the difference in expected future compensations between the military and civilian sectors based on a military personnel's career stage. This allows a tailored approach for retention incentives based on variables such as grade and time-in-grade, and has the potential to generate substantial savings for the military.

The balance of the paper is organized as follows. Section 3.2 discusses some relevant literature. Section 3.4 presents the Markov chain model and its estimation from data. Section 3.5 presents an aggregated Markov chain model based on grouped states. This section also presents functional manipulations of the transition matrix that provide insightful steady state interpretations. In Section 3.6, we present a dynamic programming model for the stay-or-leave decision and then in Section 3.7 we integrate the results from dynamic programming with logistic regression results to assess the impact on predictive modeling. Section 3.8 concludes.

3.2 Related Literature

Discrete-time Markov chains are frequently used to model and evaluate the long-term behavior of individuals in a variety of disciplines. Describing a process as a Markov chain and understanding its properties enables informed decision making and policy guidance. Constructing and evaluating Markov chains has a precedence in the medical industry, particularly in describing the progression and treatment of chronic diseases and illness. A Markov chain model can be used

to study long-run behavior even when only data with limited time range is available. This property is important for certain applications, including the one in the current paper.

Beck and Pauker[8] describe how a Markov model can replace a analytical methods such as decision trees within the medical decision making process. They demonstrate the utility of the fundamental matrix, especially coupled with simulation. Craig and Sendi[22] use discrete-time Markov chains to evaluate treatment programs and health care protocols for chronic diseases. They present different types of situations that make matrix estimation techniques unique. In particular, they describe a discrete-time homogeneous Markov model in which the estimation techniques use observation intervals that coincide with the cycle length. This situation happens to align fairly accurately with the structure of the enlisted Army career network we describe in this paper.

Markov models are also applied to object-oriented systems such as the financial industry as described by Cyert, Davidson, and Thompson[23]. They developed a Markov model describing the behavior of accounts receivable balances and used matrix properties, including the fundamental matrix, to make a variety of interpretations about the behavior of accounts at different stages of a time horizon. While the context of the Cyert et al.[23] research is much different than military personnel dynamics, we demonstrate the same utility of an implementable method for interpreting a variety of system related questions.

A small subset of retention research uses dynamic programming, which can be described as an approach for sequential decision models (Puterman[76]). Using a modeling approach similar to Hall[50] and Gotz and McCall[46], the enlisted retention model can be structured as a manpower network. Hall models Army officer retirement, while Gotz and McCall model a distinct proportion of the Air Force officer. Gotz and McCall assert that, between the 10 and 20 year marks, retirement pay is the most important factor for officer retention with officers making decisions in an optimal sequential fashion. In contrast to the aforementioned, enlisted personnel behave differently than commissioned officers in the military. The enlisted career path and military experience is much different than that of officers, making it unreasonable to conclude that retirement pay is as influential without isolated analysis.

The ability of current technology to accommodate more computationally intensive models suggests that more complex dynamic programming methods are suitable for modeling optimal reenlistment behavior than the simplified Annualized Cost of Leaving model produced by Warner and Goldberg[87], which focus on first and second terms of service and do not incorporate future uncertainty. Dissimilar to Asch et al.[3], the following approach does not model the retention decision of all services or incorporate the potential transition from active duty to the reserves. Asch et al.[3] estimate model parameters that affect the decision to stay on active duty or leave, and a parameter related to the variance of the stochastic shocks affecting the alternatives of being a civilian or a reservist. Whereas Asch et al.[3] estimate model parameters associate with the means, variances, and covariance of the preference for active and reserve service, we computationally construct parameters, specifically pertaining to transitions rates, from historical data sets.

Duala and Moffitt[35] also used a stochastic dynamic programming model to estimate the effect of reenlistment incentives on military retention rates using panel data. They modeled decision points every 4 years after reenlistment up to twentieth year, but did not analyze the relevance of military occupation specialties. The modeling approach proposed in this paper can be modeled in aggregate or decompose by skill. We also incorporate the entire career horizon and different parameter estimation techniques. Rather intuitively, in the presence of unobserved heterogeneity, Duala and Moffitt[35] also conclude that a higher military-civilian pay difference significantly affects reenlistment behavior.

A distinct difference between modeling officer and enlisted retention decision lies in the gap between career expectations. Identical retirement systems may shape intuition about similar officer and enlisted retirement rate; however, the reality is that the probability of an enlisted personnel reaching retirement is significantly less than officers. Recent statistics show that less than 15% of enlisted personnel, compared to 46% of officers, will become eligible for retirement annuity (Stewart and White[82], Henning[55]). This implies the relative importance of effectively modeling the enlisted retention decision prior to retirement eligibility. The retention model proposed in the Section 3.6 assumes that the goal of enlisted personnel is to maximize pay and entitlements.

Expected civilian compensation is a key component to developing a model that evaluates an optimal decision policy for leaving the military. Generally, officers are provided equal or better compensation in the civilian workforce when they retire or separate from the military. Due to lower average civilian education levels and variance in skill levels, enlisted trends do not necessarily mirror the officer domain. Therefore, one of Hall's primary assumptions, that an officer's initial civilian pay is equivalent to their final military pay, is not reasonable for our model. Uncertainty in future civilian pay compared to current military compensation adds complexity but cannot be sacrificed for simplicity.

Duala and Moffitt[35] acknowledged that "military service is not completely substitutable for civilian work" and addressed civilian compensation by using estimates from Internal Revenue Service data on the post-service civilian earnings of veterans, allowing the civilian wage profile to depend on time spent in the military. Similarly, [3] used data to map civilian wages with respect to total years of experience using a civilian median-wage profile based on population surveys of corresponding education ranges. Neither of the aforementioned models account for uncertainty of civilian wages. The method we use to estimate expected civilian pay entitlements is described in Section 3.6.1.

Nimala and Jeeva [73] describe a dynamic programming approach to optimal manpower recruitment and promotion policies. The focus on a two grade system and like Rao [78], use a mathematical model with an objective of minimizing manpower system costs. Additionally they state that their model is a dynamic programming method that is analogous to the Wagner-Whitin model. After a detailed description of their model and assumptions, to include a numerical illustration, the conclusions appear to be the same as Rao. Aside from the obvious fact that they focus on recruiting rather than retention. the major limitation of both of these models is described as the fact that it is considered in isolation from the various constraints and operating policies under which a manpower system operates and proposes a possible extension to develop an integrated model which minimizes the manpower system costs in the presence of the system constraints and operating policies.

3.3 Enlisted Career Path

This section provides some specific insights into the career path of an individual in the enlisted manpower network. In Section 3.4, we begin the description of the Markov chain model for enlisted personnel dynamics. However, this section provides additional background details pertaining to components of the model which influence compensation and transition probabilities associated with promotion, separations and reenlistments.

3.3.1 Pay and Entitlements

Basic pay while in military service is determined by an individual's pay grade (referred from hereinafter as grade) and accumulated years-of-service (YOS). Pay increases occur with increases in grade and also occur at specified yearly intervals of service, regardless of promotion. Military pay changes occur after promotion or in two year increments. In addition to basic pay, all military members are provided additional entitlements such as tax free Basic Allowance for Housing (BAH) and a Basic Allowance for Subsistence (BAS). The value of these entitlements are categorized based on the status of dependents (spouse or child).[1] Basic military pay scales are the same for all military occupation specialties (MOS); however, some specialties may receive additional incentive or bonus pay. For simplicity, this research does not include any additional current specialty pay programs.

Expected civilian compensation is a key component to developing a model that evaluates an optimal decision policy for leaving the military. Generally, officers are provided equal or better compensation in the civilian workforce when they retire or separate from the military. Due to lower average civilian education levels, enlisted trends do not necessarily mirror the officer domain. Therefore, one of Hall's primary assumptions, that an officer's initial civilian pay is equivalent to their final military pay, is not reasonable for this model development. Uncertainty in future civilian pay compared to current military compensation adds complexity but cannot be sacrificed

[1] Entitlements are higher with dependents than without dependents.

for simplicity.

Duala and Moffit acknowledged that "military service is not completely substitutable for civilian work" and addressed civilian compensation by using estimates from Internal Revenue Service data on the post-service civilian earnings of veterans, allowing the civilian wage profile to depend on time spent in the military. Similarly, Asch et al.[3] used data to map civilian wages with respect to total years of experience using a civilian median-wage profile based on population surveys of corresponding education ranges. Neither of the aforementioned models account for uncertainty of civilian wages.

3.3.2 Rank Structure

The Army enlisted force consists of nine different pay grades. Grades are recognized as E1-E9 and correspond to a rank structure that ranges from Private (PVT) to Command Sergeant Major (CSM).[2] Grades are also categorized by five skill levels, with E-1 to E-4 composing Skill Level 1 (SL1), E-5 as Skill Level 2 (SL2) and continuing to increase by Grade until categorizing E-8 and E-9 as Skill Level 5 (SL5). Because of distinct differences in promotion criteria and retention behavior, this research analyzes each grade separately with the exception of grades in SL1, which do not generally demonstrate differences in behavior.

3.3.3 Promotions

One of the primary reasons for differentiating SL1 individuals from other grades is that they are subject to different promotion criteria. SL1 individuals are part of a decentralized promotion system, where promotions are generated by meeting promotion eligibility requirements dictated by policy. Promotions to E-2 through E-4 are managed by a Battalion commander. Grades E-5 to E-9 are all classified as Non-Commissioned Officers (NCOs); therefore, they are subject to semi-centralized or centralized Department of the Army promotion systems. Both semi-centralized and centralized promotion counts are determined by the needs of the Army. Soldiers being considered

[2] Although there are nine grades, there are more than nine enlisted ranks. For example, a Specialist (SPC) and a Corporal (CPL) are both E-4; a Master Sergeant (MSG) and a First Sergeant (MSG) are both E-8.

for promotion to Sergeant (E-5) and Staff Sergeant (E-6) are selected for promotion based upon decentralized promotion boards and subsequent monthly centralized by-name lists shaped by the Headquarters, Department of the Army (HQDA). Promotions to Sergeant First Class (E-7) through Sergeant Major (E-9) are purely centralized because all selection and promotion authority resides within HQDA.

The enlisted promotion system functions much differently than the officer promotion system, primarily in the duration which individuals may remain in a particular grade. Officers always progress through the promotion system as a member of a promotion cohort or "Year Group". Officers generally have only three promotion opportunities at each rank: Below-the-Zone, In-the-Zone, and Above-the-Zone.[3] The majority of officers are promoted at an In-the-Zone benchmark, specified by their time-in-grade (TIG). Essentially, the three zones of promotion provide a three year window of promotion opportunity, after which an individual must exit the military if unsuccessful.

Enlisted personnel have the ability to be promoted in a much larger window; therefore, they may also remain at the same grade for a much longer period. Regardless of the type of enlisted promotion system, each promotion level has minimum time-in-service (TIS)[4] and time-in-grade requirements. Enlisted personnel may be promoted to the next rank if they satisfy a given set of promotion requirements, one being that one must serve a minimum amount of time in a grade before consideration in the next grade. For example, an E-5 may be considered for promotion at the 3-year mark, but may be reevaluated at every month of every year until their 12th year without being promoted. For purposes of this research, minimum time-in-service requirements are the primary focus, since enlisted time-in-grade standards are less than one year.

Another aspect in the enlisted promotion system is retention control points (RCPs). The Army applies retention control points to each grade, which force an individual to exit the military when reached. Retention control points are enforced based on an individual's time-in-service and vary by grade. The retention control points for grades E-6 and beyond are all set at 20 years or

[3] There is only one opportunity at grades O-2, O-3, and O-7 or higher.
[4] The difference between TIS and YOS is that TIS is generally measured in months.

higher; therefore, individuals at these grades who reach an retention control points will still earn a military retirement. Retention control points for SL1 and E-5 are currently set a 12 and 13 years respectively. As a result, it is impossible to retire at one of these grades. Moreover, the only grade at which an individual can reach 30 years of service is E-9. These aspects of the promotion system become important when establishing Markov states and transitions in later sections.

As alluded to earlier, a key distinction between the management of officer and enlisted promotions is that enlisted personnel are purely promoted to meet requirements. An individual may be qualified to serve in the next rank, but if there are no unfilled requirements, the individual must remain at the current rank or leave the Army in the case of reaching a retention control point. Another option in the presence of limited promotion potential is for an individual to transfer to another military occupation specialty, if applicable. For simplicity, we will not consider a scenario in which a Soldier is allowed to transfer between skills. Moreover, we do not consider demotions in this scenario since their occurrences are relatively infrequent, especially in the higher enlisted grades.

3.3.4 Reenlistment Categories

Another significant aspect of the enlisted manpower network involves terms of service, or more literally, a classification status pertaining to an individual's number of previous reenlistments and total years of service. An individual serving under their initial enlistment contract is considered a first-term (FTM) Soldier. Upon the first instance of a re-enlistment, an individual transfers to the re-enlistee (REE) network. Many Army manpower forecasting models separate individuals into separate first-term and re-enlistee networks due to their known differences in behavior. The re-enlistee network can be further decomposed into two separate subcategories: mid-term (MID) and careerist (CAR). Both of these subcategories indicate that an individual has one or more previous reenlistments with the distinguishing factor being that careerists have more than 10 years of total service. This difference becomes important during model development, since careerists are the only re-enlistment category that can transition into military retirement after an eligible amount of

time-in-service.

3.3.5 Types of Separation

This model focuses on the decision to stay in the military or leave voluntarily. Implicit to this decision is the understanding that there are two types of voluntary separation, an expiration of term-of-service (ETS) or retirement. The difference is marked by whether an individual leaves prior to accumulating enough time-in-service to trigger retirement compensation, normally 20 years. In reality, a large portion of personnel are forced to leave the Army involuntarily due to a variety of reasons ranging from administrative medical separations to misconduct separations. In general, involuntary separations are classified as either administrative or adverse actions. The purpose of this model is not to predict involuntary losses; however, the probability to transitioning to an involuntary loss state must be incorporated. More importantly, the optimal decision policy as it pertains to evaluating military and civilian compensations only applies to the voluntary stay or leave decision.

3.4 A Markov Chain Model for Enlisted Military Personnel

This section introduces a discrete-time Markov chain model for enlisted Military Personnel. We use three years of historical data to estimate parameters for the Markov Chain.

3.4.1 Model Description

Our model considers individuals in the rank of Private (E1) through Sergeant Major (E9). However, since grades E1 through E4 belong to the same skill level category and have similar behavior patterns and decentralized[5] promotion criteria, we group all of them into skill level 1 (SL1). States refer to a combination of grade and years-in-grade. The state space is augmented by voluntary and involuntary separations, which are modeled as two separate absorbing states. Our

[5] Unlike promotions for grades E5 through E9, promotions for grades E1 through E4 are based on a set of pre-determined qualifications and are not centrally managed by an army-level promotion board.

model therefore ignores the possibility of returning to the military after voluntary and involuntary separations.

The state definition in our model should be contrasted with some previous work where state is defined as an individual's pay grade and/or time-in-service (see, e.g., Hall[50]). Note, however, that Hall[50] models officers who are typically promoted in cohort. Compared with enlisted personnel, the variation in state transition is much smaller. For enlisted promotion criterion, which is relevant in our context, a minimum time-in-service requirement is typically stipulated. To incorporate time-in-service promotion criterion, we impose minimum time-in-grade requirement in the state transition. For example, an SL1 is not eligible for promotion to E5 with less than two years in grade and a E5 is not eligible for promotion to E6 with less than two years in grade, etc. The minimum time-in-grade captures the observations from the data quite well, as there are rarely violations. It also supersedes time-in-service requirements.

Figure 3.1 shows the state transition diagram for our model. In total, there are 79 states, including voluntary and involuntary separations as two absorbing states. Note that for each grade, there is a maximum time in grade. For example, for grade E5, the maximum time-in-grade is 11 years.

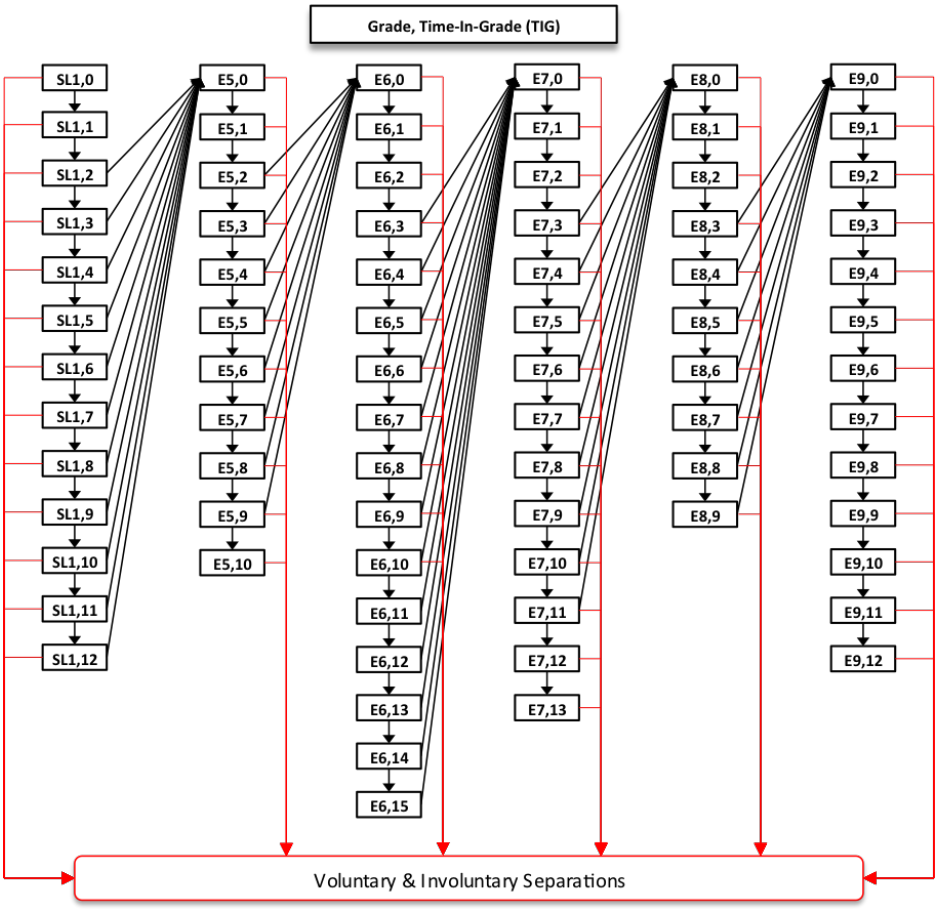


Figure 3.1: State Transition Diagram of the Markov Chain Model

3.4.2 Data Description

In order to estimate parameters for the Markov chain, we analyze three years of historical data for enlisted personnel obtained from Headquarters, U.S. Army, Deputy Chief of Staff, Army G-1 (Personnel). Our data covers the period from October 2007 to September 2009. Our main data sources are personnel inventory and transaction files in Total Army Personnel Database (TAPDB). The personnel inventory file contains end of month snapshots of the active Army enlisted force each month. It includes a large number of personal attribute fields for each enlisted personnel. The transaction file is compiled at the end of each month and includes a record for any major personnel transaction occurred during the month. We merge inventory and transaction files by social security number (SSN) and date. Table 3.1 describes sample variables from TAPDB inventory and transaction files.

Descriptive statistics of the data used for predictive modeling are provided in Appendix C. Not included are statistics for variables in Table 3.1 which are not used for prediction, such as gains, promotions and involuntary losses. Additionally, education level statistics are not presented due to the unique coding system adopted by the Army. However, 98.6% of individuals completed a level of schooling equivalent to a high school degree and 9.2% of enlisted individuals have some amount of eduction beyond high school.

3.4.3 Estimation Procedure

We use the historical data described in Section 3.4.2 to estimate the transition matrix for the discrete-time Markov chain model. Our estimation procedure follows Craig and Sendi[22]. Even though our data are collected on a monthly interval, we use a cycle length of one year. This choice allows us to accommodate seasonality in promotions observed in the data. Since we have complete observations for three full years, we use a version of the estimation procedure where the observation intervals coincide with cycle length (Craig and Sendi[22]).

An important assumption we make is that the transition probabilities are stationary over

Extract	Variable	Definition
Transactions	SSN	Social Security Number pertaining to the Soldier who executed the transaction
Transactions	Transaction Date	Actual date of transaction (DDMMYYYY)
Transactions	Transaction Category	Major category type of transaction (Gain, Loss, Extension, Promotion, Demotion)
Transactions	Gain Type	Description of specific type of gain to active duty (Prior Service, Non-Prior Service, Immediate Reenlistment
Transactions	Loss Type	Description of specific type of loss from active duty (Expiration of Term of Service, Retirement, Misconduct, Physical Disability, Dropped From Rolls, Entry Level Separation, Hardship or Parenthood, Pregnancy, Unfit, Unsatisfactory Performance, Reduction In Force, Early Release, Early Retirement, Immediate Reenlistment, Other)
Inventory	Pay Grade	Pay grade scale of individual (E1 to E9)[6]
Inventory	Time-in-Grade	Months of service at current grade (converted to years)
Inventory	Time-in-Service	Total months of active service (converted to years)
Inventory	AFQT	Armed Forces Qualification Test score
Inventory	Education	Level of education centered on high school graduate
Inventory	Marital Status	Marital status of married, divorced or single
Inventory	Dependents	Number of minor dependents
Inventory	Race	Racial category of white, black, hispanic or asian/other
Inventory	Reenlistment Quantity	Number of previous reenlistments
Inventory	Deployed	Binary category equals 1 if previously deployed or 0 if never deployed
Constructed	MSD	Months since previous deployment
Constructed	Age	Age centered on 18 years
Constructed	Employment	Employment rate of change (12 month moving average)

Table 3.1: Description of inventory and transaction data variables

time.[7] This assumption is not unreasonable as military operations are fairly consistent during the three year observational period. With the stationarity assumption, the observed transitions in all three years can be pooled together to form an observed one-year transition count matrix.

We also manipulate the data to more honestly reflect involuntary separation. Recall that in our state definition, there is a maximum time-in-grade allowed for each grade, which is also called a retention control point (RCP). When a RCP is reached for an enlisted individual, all our original records show voluntary separation. In reality, the voluntary separation decision is made with known information about promotion potential and a barrier for continued transition within the current grade. Therefore, data for individuals who reached an RCP are treated as involuntary losses even if a record depicts a voluntary loss.

For notational simplicity, let m denote the number of states. For our model, m is the constant 79. The first step in our estimation procedure is to produce the transition count matrix $C = [c_{ij}]$, which is an $m \times m$ matrix. The entry c_{ij} represents the total number of yearly transitions from state i to state j. We assume without loss of generality that the states are ordered such that the last two states are voluntary and involuntary losses, which are also the absorbing states.

Let the $m \times m$ matrix $M = [\theta_{ij}]$ denote the transition matrix, where θ_{ij} is the probability of moving from state i to state j by the end of a cycle. With the observed count matrix C, the maximum likelihood estimate of the transition probability θ_{ij} is the row proportions of the counts[8]

$$\theta_{ij} = \frac{c_{ij}}{\sum_{j=1}^{m} c_{ij}}, \quad \forall \ i, j.$$

We also have $\theta_{m-1,m-1} = \theta_{m,m} = 1$, because the last two states are absorbing states. Since we assume there is no demotion, entries below the diagonal in M are equal to zero. Moreover, the transition probabilities in the diagonal for non-absorbing states are also equal to zeros since time-in-grade uses the same measurement as the cycle length.

[7] In principle, the approach can be used to estimate a separate transition matrix for each year.
[8] For simplicity, we make no notational distinction between population parameters and their estimates.

3.4.4 The Transition Matrix

The state progression of an enlisted personnel follows the Markov chain described in Figure 3.1 with transition matrix M. There are four distinct types of transitions:

- Remain in the same grade (time-in-grade increases by 1)
- Promote to the next grade (time-in-grade equals 0)
- Exit the system due to involuntary separation
- Exit the system due to voluntary separation

Since the estimated transition matrix M is a 79×79 matrix, we choose not to report the full details in the paper. Instead, we report slices of the estimates to give a flavor of the result.

3.4.4.1 Continuation within Grade

The calculation for continuation within the same grade requires tracking individual records from one year to the next. Moreover, individuals are only included in the continuation counts if they are in the initial data set at time 0. Figure 3.2 shows the probability of an individual continuing within the same grade from one year to the next. For example, a SL1 individual has an 87.8% probability of beginning and ending their second year in grade without advancing in grade due to promotion or leaving the Army. The probability decreases to 43.4% during the fourth year in SL1.

3.4.4.2 Promotions

Promotion counts from state to state are calculated in the same manner as voluntary and involuntary losses. The Army applies retention control points to each grade, which force an individual to exit the military when reached. Retention control points are enforced based on an individual's time-in-service and vary by grade. Figure 3.3 shows the decrease in promotion probability by grade

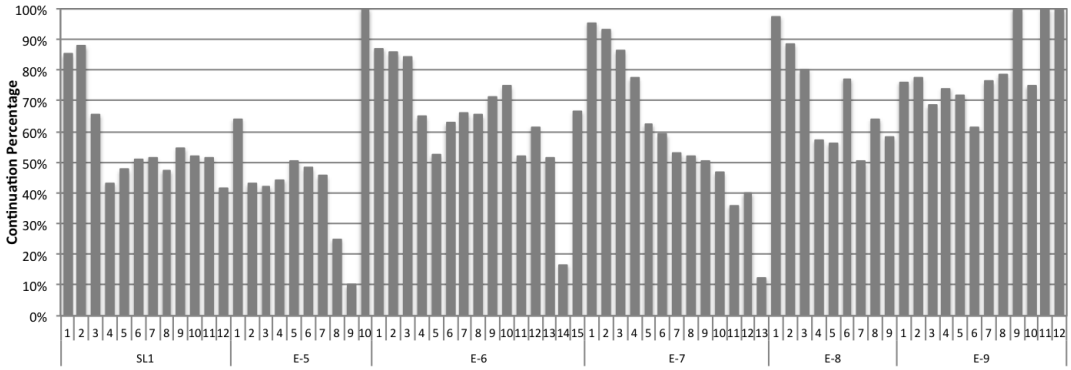


Figure 3.2: Continuation probabilities within grade

(to E6 and above) and Figure 3.4 shows the probability of promotion for each year of service for semi-centralized or centralized Army promotion systems to E6 through E8.

3.4.4.3 Involuntary Losses

Involuntary losses include all administrative and adverse losses and can be calculated from the loss transactions for the relevant periods. Involuntary losses occur primarily during the grades of E5 and lower. Involuntary loss rates from fiscal years 2006 through 2009 are shown in Table 3.2.

3.4.4.4 Voluntary Losses

Voluntary losses are composed of two loss sub-categories: non-disability retirement (NDR) and expiration of term of service (ETS). Voluntary losses occur at all grade levels with the rates increasing at higher grade and higher time-in-grade combinations. This intuitive result is depicted in Figure 3.5 where the highest voluntary loss rates occur at grade E7 with greater than 10 years in grade. Voluntary loss counts are generated in the same manner as involuntary losses as transaction counts are merged with inventory data. It can be seen from Figure 3.6 that the probability of an individual choosing to leave voluntarily outweighs the probability of any type of involuntary loss at grade E5 and beyond. It is also evident that loss probabilities from both major categories are

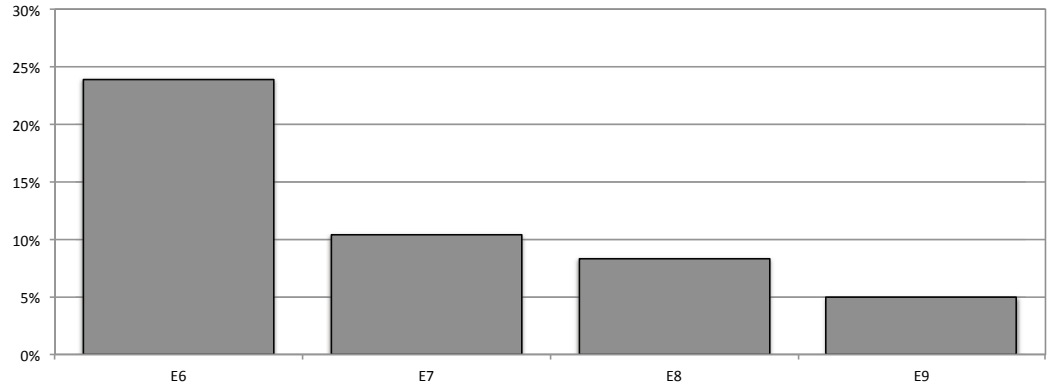


Figure 3.3: Promotion probabilities by grade

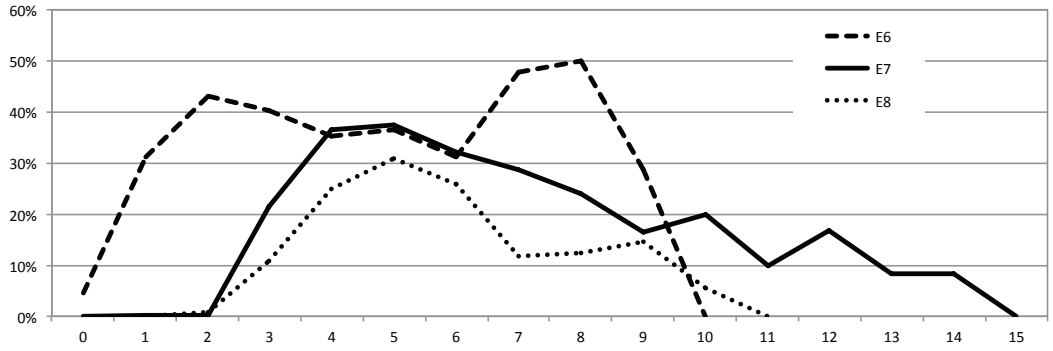


Figure 3.4: Promotion probabilities by time-in-grade

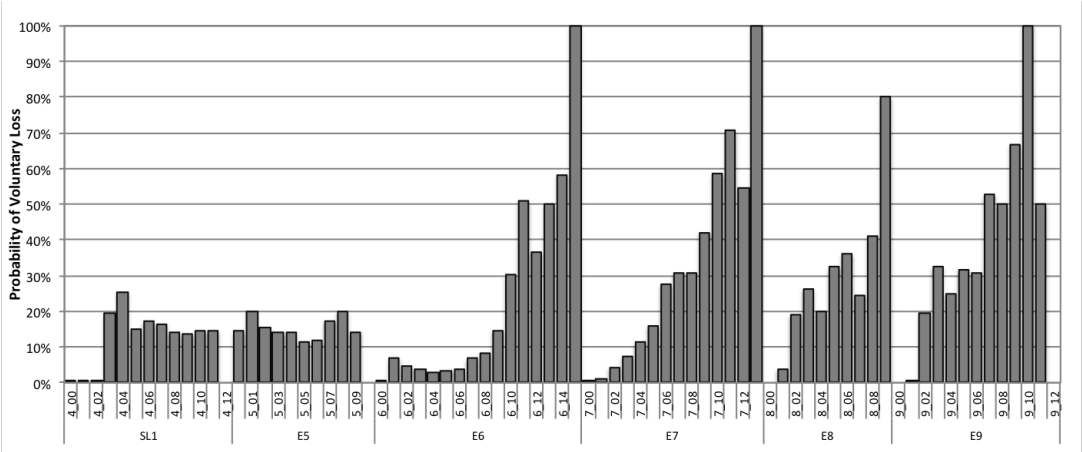


Figure 3.5: Probabilities of voluntary loss from each state in a period

TIG	E4	E5	E6	E7	E8	E9
0	27.6%	4.7%	2.4%	2.3%	0.9%	0.0%
1	11.6%	7.2%	3.6%	2.2%	0.8%	0.4%
2	10.0%	10.2%	4.0%	2.5%	0.9%	1.4%
3	14.7%	14.2%	4.0%	1.8%	0.7%	1.3%
4	18.0%	12.9%	4.9%	1.2%	1.0%	6.1%
5	20.2%	10.9%	4.0%	0.9%	0.8%	0.0%
6	18.0%	19.4%	3.5%	1.4%	2.3%	0.0%
7	20.9%	8.4%	2.5%	2.6%	0.0%	0.0%
8	22.5%	20.1%	0.7%	2.2%	0.0%	0.0%
9	21.1%	0.0%	0.9%	1.3%	RCP	0%
10	33.1%	RCP	0.7%	7.6%	-	0%
11	39.5%	-	2.3%	5.0%	-	RCP
12	RCP	-	2.9%	0.0%	-	-
13	-	-	12.5%	RCP	-	-
14	-	-	0.0%	-	-	-
15	-	-	RCP	-	-	-

Table 3.2: CMF 11 Average Involuntary Loss Rates FY06-09

noticeably low ($< 5\%$) at the grade of E6; particularly likely for voluntary separations due to the associated time-in-service requirements and the proximity to retirement eligibility.

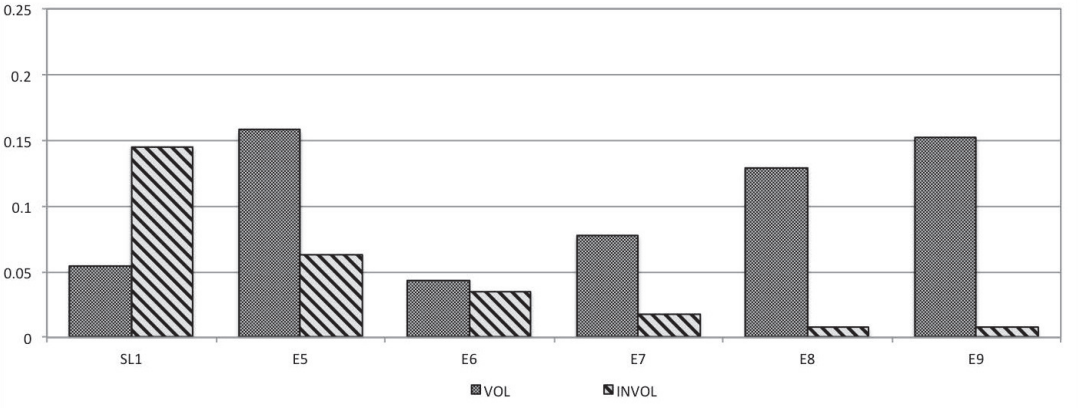


Figure 3.6: Comparison of loss probabilities in a single period by grade

3.5 An Aggregated Markov Chain Model

The Markov chain model presented in Section 3.4 defines state as grade and time-in-grade combinations. As we showed in Section 3.4, the model allows us to make many detailed observations. This section considers an aggregated model where the states are grouped to the grade level. Grade is arguably the most widely used unit of analysis for military personnel planning. The aggregated model can be used to answer many managerially relevant questions for different grades. We also use the aggregated Markov chain to validate our model using a chi-squared goodness-of-fit test.

3.5.1 State Aggregation by Grade

We aggregate states by grade to form a new Markov chain. The states for each grade level forms a state in the new model. For example, $(SL1\_0, SL1\_1, \ldots, SL1\_12)$ makes up the state s_1. In this fashion, The six Grade categories (Skill Level 1 through E9) make up states s_1 to s_6. s_7 and s_8 represent voluntary and involuntary loss states, respectively. Figure 3.7 shows the state transition diagram for the aggregated Markov chain.

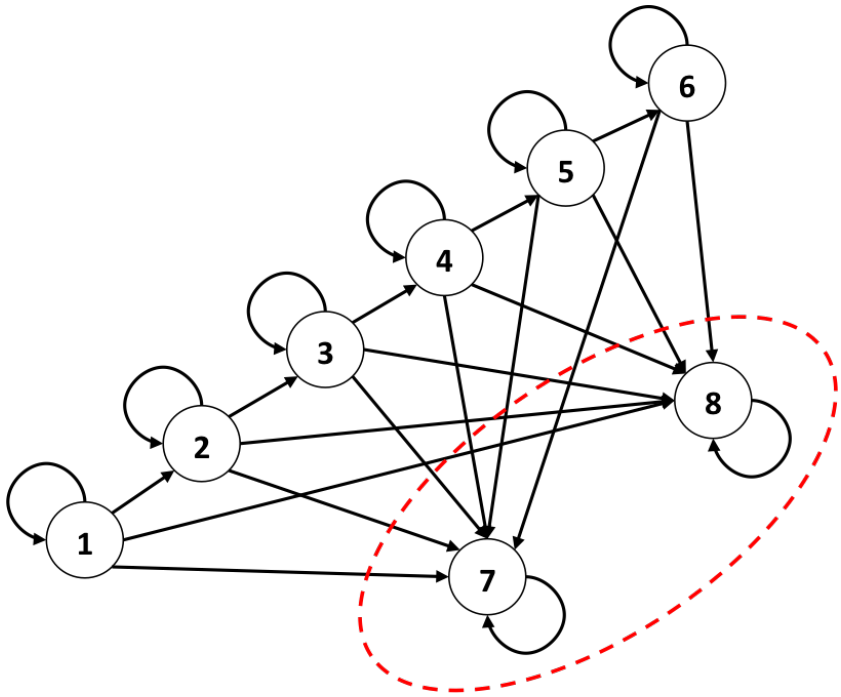


Figure 3.7: State Transition Diagram for the Aggregated Markov Chain Model

We use an 8×8 matrix $P = [p_{ij}]$ to represent the transition matrix of the aggregated Markov chain, where p_{ij} denotes the transition probability from state s_i to state s_j. The transition matrix P can be generated from the transition matrix M. The idea is that transitions within the same grade level are considered transitions back to the same state in the aggregated model, while only promotions or voluntary/involuntary separations are considered transitions between states. Clearly, we have $p_{77} = p_{88} = 1$ because states s_7 and s_8 are absorbing states. The resulting transition matrix P is given by

$$P = \begin{bmatrix} 0.6691 & 0.1588 & 0 & 0 & 0 & 0 & 0.0646 & 0.1075 \\ 0 & 0.5335 & 0.2542 & 0 & 0 & 0 & 0.1495 & 0.0627 \\ 0 & 0 & 0.6992 & 0.1110 & 0 & 0 & 0.1408 & 0.0490 \\ 0 & 0 & 0 & 0.8100 & 0.0800 & 0 & 0.1000 & 0.0100 \\ 0 & 0 & 0 & 0 & 0.8150 & 0.0499 & 0.1282 & 0.0069 \\ 0 & 0 & 0 & 0 & 0 & 0.8380 & 0.1554 & 0.0066 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 \end{bmatrix}.$$

3.5.2 Validation and Bootstrap Confidence Intervals for P

The transition matrix for the Markov chain is validated using a Chi-square goodness-of-fit test applied to our sample data (Montgomery and Runger[70]). The Chi-square goodness-of-fit test provides a useful measure of fit to our data as described by Anderson and Goodman[2]. A test of the difference between the observed counts at each state in an initial period 0 and final period 3 are evaluated using the transition probability matrix P.

Let O be the vector of initial observed counts and O' be the vector of counts after three period.

Under the estimated transition matrix P, the vector of expected count is given by $E = O^T P^3$, where E_i denotes the expected count in state i. The test statistic is given by

$$\chi_0^2 = \sum_{i=1}^{8} \frac{(O_i' - E_i)^2}{E_i}$$

Table 3.3 lists the values of vectors O, O', and E. The test statistic $\chi_0^2 = 6.7604$, which is less than $\chi_{0.05,7}^2 = 14.067$. Hence the test is insignificant at P-value 0.05, implying that the transition matrix is a good fit.

Period	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8
O	7752	1869	1692	779	258	54	0	0
O'	2259	1690	1762	862	285	67	3060	2419
E	2322.1	1624.7	1718.2	877.9	302.2	66.6	3064.6	2427.0

Table 3.3: Expected and observed counts for the goodness-of-fit test

Next, we use the bootstrap method to assess the uncertainty of the maximum likelihood estimate and to construct confidence intervals for the transition matrix. Let $\tilde{C}$ be the aggregated count matrix. Let t_r denote the total number of transitions for row r. We bootstrap row r by sampling t_r transitions with replacement from the observed t_r transitions. For example, since there are 69,835 transitions in the first row of $\tilde{C}$, 69,835 draws are taken from the corresponding empirical distribution to form a new set of transition counts for row 1. This process is repeated for each row until a new count matrix C^* is formed for each bootstrap sample. The new set of transition counts for row r of the count matrix is used to construct a new set of transition probabilities and a new transition probability matrix, P^*. The process is repeated for a number of bootstrap samples until a confidence interval can be constructed for the function.

Table 3.4 reports the confidence interval for the transition matrix based on 500 bootstrap samples. The confidence intervals are created from a collection of bootstrapped transition matrices, which approximate sampling distributions. The goal of bootstrap confidence interval theory is to calculate dependable confidence limits for a parameter of interest θ from the bootstrap distribution

of $\hat{\theta}$. Having chosen to use a particular $\hat{\theta}$, bootstrapping is a general methodology for answering how accurate is it as an estimator of θ?

We can use bootstrapping to assess the uncertainty of each probability in the transition matrix as well as any function of the transition matrix. While sensitivity analysis is a very helpful technique to investigate the behavior of a Markov model, Craig and Sendi[22] claim that it should not be used to construct a confidence interval because it does not take into account model restrictions and complex dependency of all the transition probabilities. To assess the precision of this estimate, it substitutes considerable amounts of computation in place of theoretical analysis and is more accurate than standard errors when dealing with nonparametric confidence intervals (DiCiccio and Efron[33]).

Initial State	s_1	s_2	s_3	s_4
s_1	(0.66899, 0.66929)	(0.15871, 0.15896)	-	-
s_2	-	(0.53333, 0.53390)	(0.25387, 0.25438)	-
s_3	-	-	(0.69906, 0.69961)	(0.11078, 0.11119)
s_4	-	-	-	(0.80959, 0.81028)
s_5	-	-	-	-
s_6	-	-	-	-
Initial State	s_5	s_6	s_7	s_8
s_1	-	-	(0.06452, 0.06468)	(0.10733, 0.10753)
s_2	-	-	(0.14929, 0.14971)	(0.06262, 0.06290)
s_3	-	-	(0.14046, 0.14088)	(0.04888, 0.04915)
s_4	(0.07978, 0.08027)	-	(0.09979, 0.10033)	(0.00990, 0.01007)
s_5	(0.81405, 0.81525)	(0.04975, 0.05042)	(0.12776, 0.12888)	(0.00682, 0.00707)
s_5	-	(0.83741, 0.84006)	(0.15339, 0.15603)	(0.00627, 0.00684)

Table 3.4: Confidence intervals for transition probabilities based on 500 bootstrap samples

3.5.3 Interpretations

One of the benefit of the Markov chain model is that it allows us to draw conclusions on long-run statistics, even though our model is only estimated from data with a limited time range. In this section, we expect some interesting interpretations from our model.

A useful concept in the analysis of Markov chains is the fundamental matrix, which we will use subsequently. Following Cyert et al.[23] and Ross[81], we partition the matrix, P as follows:

$$P = \left[\begin{array}{c|c} Q & R \\ \hline 0 & I \end{array}\right].$$

In the above, Q is a 6×6 matrix, R is a 6×2 matrix, 0 is a 2×6 zero matrix and I is the identity matrix. With this setup, we can compute the **fundamental matrix**

$$\begin{aligned} N &= \sum_{k=0}^{\infty} Q^k \\ &= I + Q + Q^2 + Q^3 + \cdots + Q^k + \cdots \\ &= (I - Q)^{-1} \\ &= \begin{bmatrix} 3.0221 & 1.0287 & 0.8694 & 0.5079 & 0.2196 & 0.0677 \\ 0 & 2.1436 & 1.8115 & 1.0583 & 0.4577 & 0.1410 \\ 0 & 0 & 3.3245 & 1.9422 & 0.8399 & 0.2587 \\ 0 & 0 & 0 & 5.2632 & 2.2760 & 0.7011 \\ 0 & 0 & 0 & 0 & 5.4054 & 1.6650 \\ 0 & 0 & 0 & 0 & 0 & 6.1728 \end{bmatrix}. \end{aligned} \tag{3.1}$$

The (i, j)-th entry of N, n_{ij}, is the expected number of transitions (visits) to state j before being absorbed, starting in state i. The fundamental matrix plays an important role in Markov chain theory. For example, the matrix NR provides probability estimates of being absorbed into each of the absorbing states, s_7 and s_8. The definition (3.1) of fundamental matrix assumes an infinite number of transitions. This definition gives a compact representation of N using Q. The career path of an enlisted personnel has a finite timeline, typically 30 years. Therefore, we expect that computations based on the fundamental matrix has certain level of error. In Appendix D, we show that the error from using an infinite summation is very small.

The variance of n is given by

$$V = N(2N_{dg} - I) - N_{sq} \tag{3.2}$$

where N_{dg} is a copy of matrix N with only the diagonal entries and N_{sq} is the Hadamard product of N (Beck[8]). Here, The Hadamard product is the the elementwise product of two matrices (Styan[83]). Using (3.1) in (3.2), we have

$$V = \begin{bmatrix} 6.1108 & 2.3234 & 4.1552 & 4.5804 & 2.1065 & 0.7630 \\ 0 & 2.4515 & 6.9516 & 8.9618 & 4.2805 & 1.5795 \\ 0 & 0 & 7.7276 & 14.7298 & 7.5344 & 2.8682 \\ 0 & 0 & 0 & 22.4377 & 17.1490 & 7.4624 \\ 0 & 0 & 0 & 0 & 23.8130 & 16.1183 \\ 0 & 0 & 0 & 0 & 0 & 31.9311 \end{bmatrix}.$$

where each element of V provides the variance for the number of periods one expects to be in state j given a start in state j.

Figure 3.8 shows the probability of remaining in the Army each year over a horizon of 30 years when starting in state s_1 (Skill Level 1). **what is interesting about this?** This graph is interesting because it allows us to see how changes to short-term behavior can affect long-term continuation rates with tracking long-term cohort data. From Figure 3.8, we can see that there is only a 45% probability that an enlisted individual will remain in the Army beyond 4 years. Only 17% can be expected to still be remain in the Army for 10 years. Whether these rates are acceptable is a strategic decision; however, it is useful to know whether short term policies shift the curve in a desirable direction.

From the fundamental matrix (3.1), the expected amount of time an individual spends in skill level 1 is 3.0221. The expected amount of time an individual spends as an E7 is 0.5079 years. Even though this information is interesting, it is not intuitive because it incorporates the probability of not reaching the state. A somewhat related, but perhaps more interesting question, is the expected

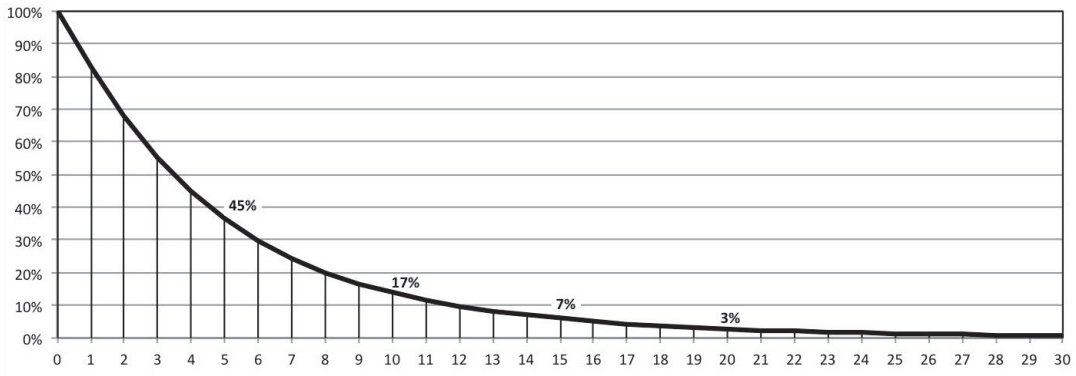


Figure 3.8: Continuation rates over 30 year time horizon

amount time someone spends in grade E7 given he/she reaches the grade. Manipulations of the fundamental matrix N can be used to answer a number of such questions. Such an approach is taken by many authors, including Doyle and Snell[34] and Cyert et al.[23].

First, the row sum of matrix N,

$$T = \begin{bmatrix} 5.7153 \\ 5.6121 \\ 6.3652 \\ 8.2402 \\ 7.0704 \\ 6.1728 \end{bmatrix}$$

gives the expected number of periods before being absorbed from each state. The interpretation of T is quite intuitive. The expected number of cycles or periods before being absorbed into a loss state is 5.7153 years. This expectation increases with an increase in grade due to the increase probability of remaining until the retirement threshold. The peak is upon reaching the grade of E7 when an individual is expected to remain for 8.2402 years until being absorbed, presumably as a voluntary loss category of retirement. Beyond the grade of E7, the expected time until absorption decreases due to the finite time horizon of mandatory retirement or reduced incentive for delaying

retirement.

Second, the probability of moving from each of the transient states to an absorbing state is given by the matrix NR, which is shown below:

$$NR = \begin{bmatrix} 0.5609 & 0.4390 \\ 0.7619 & 0.2378 \\ 0.8102 & 0.1898 \\ 0.9270 & 0.0730 \\ 0.9517 & 0.0483 \\ 0.9593 & 0.0407 \end{bmatrix}.$$

The elements in NR has interesting interpretations. For example, the probability of eventually becoming a voluntary loss 0.5609 is higher than the probability of becoming an involuntary loss 0.4390 when starting from the initial state, s_1. Despite the significantly higher involuntary loss rates of skill level 1 individuals, these probabilities account for expected future transitions to other states. We can see that the disparity between absorbing into the two loss states diverges significantly as the grade level progresses.

It is possible to use bootstrap method to estimate the confidence intervals for the absorption times from different starting states. Consider state s_1, which has expected absorption time of 5.7153 years. The results of 500 bootstrap samples are shown in Figure 3.9. Using an two-tailed confidence interval, the 95% confidence interval for the expected absorption time is $[5.715, 5.721]$.

The probability of ever making a transition into state j starting in state i is given by

$$f_{ij} = \frac{n_{ij}}{n_{jj}}$$

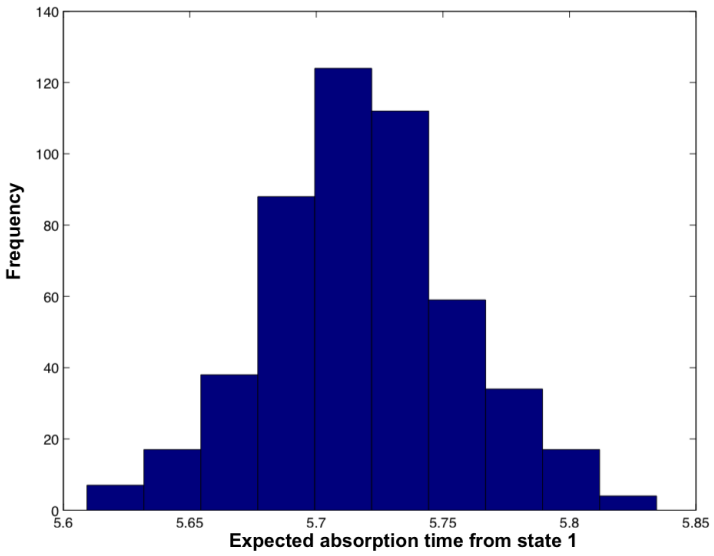


Figure 3.9: 500 bootstrap samples of absorption time from state 1

where n_{ij} is the (i,j)-entry of N (Ross[81]). Let $F = [f_{ij}]$. We have

$$F = \begin{bmatrix} 0.6691 & 0.4799 & 0.2615 & 0.0965 & 0.0406 & 0.0110 \\ 0 & 0.5335 & 0.5449 & 0.2011 & 0.0847 & 0.0228 \\ 0 & 0 & 0.6992 & 0.3690 & 0.1554 & 0.0419 \\ 0 & 0 & 0 & 0.8100 & 0.4211 & 0.1136 \\ 0 & 0 & 0 & 0 & 0.8150 & 0.2697 \\ 0 & 0 & 0 & 0 & 0 & 0.8380 \end{bmatrix}.$$

So, what is the probability of an enlisted personnel ever reaching the grade of E9? Based on F, the probability is close to 1%. In fact, an individual has less than a 5% probability of reaching the grade E8. Perhaps equally indicative of the likelihood of reaching the grade of E9 is that the probability of reaching this state is still less than 5% (0.0419) when an individual reaches the grade E6.

3.6 A Dynamic Programming Model for Military Personnel Retention

Based on the Markov chain estimated in Section 3.4, we construct a Markov decision process (MDP) model for a military personnel's stay or leave decision at each point of his/her career. The model captures career path dynamics, including military pay and career advance opportunities. We assume that in making a stay or leave decision, a military personnel weighs the expected military pay, including possible retirement benefits, against the expected pay from civilian job opportunities. Even though not explored in the present paper, our model can be used to shape and target incentives to military personnel.

The model is formulated as a finite-horizon MDP using an optimal stopping framework (Puterman[76]). The state x is the grade and time-in-grade combination in each period. Important assumptions are made in order to develop the model, such as the assumption that when an individuals decides to leave the military, they cannot return to the military at later periods in the time horizon. We assume that individuals do not change their skills during the time horizon and

that all military service is conducted while on active duty. Although promotions, involuntary losses and retention decisions occur at monthly intervals, following Section 3.4, a simplifying assumption is made to define the period length as one year. The objective of the model is to maximize total expected payoff over the finite horizon.

Since leaving the military is modeled as a decision, the transition probabilities in our model follows the transition probabilities estimated in Section 3.4, conditional on not being a voluntary loss. Let q_{xy} denotes the probability of transitioning from state x to state y in the following period, we have

$$q_{xy} = \frac{\theta_{xy}}{1 - \sum_{y \neq \mathbf{VL}} \theta_{xy}}, \qquad \forall x, y.$$

The state space includes all states for the Markov chain estimated in Section 3.4 except the voluntary loss state, which we denote by VL. Appendix B provides a notation dictionary.

3.6.1 Estimating Civilian Pay

A key input the MDP model is the expected civilian pay in the event of choosing to leave the military. As mentioned earlier, it is not reasonable to assume initial civilian pay is equal to the last military pay. We assume the MDP model is constructed for a specific military skill, which is cross-mapped to comparable civilian occupations using the Occupational Information Network (O*NET).[9] We then construct an empirical distribution of median expected civilian wages from earnings data of full-time wage and salary workers as reported by the Bureau of Labor Statistics.

Table 3.5 illustrates a distribution of potential civilian employment for an enlisted personnel with a military occupation of Infantry. The second column list the comparable civilian occupations for an infantryman. The third column reports the number of workers employed in each of the civilian occupation. The empirical distribution of the different occupations is reported in the second-to-last column, where we use α_i to denote the proportion of workers employed in occupation i. The compensation for each occupation is reported as a multiple, δ_i of the military compensation $m_{x,t}$. By using a multiplier to military compensation rather than a constant value (i.e median of civilian

[9] http://www.onetonline.org/crosswalk/

occupation i), we account for an increase in expected pay as military experience increases. It is reasonable to assume that two individuals with the same military occupation specialty and expected civilian occupation can expect different levels of pay attributed to technical acumen and leadership experience.

i	Civilian Occupation	Workers (1,000)	α_i (%)	δ_i
1	Training and Development Specialists	226	3.5	2.10
2	First-Line Supervisors of Police and Detectives	88	1.4	2.13
3	Correctional Officers and Jailers	312	4.8	1.32
4	Police Patrol Officers	578	8.9	1.77
5	Security Guards	572	8.8	0.95
6	First-Line Supervisor of Construction Trades and Extraction Workers	517	8.0	1.75
7	Construction Laborers	937	14.4	1.08
8	Operating Engineers and Other Construction Equipment Operators	315	4.9	1.36
9	First-Line Supervisor of Production and Operating Workers	574	8.8	1.63
10	Heavy and Tractor-Trailer Truck Drivers	2,368	36.5	1.25

Table 3.5: Cross-Mapping of Military to Civilian Skills (Enlisted Infantryman)

3.6.2 Military Compensation and Retirement Pay

Individuals have the choice to stay in the military and continue to receive military pay or leave the military before or after retirement eligibility. Enlisted personnel are compensated based on a military pay scale, which increases by pay-grade and years-of-service. Individuals receive additional compensation pay that includes a housing compensation based on local civilian housing markets and a basic allowance intended to pay for food. Individuals who leave the Army prior to retirement eligibility, do not receive any type of pension compensation. However, after reaching retirement eligibility, individuals can leave the military and receive long-term compensation.

Upon retirement, a military personnel receives a proportional amount of the so-called high-3 average retirement compensations (Hall[50]). Let $m_{x,t}$ denote the base military compensation for an individual in state x in period t. The high-3 average retirement compensations is calculated as

$$\widehat{m}_{x,t} = \frac{m_{x'',t-3} + m_{x',t-2} + m_{x,t-1}}{3}, \tag{3.3}$$

where x' and x'' are the states in two years preceding retirement. Note that the the calculation

of retirement compensation does not violate the Markovian property because it is reasonable to assume that an individual can be promoted with less than or equal to one year of time-in-grade, but not in two consecutive years. Therefore, the pay scale can be backed out from the state upon retirement.

The proportion of the high-3 average compensations received by an individual upon retirement depends on the total number of years in service. Let w_t denote the proportion received by an individual with total number of years of service t. Since military retirement pay is only earned if a Soldier serves a total of 20 years of active duty, $w_t = 0$ for $t < 20$. The standard retirement format (Mattock et al.[68]), an individual receives a base 50% plus 2.5% for each additional year served beyond 20 years. Hence w_t is defined as[10]

$$w_t = \begin{cases} 0, & \text{if } t < 20, \\ 0.5 + 0.025(t - 20), & \text{if } t \geq 20. \end{cases} \tag{3.4}$$

3.6.3 Value Function

First, consider the value of remaining in the Army for one more year. The expected compensation in period t for an individual in state x is given by

$$m_{x,t} + c_{x,t} + \beta \sum_{y=1}^{S} P_{x,y} V_{t+1}(y), \tag{3.5}$$

where S represents the total number of states and β denotes a personal discount factor such that $\beta = \frac{1}{1+\gamma}$ and γ is the personal discount rate.

In case the decision is to leave the army, we can use (3.6) to represents an individual's future total cash flow from that decision

$$w_t \widehat{m}_{x,t} \sum_{k=t}^{\tau-\eta} \beta^{k-t} + \delta_i m_{x,t} \sum_{k=t}^{T-1} \beta^{k-t}(1 + \xi_{k-t}) \tag{3.6}$$

The first term in the above cash flow equation represents an individual's expected retirement pay at $\tau - \eta$, where τ represents life expectancy and η represents an individual's age upon entering the

[10] Alternatively, an individual can choose a REDUX retirement option with a slightly different formula. Recent research (Hattiangandi et al.[53]) suggests that the REDUX option is not a good choice for most people. The REDUX option is further detailed in Appendix E.

Army (we assume 21 years). To simplify the model, we assume an individual's life expectancy to be the same; however, the model may be refined with the use of applicable data pertaining to life expectancies by selected attribute to better represent the military retirement time horizon. The second part of equation takes into account individual's initial civilian sector pay plus civilian pay raises ξ_{k-t} and time T is considered the ceiling for full retirement benefits.

Now, we are ready to construct the value functions. In the terminal period T, the only viable option is to retire, and no civilian pay will be received. By simplifying the first term in (3.6), we obtain the value function

$$V_T(x) = w_T \widehat{m}_{x,T} \frac{1-\beta^{\tau-\eta+1}}{1-\beta}. \tag{3.7}$$

For $t < T$, the decision is whether to stay in the army. We also incorporate the possibility of becoming an involuntary loss. For notational purposes, let IL denote the state of involuntary loss. Then $q_{x,\mathrm{IL}}$ is the probability of becoming an involuntary loss in state x. The optimality equations are given by

$$\begin{aligned} V_t(x) = (1-q_{x,\mathrm{IL}})E\Bigg[\max\Bigg\{ & m_{x,t} + c_{x,t} + \beta \sum_{y \neq \mathrm{IL}} \frac{q_{xy} V_{t+1}(y)}{(1-q_{x,\mathrm{IL}})}, \\ & w_t \widehat{m}_{x,t} \sum_{k=t}^{\tau-\eta} \beta^{k-t} + \delta_i m_{x,t} \sum_{k=t}^{T-1} \beta^{k-t}(1+\xi_{k-t}) \Bigg\}\Bigg] \\ & + q_{x,\mathrm{IL}} \left(w_t \widehat{m}_{x,t} \sum_{k=t}^{\tau-\eta} \beta^{k-t} + \delta_i m_{x,t} \sum_{k=t}^{T-1} \beta^{k-t}(1+\xi_{k-t}) \right), \qquad \forall x, t < T. \end{aligned} \tag{3.8}$$

The right-hand side of equation 3.8 is comprised of two parts. The first represents the probability of not being an involuntary loss multiplied by the expected maximum value of staying in the military or leaving the military. The second represents the probability of an involuntary loss multiplied by the value of leaving the Army. The value of of staying in the military, $m_{x,t} + c_{x,t} + \beta \sum_{y \neq \mathrm{IL}} \frac{q_{xy} V_{t+1}(y)}{(1-q_{x,\mathrm{IL}})}$, consists of military compensation plus additional compensation plus the conditional value function in the next period. The value of leaving the Army, $w_t \widehat{m}_{x,t} \sum_{k=t}^{\tau-\eta} \beta^{k-t} + \delta_i m_{x,t} \sum_{k=t}^{T-1} \beta^{k-t}(1+\xi_{k-t})$, consist of the value of future retirement pay plus the value of future civilian pay.

3.6.4 Optimal Policy Index

One method of evaluating the stay or leave decision between individuals is to compare the expected values of compensation for each decision at different states and time periods. The total expected value of compensation will change in respect to each variable and it is worth noting the scale in which the values change. For the following computational experiments, we use a life expectancy of 75 years. We assume that the average age of individuals entering the Army is 22 years, the total time an individual works in the military and civilian sector is 40 years, and the civilian pay raise per period is 2.5%.

Applying the assumptions above to the value function, the following time parameters are used for computation: $\tau = 75$, $\eta = 22$, and $T = 40$. The parameters $\beta = 0.8696$ and $\xi = 0.025$ relate to the personal discount factor and civilian pay raise. Probability matrices are used for q and P.

Because of variations across occupations in civilian pay, the stay or leave decision depends on the realization of civilian occupation. We use a policy index to represent the propensity to leave. For example, an optimal policy index of 0.3 would indicate a 30% probability that the expected compensation value of leaving is greater than the value of staying, given an empirical distribution of expected civilian pay. A depiction of the indices for each state-time combination within the Infantry skill are shown in Figure 3.10. Darker shading levels represent a higher index values. The shading clearly depicts the impact of retirement benefits after 20 years as well as the forced decisions at each retention control point.

Individuals are characterized by different attributes and preferences which lead to a variance in personal discount rates. The policy indices in Figure 3.10 are calculated using a constant personal discount rate of 15%. This personal discount rate is consistent with the average rate simulated by Asch et al.[3], which they found to be the best fit and most reasonable approximation of enlisted Army behavior. It is worth noting that a personal discount rate of 15% is significantly lower than the mean nominal enlisted discount rates reported by Warner and Peeter[88]. Moreover, Warner

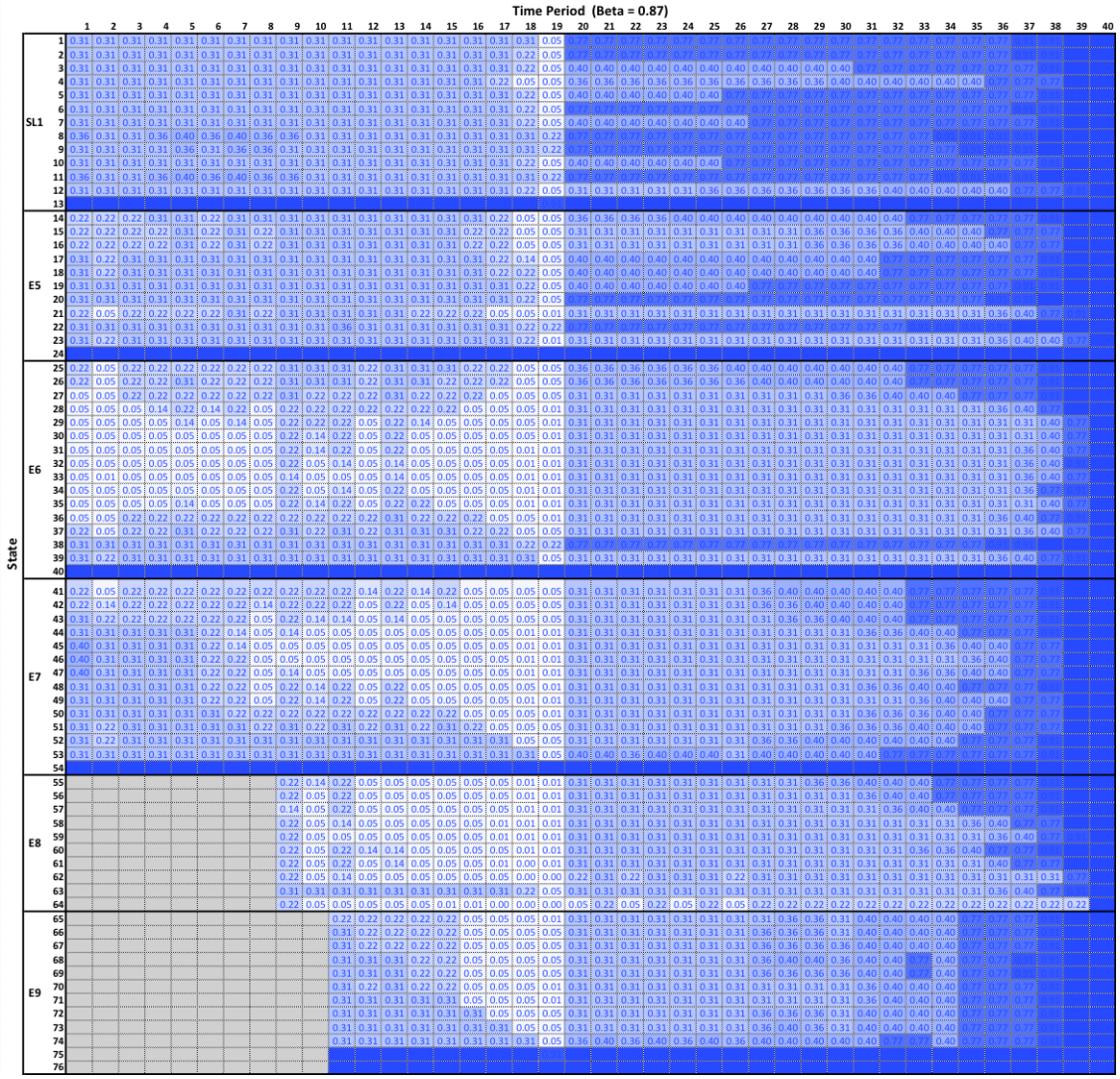


Figure 3.10: Retention Index by State and Time Period

and Pleeter[88] suggest that there is a significant difference between officer and enlisted discount rates that is significantly attributable to observable demographic and characteristics differences.

Table 3.6 shows how changes to the personal discount factor affect the average index values for a selection of grade and YOS combinations. The average index is calculated over all possible time-in-grade values. Intuitively, higher discount rates result in increased compensation values for leaving. Note that an individual with he grade of E-5 and 7 years-of-service has an average index value of 0.090 when $\beta = 0.99$. Therefore, they have strong propensity for continued service. When then personal discount factor decreases to 0.75, the index increase to 0.373 and the preference is less strong.

Grade	YOS	Personal Discount Factors (β)					
		0.99	0.95	0.90	0.85	0.80	0.75
E4	5	0.077	0.085	0.246	0.277	0.315	0.415
E5	7	0.090	0.090	0.191	0.273	0.282	0.373
E5	9	0.090	0.090	0.218	0.273	0.336	0.445
E6	12	0.088	0.113	0.206	0.275	0.331	0.419
E7	15	0.179	0.186	0.271	0.336	0.450	0.485

Table 3.6: Average Index values for all Time-in-Grade.

Figure 3.11 shows the change in expected value for each combination of state and years-of-service for two different discount factors. The impact of the personal discount rate is most evident in the states associated with lower pay grades.

3.7 Integrating Dynamic Programming Results in Predictive Modeling

The dynamic programming model in the previous section tries to rationalize the stay-or-leave decision for military personnel by taking into account long-term payoffs from stay or leave decisions. There are two major differences between the model and more commonly used classification approach. First, the model captures long-run personnel dynamics through its use of the estimated transition probabilities. Second, it does not use demographic variables typically employed in clas-

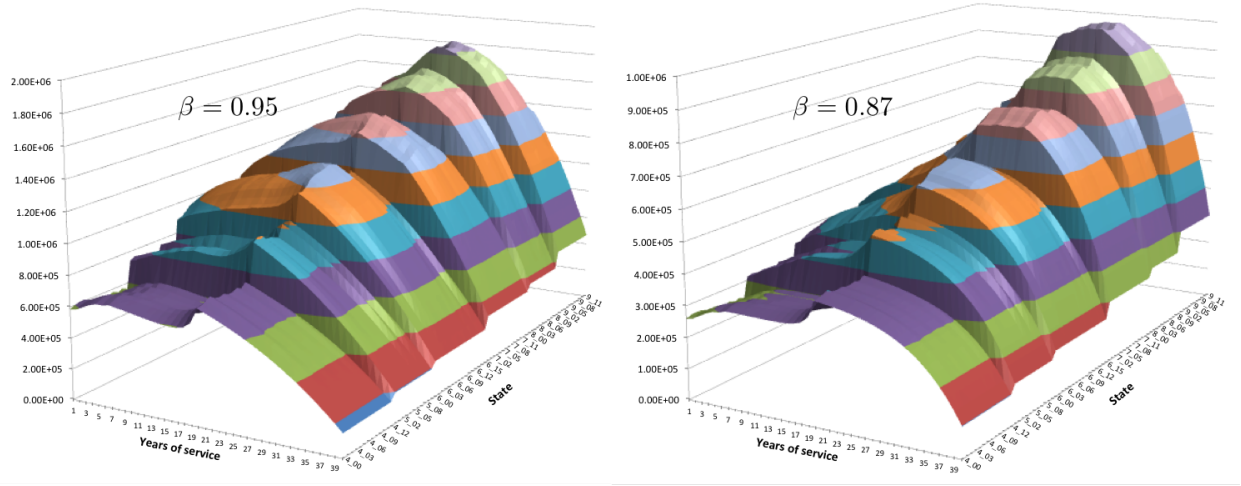


Figure 3.11: Value by State and Time Period

sification approaches. Because of these difference, we believe the result from the model can be used to supplement popular predictive modeling approaches. In this section, we discuss how to integrate the results of the dynamic programming model with one of the most popular binary classification method, the logistic regression.

We first discuss a logistic regression model based on the same data as the Markov chain model in Section 3.4. The variables used in the binary logistic regression equation are described earlier in Table 1. In addition to variables included in the original database we construct additional endogenous and exogenous variables such as **age**, **months since last deployment**, and **employment rate of change**. The employment exogenous variable is constructed from data obtained from the Bureau of Labor Statistics by calculating a twelve month moving average of the monthly change in employment rate. Table 3.7 shows the Logit results. Most of the results are aligned with tuition, including the impact of experience and grade being more influential than marital status or number of dependents. The table only includes the racial category of **Black** because there is not a significant difference in behavior between **White**, **Hispanic** and **Asian/Other**.

There are two approaches to improve the classification accuracy of the logistic regression model. The first one is to incorporate the policy indices from the dynamic programming model as an additional variable in the logit model. The second one is to use a combination of the DP policy indices and the logistic regression classification as a classifier. That is, we use a weighted index $\alpha I + (1 - \alpha)L$ as a classifier, where I is the DP policy index, L is the logit index, and α is the weight assigned to the DP policy index I.

Table 3.8 shows the cross-validation results of classifying the stay-or-leave decision using binary logistic regression. The overall results provide a marginal lift in comparison to predicting a stay decision for all individuals. Moreover, the majority of the error is a result of misclassifying the observed leave decisions. The best fit logit model accurately predicts the stay decision in 89.9% of observations and classifies 58.7% of the leave decisions correctly. This finding is significant, since the focus of incentive programs should is directed at targeting individuals who are predicted to leave. Table 3.8 also shows that the additional parameter does not provide noticeable improvement

	Coefficients	
Variables	Logit	Logit w/ Index
Constant	-3.025***	-4.073***
	(0.053)	(0.087)
MINOR_DEP	-0.413***	-0.417***
	(0.018)	(0.018)
E5	-0.093*	-
	(0.031)	-
E6	-2.134***	-2.003***
	(0.061)	(0.055)
E7	-2.386***	-2.573***
	(0.113)	(0.118)
E8	-1.081***	-1.392***
	(0.286)	(0.307)
E9	-2.120*	-2.560***
	(0.690)	(0.683)
AFQT	-0.001*	-0.001
	(0.001)	(0.001)
REENLIST_QY	-2.212***	-2.177***
	(0.034)	(0.034)
TIME_IN_SERVICE	0.809***	0.769***
	(0.011)	(0.011)
AGE_CENTER	0.130***	0.163***
	(0.008)	(0.008)
AGE_CENTER^2	-0.009***	-0.011***
	(0.000)	(0.000)
MARRIED	-0.564***	-0.556***
	(0.030)	(0.030)
DIVORCED	-0.686***	-0.670***
	(0.090)	(0.091)
BLACK	-0.430***	-0.440***
	(0.057)	(0.058)
EDUCATION_CENTER	0.079***	0.090***
	(0.017)	(0.017)
EMPLOYMENT	-0.005***	-0.005***
	(0.001)	(0.001)
MONTHS_SINCE_DEPLOY*DEPLOY	0.004***	0.004***
	(0.001)	(0.001)
INDEX	-	0.563***
	-	(0.035)

Standard errors in parentheses; $*\ p < .05$, $**\ p < .01$, $***\ p < .001$

Table 3.7: Estimated Logistic Regression Coefficients

to the overall classification accuracy. The misclassification error is not significantly improved when predicting the leave decision; however, it continues to be the primary source of error. When performing logit with the Index parameter and a cut for the decision at 0.3, the overall classification percentage slightly decreases. However, the classification accuracy for the leave decision increases to 84.5%.

Method	Cut	Overall %	Stay %	Leave %
Logit	0.5	79.6%	89.9%	58.7%
Logit w/ Index	0.5	79.6%	89.8%	59.0%
Logit w/ Index	0.3	76.9%	73.1%	84.5%
Index	0.3	71.1%	99.9%	12.9%
Index	0.5	70.3%	99.9%	10.5%

Table 3.8: Classification Results of Stay or Leave Decisions (FY 06-09)

Table 3.9 shows the forecasting accuracy for different parameter combinations using the weighted approach, where we also experimented with different cut-off value z. The best overall accuracy with a personal discount factor of 87% is achieved with weights of $\alpha = 0.40$ for the retention index value and $1 - \alpha = 0.60$ for the logit value. The best cut line between stay and leave for the weighted index is $z = 0.33$. The two scenarios shown below the top line show the accuracy levels when z and α are fixed at 0.5 respectively. A slightly better accuracy specifically regarding the leave decision is achieved with $\alpha = 0.5$ and $z = 0.30$.

Predicting a leave decision based on a dynamic programming approach filters a portion of leave predictions from the hybrid model. However, the smaller isolated set is predicted purely on expected compensation values. Predicted leaves decisions in states and time combinations with a positive value for stay versus leave compensation are not as likely to be influenced by monetary incentive.

The results of this model are significant beyond the integration of both modeling techniques because, individually, we do not significantly sacrifice predictive capability using the dynamic programming model. In fact, we can achieve slightly better results while requiring only three factors:

β	Overall Accuracy%	Stay%	Leave%	α	z
0.87	80.8%	85.1%	72.2%	0.40	0.33
0.87	79.6%	89.9%	58.7%	0.00	0.50
0.87	80.7%	84.1%	73.9%	0.50	0.30
0.95	80.75%	85.80%	70.6%	0.40	0.26
0.95	79.61%	89.94%	58.7%	0.00	0.50
0.95	80.72%	85.48%	71.1%	0.50	0.30

Table 3.9: Classification Accuracy for Different Combinations of α and z Values

grade, TIG and TIS. This eliminates the need for collection and analysis of specific information about individuals. More importantly, it significantly reduces the potential perception of individual bias or discrimination that would exist if incentives are tailored by more descriptive factors. This result is important because of the nature of policy making. Many policies, particularly regarding incentive options, are naturally broad due to an aversion to policies that are based on demographic characteristics. Targeting individuals based on grade and time-in-grade states, present a framework in which previously unpalatable policy frameworks involving explanatory demographic characteristic are replaced by feasible state criterion.

3.8 Conclusion

We represent the enlisted career network as a discrete-time homogeneous Markov chain and estimate its transition probabilities using three years of personnel data from the US Department of Defense. The model allows us to answer a wide variety of questions related to personnel behavior. Understanding the progression of individuals throughout a career timeline can shape policies that influence the composition of the personnel inventory such as promotion criteria, retention control points, and forced reduction measures. The estimated parameters for the model are used to construct a stochastic dynamic programming model to understand individual stay-or-leave decisions.

The stochastic dynamic programming model provides an alternative to classical classification approaches to evaluating the stay-or-leave decisions. This approach can predict retention behavior

more accurately than some existing approaches, such as logistic regression, which omit expected future compensation. Another advantage is that the dynamic programming model does not use personal attributes, which can lead to perceived discriminating in personnel actions. Future research can build on our work to optimize resource allocations and tailor retention incentives.

Chapter 4

A Graph Coloring Approach to the Deployment Scheduling and Unit Assignment Problem

4.1 Introduction

This paper proposes a graph coloring approach to the deployment scheduling and unit assignment (DSUA) problem and addresses one of the external factors of personnel inventory behavior, deployments. The configuration of persistent unit deployments affect both individual perceptions of service palatability and operational effectiveness. We focus on the unit deployment problem faced by the U.S. Army. There is little evidence to suggest any analytical underpinnings to U.S. Army deployment scheduling patterns. Moreover, a variety of personnel trends, ranging from attrition rates to medical and family issues, have been linked to deployment frequency. While certain levels of persistent operational demand inevitably lead to an increased stress on the force, it is reasonable to conclude that improving deployment measures known to affect personnel is of vital importance.

The U.S. Army frequently acknowledges the importance of stabilization time between deployments and the impact of frequent long deployments. The ratio of the time a unit is deployed in theater or "boots-on-the-ground" (BOG) to the time not deployed (Dwell) is referred to as a BOG:Dwell ratio. The Secretary of Defense has historically set 1:2 as an objective, whereas the Army prefers a sustained ratio of 1:3. From 2003 to 2011, neither of these goals was met, as the Army deployed soldiers to Iraq and Afghanistan at BOG:Dwell ratios closer to 1:1 than 1:2 (Bonds et al. [12]). During that period, deployments for General Purpose Forces (Division and below) supporting named operations outside the continental United States was standardized at 12

months. In an effort to increase the quality of life for Soldiers and families, rebalance the force, and provide better alignment of units the Army began transitioning to a 9-month deployment period (Department of the Army [30]).

> The mental health advisory teams sent to Operations Iraqi Freedom and Enduring Freedom have conducted investigations and provided insight and recommendations to improve force health. In November 2006, Mental Health Advisory Team IVs central findings included the following observations: Overall, Soldiers had higher rates of mental health problems than Marines, and Deployment length was related to higher rates of mental health problems and marital problems. Key recommendations included extending the interval between deployments and decreasing deployment length. The Army has only focused on the first recommendation.
>
> — Lieutenant Colonel Heather Reed, Military Review May-June 2011

Numerous studies focus on measuring the stress on the force caused by deployments, to include physical and mental health of individuals. However, there is little evidence that current unit sourcing and scheduling methods focus on improving these issues. Sourcing requirements complicate unit scheduling because maximizing the level of operational readiness by seeking to rotate units to familiar regions can conflict with deployment timing goals. This paper proposes a deployment scheduling method, which when compared to traditional deployment scheduling techniques simultaneously improves multiple simultaneous objectives ranging from length and frequency of deployments to operational consistency of locations.

4.2 Scheduling Problems

The scheduling literature is one of the richest in operations research. However, traditional scheduling problems such as job scheduling, machine scheduling and employee scheduling problems do not provide a convenient fit to achieve the goals of military deployment scheduling. Moreover, the small amount of specialized literature related to military deployment scheduling is focused on lower level planning issues such as route and logistics scheduling.

Machine scheduling and production scheduling problems vary in complexity. Lenstra et al.[64]

show that while some classical machine scheduling problems are efficiently solvable, others are NP-hard. For instance, Cheng and Chen[17] consider sequencing a set of simultaneously available jobs on several identical parallel-machines with earliness and tardiness penalties. The objective is to minimize some penalty function of earliness, tardiness and due-date values. This NP-hard problem is similar but not exactly the same as the deployment scheduling and unit assignment problem where jobs represent units and machines may be viewed as locations.

Modeling of personnel or employee scheduling problems appear to have structural requirements and constraints more similar to the deployment scheduling problem. Personnel scheduling problems have been studied extensively. According to Baker[7], three main groups can be distinguished: shift scheduling, days off scheduling and tour scheduling. The latter combines the first two types. Alfares[1] proposes ten categorical approaches for solving tour scheduling problems, ranging from integer/linear/goal programming to metaheuristics.

Glover and McMillan[44] present the general employee scheduling problem which extends the standard shift scheduling problem by discarding key limitations and overcoming large scale complexity by generating solutions with a combination of management science and artificial intelligence methods. Glover[42] also shows tabu search algorithms are an applicable strategy for combinatorial optimization problems that involve scheduling. Tabu search algorithms have been applied to scheduling problems ranging from distributing workloads among machines to large scale employee scheduling. As mentioned by Glover[42], tabu search methods possess the capability of being merged with other methods. Like machine scheduling problems, employee scheduling problems do not precisely fit the DSUA problem because of unique constraints and objectives.

Complex combinatorial scheduling problems have been approached with graph coloring methods. As we identify later in this paper, the structural nature of some scheduling problems makes graph coloring an attractive formulation. Gamach et al.[41] use graph coloring methods to determine a feasible schedule for crew scheduling problem within the airline industry. Moreover, they propose a new methodology to determine the existence of a feasible solution based on a graph coloring model and and a tabu search algorithm. Building upon observations by Gamach et al.[41], we

adapt graph coloring to the unique structure, goals and constraints of the deployment scheduling described below.

4.2.1 Deployment Scheduling and Unit Assignment Problems

Hodgson et al.[56] developed a deployment scheduling analysis tool (DSAT) for studying military deployment scenarios and optimizing the scheduling of transportation assets. The DSAT enhances exiting deployment schedules with heuristics for equipment and transportation routing, but does not optimize the top-level unit deployment schedule. McKinzie and Barnes[69] surveyed the numerous legacy and current strategic mobility models of the time period. While varying in intent, each focuses on some aspect of the defense transportation system and treat unit deployment schedules or time phased force deployment data (TPFDD) as an input. Aviles[5] created an integer programming model to develop deployment schedules from two nearly simultaneous major regional conflicts using a sequential heuristic. This method meets the initial demands but does not extend to unit deployment cycles during persistent conflict.

4.2.2 Unit Deployment Scheduling

Currently, Forces Command (FORSCOM) produces a deployment schedule from the ARFORGEN Synchronization Tool schedule which informs subsequent sourcing tools. The output from FORSCOMs ARFORGEN Synchronization Tool provides critical arrival and return dates for each rotating unit (Hughes et al.[59]). However, units are assigned at rolling time intervals and do not incorporate analytical methods to optimize manning objectives, which will be described in Section 4.3.1.

The ARFORGEN Synchronization Tool simulates discrete events to achieve a predictive view of the Army inventory moving through the ARFORGEN process over time. It integrates data warehousing, discrete event modeling, scheduling, optimization algorithms and data visualization into a scenario management infrastructure for sourcing. However, this tool requires a human-interaction for selecting sourcing procedures and is most effective for "what-if" analysis or synchronizing pro-

cesses without necessarily adhering to a strategic optimization goal.

One of the most common measurements for stress on the force is the Boots-on-the-Ground (BOG) over dwell ratio. For the active component of the U.S. Army, the BOG:Dwell ratio is the ratio of deployed periods to non-deployed periods. Theoretical goals are often defined in *steady-state* scenarios. A steady-state rotation occurs when the amount of forces in the Available Force Pool exceeds requirements (supply exceeds demand). A steady-state rotation roughly corresponds to the global steady-state security posture and enables the Army to generally achieve operational cycle rotations (Department of the Army [29]). Most deployment polices are generally tailored to steady-state parameters unless specified otherwise.

The Army Force Generation regulation, Department of the Army[29], also describes a *surge* rotation as a scenario where demand exceeds forces in the available force pool (at steady-state rotational rates). The Army responds through a surge of additional deploying units from an otherwise non-available state (training phase). A surge rotation is characterized by operational cycle rotation ratios shorter than described in the steady-state rotation and by reduced capabilities due to shortened preparation times before deployment.

As mentioned earlier, the Army's steady-state unit BOG:Dwell ratio goal is 1:3. In a 36-month cycle, active Army units will be available for deployment for 9 months (1 period) and be in Dwell for 27 months (3 periods). The Army's unit BOG:Dwell ratio goal during surge rotations is 1:2. In a 36-month cycle, active Army units will be in the available phase for 12 months (1 period) and be in Dwell for 24 months (2 periods) (Department of the Army[29]). Historically, and specifically in the period of this analysis (2003-2011), the Army has failed to consistently achieve these goals.

Competing with the above mentioned goal of reduced frequency is the desire to reduce deployment lengths for individual units. The Army's standard deployment during the period of this analysis was a 12-month deployment. However, during a period of surge (2007-2008), deployments were increased to 15 months. Beginning on 1 January 2012, Army policy redefined the standard deployment length to 9 months in an effort to reduce the impact of lengthy deployments on individ-

uals and families (Department of the Army[30]). While the feasibility was dictated by an observed and projected decrease in demand, we demonstrate in Section 4.4 that reduced deployment lengths impact the theoretical lower bounds for achievable unit BOG:Dwell ratios.

4.3 Scheduling Objectives and Constraints

Modeling methods for DSUA efficiency extend beyond just improving BOG:Dwell ratio goals. Analytical methods and heuristics can incorporate other desired objectives such as improved location consistency and reduced unit supply criteria. In order to improve deployment scheduling metrics, it is useful to identify the key data, constraints and objectives of the problem in a similar manner as modeling staff scheduling problems as presented by (Blöchliger[11]). The DSUA problem incorporates objectives similar to a general employee scheduling problem, such as minimizing the number of resources (employees versus units); however, it introduces some unique components in terms of variable deployment lengths and location-based operational readiness.

4.3.1 Deployment Scheduling Objectives

Army regulations and publications stipulate specific goals and policies regarding deployment scheduling metrics. Most publicly addressed are goals pertaining to length of deployments and BOG:Dwell ratios. However, location consistency is equally important in terms of readiness and enveloping operational and strategic objectives. Moreover, location consistency can be at direct odd with heuristics and algorithms that are intended to improve other scheduling objectives. The following objectives are currently identified as important metrics for evaluating a deployment schedule:

(1) Minimize the number of units required to source a deployment schedule,

(2) Minimize the number of deployed locations per unit,

(3) Minimize the length of deployments.

The first two objectives are directly targeted using optimization methods, while the third objective is manipulated as an input parameter to our approaches. Because the length of deployment varies in a very limited integer range, optimizing on the parameter is rather straight forward. For example, assuming a maximum range of 9-15 months, we can formulate each instance of the parameter for evaluation.

We can measure the merit of a schedule by one or more of these objectives. While we discuss each of these objectives in other sections, we emphasize the importance of the the first stated objective, which minimizes the size of the required sourcing pool. Minimizing the number of units required has significant structural, policy and budget implications. Clearly, demonstrating theoretical capabilities given a set of constraints is useful for comparison with historical data and in guiding future decisions about the size of a force required to meet potential demand scenarios.

Lastly, we are concerned about the ratio of deployment time to dwell time per unit. It is not included above as an optimization objective; however, it is critical that we measure and assess the performance of each approach in regards to BOG:Dwell ratio goals dictated by policy. Minimum BOG:Dwell ratios exist as a constraint for the unit assignments during souring, but we also assess each approach related to the average ratio for all units as described earlier for surge and steady-state scenarios.

4.3.2 Demand Constraints

Arguably, the most important constraint in any scheduling problem is the ability to meet demand. In the case of the deployment scheduling problem, demand is defined by mission requirements that are often difficult to predict. Previously, we discussed how deployment goals can change under steady-state versus surge demand scenarios. While steady-state scenarios are commonly used for strategic planning and theoretical limits, they are rarely observed in practice.

Figure 4.1 shows the actual deployment strength of battalion sized combat units during 86 months of Operation Iraqi Freedom. For analysis purposes, we use the historical deployment pattern as a proxy for demand and assume an excess in unit supply was not deployed. This demand

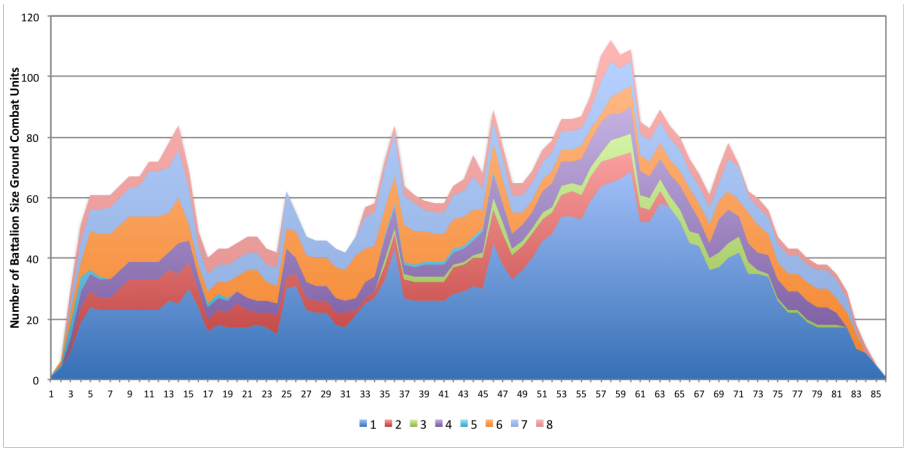


Figure 4.1: Monthly Unit Demand (Colored by Location)

figure is limited to infantry, armor, cavalry, field artillery and engineer units. The colors depict the separation in demand by province location. It is clear that during this period of persistent conflict, steady-state demand did not exist.

4.3.3 Deployment Constraints

The Army established a dwell versus deployment policy to compliment the previously mentioned ratio goals. Prior to the 2012 calendar year, individuals deployed for a period of twelve months or more were required to subsequently dwell for at least one month per month deployed (Department of the Army[30]). In other words, the upper bound BOG:Dwell ratio is 1:1. For the purposes of developing methods to compare with historical data, a modeling assumption is made that units must dwell for a duration of time greater than or equal to the previous length of deployment.

4.4 Scheduling Management

Prior to the development and evaluation of methods to optimize the objectives in Section 4.3.1, it is useful to understand the limits which can be derived based on the relationship of the first two objectives when presented as variables with demand and units of supply. Section 4.4.1 presents a closed-form solution of the best average BOG:Dwell ratio for units given a set of fixed parameters in steady-state rotation. Section 4.4.2 presents a lemma to identify the unique combination of group sizes that maximize BOG:Dwell when units are rotated within groups rather than mixed independently.

4.4.1 Closed-Form Constraints

One of the important metrics for evaluating deployment scheduling is the ratio of deployment time to dwell time for each unit. This ratio is referred to as the BOG:Dwell ratio, where BOG is an acronym for Boots on the Ground. A unit's BOG time is calculated at the time it arrives in a deployment theater until the time it departs. Mentioned previously, Army leadership frequently

references a goal of maintaining a BOG:Dwell ratio of less than $1:2$ during other than steady-state operations.

The BOG:Dwell ratio is a supply and demand issue. It is easy to prove that when demand is constant the ratio converges to a theoretical limit (Dabkowski et al.[24]). The following notation is used for the derivation of BOG:Dwell ratios and examining the limits:

N**:** number of rotational units

M**:** number of sustained units required to support deployment theater

x**:** duration of deployment ($x \in R+$)

n**:** duration of the overlap between a deployed unit and its replacement unit (same time units as x; $n \in R+, n < x$)

T**:** length of the analysis period (same time units as x; $T \in R+$, $T > x - n$)

D**:** total number of days

D^d**:** total number of deployed days

D^w**:** total number of dwell days

In a previous report in which we analyze the unit and individual BOG:Dwell rates during periods of steady-state demand, Dabkowski et al.[24] construct the following theorem and proof to show the theoretical limits:

Theorem 1. *As T tends to infinity, if $\frac{N}{M} > 1 + \frac{n}{x-n}$, the unit BOG:Dwell ratio is given by the relation:*

$$r(T) = 1 : \frac{N(x-n)}{Mx} - 1 \tag{1}$$

Proof. (by construction). Without loss of generality, we start by noting that the left-hand side of the Army's average AC BCT BOG:Dwell ratio during a T-day analysis period is simply the aggregate number of BOG days during T. With a requirement for M deployed BCTs throughout T, the left-hand side is minimally M T . Additionally, we note that in order for a BCT to be replaced at the termination of its x-day rotation, it must complete an n-day Relief in Place-Transition of Authority (RIP-TOA) cycle. Assuming that every RIP-TOA cycle begins n-days prior to the end of a deployed BCT's rotation, there are minimally $\left\lfloor \frac{T}{x-n} \right\rfloor$ overlaps for each of the $i = 1, 2, ..., M$ BCTs required during T. Finally, based on the Army's AC BCT deployment history prior to the beginning of the analysis period, it is possible that a full or partial RIP-TOA cycle terminates for an incoming BCT at some time t_i, where $t_i \in (0, xn)$, $i = 1, 2, ..., M$. We denote this potential overlap time as β_i, where $0 < \beta_i \leq n$. Accordingly, as we have a requirement for M units, it follows that the total number of overlap days is $Mn \left\lfloor \frac{T}{x-n} \right\rfloor + \sum_{i=1}^{M} \beta_i$. Aggregating the above, the left-hand side of the BOG:Dwell ratio is:

$$D^d = MT + Mn \left\lfloor \frac{T}{x-n} \right\rfloor + \sum_{i=1}^{M} \beta_i \tag{2}$$

Substituting $\left\lfloor \frac{T}{x-n} \right\rfloor$ with $\frac{T}{x-n} - \alpha$, where $\alpha \in [0, 1)$ and represents the decimal portion of $\frac{T}{x-n}$, yields:

$$D^d = MT + Mn \left(\frac{T}{x-n} - \alpha \right) + \sum_{i=1}^{M} \beta_i \tag{3}$$

Since the right-hand side of the ratio in (1) is simply the aggregate number of dwell days during T, we can derive the aggregate number of dwell days as any number of days not deployed. The total number of dwell days is given by:

$$D^w = D - D^d = NT - MT - Mn \left(\frac{T}{x-n} - \alpha \right) - \sum_{i=1}^{M} \beta_i \tag{4}$$

Placing (3) and (4) into the BOG:Dwell ratio, treat- ing x, n, N, M, α, and β_i as constants, and collecting terms with respect to T we obtain:

$$r_q(T) = \frac{T\left(M_q + \frac{Mn}{x-n}\right) - Mn\alpha + \sum_{i=1}^{M}\beta_i}{T\left(N - M - \frac{Mn}{x-n}\right) + Mn\alpha - \sum_{i=1}^{M}\beta_i} \tag{5}$$

Taking the limit of $g(T)$ as $T \to \infty$ it is obvious that $M + \frac{Mn}{(x-n)} > 0 \ \forall M, n$, and x, which implies the limit of the numerator as $T \to \infty$ is ∞. Similarly, in the denominator $N - M - {}^{Mn}/_{(x\,-\,n)} > 0$ when $\frac{N}{M} > 1 + \frac{n}{x-n}$. Assuming this relation holds, the limit of the denominator as $T \to \infty$ is ∞. With this in mind, we have the indeterminate form ${}^{\infty}/_{\infty}$.

Applying L'Hôpital's rule, we take the derivative of the numerator and the denominator with respect to T, which yields:

$$\lim_{T\to\infty} r(T) = \frac{M + \frac{Mn}{x-n}}{N - M - \frac{Mn}{x-n}} \tag{6}$$

Multiplying the numerator of (6) by $\left(M + {}^{Mn}/_{(x\,-\,n)}\right)^{-1}$ results in

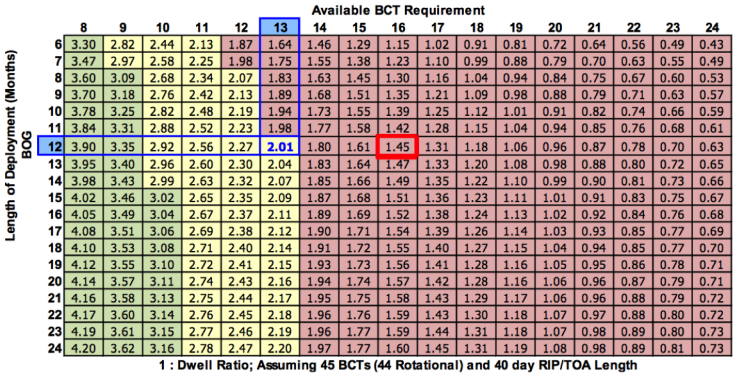
$$\lim_{T\to\infty} r(T) = \frac{1}{\frac{N}{M+\frac{Mn}{x-n}} - 1} \tag{7}$$

Finally, equation (7) can be simplified by collecting terms with respect to M.

$$\lim_{T\to\infty} r(T) = \frac{1}{\frac{N(x-n)}{Mx} - 1} \tag{8}$$

□ □

We can demonstrate an application of the limiting theorem presented above. Consider a scenario in which we have 44 units available in supply for rotation and a mandatory overlap of 40 days among rotating units. Figure 4.2 presents the results of the theorem when parameters of demand and length of deployment are adjusted. For example, as highlighted, the only way that a ratio of 1:2 or better can theoretically be achieved with 12 month deployments is if demand is less than or equal to 13 units in steady-state. Highlighted in red, the figure shows the best theoretical right-hand side (RHS) of a BOG:Dwell ratio (1:RHS) given the existing deployment-length policy and "Available" demand planning requirement in 2009.



Available BCT Requirement

Length of Deployment (Months) BOG	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
6	3.30	2.82	2.44	2.13	1.87	1.64	1.46	1.29	1.15	1.02	0.91	0.81	0.72	0.64	0.56	0.49	0.43
7	3.47	2.97	2.58	2.25	1.98	1.75	1.55	1.38	1.23	1.10	0.99	0.88	0.79	0.70	0.63	0.55	0.49
8	3.60	3.09	2.68	2.34	2.07	1.83	1.63	1.45	1.30	1.16	1.04	0.94	0.84	0.75	0.67	0.60	0.53
9	3.70	3.18	2.76	2.42	2.13	1.89	1.68	1.51	1.35	1.21	1.09	0.98	0.88	0.79	0.71	0.63	0.57
10	3.78	3.25	2.82	2.48	2.19	1.94	1.73	1.55	1.39	1.25	1.12	1.01	0.91	0.82	0.74	0.66	0.59
11	3.84	3.31	2.88	2.52	2.23	1.98	1.77	1.58	1.42	1.28	1.15	1.04	0.94	0.85	0.76	0.68	0.61
12	3.90	3.35	2.92	2.56	2.27	**2.01**	1.80	1.61	1.45	1.31	1.18	1.06	0.96	0.87	0.78	0.70	0.63
13	3.95	3.40	2.96	2.60	2.30	2.04	1.83	1.64	1.47	1.33	1.20	1.08	0.98	0.88	0.80	0.72	0.65
14	3.98	3.43	2.99	2.63	2.32	2.07	1.85	1.66	1.49	1.35	1.22	1.10	0.99	0.90	0.81	0.73	0.66
15	4.02	3.46	3.02	2.65	2.35	2.09	1.87	1.68	1.51	1.36	1.23	1.11	1.01	0.91	0.83	0.75	0.67
16	4.05	3.49	3.04	2.67	2.37	2.11	1.89	1.69	1.52	1.38	1.24	1.13	1.02	0.92	0.84	0.76	0.68
17	4.08	3.51	3.06	2.69	2.38	2.12	1.90	1.71	1.54	1.39	1.26	1.14	1.03	0.93	0.85	0.77	0.69
18	4.10	3.53	3.08	2.71	2.40	2.14	1.91	1.72	1.55	1.40	1.27	1.15	1.04	0.94	0.85	0.77	0.70
19	4.12	3.55	3.10	2.72	2.41	2.15	1.93	1.73	1.56	1.41	1.28	1.16	1.05	0.95	0.86	0.78	0.71
20	4.14	3.57	3.11	2.74	2.43	2.16	1.94	1.74	1.57	1.42	1.28	1.16	1.06	0.96	0.87	0.79	0.71
21	4.16	3.58	3.13	2.75	2.44	2.17	1.95	1.75	1.58	1.43	1.29	1.17	1.06	0.96	0.88	0.79	0.72
22	4.17	3.60	3.14	2.76	2.45	2.18	1.96	1.76	1.59	1.43	1.30	1.18	1.07	0.97	0.88	0.80	0.72
23	4.19	3.61	3.15	2.77	2.46	2.19	1.96	1.77	1.59	1.44	1.31	1.18	1.07	0.98	0.89	0.80	0.73
24	4.20	3.62	3.16	2.78	2.47	2.20	1.97	1.77	1.60	1.45	1.31	1.19	1.08	0.98	0.89	0.81	0.73

1 : Dwell Ratio; Assuming 45 BCTs (44 Rotational) and 40 day RIP/TOA Length

Figure 4.2: BOG:Dwell Ratios with fixed unit supply and overlap

4.4.2 Unit Grouping Lemma

Another way to address deployment rotations is to deploy units as groups. More simply, a unit would belong to a group of units that deploys in a rotational cycle such that the units ensure consistency in the same locations. Another closed-form best case solution for rotational frequency can be obtained if units are grouped together in this manner. This rotational pattern can not achieve a better theoretical limit than Theorem 1, but it does allow us to increase operational predictability and address location consistency. The following Lemma is constructed and proven in the document by Dabkowski et al.[24]:

Lemma 4.4.1. ***Unit-Grouping Lemma.*** *Given N rotational units and a sustained requirement of M units such that $M > 0$, there exists a unique combination of $(M-s)s$ and $r(s+1)$ groups of units that fulfill the requirement, where $r = N - M\lfloor N/M \rfloor$ and $s = \lfloor N/M \rfloor$.*

Proof. (by construction). In order to prove this lemma, we introduce the Division Algorithm, an important theorem from number theory. Succinctly, the Division Algorithm states: "If a and b are integers such that $b > 0$, then there are unique integers r and s such that $a = sb + r$ with $0r < b$"(Rosen, 2000). Using this result, we substitute N and M for a and b respectively yielding: $N = sM + r$. Adding $(sr - sr)$ to the right-hand side gives $N = sM + r + sr - sr$, and we rearrange terms to obtain $N = s(M - r) + (s + 1)r$. At this point we have respectively added zero

to right-hand side, thus preserving the existence and uniqueness of the integers r and s. Moreover, as a matter of interpretation, this equation implies that our N units can always be distributed into a unique combination of $(Mr)s$ and $r(s+1)$ groups. Finally, we note that the values for r and s are explicitly given by the expressions $r = a - b\lfloor a/b \rfloor$ and $s = \lfloor a/b \rfloor$ which equates to $r = N - M\lfloor N/M \rfloor$ and $q = \lfloor N/M \rfloor$ following substitution (Rosen, 2000). □

4.4.3 Location Consistency

During persistent conflict, maintaining a certain level of consistency for units deploying to a location has obvious tactical and operational advantages. Familiarity with a location provides a benefit to operational readiness. As stated by Reed[79], a unit returning to the same location (with sufficient continuity in the organization) has a more positive effect in maintaining continuity of operations than a new unit or one with high turnover. Moreover, if units return to a location where they have previously served and they retain some of their leaders, they have the knowledge necessary to anticipate the second-order and third-order effects of their actions and are less likely to derail efforts of a previous unit.

It is important to include location consistency with the DSUA problem because a goal of location consistency can be at odds with other objectives. Section 4.4.2 showed that maintaining unit grouping reduces the theoretical limitations of BOG:Dwell ratios. Location consistency also has the effect of increasing the lower bound of an objective function that minimizes the number of units used for scheduling.

Deployment scheduling logistics articles reference parameters for general theater locations during simultaneous conflicts but in general do not address location scheduling parameters within a theater of operations. Location consistency is generally associated as a readiness issue; its impact to scheduling logistics is primarily isolated to the high-level unit scheduling problem addressed here.

4.5 Scheduling Optimization

The DSUA problem can be set up as an integer programming problem similar to those used in employee scheduling (Glover and McMillan[44]) and staff scheduling problems (Blöchliger[11]). While it is useful to demonstrate the ability to present the deployment scheduling problem in terms of integer programming properties, later we show that specific characteristics of this problem lend it to be solved more efficiently with graph coloring approaches. Initially, we define the following scheduling parameters for the binary integer program:

q: Unit identifier,

L: Number of deployment locations (by Province),

w: Minimum dwell time policy ($w \in \mathbb{Z}+$),

d: Length of deployment in months ($d \in \mathbb{Z}+$),

T: Length of deployment horizon ($T \in \mathbb{Z}+, T > d$),

r_{lm}: Required number of units in location l in month m ($r \in \mathbb{Z}+$).

The binary variables of this formulation are as follows:

$$x_{qlm} = \begin{cases} 1 & : \text{if unit } q \text{ is assigned to location } l \text{ in month } m, \\ 0 & : \text{otherwise,} \end{cases}$$

$$y_{qlm} = \begin{cases} 1 & : \text{if unit } q \text{ starts deployment in location } l \text{ in month } m, \\ 0 & : \text{otherwise,} \end{cases}$$

$$v_{ql} = \begin{cases} 1 & : \text{if unit } q \text{ is deployed to location } l \text{ at any time,} \\ 0 & : \text{otherwise,} \end{cases}$$

$$z_q = \begin{cases} 1 & : \text{if unit } q \text{ is ever scheduled for deployment,} \\ 0 & : \text{otherwise.} \end{cases}$$

The variable x_{qlm} is used to track the utilization of a particular unit at a given time and location. The variable y_{qlm} identifies the starting point of each deployment by time and location. We use v_{ql} to track the number of deployments by unit and location and incorporate z_q to identify if a unit is ever scheduled for deployment.

4.5.1 Optimization Objectives

The deployment scheduling problem can be optimized on multiple objectives, particularly with regard to the number of units required to source the schedule and the number of deployed locations per unit. As mentioned previously with scheduling objective, we treat deployment length, d, as an input parameter rather than an objective due to its limited integer range.[1]

Suppose we have a sourcing pool and want to minimize the number of units used to source the the demand. An optimization objective can be applied to the number of units used in the schedule, which minimizes $\sum_q z_q$. While this objective function can be deemed inequitable for an existing sourcing pool, a unit utilization objective allows us to determine a lower bound for structure requirements, given an expected demand.

Two of the metrics that are useful in measuring readiness and stress on the force are location consistency and BOG:Dwell ratios. The number of locations per unit can be minimized by optimizing on the objective function, $\sum_q \sum_l v_{ql}$, which intuitively works against an objective to minimize units. Creating an objective function that minimizes BOG:Dwell ratios causes complications for an integer programming approach. Assume we can substitute the average deployed time per unit for the average BOG:Dwell ratio since we are optimizing in a finite time horizon. An objective

[1] Unit deployments in persistent conflict range from 9 to 15 months in recent history. It is reasonable to assume a 9-month lower bound since shorter deployments continue to worsen BOG:Dwell ratios and can cause logistics and operations issues that make solutions infeasible in other domains.

function minimizing the average time deployed

$$\frac{\sum_{q}\left(\sum_{l}\sum_{m} y_{qlm}d\right)}{\sum q}$$

poses nonlinear issues that make it difficult to solve.

4.5.2 Constraints

The DSUA problem seeks to optimize single or multiple objectives while meeting mission demands at multiple locations over a finite time horizon. For the purposes of this research, time periods are measured in months. The scheduling optimization must meet minimum demand requirements while adhering to deployments policies, specifically pertaining to the length of deployments and minimum dwell time for units. The following constraints are created to depict deployment demand and unit assignment policies:

$$\sum_{q} x_{qlm} \geq r_{lm} \qquad \forall\, l, m, \tag{4.1}$$

$$\sum_{t=\max\{1,m-d+1\}}^{m} y_{qlt} = x_{qlm} \qquad \forall\, q, l, m, \tag{4.2}$$

$$\sum_{l=1}^{L}\sum_{t=m}^{m+2d} y_{qlt} \leq 1 \qquad \forall\, q, m, \tag{4.3}$$

$$z_q \geq x_{qlm} \qquad \forall\, q, l, m, \tag{4.4}$$

$$\sum_{m} y_{qlm} \leq T v_{ql} \qquad \forall\, q, l, m, \tag{4.5}$$

$$x_{qlm}, y_{qlm}, z_q \in \{0,1\} \qquad \forall\, q, l, m.$$

Constraint (4.1) ensures that demand is met during each time period at each location. Constraints (4.2) and (4.3) respectively align monthly utilization after deployment start dates and establish minimum dwell times between deployment start dates for a unit. Constraint (4.4) essentially turns on the binary z_q variable when a unit deploys in any time period or location. Lastly, constraint (4.5) activates v_{ql} if unit q is deployed to location l at least once.

The scheduling problem formulated as an integer programming problem is difficult to solve due to the size and scale. We use an historical data set from Iraq as a basis for the variable indices. Excluding locations with less than five deployments, there are 8 locations and 86 months in the deployment scenario being modeled. If demand is consistent with the historical use of over 200 different units, the problem requires more than 275,000 binary x_{qlm} and y_{qlm} variables combined. Moreover, the number of row constraints associated with (4.2) and (4.4) exceed 275,000 and constraints (4.1) and (4.5) have 688 and 1,600 rows respectively. Overall, there are more than 275,000 variables and 300,000 constraints.

The optimization problem is complicated with variable deployment lengths. However, Army deployment policies generally dictate fixed deployment lengths. Deployment length may change at different points in the time horizon but remain consistent among units. Therefore, it is reasonable to assume that deployment lengths cannot vary for each deployment during this analysis. This assumption is important because it lends the deployment scheduling problem to other solution methods, as shown below.

4.6 Schedule Coloring Problem

When deployment lengths are fixed, the DSUA problem lends itself to be solved more efficiently as a graph coloring problem. Graph coloring is a method that has useful application in many types of complex problems that involve optimization. As discussed earlier, the Army has policies which standardize the length of deployments. While it is possible for two deployed units to have different deployment lengths, it is a reasonable assumption that deployment lengths are the same for all units during a specified time window. Because this assumption, we can use graph coloring techniques to assign a minimal number of units to deployment schedules. Moreover, we will demonstrate how graph coloring methods can be modified to improve specific objective measures that are unique to the DSUA problem.

Consider a very small-scale example of a deployment schedule in which there is a demand for units in three different locations, $l := 1, 2, 3$ over a time horizon of 10 months. Suppose the demand

Location	Month									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	1	2	2	0	0	0	1	1	1	1
2	0	1	1	1	1	0	0	0	1	1
3	0	0	0	1	1	2	2	2	0	0

Table 4.1: Example Problem: Demand by location and month

for each location and time period, r_{lm}, is given as the values in Table 4.1. We then create a simple algorithm which builds a feasible deployment schedule by assigning deployments sequentially by location to meet each demand one month at a time. Hence, if viewed in a matrix format, the schedule is constructed and more importantly labeled by row then column.

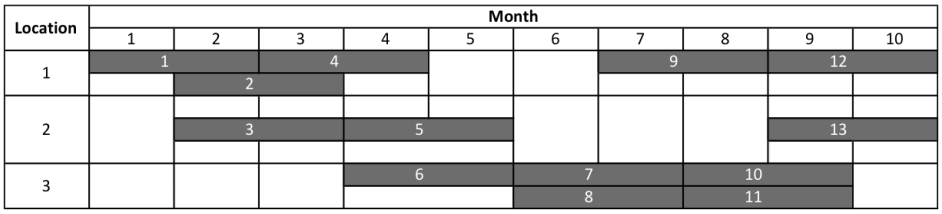


Figure 4.3: Feasible deployment schedule without assignment

Using the demand from Table 4.1, a simple deployment schedule is resented in Figure 4.3 given a standard deployment length of two time periods. The importance of numbering the deployments in a left to right time sequence proves to be critical to the performance of particular graph coloring algorithms. In terms of graph coloring notation, the numbered deployments are the same as graph vertices.

The basic objective of graph coring problems is to assign colors to a set of vertices in a graph such that all vertices are colored with a minimum number of different colors. A constraint that limits two vertices from being colored with the same color is depicted by a connecting edge between the two vertices.

Once again, we refer to the example problem where there are 13 vertices that must be colored. We apply a simple constraint in which two vertices cannot be colored the same unless they are

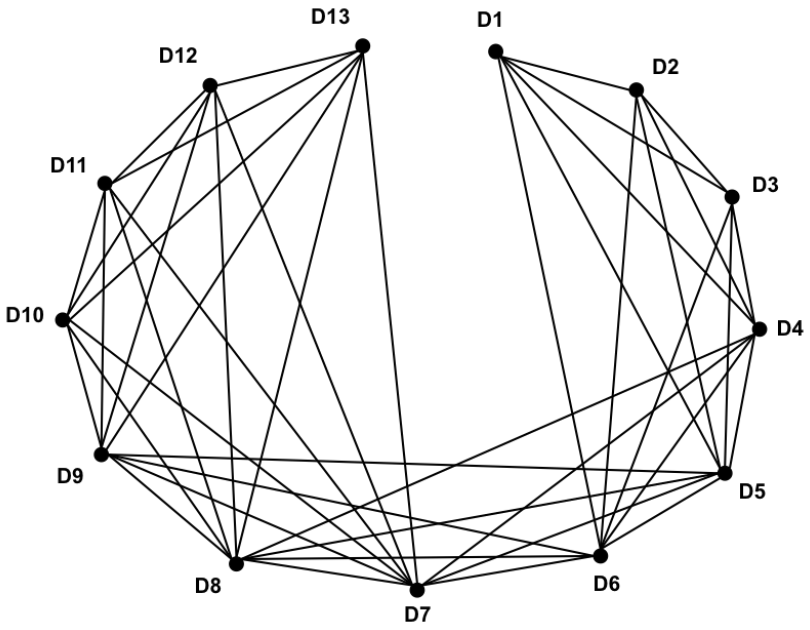


Figure 4.4: Graph representation of a deployment schedule

separated by at least two time periods. In deployment terms, this constraint enforces a policy that, if a unit deploys, it cannot be assigned another deployment until is has dwelled for the same amount of time as the previous deployment length. An edge between two vertices represents deployments that are separated by less than two time periods. Figure 4.4 shows the graph corresponding to the deployments in Figure 4.3. The remainder of this section uses this example problem as a basis for describing graph coloring algorithms that are most applicable to the DSUA problem.

4.6.1 Definitions and Notations

Given a graph $G = (V, E)$ with vertex set V and edge set E, and given an integer k, a k-coloring of G is a function $c : V \to \{1, \cdots, k\}$. The value $c(v)$ of a vertex v is called the color of v. The goal of the standard graph coloring problem is to find an assignment of colors to the vertices of G that minimizes the number of colors used such that no two adjacent vertices receive the same color. Colors are defined as positive integers and the number of colors in an optimal coloring solution is called the chromatic number of the graph, denoted by $\chi(G)$. Finding $\chi(G)$ in general graphs is known to be one of the most difficult optimization problems on graphs (Palubeckis[75]).

4.6.2 Coloring Algorithms for Scheduling

Recursive Largest First (RLF) is one of the best known coloring algorithms. Developed by Leighton[63], RLF has been show to be suitable for large-scale practical problems. The RLF algorithm is described in Appendix I. First-Fit can be considered a special version of RLF that produces optimal solutions for the problem of minimizing the number of colors in an interval graph. We first review how First-Fit operates and then describe how we apply it to the DSUA problem.

4.6.2.1 First-Fit Algorithm

Let U_1 be the set of uncolored vertices that are not adjacent to any of the vertices colored with color c and U_2 be the set of uncolored vertices adjacent to a least one of the vertices colored with c. Given G from above, the First-Fit Algorithm chooses the first vertex, v, in U_1 and colors it

with the current color c. Upon updating U_1 and U_2, we again choose the first v in U_1. The process continues until U_1 is empty and then is repeated for new colors until all vertices in G are colored.

The First-Fit algorithm produces heuristic solutions for general graphs. However, with a specific structuring of the vertices being colored, the First-Fit algorithm produces optimal solutions to the problem of minimizing the number of colors. Specifically, this occurs in problems possessing the same properties of a interval graph. If C is defined as the set of vertices that have already been colored and V' is a subset of uncolored vertices in V, the First-Fit algorithm can be depicted as in Algorithm 1.

```
Set C = ∅, U1 := V, U2 = ∅, c := 0;
while |C| < |V| do
    if U1 ≠ ∅ then
        Identify first v (lowest integer) in the subgraph of U1;
        Increment c by 1;
        Assign color c to v;
        Move v from U1 to C;
        Move all v' ∈ U1 that are adjacent to v from U1 to U2;
    else
        Increment c by 1;
        Set U1 := U2, U2 := ∅;
    end
end
```

Algorithm 1: First-Fit Algorithm

Kierstead[61] proposed that a graph G is an interval graph if its vertex set can be represented by a collection of closed intervals V of the real numbers so that for all $i, j \in V$, $i \cap j \neq \varnothing$ if and only if the vertex represented by i is adjacent to the vertex represented by j such that the V intervals are a representation of G.[2] Kierstead describes interval graphs as perfect in that the chromatic number $\chi(G)$ of an interval graph G is equal to its maximum clique size $\omega(G)$.

Because of the known interval graph structure of the DSUA problem, the First-Fit algorithm is chosen as the basis for solving the variants of the problem that we consider. Further validation of the DSUA problem with the First-Fit algorithm in terms of interval graph theory is described

[2] Intervals in the graph may have the same endpoints.

in Section 4.6.3.1. Figure 4.5 shows the coloring solution for the schedule in Figure 4.3 using the First-Fist algorithm to minimize the number of colors used. First-Fit provides an optimal solution of 7 colors; however, it is important to note that it is not the only optimal solution.

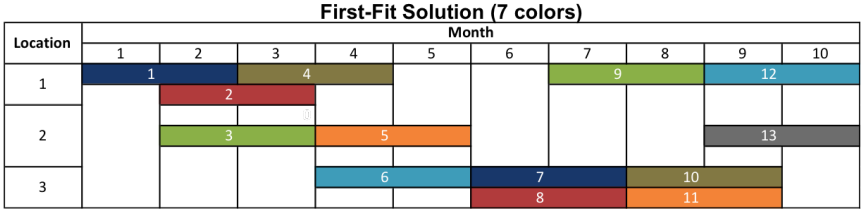


Figure 4.5: Minimize the number of units (colors)

4.6.2.2 Color Swapping

As mentioned above, First-Fit optimizes objective 1, that is, if minimizing the number of units required to source a deployment schedule. However, it does not produce optimal solutions when considering the minimization of deployed locations per unit. Therefore, we search for solutions that minimize our second objective by swapping colors in colored graphs. So, color swapping is applied as a post-processing local search to a colored graph. Swaps are made within a solution of fixed $|C|$ colors and a new solution is evaluated based on a metric other than the number of colors used. In the case of the deployment scheduling problem, we use a local search to improve upon the metric of number of deployed locations per unit (color).

Consider a solution produced by the First-Fit algorithm. Each deployment is represented by a vertex, which is assigned a color. A location index is also associated with each vertex. The local search iterates with each vertex, v, first identifying the color, $c(v)$ and location $l(v)$ indices. With each iteration, a search is conducted to identify another vertex, v' with the same color assignment that is in a different location; therefore, $c(v) = c(v')$ and $l(v) \neq l(v')$. Another neighborhood search is conducted to find a vertex, v'', that is assigned a different color but in the same location as v; therefore, $c(v) \neq c(v'')$ and $l(v) = l(v'')$. If v' and v'' are not connected by an edge and if v and

v'' are also not connected by an edge, the colors assigned to v' and v'' are swapped. Using our notation:

$$\text{if } (v', v'') \notin E \text{ and } (v', v'') \notin E$$

$$\text{then } c(v') \leftrightarrow c(v'').$$

Note that after the swap, v and v'' have the same color. Since these deployments occur in the lame location, the swap has assigned the unit to be deployed to a single location instead of two different locations.

The process is repeated for every vertex v until no swaps are possible. The new solution is then subject to the swapping heuristic beginning at the first deployment vertex. If the heuristic passes through a full iteration of all vertices without making any swaps , the heuristic is terminated and the current solution is returned as the best solution after swapping. Consider a solution in which C represents an initial assignment and n is the number of deployments, then the color swapping algorithm is shown in Algorithm 2.

```
Given C;
while Terminate = False do
    Terminate ← True;
    for v = 1, ..., n do
        for v' ≠ v do
            if c(v) ≠ c(v') and l(v) ≠ l(v') then
                for v'' ≠ v and v'' ≠ v' do
                    if c(v) ≠ c(v'') and l(v) ≠ l(v'') then
                        c(v') ↔ c(v'');
                        Terminate ← False;
                    end
                end
            end
        end
    end
end
```

Algorithm 2: Color Swapping Heuristic

4.6.2.3 First-Fit by Location (FFL) Algorithm

In the previous section, the First-Fit algorithm is followed by a color swapping procedure. Sequencing the First-Fit and color swapping steps is a form of lexicographic multi-objective optimization. Specifically, with the lexicographic method, the objective functions are sequenced in order of importance and the optimization problems are solved one at a time (Marler[66]).

The First Fit by Location (FFL) algorithm is a modification of the First-Fit algorithm and seeks to minimize both objectives at the same time: the number of colors used to color the vertices of G, and the number of locations assigned to each unit. In each color iteration, the first uncolored vertex, v is colored with the next color, c. Subsequently, rather than choosing the first vertex in the U_1 set, we choose the first vertex in U_1 that has the same location as v and color it with c. In the case where U_1 is not empty but does not contain a vertex with the same location as v, the first vertex in U_1 is colored c. The U_1 and U_2 sets are updated after each coloring and the process is repeated until U_1 is empty and the next iteration begins with the next color, (i.e. $c = c + 1$).

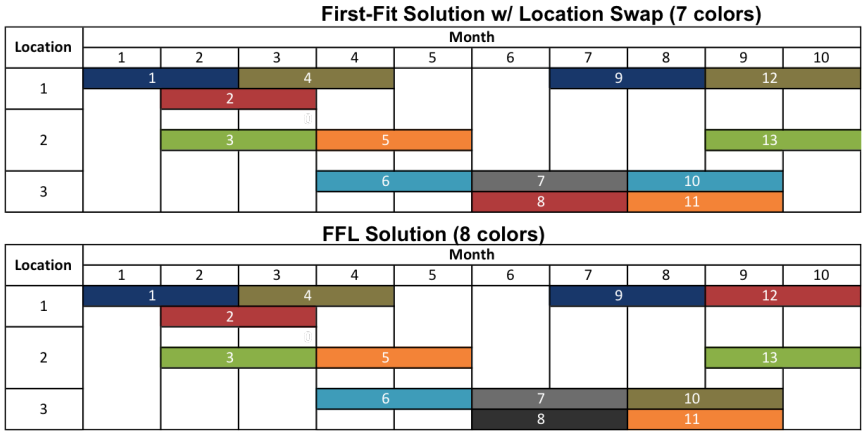


Figure 4.6: Minimize the number of units (colors) and locations

Figure 4.6 demonstrates the different coloring solutions for the uncolored schedule in Figure 4.3 using First-Fit with swapping and FFL algorithms. Despite the small scale of this example (13 vertices), the two algorithms clearly produce different solutions, as First-Fit with swapping addresses the objectives lexicographically, while the FFL seeks to improve objectives simultaneously.

FFL can also be combined with the swapping heuristic to seek an improved solution on the location objective. A FFL swapping solution is not presented in Figure 4.6 since the scale is too small for swapping to produce an improvement.

4.6.3 Algorithm Comparisons

In this section, we discuss the theoretical characteristics of the above-mentioned algorithms and demonstrate their performance with the initial toy problem before comparing results in the computational experiments in Section 4.7. We begin with the observation that the DSUA problem falls into a subset of interval graph problems. Accordingly, the First-Fit algorithm provides an optimal solution for the unit minimization problem. Extending the theoretical implications of First-Fit, it is clear that the FFL algorithm produces a solution which is always equal or better than the First-Fit with swapping solution when measured against the additional location objective. However, FFL demonstrates an ability to meet or exceed the performance of First-Fit when measuring the performance of average BOG:Dwell ratios prior to any swapping heuristics.

4.6.3.1 Interval Graph Theory

As previously mentioned, graphing problems often have multiple optimal solutions. With an objective of minimizing colors, other algorithms such as RLF and metaheuristics adaptations such as Greedy Randomized Adaptive Search Procedure (GRASP) are proven to provide optimal solutions as well. However, the First-Fit algorithm is the least computationally intensive. Given that we show the deployment scheduling problem meets the characteristics of an interval graph, other methods can be omitted from our analysis due to excessive computation.

The deployment schedule, as represented earlier in Figure 4.3, has the characteristics of an interval graph. Using time periods as an index, the vertices in the graph are structured with a left end point ordering. Moreover, the graph can be described as a collection of closed intervals ordered in a relation such that $i < j$ iff the right end point of i is less than the left end point of j (Kierstead and Qin[62]). As such, the First-Fit provides an optimal solution equal to the maximum clique

size.

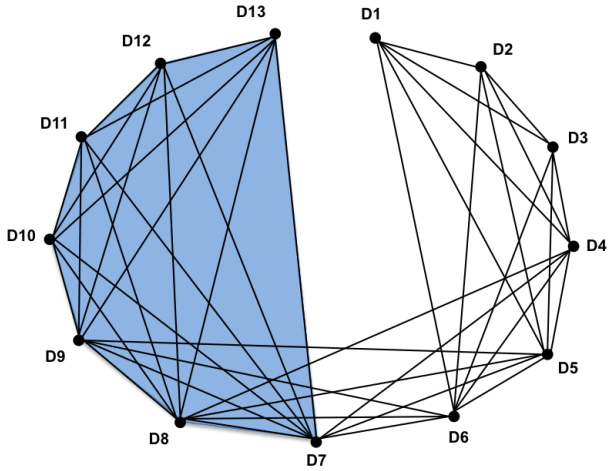


Figure 4.7: Optimal coloring of G equals maximum clique size ($\omega(G) = 7$)

Returning to the example problem, the maximum clique size of the the graph is simple to calculate. Graphically it is represented in Figure 4.7 as the set of vertices that are shaded in blue. The maximum clique size, $\omega(G) = 7$, is equal to the optimal solution when minimizing colors with First-Fit.

4.6.3.2 Performance on Objectives

This section shows how the different algorithms perform against the measured objectives, prior to analysis with large scale and historical data sets in the following section. Using the example problem we replace the terms vertices and colors for deployments (deploy) and units respectively. Although the swapping heuristic is an extension to the First-Fit algorithm, First-Fit and First-Fit Swapping are compared as separate algorithms since the first has a single objective and the latter deals with two objectives lexicographically. First-Fit by Location is the third comparison.

Naturally, the performance of these algorithms is evaluated like all graph coloring problems in terms of minimizing the number of units (colors) used to meet deployment (vertex coloring) requirements. Performance is compared by the average number of locations assigned per unit

during the deployment horizon. It is also worth noting the maximum number of locations for any unit, which could indicate a skewed distribution among units. The algorithms are also evaluated based on BOG:Dwell ratio statics, most importantly the average unit ratio. BOG:Dwell ratios are only measured when a unit has more than one deployment. The BOG:Dwell statistic, as shown in Table 4.2, is the right-hand side of the BOG:Dwell ratio with the left-hand side equal to 1; therefore, in the format of 1:Dwell.

Method	Deploy	Edges	Units	Loc/Unit	Diff (%)	Avg Dwell	Diff (%)
First-Fit	13	44	7	1.8571	-	1.3333	-
First-Fit Swap	13	44	7	1.2857	-30.77	1.5833	+18.75
FFL	13	44	8	1.2500	-32.69	1.9000	+42.50

Table 4.2: Minimizing the number of units and locations

It is clear from Table 4.2 that even with a small example, FFL does not guarantee the same minimum unit optimality as First-Fit. First-Fit Swapping and FFL significantly improve the locations per unit statistic, but FFL intuitively dominates due to the availability of additional unit resources. The average dwell time is also improved by both algorithms as residual benefit from the location improvement. Focussing on FFL, it never produces a worse BOG:Dwell statistic than First-Fit. This is obvious because First-Fit colors vertices based on the parameter of interval size, which is essentially a lower bound. FFL can extend into the next interval to find a better location, thereby increasing dwell time. Generally, FFL performs better than First-Fit Swapping in average dwell time but is not guaranteed a better result. For the example shown in Table 4.2, the maximum dwell time is 1.5 using First-Fit and 2.5 using both location improving heuristics.

4.7 Computational Experiment

The computational experiment applies the above-mentioned algorithms to a historical data set that represents an 86-month demand scenario in Iraq from 2003 to 2010. Historical deployments are used as a proxy for demand in 8 separate provinces which separate demand by location. The

historical demand, depicted earlier in Figure 4.1, represents a selection of unit types and is not a steady-state scenario. In this section we compare the historical deployment schedule, which used 203 units, with schedule solutions produced by each of the graph coloring algorithms presented earlier.

In addition to comparing results with the historical scenario, we evaluate the performance and sensitivity of solutions when we adjust parameters such as deployment lengths and minimum dwell policies. Subsequently we present an analysis of the algorithms with different demand scenarios, to include steady-state demand. The analysis is concluded with a validation of First-Fit as an optimal procedure for the objective of minimizing units.

4.7.1 Historical Comparison

The historical deployment schedule actually assigned 460 individual deployments to satisfy the demand requirements. The deployments were fulfilled by 203 different operational units, which were all battalion size elements. During the time horizon of this analysis the standard length of deployments was 12 months; however, during a surge period, deployments lengths extended to 15 months. Assuming that the demand is known and deployments are fixed at 12/15 months depending on the policy at the time, we are able to construct a deployment scenario that meets the same demand using 435 deployment assignments. Using a maximum BOG:Dwell policy of 1:1, the schedule's corresponding graph has 52,359 edges.

The results in Table 4.3 indicate that demand could be met with a minimum of 192 units by adhering to a 1:1 BOG:Dwell ratio policy. Moreover, the number of locations per unit could be minimized using a sourcing plan of 196 units and the FFL heuristic with swapping. We also observe that when the swapping heuristic is applied to First-Fit, the average number of locations per unit is less than 0.02 greater than FFL Swap. The swapping heuristic appears to introduce some inconsistencies in the comparison of First-Fit Swap and FFL Swap as measured against average dwell, but the differences are relatively small.

In the lower half of the table, the minimum dwell policy is increased to twice the length of

deployment. The number of edges increase in the graph to 69,067 and the number of units required increases accordingly. Most importantly, in order to adhere to BOG:Dwell constraint of 1:2 for all units, the number of required units increases to 249 or 258 for First-Fit and FFL solutions respectively. Another less intuitive dynamic, which remains consistent in subsequent scenarios, relates to the locations per unit statistic. As the minimum dwell policy increases, the number of locations per unit can be further reduced.

4.7.2 Deployment Length and Minimum Dwell

This section examines the sensitivity of the coloring solutions to changes in the deployment length parameter and minimum dwell constraint. Unlike the solutions in Table 4.3, the deployment length is constant throughout the time horizon. Table 4.4 shows the number of units used to color the deployment schedules using the First-Fit and FFL algorithms. As the minimum RHS dwell constraint increases from 1 to 3, the number of units increases. However, the direction of change varies with increases to deployment lengths.

In Table 4.5, the location objective is also measured with changes to the deployment length parameter and dwell constraint. Results are shown post-swapping for First-Fit and FFL and compared to the base First-Fit results which only target the number of units objective. Both algorithms with post-swapping heuristics significantly outperform standard First-Fit. Moreover,

	Method	Deploy	Units	Time (sec)	Loc/Unit	Avg Dwell	Max Dwell
$w = 1{:}1$	First-Fit	435	192	17	1.7656	1.3770	1.7500
	First-Fit Swap	-	-	17	1.2343	1.4893	2.9167
	FFL	435	196	19	1.2245	1.4843	2.4167
	FFL Swap	-	-	17	1.2194	1.5485	3.1667
	Actual	460	203	-	1.8252	1.4166	3.0909
$w = 1{:}2$	First-Fit	435	249	27	1.5301	2.3109	2.6667
	First-Fit Swap	-	-	6	1.1606	2.3888	4.1667
	FFL	435	258	35	1.1395	2.4983	4.5000
	FFL Swap	-	-	3	1.0969	2.4826	4.4167

Table 4.3: Comparison of Algorithms to Historical Data

		Deployment Length (d = months)						
Method	w	9	10	11	12	13	14	15
First-Fit	1:1	206	210	204	207	209	213	192
FFL		209	218	213	212	214	216	192
First-Fit	1:2	292	297	283	287	283	274	249
FFL		296	305	306	295	285	284	267
First-Fit	1:3	369	375	355	351	343	338	341
FFL		387	393	378	356	343	339	341
Number of Deployments		693	642	590	507	468	436	406

Table 4.4: Sensitivity of deployment lengths

FFL Swap almost always produces a marginally better solution than First-Fit Swap and never does worse.

The unit and location objectives are clearly in conflict with each other. The average dwell statistic is not specifically targeted by either of the coloring algorithms or the color swapping heuristic. However, we are able to measure the performance of our solutions in terms of the average dwell (right-hand side of BOG:Dwell) statistic. Arguably, BOG:Dwell statistics are weighed with equal importance as unit and location statistics. Therefore, it is significant that our FFL and swapping solutions generally improve dwell statistics beyond First-Fit as a by-product of targeting other objectives.

Table 4.6 shows the results of First-Fit and FFL as well as post-swapping results from both. The result which provides the best solution for each combination of deployment length and minimum dwell time is highlighted in bold. Results from FFL before and after swapping are displayed since the swapping heuristic does not guarantee any improvement to average dwell. In fact, as the minimum dwell policy constraint is increased from 1:1 to 1:2, the swapping heuristic appears to shift from increasing to decreasing the average dwell time. The three latter solutions dominate First-Fit in all parameter and constraint scenarios until the maximum deployment length and dwell policy limit is reached.

The dwell statistic is significant because, as mentioned in Section 4.3, success is measured based on the average BOG:Dwell ratios of units rather than an enforced minimum dwell policy. For example, the minimum dwell policy could be a ratio of 1:1 yet the Army may have a goal of maintaining an average BOG:Dwell ratio of 1:2. Figure 4.8 demonstrates the relationship of our coloring methods, minimum dwell constraint and published BOG:Dwell ratio goals.

As the minimum dwell constraint is raised as depicted by the horizontal axis, the top half of Figure 4.8 shows the average dwell statistics after First-Fit and First-Fit Swap. The lower half of the figure shows the same trend for FFL and FFL Swap. The vertical grid lines identify the points at which solutions achieve BOG:Dwell goals of 1:2 and 1:3. Both algorithms indicate that the minimum dwell policy must be close to 1:1.6 in order to achieve an average ratio of 1:2 while

				First-Fit	First-Fit Swap		FFL Swap	
	d*	Vertices	Edges	Loc/Unit	Loc/Unit	Diff(%)	Loc/Unit	Diff(%)
Min Dwell = 1:1	9	693	90,706	2.3495	1.6068	-31.61	1.3493†	-42.57
	10	642	80,806	2.1762	1.4667	-32.60	1.2661†	-41.82
	11	590	74,246	1.9853	1.5098	-23.95	1.2207†	-38.51
	12	507	64,643	1.9034	1.2705	-33.25	1.2358†	-35.07
	13	468	59,949	1.8373	1.2440	-32.29	1.1542	-37.18
	14	436	55,297	1.7670	1.1690	-33.84	1.1343	-35.81
	15	406	49,098	1.7240	1.2448	-27.80	1.1876	-31.11
Min Dwell = 1:2	9	693	132,466	1.9178	1.2877	-32.86	1.2162†	-36.58
	10	642	122,454	1.7710	1.2660	-28.51	1.1377	-35.76
	11	590	112,071	1.6254	1.2367	-23.91	1.0686	-34.26
	12	507	88,439	1.5436	1.1533	-25.29	1.0814	-29.94
	13	468	80,439	1.4276	1.1201	-21.54	1.0912	-23.57
	14	436	74,045	1.4051	1.1533	-17.92	1.0845	-22.82
	15	406	66,159	1.4819	1.2329	-16.80	1.0899	-26.45
Min Dwell = 1:3	9	693	167,920	1.6043	1.1626	-27.53	1.0801	-32.68
	10	642	153,590	1.5253	1.1520	-24.47	1.0483	-31.27
	11	590	137,213	1.4282	1.1690	-18.15	1.0423	-27.02
	12	507	106,895	1.3020	1.1026	-15.31	1.0618	-18.45
	13	468	95,272	1.3120	1.0729	-18.22	1.0670	-18.67
	14	436	87,664	1.2426	1.0947	-11.90	1.0885	-12.40
	15	406	77,219	1.1554	1.0293	-10.91	1.0293	-10.91

* Deployment length in months

† No improvement to the FFL solution

Table 4.5: Locations/Unit for deployment lengths and minimum BOG:Dwell ratios

				First-Fit	First-Fit Swap		FFL		FFL Swap	
	d*	Vertices	Edges	Avg Dwell	Avg Dwell	Diff(%)	Avg Dwell	Diff(%)	Avg Dwell	Diff(%)
Min Dwell = 1:1	9	693	90,706	1.2046	1.2275	1.90	1.3179	9.41	1.3935	**15.68**
	10	642	80,806	1.1315	1.1376	0.54	1.2683	12.09	1.3055	**15.38**
	11	590	74,246	1.1965	1.1980	0.13	1.3667	14.22	1.3938	**16.49**
	12	507	64,643	1.3039	1.3478	3.37	1.4119	8.28	1.4361	**10.14**
	13	468	59,949	1.2511	1.3837	10.60	1.3543	8.25	1.3911	**11.19**
	14	436	55,297	1.2514	1.4038	**12.18**	1.3539	8.19	1.4001	11.88
	15	406	49,098	1.1579	1.2333	6.51	1.2636	9.13	1.2796	**10.51**
Min Dwell = 1:2	9	693	132,466	2.2860	2.3091	1.01	**2.3987**	4.93	**2.3987**	4.93
	10	642	122,454	2.1373	2.1893	2.43	2.3417	9.56	2.3436	**9.65**
	11	590	112,071	2.1872	2.2519	2.96	2.4531	**12.16**	2.4321	11.20
	12	507	88,439	2.2940	2.4348	6.14	2.4978	**8.88**	2.4949	8.76
	13	468	80,439	2.1978	2.3946	**8.95**	2.3930	8.88	2.3926	8.86
	14	436	74,045	2.1737	2.3034	5.97	2.4107	**10.90**	2.3882	9.87
	15	406	66,159	2.1520	2.1563	0.20	2.3453	**8.98**	2.3209	7.85
Min Dwell = 1:3	9	693	167,920	3.3099	3.4129	3.11	3.5285	**6.60**	3.5199	6.34
	10	642	153,590	3.1643	3.2700	3.34	3.4101	**7.77**	3.3685	6.45
	11	590	137,213	3.1799	3.2317	1.63	3.4717	**9.18**	3.4207	7.57
	12	507	106,895	3.2943	3.4161	3.70	3.4327	**4.20**	3.4238	3.93
	13	468	95,272	3.1458	3.2849	**4.42**	3.2732	4.05	3.2689	3.91
	14	436	87,664	3.2784	3.2456	-1.00	3.2931	**0.45**	3.2666	-0.36
	15	406	77,219	3.4185	3.3251	-2.73	3.4185	0.00	3.3262	-2.70

* Deployment length in months

Table 4.6: Average Dwell for deployment lengths and minimum BOG:Dwell ratios

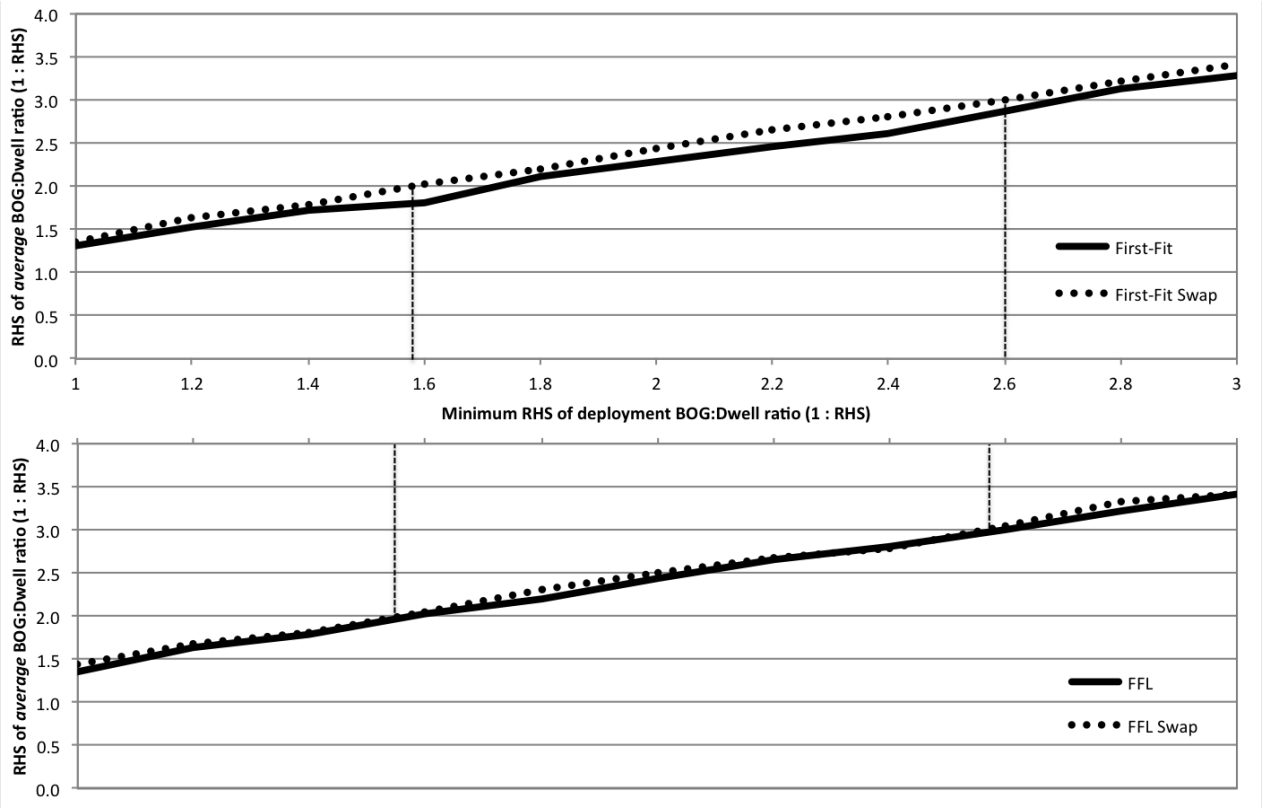


Figure 4.8: Average BOG:Dwell ratios with changes to the dwell policy constraint

the minimum dwell policy must be at or near 1:2.6 in order to achieve an average ratio of 1:3.

4.7.3 Demand Scenarios

The demand scenario which drives the parameters for demand by location and month is a critical component to this analysis. Clearly, different demand scenarios will produce different solutions. In this section we evaluate some different generated scenarios other than the historical scenario. We also acknowledge the use of steady-state demand scenarios, used for theoretical planning and the absence of reasonable demand forecasting mechanism, and demonstrate the expected consistency in results when coloring a steady-state deployment schedule.

4.7.3.1 Demand Excursion

Alternate demand scenarios are introduced in order to determine if the performance of the coloring algorithms and heuristics is attributable to the characteristics of the historical demand input. Recalling that the historical demand is not evenly distributed by location, our initial excursion implements a similar aggregate demand pattern in which demand is evenly distributed among the eight locations. Over time, the excursion demand is not steady-state and is depicted in Figure 4.9.

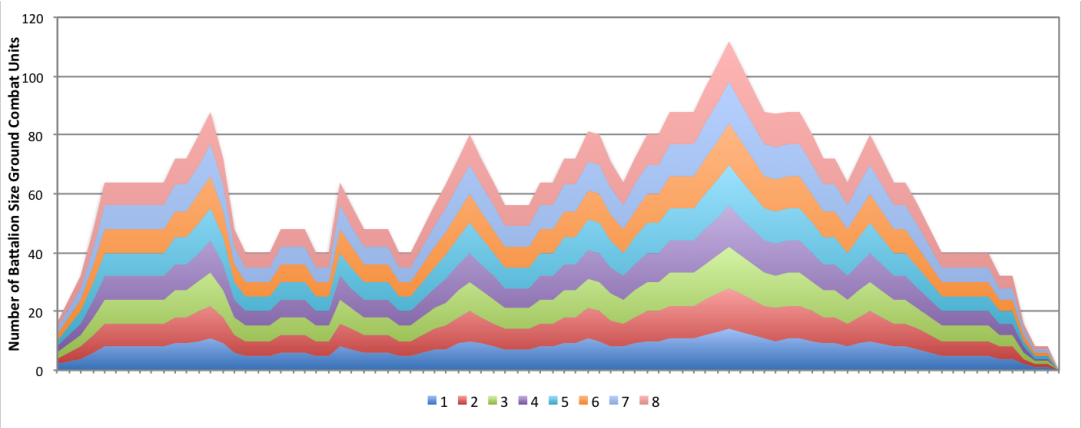


Figure 4.9: Demand excursion evenly distributed by location ($T = 86$)

With the results from the excursion, we are interested in determining if some of the early trends still hold, namely if FFL continue to produce better solutions in terms of location than First-Fit and if swapping heuristics improve the solution. Also of interest is any change to relationship between location improved solutions and dwell ratio measurements. Scheduling solutions are produced with each method while varying the deployment length parameter and minimum dwell constraint.

	Locations per Unit			Max Locations			Average Dwell			w
Method	9	12	15	9	12	15	9	12	15	
First-Fit	2.6373	2.6373	1.8296	5	4	3	1.2245	1.2926	1.1510	1:1
First-Fit Swap	1.1762	1.1762	1.0511	3	2	3	**1.2428**	**1.3137**	**1.1902**	
FFL	1.0518	1.0518	1	2	1	1	1.2238	1.2926	1.1510	
FFL Swap	1	1	1	1	1	1	1.2264	1.2926	1.1510	

Table 4.7: Demand Excursion with $d := \{9, 12, 15\}$ and $w = d$

Table 4.7 shows the results of using a BOG:Dwell constraint of 1:1 and deployment lengths of 9, 12 and 15 months. The number of units required is the same for all four assignment methods. Interestingly, the number of units is 193 for both the 9-month and 12-month deployment solutions. With 15-month deployments, the number of units decreases to 176. FFL Swap is clearly capable of more successfully minimizing the number of locations per units when demand is more evenly distributed, regardless of deployment length. First-Fit Swap tends to consistently produces slightly better average dwell results, but the difference from other methods is relatively insignificant.

	Locations per Unit			Max Locations			Average Dwell			w
Method	9	12	15	9	12	15	9	12	15	
First-Fit	1.9707	1.6117	1.2208	3	3	2	2.3279	2.3300	2.0843	1:2
First-Fit Swap	1.0256	1	1	3	1	1	**2.3581**	**2.3964**	**2.1260**	
FFL	1	1	1	1	1	1	2.3279	2.3300	2.0843	
FFL Swap	1	1	1	1	1	1	2.3279	2.3300	2.0843	

Table 4.8: Demand Excursion with $d := \{9, 12, 15\}$ and $w = 2d$

When the BOG:Dwell ratio constraint is tightened to 1:2, the general observations remain

the same, as seen in Table 4.8. Of course, the number of units required for the scheduling solutions increases; however, the requirement of 273 units remains equal for all 9-month and 12-month solutions. The 15-month solutions require 240 units. FFL Swap continues to produce perfect solutions when measured against the location objective, while the swapping heuristic improves the First-Fit solution to a nearly optimal location solution. First-Fit still dominates the average dwell statistic on a relatively insignificant scale.

4.7.3.2 Steady-State Demand

Often, steady-state demand scenarios are used for theoretical planning or in the absence of demand forecasts. Naturally, the number of units required to fulfill a steady-state scenario will change if minimum dwell policies are adjusted. For the steady-state computations, we generated a demand scenario in which demand remains constant for 64 total units. Demand requirements are distributed unevenly by location but remain constant over time in each location.

The results show that changes in deployment lengths have no impact on the number of units required when demand represents a steady-state scenario. As the deployment length increases, the number of deployments decreases; however, the number of units remains constant. The solution for all algorithms assigns every unit to only one location. Moreover, the minimum, maximum and average dwell statistics are always equal to and bound by the minimum dwell policy constraint. In fact, the steady-state solutions are the same regardless of the distribution by location.

4.7.4 First-Fit Validation

In order to validate the algorithms in terms of optimality in Section 4.7, we apply a version of the Bron-Kerbosch algorithm to identify the maximum cliques of our computational experiment. As mentioned earlier, Kierstead[61] states that minimum color optimality of an interval graph is equal to maximum clique size.

Cazals and Karande[16] describe the algorithm which maintains three disjoint sets of vertices R, P, and X. Set R stands for the currently growing clique, P represents prospective vertices which

are connected to all vertices in R, and X contains vertices already processed. Since all maximal cliques containing the vertices in X have been reported, those vertices are no longer used in the current iteration. The Bron-Kerbosch algorithm is shown in Algorithm 3.

Call $BK*$ $(R = \varnothing,\ P := V,\ X = \varnothing)$;
if $P = \varnothing$ **and** $X = \varnothing$ **then**
  Report R as a maximum clique;
else
  Let u_p be the pivot vertex;
  Assume $P = \{u_1, u_2, \ldots, u_k\}$;
  for $i \leftarrow 1$ **to** k **do**
    if u_i **is not a neighbor of** u_p **then**
      $P = P - \{u_i\}$;
      $R_{new} = R \cup \{u_i\}$;
      $P_{new} = P \cap N[u_i]$;
      $X_{new} = X \cap N[u_i]$;
      $P_{new} = P \cap \{u_i\}$;
      $BK * (R_{new}, P_{new}, X_{new})$;
      $X = X \cup \{u_i\}$;
    end
  end
end

Algorithm 3: Bron-Kerbosch Algorithm

Since we use solutions from First-Fit as a base case for comparison in earlier results, it is important that we show these solutions to be optimal in reference to the number of units used. Therefore a perfect solution in terms of the location objective is one which uses the minimum number of units and assigns all units to only one location. The Bron-Kerbosch successfully validates our results as the maximum clique from each of the constrained demand scenarios is equal to our First-Fit solutions.

4.8 Conclusion

We demonstrate how the DSUA problem can be efficiently structured and solved using modifications to existing graph coloring techniques. Unlike traditional coloring problems, the DSUA problem seeks to improve multiple objectives. Section 4.4 proves the theoretical limitations of

BOG:Dwell in steady-state scenarios relative to the number of units available for sourcing and deployment length.

In Section 4.5, we show that the DSUA problem can be formulated as an integer programming problem. Due to the special characteristics of the DSUA problem in which we can assume fixed deployment lengths and the set of variable and constraints is extremely large, graph coloring methods reduce the solution complexity and provide a more efficient method for targeting multiple objectives at once. Moreover, the scheduling structure, which builds upon an indexed time horizon lends itself to algorithms that are proven to optimize the coloring solutions of interval graphs in respect to the minimum number of colors.

Two primary algorithms, First-Fit and FFL, are presented along with a swapping heuristic which seeks to improve their solutions against the objective which minimizes locations per unit. Solutions are also evaluated based on BOG:Dwell performance, which while not targeted by the algorithms, seems to perform in concert with the location objective. Perhaps most importantly, we show that a deterioration in BOG:Dwell ratio performance is not sacrifice for improvements with the location objective. Naturally, BOG:Dwell conflicts with an objective to minimize units because an increase in the supply for unit sourcing can be used to increase average dwell time.

Using a historical demand scenario and other demand scenario excursions, computational experiments demonstrate the effectiveness of First-Fit Swap and FFL Swap in producing DSUA problem solutions that are optimal in respect to the unit objective. Depending on the complexity of the demand scenario and policy constraints, solutions are reasonably close to a goal of one location per unit and always average less than two locations per unit. FFL Swap is the most effective method to target the location objective and First-Fit Swap always provides an optimal solution against the unit supply objective while significantly targeting the location objective. No method provides a significant advantage over another in terms of BOG:Dwell ratios; however, the results from tightening the minimum BOG:Dwell policy constraint provide an expectation for average BOG:Dwell results that is important for consideration when balancing the implication of budget decision with the size of the sourcing pool with statistical goals such as average BOG:Dwell.

The DSUA problem is of significant importance to the Army from multiple perspectives. Clearly, dwell and location performance measures that are indicators of stress on the force and operational consistency, which can be used to drive palatable DSUA solutions and influence policies. For example, since individuals do not align with single units throughout their careers, we cannot expect individuals dwell policies to meet or exceed unit capabilities. Moreover, the ability to minimize the number of units required to meet expected demand has even broader and more direct budget and force structure implications. In an era of tightening budget constraints, the ability to meet demand with a minimum size of force is of great importance. This research demonstrates how the Army can use these methods to minimize the size of its force structure requirements and reduce costs while simultaneously understanding the impact in regards to other measures that affect operational readiness and individual experiences that influence retention.

An important component to this research is the ability to use forecasted demand to drive DSUA problem solutions. Because of the uncertainty of demand, the time horizon for a reliable solution may be limited. Future research of these methods could provide alternative methods that would account for additional uncertainty in demand by incorporating theory from online interval graphs; therefore, producing a solution for a shorter time interval before demand further along the time horizon is observed.

Bibliography

[1] Hesham Alfares. Survey, categorization, and comparison of recent tour scheduling literature. Annals of Operations Research, 127, 2004.

[2] T.W. Anderson and Leo A. Goodman. Statistical inference about markov chains. The Annals of Mathematical Statistics, 28(1), 1957.

[3] Beth J. Asch, James Hosek, Michael Mattock, and Christina Panis. Assessing compensation reform. Technical report, RAND National Defense Research Institute, 2008.

[4] Beth J. Asch and John T. Warner. A theory of compensation and personnel policy in hierarchical organizations with application to the united states military. Journal of Labor Economics, 2001.

[5] Steven M. Aviles. Scheduling army deployment to two nearly simultaneous major regional conflicts. Master's thesis, Naval Postgraduate School, 1995.

[6] James Bailey, Hesham Alfres, and Win Yuan Lin. Oprimization and heuristic models to integrate project task and manpower scheduling. Computers and Industrial Engineering, 29(1-4):473–476, 1995.

[7] Kenneth R. Baker. Workforce allocation in cyclical scheduling problems: A survey. Operational Research Quarterly, 27, 1976.

[8] Robert J. Beck and Stephen G. Pauker. The markov process in medical prognosis. Medical Decision Making, 3(4), 1983.

[9] Richard Bellman. Introduction to Matrix Analysis. Society for Industrial and Applied Mathematics, 1997.

[10] Tanja F. Blackstone. Recruiting and retention of military personnel.pdf. Technical report, Navy Personnel Research, Studies, and Technology, 2007.

[11] Ivo Blöchliger. Modeling staff scheduling problems. a tutorial. European Journal of Operational Research, 158, 2004.

[12] Timothy M. Bonds, Dave Baiocchi, and Laurie L. McDonald. Army deployments to oif and oef. Technical report, RAND Corporation, 2010.

[13] Peter Brucker, Rong Qu, and Edmund Burke. Personnel schedling: Models and complexity. European Jounral of Operational Research, 210, 2011.

[14] Michael J. Brusco and Tony R. Hohns. Staffing a multiskilled workforce with varying levels of productivity: An analysis of cross-training policies. Decision Sciences, 29(2), 1998.

[15] Richard J. Buddin. Success of first- term soldiers: The effects of recruiting practices and recruit characteristics. Technical report, Rand Corporation, 2005.

[16] F. Cazals and C. Karande. A note on the problem of reporting maximal cliques. Theori=etical Computer Science, 407, 2008.

[17] T.C.E. Cheng and Z.-L. Chen. Parallel-machine scheduling problems with earliness and tardiness penalties. Journal of the Operational Research Society, 45(6), 1994.

[18] S. Chick, P.J. Sanchez, D. Ferrin, and D.J. Morrice, editors. Practical Introduction to Simulation Optimization. Proceedings of the 2003 Winter Simulation Conference, 2003.

[19] S. Chick, P.J. Sanchez, D. Ferrin, and D.J. Morrice, editors. A Simulation Approach to Manpower Planning. Proceedings of the 2003 Winter Simulation Conference, 2003.

[20] Hunter R. Coates, Teresa S. Silvernail, Lawrence V. Fulton, and Lana Ivanitskaya. The effectiveness of the recent army captain retention program. Armed Forces & Society, 37(1), October 2010.

[21] Daniel Costa, Alain Hertz, and Olivier Dubuis. Embedding a sequential procedure within an evolutionary algorithm for coloring problems in graphs. Journal of Hueristics, 1995.

[22] Bruce A. Craig and Peter P. Sendi. Estimation of the transition matrix of a discrete-time markov chain. Health Economics, 11, 2002.

[23] R. M. Cyert, H. J. Davidson, and G. L. Thompson. Estimation of the allowance for doubtful accounts by markov chains. Management Science, 8(3), 1962.

[24] Matthew Dabkowski, Michael J. Kwinn, Kent Miller, and Mark Zais. Unit bog:dwell... a closed-form approach. Phalanx, 42(4), December 2009.

[25] Department of Defense. Defense Budget Priorities and Choices - Fiscal Year 2014, April 2014.

[26] Department of the Army. Army Posture Statement: Addendum H (Army Force Generation), 2007.

[27] Department of the Army. Personnel Policy Guidance for Overseas Contingency Operations, July 2009.

[28] Department of the Army. Army Regulation 600-8-19: Enlisted Promotions and Reductions, 2010.

[29] Department of the Army. AR 525-29: Army Force Generation, March 2011.

[30] Department of the Army. Army Deployment Period Policy, August 2011.

[31] Department of the Army. Army Regulation 601-280: Army Retention Program, 2011.

[32] D. D DeWolfe, J. G Stevens, and R. K Wood. Setting military reenlistment bonuses. Naval Research Logistics, 40, 1993.

[33] Thomas J. DiCiccio and Bradley Efron. Bootstrap confidence intervals. Statistical Science, 11(2), 1996.

[34] Peter G. Doyle and Laurie J. Snell. Random walks and electric networks. Free Software Foundation, 2000.

[35] Thomas Duala and Robert Moffitt. Estimating dynamic models of quit behavior: The case of military reenlistment. Journal of Labor Economics, 13(3):499–523, July 1995.

[36] Rick Durrett. Essentials of Stochastic Processes. Springer Science, 1999.

[37] B. Efron and Tibshirani. Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. Statistical Science, 1(1), 1986.

[38] A.T. Ernst, H. Jiang, M. Krishnamoorthy, and D. Sier. Staff scheduling and rostering: A review of applications, methods and models. European Jounral of Operational Research, 153, 2004.

[39] Michael C. Fu. Optimization for simulation: Theory vs. practice. INFORMS Journal on Computing, 14(3), 2002.

[40] Philippe Galinier and Alain Hertz. A survey of local search methods for graph coloring. Computers and Operations Research, 33, 2006.

[41] Michel Gamach, Alain Hertz, and Jerome Olivier Ouellet. A graph coloring model for a feasibility problem in monthly crew scheduling with preferential bidding. Computers and Operations Research, 34, 2007.

[42] Fred Glover. Tabu search - part i. ORSA Journal of Computing, 1(3), 1989.

[43] Fred Glover, Randy Glover, Joe Lorenzo, and Claude McMillan. The passenger-mix problem in the scheduled airlines. Interfaces, 12, 1982.

[44] Fred Glover and Claude McMillan. The general employee scheduling problem: An integration of ms and ai. Computers and Operations Research, 13, 1986.

[45] Jonathan Goodman. Principles of scientific computing. [unpublished] www.cns.nyu.edu, 2006.

[46] Glenn A. Gotz and John J. McCall. Sequential analysis of the stay/leave decision: U.S. air force officers. Management Science, 29(3), March 1983.

[47] Glenn A. Gotz and John J. McCall. A dynamic retention model for air force officers. Project air force report, Rand Corporation, December 1984.

[48] R.L. Graham, E.L. Lawler, J.K. Lenstra, and A.H.G. Rinnooy Kan. Annals of Discrete Mathematics 5: Optimization and Approximation in Deterministic Sequencing and Scheduling: A Survey. North-Holland Publishing Company, 1979.

[49] Richard C. Grinold. Manpower planning with uncertain requirements. Operations Research, 24(3), 1976.

[50] Andrew O. Hall. Simulating and Optimizing: Military Manpower Modeling and Mountain Range Options. PhD thesis, University of Maryland, 2009.

[51] PR Harper, NH Powell, and JE Williams. Modelling the size and skill-mix of hospital nursing teams. Journal of the Operational Research Society, 61, 2010.

[52] Glenn W. Harrison, Morten I. Lau, and Melonie B. Williams. Estimating individual discount rates in denmark: A field experiment. The American Economic Review, 2002.

[53] Anita U. Hattiangandi, Gary Lee, Robert W. Shuford, and Aline O. Quester. Retirement choices: 2013. Technical report, Center for Navy Analyses, 2013.

[54] Charles A. Henning. Military retirement: Background and recent developments. Technical report, Congressional Research Service, 2008.

[55] Charles A. Henning. Military retirement reform: A review of proposals and options for congress. Technical report, Congressional Research Service, 2011.

[56] T.J. Hodgson, B. Melendez, K.A. Thoney, and T. Trainor. The deployment scheduling anayisis tool (dsat). Mathematical and Computer Modelling, 39:905–924, 2004.

[57] James Hosek, Michael G. Mattock, Christine Fair, Jennifer Kavanagh, Jennifer Sharp, and Mark Totten. Attracting the best: How the military competes for information technology personnel. Technical report, RAND National Defense Research Institute, 2004.

[58] Huei-Chueng Huang, Loo-Hay Lee, Haiqing Song, and Brian Thomas Eck. SimMan - a simulation model for workforce capacity planning. Computers Operations Research, 36, 2008.

[59] David W. Hughes, Mark M. Zais, Paul Kucik, and Fernando M. Huerta. Arforgen bog:dwell simulation. Technical report, Operations Research Center of Excellence, 2011.

[60] Karl Jehle. Cost of doing business: Friction in the personnel structure system. British and American Forescasting Exchange, 2001.

[61] H.A. Kierstead. The linearity of first-fit coloring of interval graphs. Society for Industrial and Applied Mathematics, 1(4), 1988.

[62] H.A. Kierstead and Jun Qin. Coloring interval graphs with first-fit. Discrete Mathematics, 144, 1995.

[63] Frank Thomson Leighton. A graph coloring algorithm for large scheduling problems. Journal of Research of the National Bureau of Standards, 84(6), 1979.

[64] J.K. Lenstra, A.H.G. Rinnooy Kan, and P. Brucker. Complexity of machine scheduling problems. Annals of Discrete Mathematics, 1, 1977.

[65] Enrico Malaguti and Paolo Toth. A survey on vertex coloring problems. International Transactions in Operational Research, 17, 2010.

[66] R.T. Marler and J.S. Arora. Survey of multi-objective optimization methods for engineering. Structural Multidisciplinary Optimization, 26, 2004.

[67] Michael Mattock and Jeremy Arkes. The dynamic retention model for air force officers: New estimates and policy simulations of the aviator continuation pay program. Technical report, Rand Corporation, 2007.

[68] Michael G. Mattock, James Hosek, and Beth J. Asch. Reserve participation and cost under a new approach to reserve compensation. Technical report, RAND National Defense Research Institute, 2012.

[69] K. McKinzie and J.W. Barnes. A review of strategic mobility models supporting the defense transportation system. Mathematical and Computer Modelling, 39, 2004.

[70] Douglas C. Montgomery and George C. Runger. Applied Statistics and Probability for Engineers. John Wiley & Sons, Inc., 2003.

[71] Carol S. Moore and Anita U. Hattiangadi. Inside the black box: Assessing the navy's manpower requirements. Technical report, Center for Navy Analyses, 2002.

[72] Tomomichi Nakamura, Kevin Judd, Alistair I. Mees, and Michael Small. A comparative study of information criteria for model selection. International Journal of Bifurcation and Chaos, 16(8), 2006.

[73] S. Nirmala and M. Jeeva. A dynamic programming apporach to determine optimal manpower recruitment and promotion policies for the two grade system. African Journal of Mathematics and Computer Science Research, 3(12), 2010.

[74] Linda O'Brien-Pallas, Stephen Birch, Andrea Baumann, and Gail Tomblin Murphy. Integrating workforce planning, human resources, and service planning. Technical report, World Health Organization, 2001.

[75] G. Palubeckis. On the recursive largest first algorithm for graph colouring. International Jounal of Computer Mathematics, 85(2), 2088.

[76] Martin L. Puterman. Markov Decision Processes: Discrete Stochastic Dynamic Programming. John Wiley & Sons, Inc., 2005.

[77] Xiangtong Qi and Jonathan F. Bard. Generating labor requirements and rosters for mail handlers using simulation and optimization. Computers Operations Research, 33, 2006.

[78] P.P. Rao. A dynamic programming apporach to determine optimal manpower recruitment policies. Journal of the Operational Research Society, 1990.

[79] Heather Reed. Wartime sourcing: Building capability and predictability through continuity. Military Review, pages 62–69, May-June 2011.

[80] K.H. Rosen. Elementary Number Theory and Its Applications. Addison Wesley Longman, 6th edition, 2011.

[81] Sheldon M. Ross. Stochastic Processes. John Wiley & Sons, Inc., 1996.

[82] James R. Stewart, Derek B. ad White. Military personnel: Active duty compensation and its tax treatment. Technical report, U.S. Government Accountability Office, 2004.

[83] George P.H. Styan. Hadamard products and multivariate statistical analysis. Linear Algebra and Its Applicatioms, 6, 1973.

[84] Gary M. Thompson. Accounting for the multi-period impact of service when determining employee requirements for labor scheduling. Journal of Operations Management, 1993.

[85] Jorne Van den Bergh, Jeroen Beliën, Philippe De Brueker, and Erik Demeulemeester. Personnel scheduling: A literature review. European Journal of Operational Research, 226, 2013.

[86] Jorne Van den Bergh, Joroen Beliën, Philippe De Brueker, and Erik Demeulemeester. Personnel scheduling: A literature review. European Jounral of Operational Research, 226, 2012.

[87] John T. Warner and Matthew S. Goldberg. The influence of non-pecuniary factors on labor supply: The case of navy enlisted personnel. The Review of Economics and Statistics, 1984.

[88] John T. Warner and Saul Pleeter. The personal discount rate: Evidence from military downsizing programs. The American Economic Review, 2001.

[89] Kum-Khiong Yang, Scott Webster, and Robert A. Ruben. An evaluation of worker cross training and flexible workdays in job shops. IIE Transactions, 39, 2007.

[90] Mark M. Zais. Personnel friction analysis: Quantifying the cost of doing business. Technical report, Army G1, Military Strength Analysis Forecasting Division, May 2011.

Appendix A

SAS Code for Transition Counts

```
libname my_sas '[File Location]';
* Reduce Inventory data set to single CMF;
data my_sas.Inv_200610_200910_Infantry;
set my_sas.Inv_200610_200910_Secure;
if PMOS_CD in ('11B','11C','11X','11Z');
GRADE = PAY_GRADE_ID;
if GRADE le 4 then GRADE = 4;
TIG = floor(TIG_QY/12);
if GRADE = 4 then TIG = floor(MSV_QY/12);
if TIG < 10 then STATE = cat(GRADE,'_0',TIG);
if TIG ge 10 then STATE = cat(GRADE,'_',TIG);
run;

data work.inv_temp1;
set my_sas.Inv_200610_200910_infantry;
trans_num=substr(transaction_number,1,length(transaction_number)-3);
if TIG_QY ne .;
if MSV_QY ne .;
run;

proc sort data=work.inv_temp1;
by trans_num YM_DT;
quit;

data work.inv_temp1a;
set work.inv_temp1;
TRAINING = 0;
retain RAND;*prevents RAND from being reset to missing for each iteration. ;
first=trans_num;
second=lag1(trans_num);
if second ne first then RAND = ranuni(7);
if RAND < 0.70 then TRAINING = 1;
drop RAND first second;
run;
```

```
data work.inv_training1;
set work.inv_temp1a;
if TRAINING = 1;
run;

data my_sas.inv_training1;
set work.inv_training1;
run;

data work.inv_test;
set work.inv_temp1a;
if TRAINING = 0;
run;

* Only keep inventory from 3 snapshots;
data work.inv_temp2;
set inv_training1;
if YM_DT in (200709,200809,200909);
run;

proc sort data=work.inv_temp2;
by trans_num YM_DT;
quit;

data work.inv_test2;
set inv_test;
if YM_DT = 200610 then TIG = floor((TIG_QY-1)/12);*calculate 1 month back;
run;

proc sort data=work.inv_test2;
by trans_num YM_DT;
quit;

* Check to see if transaction number is the same as previous transaction number;
* Check to see if the individual is still in the same Grade;
* If number is the same, continue = 1.  If grade is same, same_grade = 1;

data work.inv_firstperiod;
set work.inv_temp2;
if YM_DT in (200709); /*CHANGE THIS FOR EACH YEAR*/
run;

/*Track continuation of individuals who are in first period*/
/*proc sql;
create table work.inv_continuation as
select b.*
```

```
from work.inv_firstperiod a,
 work.inv_temp2 b
where a.trans_num = b.trans_num
order by b.trans_num, YM_DT;
quit;*/

proc sort data=work.inv_temp2;
by trans_num YM_DT;
quit;

data work.inv_temp2b;
set work.inv_temp2;
continue=0;
same_grade=0;
first=trans_num;
second=lag1(trans_num);
if second=first then continue=1;
grade1=GRADE;
grade2=lag1(GRADE);
if grade1=grade2 then same_grade=1;
drop first second grade1 grade2;
run;

proc sql;
create table work.inv_temp2c as
select YM_DT as DATE, STATE, COUNT(continue)as Count
from work.inv_temp2b
where (EXCLUDABLE_STAT_CD = 'N' and RECSTA_CD = 'G' and continue=1 and same_grade=1)
group by DATE, STATE;
quit;

proc sort data = work.inv_temp2c;
by STATE;
run;

* Create table that displays counts State (row) and Date (column);
proc transpose data = work.inv_temp2c out = work.inv_temp2d (DROP = _NAME_);
by STATE; *row;
id DATE; *column;
var COUNT; *value;
run;

*Create new field to match current INV with previous month TRANS;
*Only keep voluntary loss transactions;
data work.trans_temp1;
set my_sas.Trans_200610_200910_secure;
MONTH=1*substr(put(DATE,z6.),5,2);
```

```
if MONTH = 01 then NEWMONTH = 12;
else NEWMONTH = MONTH-1;
NEWMONTH2 = put(NEWMONTH,z2.);
YEAR=1*substr(put(DATE,z6.),1,4);
if MONTH = 01 then NEWYEAR = YEAR-1;
else NEWYEAR = YEAR;
PREV_DATE = 1*cat(NEWYEAR, NEWMONTH2);
drop MONTH YEAR NEWMONTH NEWMONTH2 NEWYEAR;
where LOSS_TYPE_CD = 'LETS' or LOSS_TYPE_CD = 'LNDR';
*where GAIN_TYPE_CD ne 'GIMR';
run;

* remove last 3 characters from transaction number;
data work.trans_temp2;
set work.trans_temp1;
trans_num=substr(transaction_number,1,length(transaction_number)-3);
run;

proc sql;
create table work.trans_temp2b as
select a.DATE, a.TRANS_CAT_CD, a.LOSS_TYPE_CD, a.TRANS_DT, b.*
from work.trans_temp2 a,
 work.inv_firstperiod b
where a.trans_num = b.trans_num
order by DATE, a.trans_num;
quit;

proc sql;
create table work.trans_temp2d as
select DATE, STATE, COUNT(distinct(trans_num))as Count
from work.trans_temp2b
where (EXCLUDABLE_STAT_CD = 'N' and RECSTA_CD = 'G' and DATE > 200709...
   and DATE < 200810)
group by DATE, STATE;
quit;

proc sort data=work.trans_temp2d;
by STATE DATE;
quit;

* Create table that displays counts State (row) and Date (column);
proc transpose data = work.trans_temp2d out = work.trans_temp2e (DROP = _NAME_);
by STATE; *row;
id DATE; *column;
var COUNT; *value;
run;
```

```
*Create new field to match current INV with previous month TRANS;
*Only keep involuntary loss transactions;
data work.trans_temp3;
set my_sas.Trans_200610_200910_secure;
MONTH=1*substr(put(DATE,z6.),5,2);
if MONTH = 01 then NEWMONTH = 12;
else NEWMONTH = MONTH-1;
NEWMONTH2 = put(NEWMONTH,z2.);
YEAR=1*substr(put(DATE,z6.),1,4);
if MONTH = 01 then NEWYEAR = YEAR-1;
else NEWYEAR = YEAR;
PREV_DATE = 1*cat(NEWYEAR, NEWMONTH2);
drop MONTH YEAR NEWMONTH NEWMONTH2 NEWYEAR;
where TRANS_CAT_CD = 'LOSS' and LOSS_TYPE_CD not in ('LETS', 'LNDR', 'LIMR');
run;

* remove last 3 characters from transaction number;
data work.trans_temp4;
set work.trans_temp3;
trans_num=substr(transaction_number,1,length(transaction_number)-3);
run;

proc sql;
create table work.trans_temp4b as
select a.DATE, a.TRANS_CAT_CD, a.LOSS_TYPE_CD, a.TRANS_DT, b.*
from work.trans_temp4 a,
 work.inv_firstperiod b
where a.trans_num = b.trans_num
order by DATE, a.trans_num;
quit;

proc sql;
create table work.trans_temp4d as
select DATE, STATE, COUNT(distinct(trans_num))as Count
from work.trans_temp4b
where (EXCLUDABLE_STAT_CD = 'N' and RECSTA_CD = 'G' and DATE > 200709...
     and DATE < 200810)
group by DATE, STATE;
quit;

proc sort data=work.trans_temp4d;
by STATE DATE;
quit;

* Create table that displays counts State (row) and Date (column);
proc transpose data = work.trans_temp4d out = work.trans_temp4e (DROP = _NAME_);
by STATE; *row;
```

```
id DATE; *column;
var COUNT; *value;
run;

data work.prom_temp1;
set my_sas.Trans_200610_200910_secure;
MONTH=1*substr(put(DATE,z6.),5,2);
if MONTH = 01 then NEWMONTH = 12;
else NEWMONTH = MONTH-1;
NEWMONTH2 = put(NEWMONTH,z2.);
YEAR=1*substr(put(DATE,z6.),1,4);
if MONTH = 01 then NEWYEAR = YEAR-1;
else NEWYEAR = YEAR;
PREV_DATE = 1*cat(NEWYEAR, NEWMONTH2);
drop MONTH YEAR NEWMONTH NEWMONTH2 NEWYEAR;
where TRANS_CAT_CD = 'PROM';
run;

* remove last 3 characters from transaction number;
data work.prom_temp2;
set work.prom_temp1;
trans_num=substr(transaction_number,1,length(transaction_number)-3);
run;
proc sql;

create table work.prom_temp2b as
select a.DATE, a.TRANS_CAT_CD, a.LOSS_TYPE_CD, a.TRANS_DT, b.*
from work.prom_temp2 a,
 work.inv_firstperiod b
where a.trans_num = b.trans_num
order by DATE, a.trans_num;
quit;

proc sql;
create table work.prom_temp2d as
select DATE, STATE, COUNT(distinct(trans_num))as Count
from work.prom_temp2b
where (EXCLUDABLE_STAT_CD = 'N' and RECSTA_CD = 'G' and DATE > 200709...
     and DATE < 200810 and PAY_GRADE_ID > 4)
group by DATE, STATE;
quit;

proc sort data=work.prom_temp2d;
by STATE DATE;
quit;

* Create table that displays counts State (row) and Date (column);
```

```
proc transpose data = work.prom_temp2d out = work.prom_temp3 (DROP = _NAME_);
by STATE; *row;
id DATE; *column;
var COUNT; *value;
run;

* Create table to calculate number of people by State.;
proc sql;
create table work.state_temp as
select YM_DT as DATE, STATE, COUNT(distinct(trans_num)) as Count
from work.inv_temp2
where (EXCLUDABLE_STAT_CD = 'N' and RECSTA_CD = 'G')
group by DATE, STATE;
quit;
proc sort data = work.state_temp;
by STATE;
run;

proc transpose data = work.state_temp out = work.state_temp2 (DROP = _NAME_);
options missing ='0';
by STATE; *row;
id DATE; *column;
var COUNT; *value;
run;

*Analyze Test Sample;
data work.trans_test1;
set my_sas.Trans_200610_200910_secure;
MONTH=1*substr(put(DATE,z6.),5,2);
if MONTH = 01 then NEWMONTH = 12;
else NEWMONTH = MONTH-1;
NEWMONTH2 = put(NEWMONTH,z2.);
YEAR=1*substr(put(DATE,z6.),1,4);
if MONTH = 01 then NEWYEAR = YEAR-1;
else NEWYEAR = YEAR;
PREV_DATE = 1*cat(NEWYEAR, NEWMONTH2);
drop MONTH YEAR NEWMONTH NEWMONTH2 NEWYEAR;
where TRANS_CAT_CD = 'LOSS' and LOSS_TYPE_CD not = 'LIMR';
run;

* remove last 3 characters from transaction number;
data work.trans_test2;
set work.trans_test1;
trans_num=substr(transaction_number,1,length(transaction_number)-3);
run;

proc sql;
```

```
create table work.trans_test3 as
select a.*, b.TRANS_CAT_CD, b.LOSS_TYPE_CD, b.TRANS_DT
from work.inv_test2 a,
 work.trans_test2 b
where a.trans_num = b.trans_num and YM_DT=PREV_DATE
order by a.trans_num, YM_DT;
quit;

/*Evaluate inventory at period 0 and period 3*/
data work.test_firstperiod;
set work.inv_test2;
if YM_DT in (200709);
run;

proc sql;
select GRADE, COUNT(distinct(trans_num)) as Count
from work.test_firstperiod
where (EXCLUDABLE_STAT_CD = 'N' and RECSTA_CD = 'G')
group by GRADE;
quit;

proc sql;
create table work.start_test as
select STATE, COUNT(distinct(trans_num)) as Count
from work.test_firstperiod
where (EXCLUDABLE_STAT_CD = 'N' and RECSTA_CD = 'G')
group by STATE;
quit;

data work.test_lastperiod;
set work.inv_test2;
if YM_DT in (200909);
run;

proc sql;
create table work.test_still_in as
select b.*
from work.test_firstperiod a,
 work.test_lastperiod b
where a.trans_num = b.trans_num
order by YM_DT, b.trans_num;
quit;

proc sql;
select GRADE, COUNT(distinct(trans_num)) as Count
from work.test_still_in
where (EXCLUDABLE_STAT_CD = 'N' and RECSTA_CD = 'G')
```

```
group by GRADE;
quit;

proc sql;
create table work.end_test as
select STATE, COUNT(distinct(trans_num)) as Count
from work.test_still_in
where (EXCLUDABLE_STAT_CD = 'N' and RECSTA_CD = 'G')
group by STATE;
quit;

/*Count the number of voluntary losses*/
proc sql;
create table work.test_vol as
select a.*, b.DATE, b.TRANS_CAT_CD, b.LOSS_TYPE_CD, b.TRANS_DT
from work.test_firstperiod a,
 work.trans_temp2 b
where a.trans_num = b.trans_num
order by DATE, a.trans_num;
quit;

create table work.test_vol2 as
select STATE, COUNT(distinct(trans_num))as Count
from work.test_vol
where (EXCLUDABLE_STAT_CD = 'N' and RECSTA_CD = 'G' and DATE > 200709...
     and DATE < 200910)
group by STATE;
quit;

/*Count the number of involuntary losses*/
proc sql;
create table work.test_invol as
select a.*, b. DATE, b.TRANS_CAT_CD, b.LOSS_TYPE_CD, b.TRANS_DT
from work.test_firstperiod a,
 work.trans_temp4 b
where a.trans_num = b.trans_num
order by DATE, a.trans_num;
quit;

proc sql;
create table work.test_invol2 as
select STATE, COUNT(distinct(trans_num))as Count
from work.test_invol
where (EXCLUDABLE_STAT_CD = 'N' and RECSTA_CD = 'G' and DATE > 200709...
     and DATE < 200910)
group by STATE;
quit;
```

Appendix B

Alternate Method of Calculating Transition Matrix

B.1 Personnel Inventory Calculations

The Army personnel system is a dynamic system, which acts similarly to a continuous process. The calculation of transition probabilities for annual time periods calls upon traditional inventory theory in order to construct discrete-time Markov chain probabilities from continuous-time Markov chain behavior. A person who becomes eligible for promotion halfway during a period is less likely to be promoted than an individual who is eligible at the beginning of the period. Because of these characteristics, average inventory theory contains the most reasonable method for calculating the average personnel inventory.

Because the transition probabilities in $P_{x,y}$ are conditional with respect to involuntary losses, all personnel records for individuals subject to an involuntary loss are removed from the database during the year of the loss. The same is done for voluntary losses; however, this is due to the voluntary decision we are modeling rather than the conditional probability of an involuntary loss. The average eligible inventory for promotion probabilities is calculated based on the eligible inventory at the beginning of period t and a time-based ratio of the newly eligible inventory throughout period t (Figure 1).

B.2 MDP Transition Probabilities

Not included in the transition probabilities is the opportunity to exit the system as a voluntary loss, which is modeled as a decision in the MDP model. Because voluntary losses are removed from the inventory and transitions are conditional upon avoiding an involuntary loss, the transition probability for promotions is calculated as a proportion of fiscal year promotions and the average eligible inventory (Figure 2). Transition to the next state in the current grade is a result of not being promoted during the time period, or $1 - P(\text{Promotion})$.

The conditional transition probability, $P(x, y|1 - \rho_x)$, where ρ_x depicts the probability of an involuntary loss in state x.

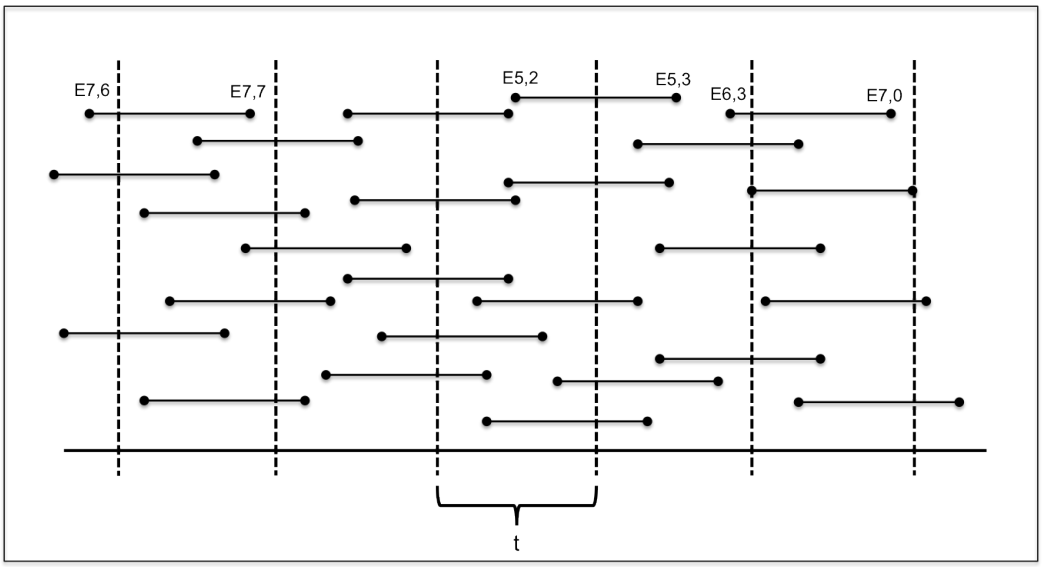


Figure B.1: Constructing discrete-time MC probabilities from continuous-time MC

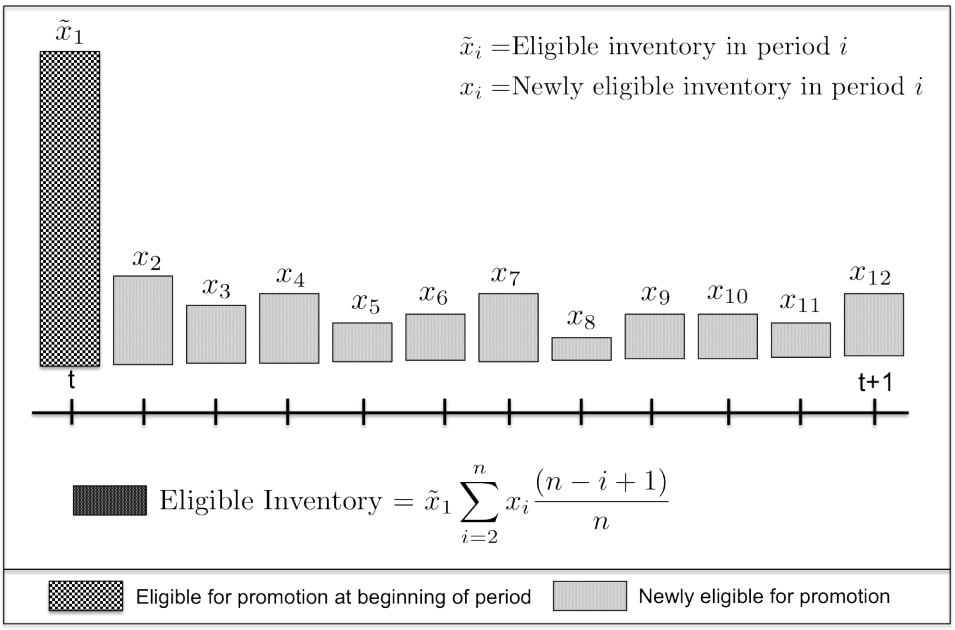


Figure B.2: Eligible Promotion Inventory

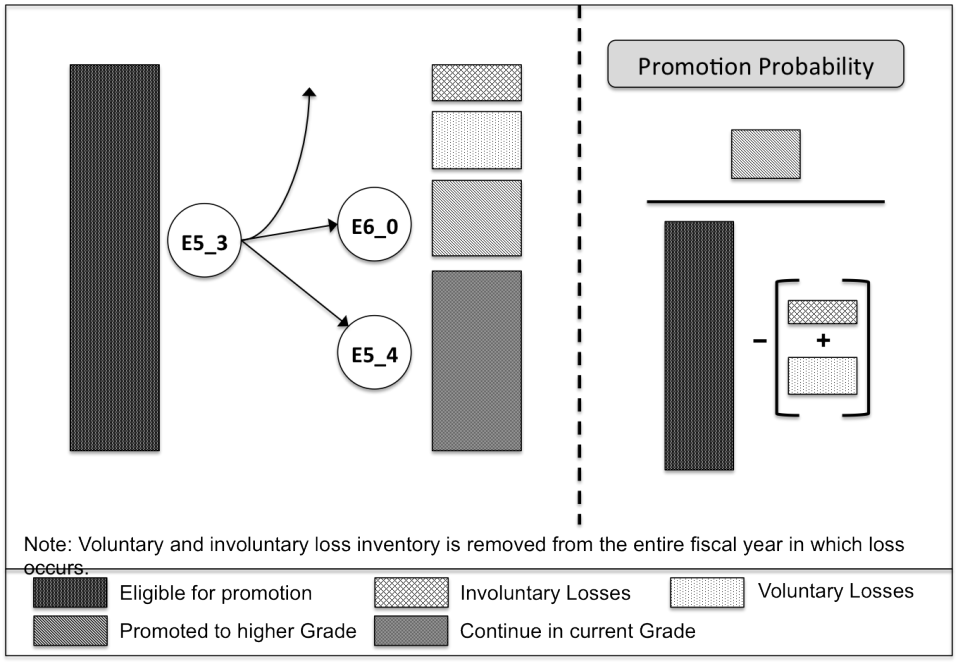


Figure B.3: Calculating Promotion Probabilities

Appendix C

Descriptive Statistics of Data

Marital Status	Percent	Race	Percent	Grade	Percent
Married	49.4%	White	73.7%	SL1	48.8%
Single	47.7%	Hispanic	12.7%	E-5	25.8%
Divorced	2.9%	Black	7.2%	E-6	19.0%
		Asian/Other	6.3%	E-7	4.8%
				E-8	1.3%
				E-9	0.3%
Prior Reenlist	Percent	Transaction	Percent	Deployed	Percent
0	58.6%	ETS	28.9%	Yes	88.2%
1	19.4%	Reenlist	66.9%	No	11.8%
2	11.4%	Retire	4.3%		
3	10.6%				

N = 43,242; Career Management Field = 11 (Infantryman)

Table C.1: Frequency percentiles of ordinal data

Variable	Minimum	Maximum	Mean	Std. Dev.
Age	18	58	26.04	5.454
Time-in-Service (Years)	0.2	30	5.67	4.43
Time-in-Grade (Months)	1	250	20.32	19.22
Months Since Deployment	0	228	12.07	16.26
AFQT	0	99	57.31	20.16
Education Minor Dependents	0	10	0.73	1.097

N = 43,242; Career Management Field = 11 (Infantryman)

Table C.2: Descriptive Statistics of scaled data

Appendix D

Approximation Error of the Fundamental Matrix N

The definition (3.1) of fundamental matrix assumes an infinite number of transitions. This definition gives a compact representation of N using Q. The career path of an enlisted personnel has a finite timeline, typically 30 years. Therefore, we expect that computations based on the fundamental matrix has certain level of error. For this reason, we would like to investigate the possible error from using an infinite summation in (3.1). To that end, we compare the fundamental matrix with a finite summation of the sequence $N_t = \sum_{k=0}^{30} Q^k$. The approximation error of using N instead of N_30 can be evaluated using the absolute and relative errors:

$$||N - N_{30}||_p, \quad \frac{||N - N_{30}||_p}{||N_{30}||_p} \times 100\%.$$

Here, $||\cdot||_p$ represents matrix norm for given constant p.

Table D.1 presents the absolute and relative errors for three different matrix norms with $p = 1, p = 2$, and $p = \infty$. The table shows that the approximation error is quite small, with the largest relative error less than 3%.

Norm	Absolute error	Relative error
1-Norm	0.2867	2.64%
2-Norm	0.1839	2.07%
∞-Norm	0.2292	2.22%

Table D.1: Absolute and Relative Errors of Using N to Approximate N_{30}

Appendix E

REDUX Retirement Option

Alternatively, an individual can choose a REDUX retirement option. The REDUX option can be selected at year 15, upon which an individual receives a $30,000 career retention bonus and commits to serve until at least year 20. However, as a tradeoff for receiving an early lump sum, the retirement pay multiplier initially begins at a lower threshold of 40% and is calculated with the following formula:

$$w_t^R = 0.4 + 0.035(t - 20),\ \forall t \geq 20. \tag{E.1}$$

We can represent the retirement multiplier as a single formula with a binary decision parameter, θ, in which θ equals 1 if an individual selects the REDUX option and equal 0 for the default High-3 Average Retirement System. The resulting formula, equation (5), is applicable for calculating the proportional amount of retirement benefits with an imbedded parameter for choosing a retirement system.

$$\hat{w_t} = 0.5 - 0.1\theta + [0.025 + 0.01\theta](t - 20),\ \forall t \geq 20 \tag{5}$$

Despite the ability to model both current options in the retirement system, an abundance of recent literature suggests that the High-Three option is economically preferable to the REDUX option, at most retirement decision points. The initial bonus of $30,000 at the 15th year of service may seem appealing from a context of time-value of money; however, the immediate benefit does not outweigh a significant reduction in retirement income. The reduction of retirement income is magnified as grade increases and years of service at retirement decreases. The reduction becomes greater the longer an individual lives, and as each year passes, the difference between REDUX and High-3 retirement income increases (Hattiangandi et al., 2013). For context, analysis conducted by Hattiangandi et al. shows that an E8 who retires at age 42 with 24 years of service will see a reduction in after-tax retirement pay of over $382,000 if they live until age 79.[1] Also, because there is no provision for automatically increasing the career status bonus to adjust for inflation, the relative worth of the $30,000 continues to decrease. Despite historically higher rates for enlisted personnel versus officers, recent evidence of a reduction in take rates suggests, at a minimum, an increase in the awareness towards the negative aspects of choosing the REDUX retirements option. The Department of Defense has collected limited data pertaining to Army REDUX take rates; however, if Army trends are consistent with data internally maintained by the Navy and Marines, a precipitous decline in popularity since 2003 may be assumed (Henning, 2011). For modeling simplicity and analytical practicality, we assume all individuals choose the High-3 retirement option.

[1] Assumes a 15% tax bracket and a 3.5% growth per year in military pay until retirement.

Setting $\theta = 0$, implies that we consistently use equation (3.4) for the enlisted retention model such that

$$w_t = w_t^A = 0.5 + 0.025(t - 20),\ \forall t \geq 20 \tag{6}$$

The parameter-based equation (5) for calculating retirement pay is not completely discarded in favor of preserving potential future research oriented on the implementation of optimal career status bonus values, which could make REDUX viable.

Appendix F

Glossary of Notations for DP Model

$m_{x,t}$: Military base pay in state x and period t

$c_{x,t}$: Additional compensation pay in state x and period t

$\widehat{m}_{x,t}$: High-3 average retirement pay in state x and period t

$q_{x,y}$: Probability of becoming an involuntary loss in state x

w_t: Proportional amount of $\widehat{m}_{x,t}$ expected to receive in retirement

δ_i: Multiplier of final military pay to calculate expected civilian pay

T: Length of military and civilian career time horizon

β: Personal discount factor

τ: Individual life expectancy

η: Age upon entering military

ξ: Expected civilian pay raises

Appendix G

MATLAB Code for Markov Chain Estimation

```
% Value Iteration Algorithm for Determining Enlisted Personnel Propensity
  to Reenlist

load P_matrix.mat % transition matrix;
load m_monthly.mat % basic military pay (by grade and years-of-service)
load c_monthly.mat % military compensation pay
load rho_matrix.mat % probability of involuntary loss at each state
load alpha.mat % empirical probability of civilian cross-skill job
load delta.mat % civilian pay mulitplier

%load m_monthly2.mat  % used for composite rates
%load delta_composite.mat % used for composite rates

P = P_matrix
S = 77;          % Number of states
T = 40;          % Number of time periods
C = 10;          % Number of civilian coss-skill jobs
rho = rho_matrix;  % Conditional probability of involuntary loss
eta=22;           % Represents expected age entering Army
tau=75;          % Represents expected life expectancy
beta=0.8696;        % Personal discount factor
ksi=0.025;       % Civilian pay raise per period

% Check matrix dimensions
size(P);
if (size(P) ~= [S, S])
   warning('Check P_matrix or state size')
end

% Convert monthly basic pay to annual value ;
m_pay=m_monthly*12;

% Convert monthly additional compensation pay to annual value;
c_pay=c_monthly*12;
```

```
% Calculate retirement percentage
mhat=zeros(S,T);
w=zeros(T);

  for t=1:19
     w(t) = 0;
  end
  for t=20:T
     w(t) = min(.75,(0.5+.25*(t-3)));
  end

  % mhat will be 0 for anything less than 3
  for t=1:19
      for x = 1:S
          mhat(x,t) = w(t)*m_pay(x,t);
      end;
  end

  for t=20:T
      for x = 1:S
          mhat(x,t) = w(t)*(m_pay(x,t-2)+m_pay(x,t-1)+m_pay(x,t))/3;
      end;
  end

% Initialization.
v=zeros(S,T+1);
vstay=zeros(S,T+1);
vleave=zeros(S,T+1);
ratio=zeros(S,T);
decision=zeros(x,t);
index=zeros(S,T);

% Value iteration;
stay1=0;
stay2=0;
policy=zeros(S,1);
temp=zeros(S,T);

%Main Value Iteration:
for t=T:-1:1
   for x = 1:S
       stay1 = m_pay(x,t) + c_pay(x,t);
       stay2 = 0;
       for y = S:-1:1
           stay2 = stay2+beta*P(x,y)*v(y,t+1);
       end;
       count(x,t)=0;
```

```
for i = 1:C  %For empirical distribution of civilian pay

         civcomp1=0;
         for k1 = t:tau-eta
             civcomp1 = civcomp1 + w(t)*mhat(x,t)*beta^(k1-t);
         end;
         civcomp2=0;
         for k2 = t:T-1
             civcomp2 = civcomp2 + delta(i)*m_pay(x,t)*beta^(k2-t)*(1+ksi);

         end;

         v(x,t) = v(x,t)+alpha(i)*((1-rho(x))*(max(stay1 + stay2, civcomp1 +...
    civcomp2)) + rho(x)*(civcomp1 + civcomp2));

         leave(x,t) =  civcomp1 + civcomp2;
         stay(x,t) = stay1 + stay2;

         if leave(x,t) > stay(x,t)
             count(x,t)=count(x,t)+alpha(i);
         end;

         vstay(x,t) = stay(x,t);
         vleave(x,t) = vleave(x,t)+alpha(i)*(civcomp1 + civcomp2);

       end;

       index(x,t) = count(x,t)/1;

       if index(x,t) > 0.5
         policy(x)=t;
         decision(x,t)=1;
       end;

   end;
end;

vstar = max(v);
display(vstay);
display(vleave);
display(v);
display(vstar);
display(decision);
csvwrite('decision.csv', decision);
csvwrite('index.csv', index);
```

Appendix H

Java Code - Schedule Build and Algorithm Call

```
import java.io.File;
import java.io.FileNotFoundException;
import java.math.*;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Comparator;
import java.util.Scanner;

public class Coloring {

    public static int d1 = 15; //deployment length
    public static int d2 = 15; //deployment length
    public static int policyChange = 45; //deployment length
    public static int T = 86; //time horizon
    public static int L = 8; //number of locations
    public static int r = 0; //number of deployment schedule rows
    public static int Max = 1000; //create array to contain all deployments
    public static double dw = 1; //dwell multiplier

public static void main(String[] args) throws FileNotFoundException {

//  Import deployment demand data
Scanner scanner1 = new Scanner(new File("/[File Location]/[Demand File]"));

int [][] demand = new int [L][T];//specify array length. MZ

   while(scanner1.hasNextInt()){
       for (int m = 0; m < L; m++){
       for (int n = 0; n < T; n++)
           demand[m][n] = scanner1.nextInt();
       }}

   //Find the maximum value in each demand row;
   int[] maxDemands = new int [L];
   for (int i = 0; i < L; i++){
```

```
        maxDemands[i] = 0;
        for (int j = 0; j < T; j++){
            if (demand[i][j] > maxDemands[i]){
                maxDemands[i] = demand[i][j];
            }
        }
    }

   //Number of schedule rows equal the hgihest of the location max demands
   int maxValue = 0;
   for (int i = 0; i < L; i++){
       if (maxDemands[i] > maxValue ){
           maxValue = maxDemands[i];
       }
   }
   r = maxValue;

   int[][][] initialSolution = new int[r][T][L];

   for (int i = 0; i < r; i++) {
       for (int j = 1; j < T;j++){
           for (int k = 0; k < L; k++){
           initialSolution[i][j][k] = 0;
           }
       }
   }
   int[][][] currSolution = initialSolution;

   int deployCounter = 1;//Used to count the number of deployments needed.
   int nextBeginRow = 0;//The next row to start copying location schedule.
   int[][] locationTracker = new int [Max][2];

//Build a schedule one column at a time
for  (int month = 0; month < T; month++) {
 for (int loc = 0; loc < L; loc++){
  for (int j = 0; j < maxDemands[loc]; j++){

    // currCount counts the number units deployed in a month
    int[] currCount = Coloring.columnCount(currSolution, loc);

//Create deployment schedule
if ((currCount[month] < demand[loc][month])&&(currSolution[j][month][loc] < 1)){
    if(month < policyChange){
        for (int i = 0; i < d1; i++){
        currSolution[j][month+i][loc]=deployCounter;

    }}
```

```
    if((month >= policyChange)&&(month < T-d2)){
        for (int i = 0; i < d2; i++){
        currSolution[j][month+i][loc]=deployCounter;
    }}
    if(month >= T-d2){
        for (int i = 0; i < T-month; i++){
        currSolution[j][month+i][loc]=deployCounter;
    }}
    //Create a separate array to track deployment locations
    locationTracker[deployCounter][0] = deployCounter;
    locationTracker[deployCounter][1] = loc;
    deployCounter ++;
}}}}

// Create a table of the start and end dates for each deployment
int deployment = 0;
int[][] deploymentDates = new int[deployCounter][3];

for (int i = 0; i < r; i++) {
   for (int k = 0; k < L; k++) {
   for (int j = 0; j < T; j++) {

    if ((currSolution[i][j][k] != deployment)&&(currSolution[i][j][k] > 0)){
        deploymentDates[deployment][2] = j;
        deployment = currSolution[i][j][k];
        deploymentDates[deployment][0] = deployment;
        deploymentDates[deployment][1] = j;
    }
    if ((currSolution[i][j][k] != deployment)&&(currSolution[i][j][k] == 0)){
        deploymentDates[deployment][2] = j;
        deployment = currSolution[i][j][k];
}}}}

  //Shrink the array by eliminating unused cells
  int[][] locationList = Arrays.copyOfRange(locationTracker, 0, deployCounter);
  int[][] List = Coloring.IncompatibleList(deployCounter, currSolution, d1, d2);

  sort2(List);//Sort by column number 2
  sort1(List);//Sort by column number 1
  //Find the first row without zeros in the sorted list.
  int cut = Coloring.findFirstNum(List);
  //Copy the array elements that aren't zero.
  int[][] shortList = Arrays.copyOfRange(List, cut, List.length);

  FirstFitAlgorithm lessGreedyAlgorithm = new FirstFitAlgorithm(shortList,
locationTracker, deployCounter, L, deploymentDates, d1);
```

```
  List = Coloring.IncompatibleList(deployCounter, currSolution, d1, d2);
  sort2(List);//Sort by column number 2
  sort1(List);//Sort by column number 1
  cut = Coloring.findFirstNum(List);//Find the first row without zeros.
  shortList = Arrays.copyOfRange(List, cut, List.length);

  FirstFitLocationAlgorithm minUnitsLocationAlgorithm =
new FirstFitLocationAlgorithm(shortList, locationTracker, deployCounter,
L, deploymentDates, d1);
}

public static int[][] IncompatibleList(int numDeploy, int[][][] currSolution,
...int deploylength1, int deployLength2){

    int [][] incompatibleList = new int[numDeploy*numDeploy][2];
    int tracker = 0;
    int listTracker = 0;
    int deploy1 = 0;

    for (int i = 0; i < currSolution.length; i++) {
       for (int j = 0; j < policyChange; j++) {
           for (int k = 0; k < L; k++) {

         if (currSolution[i][j][k] > 0){

      for (int ii=0; ii < r; ii++ ){
       int deploy2 = 0;
       for (int jj = 0; jj < T; jj++ ){
           for (int kk = 0; kk < L; kk++ ){

     if ((currSolution[i][j][k] < currSolution[ii][jj][kk])&&(deploy2 <...
currSolution[ii][jj][kk])&&(jj-j<(dw*deploylength1)+1)){

         deploy1 = currSolution[i][j][k];
         deploy2 = currSolution[ii][jj][kk];
         boolean isDifferent = compare(incompatibleList, deploy1, deploy2);

         if (isDifferent == true){
         incompatibleList[listTracker][0] = deploy1;
         incompatibleList[listTracker][1] = deploy2;
         listTracker ++;
         }
}}}}}}}}

for (int i = 0; i < r; i++) {
   for (int j = policyChange; j < T; j++) {
       for (int k = 0; k < L; k++) {
```

```
     if (currSolution[i][j][k] > 0){

      for (int ii = 0; ii < r; ii++ ){
      int deploy2 = 0;
       for (int jj = 0; jj < T; jj++ ){
          for (int kk = 0; kk < L; kk++ ){

      if ((currSolution[i][j][k] < currSolution[ii][jj][kk])&&(deploy2 <...
currSolution[ii][jj][kk])&&(jj-j<dw*(deployLength2)+1)){
          deploy1 = currSolution[i][j][k];
          deploy2 = currSolution[ii][jj][kk];

          boolean isDifferent = compare(incompatibleList, deploy1, deploy2);

          if (isDifferent == true){
          incompatibleList[listTracker][0] = deploy1;
          incompatibleList[listTracker][1] = deploy2;
          listTracker ++;
          }
    }}}}}}}}
return incompatibleList;
}

public static int[] columnCount(int [][][] array, int location){
int size = array[0].length;
int temp[] = new int[size];

for (int i = 0; i < array.length; i++){
            for (int j = 0; j < array[i].length; j++){
        if (array[i][j][location] != 0){
        temp[j] ++;}
            }}
return temp;
}

public static boolean compare (int[][] solution1, int solution2, int solution3) {

        boolean isDifferent = true;
        for (int i = 0; i < solution1.length; i++) {

        if ((solution1[i][0] == solution2)&&(solution1[i][1] == solution3)){
            isDifferent = false;
            break;
            }
        }
        return isDifferent;
```

```
}

// Sort the array by column 1
public static void sort1(int[][] theArray)
{
Arrays.sort(theArray, new Comparator<int[]>()
{
    @Override
    public int compare(int[] int1, int[] int2) {
        Integer numOfKeys1 = int1[0];
        Integer numOfKeys2 = int2[0];
        return numOfKeys1.compareTo(numOfKeys2);
    }
});
}

// Sort the array by column 2
public static void sort2(int[][] theArray)
{
Arrays.sort(theArray, new Comparator<int[]>()
{
    @Override
    public int compare(int[] int1, int[] int2) {
        Integer numOfKeys1 = int1[1];
        Integer numOfKeys2 = int2[1];
        return numOfKeys1.compareTo(numOfKeys2);
    }
});
}

public static int findFirstNum(int[][] array)
{
    int marker = 0;
    for(int i = 0; i < array.length; i++)
    {
        if (array[i][0]>0){
            marker = i;
            break;
        }
    }
return marker;
}}
```

Appendix I

Recursive Largest Fit (RLF) Algorithm

The Recursive Largest Fit (RLF) algorithm was first introduced by Leighton[63] and shown to be suitable for large-scale practical problems. It exhibits $O(n^2)$ time behavior, such that if k is the number of colors used, e is the number of edges, and n is the number of vertices, then $k \cdot e \approx n^2$. The RLF algorithm combines the strategy of a largest first algorithm and the structure of a worst-case behavior algorithm. Given $G = (V, E)$, the RLF graph coloring algorithm recursively assigns one color at a time until all vertices have been colored by evaluating the *degree*, or number of adjacent vertices, to each vertex. Leighton[63] claims that the Recursive Largest First (RLF) algorithm colors graphs with substantially fewer colors than other algorithms that do not involve interchanges.

The RLF algorithm assigns a color to the vertex with the maximum degree in G. Consider a set of vertices that are already colored c. Vertices are categorized in two sets, U_1 and U_2, where U_1 is the set of uncolored vertices that are not adjacent to any of the c_i colored vertices and U_2 is the set of uncolored vertices adjacent to at least one of the vertices colored c. The next vertex colored c is the vertex in U_1 with the highest degree of connections to vertices in U_2. Potential ties are subsequently broken by choosing the vertex with lowest degree of connections to vertices within U_1. When U_1 is empty the process is repeated with the next color, $c + 1$, until all vertices in G are colored. The RLF algorithm can be further described as follows as shown in the construct of Algorithm 4.

```
Set C = ∅, U_1 := V, U_2 = ∅, c := 0;
while |C| < |V| do
    if U_1 ≠ ∅ then
        Calculate the degree, d(v), of each vertex, v, in the subgraph of V';
        Increment c by 1;
        if d(v) > d(v') ∀ v, v' ⊂ U_1 then
            Assign color c to v;
            Move w from U_1 to C;
            Move all v' ∈ U_1 that are adjacent to v from U_1 to U_2;
        end
    else
        Evaluate |C|;
        Set U_1 := U_2, U_2 := ∅;
    end
end
```

Algorithm 4: RLF Algorithm

Appendix J

Java Code First-Fit Algorithm

```
import java.util.Random;

public class FirstFitAlgorithm {

    long startTime = System.currentTimeMillis();

    public static int [] U1;
    public static int [] U2;
    public static int [] E;
    public static int [] F;
    public static int [] dw;
    public static int [][] colorAssignments;

    public FirstFitAlgorithm(int[][] edges, int[][] locationTracker,
int numDeployments, int locations, int[][]deploymentDates, int d){

        int [][] colorAssignments = new int [numDeployments][3];
        int [] U1 = new int[numDeployments];
        int [] U2 = new int[numDeployments];
        int [] dw = new int[numDeployments];
        int size = edges.length;

        int [][] edges2 = new int[edges.length][2];
            //Make a copy of the original edges list; need to preserve original
            for (int i = 0; i < edges.length; i++){
                edges2[i][0] = edges[i][0];
                edges2[i][1] = edges[i][1];
        }

        //****Inititialization****
        for (int i = 1; i < numDeployments; i++){//set up Assignment list
            colorAssignments[i][0] = i;
            colorAssignments[i][1] = 0;
            U1[i] = 1;
            U2[i] = 0;
```

```
}

int counter = 0;
int color = 0;
int justColored = 0;
int justColoredLocation = 0;

//**** Coloring Iteration ************************************//
while (counter < numDeployments-1){
    for (int i = 1; i < numDeployments; i++){
    U1[i] = 1;
    U2[i] = 0;
    }
    //Identify first uncolored deployment;
    for (int i = 1; i < numDeployments; i++){
        if (colorAssignments[i][1] == 0){
            color++;
            colorAssignments[i][1] = color;
            counter++;
            justColored = i;
            justColoredLocation = locationTracker[justColored][1];
            colorAssignments[i][2] = justColoredLocation;
            break;
        }
    }
boolean emptyU1 = false;

while (emptyU1 == false){
for (int i = 1; i < numDeployments; i++){//set up Assignment list

    boolean adjacent = false;

    //Already colored
    if  (colorAssignments[i][1] > 0){

         U1[i] = 0;
         U2[i] = 0;
    }

    //Determine if i is adjacent
    if  (colorAssignments[i][1] == 0){//Not colored
        for (int j = 0; j < size; j++){
            if ((edges2[j][0] == i)&&(edges2[j][1] == justColored)){
            adjacent = true;
            }
            if ((edges2[j][0] == justColored)&&(edges2[j][1] == i)){
                adjacent = true;
```

```
            }//If uncolored and adjacent update U2 and E
            if (adjacent == true){
                U1[i] = 0;
                U2[i] = 1;
            }
            }
        }
}

// Delete the edges with nodes already colored by changing to zeros
for (int j = 0; j < size; j++){
    if ((edges2[j][0] == justColored)||(edges2[j][1] == justColored)){
           edges2[j][0] = 0;
           edges2[j][1] = 0;
    }
}

int temp = 0;

//Choose the first node in U1
for (int i = 1; i < U1.length; i++){
    if (U1[i] > 0){
    temp = i;
        break;
    }
}

// If true, then U1 is empty.  Re-Initialize
if (temp == 0){
    emptyU1 = true;

    //****Re-Inititialization****
        for (int i = 1; i < numDeployments; i++){
            dw[i] = 0;
            //Count the number of adjacent nodes to node i
            for (int j = 0; j < size; j++){
                if (edges2[j][0] == i){
                    dw[i]++;
                }
                if (edges2[j][1] == i){
                    dw[i]++;
                }
            }
            }
    break;
}
 justColored = temp;
```

```
    colorAssignments[justColored][1] = color;
    justColoredLocation = locationTracker[justColored][1];
    colorAssignments[justColored][2] = justColoredLocation;
    counter++;
   }
}

long totalTime = (System.currentTimeMillis()-startTime)/1000;

//Generate Locations per Unit statistic
int [][] locationCounts = new int [color][locations+1];
for (int i = 0; i < color; i++){
    for (int j = 1; j < counter+1; j++){
        if (colorAssignments[j][1] == i+1){
           locationCounts[i][colorAssignments[j][2]] = 1;
        }
    }
}
int[] unitStatistics = new int [color];
double aggregateSum = 0;
double aggregateStatistic = 0;
int maxLocations = 0;
for (int i = 0; i < color; i++){
    for (int j = 0; j < locations+1; j++){
        unitStatistics[i] = unitStatistics[i]+locationCounts[i][j];
    }
    aggregateSum = aggregateSum + unitStatistics[i];
     if (unitStatistics[i] > maxLocations){
        maxLocations = unitStatistics[i];
    }
}
   aggregateStatistic = aggregateSum/color;

   //Calculate Average BOG:Dwell Ratio
   double [] ratio = new double [color+1];
   double eligibleUnits = 0;
   double sumRatios = 0;
   double bogdwellAverage = 0;
   double minRatio = 99999;
   double maxRatio = 0;
   for (int i = 1; i < color+1; i++){
       int start = -1;
       int previousEnd=0;
       int end = 0;
       int totalDwell = 0;
       double count = 0;
       int eligible = 0;
```

```
            for (int j = 0; j < numDeployments; j++){
                if (colorAssignments[j][1] == i){
                    count++;
                    previousEnd = end;
                    start = deploymentDates[j][1];
                    end = deploymentDates[j][2];
                    if (count > 1){
                        totalDwell = totalDwell + (start-previousEnd);
                        eligible = 1;
                    }
                }

            }

            if (eligible == 1){
            ratio[i] = totalDwell/((count-1)*d);
            eligibleUnits++;
            sumRatios = sumRatios + ratio[i];
            if (ratio[i] > maxRatio){
                maxRatio = ratio[i];
            }
            if (ratio[i] < minRatio){
                minRatio = ratio[i];
            }
            }
        }
        bogdwellAverage = sumRatios/eligibleUnits;

    LocationSwap LocationSwap = new LocationSwap(colorAssignments, edges,
color, locations, deploymentDates, d);
    }
```

Appendix K

Java Code FFL Algorithm

```
import java.util.Random;

public class FirstFitLocationAlgorithm {

long startTime = System.currentTimeMillis();

public static int [] U1;
public static int [] U2;
public static int [] E;
public static int [] F;
public static int [] dw;

public static int [][] colorAssignments;


public FirstFitLocationAlgorithm(int[][] edges, int[][] locationTracker,
  int numDeployments, int locations, int[][]deploymentDates, int d){

        int [][] colorAssignments = new int [numDeployments][3];
        int [] U1 = new int[numDeployments];
        int [] U2 = new int[numDeployments];
        int [] dw = new int[numDeployments];
        int size = edges.length;

        int [][] edges3 = new int[edges.length][2];
            //Make a copy of the original edges list; need to preserve original
            for (int i = 0; i < edges.length; i++){
                edges3[i][0] = edges[i][0];
                edges3[i][1] = edges[i][1];
        }

        //****Inititialization****
        for (int i = 1; i < numDeployments; i++){//set up Assignment list
            colorAssignments[i][0] = i;
            colorAssignments[i][1] = 0;
```

```
    U1[i] = 1;
    U2[i] = 0;
}

int counter = 0;
int color = 0;
int justColored = 0;
int justColoredLocation = 0;

//**** Coloring Iteration ************************************//
while (counter < numDeployments-1){
    for (int i = 1; i < numDeployments; i++){
    U1[i] = 1;
    U2[i] = 0;
    }
    //Identify first uncolored deployment;
    for (int i = 1; i < numDeployments; i++){
        if (colorAssignments[i][1] == 0){
            color++;
            colorAssignments[i][1] = color;
            counter++;
            justColored = i;
            justColoredLocation = locationTracker[justColored][1];
            colorAssignments[i][2] = justColoredLocation;
            break;
        }
    }
boolean emptyU1 = false;

while (emptyU1 == false){
for (int i = 1; i < numDeployments; i++){//set up Assignment list

    boolean adjacent = false;

    //Already colored
    if  (colorAssignments[i][1] > 0){

         U1[i] = 0;
         U2[i] = 0;
    }

    //Determine if i is adjacent
    if  (colorAssignments[i][1] == 0){//Not colored
        for (int j = 0; j < size; j++){
            if ((edges3[j][0] == i)&&(edges3[j][1] == justColored)){
            adjacent = true;
            }
```

```
            if ((edges3[j][0] == justColored)&&(edges3[j][1] == i)){
                adjacent = true;
            }//If uncolored and adjacent update U2 and E
            if (adjacent == true){
                U1[i] = 0;
                U2[i] = 1;
            }
            }
        }
}

// Delete the edges with nodes already colored by changing to zeros
for (int j = 0; j < size; j++){
    if ((edges3[j][0] == justColored)||(edges3[j][1] == justColored)){
           edges3[j][0] = 0;
           edges3[j][1] = 0;
    }
}

int temp = 0;

//Choose the first node in U1 with same location, else first node
for (int i = 1; i < U1.length; i++){
    if ((U1[i] > 0)&&(locationTracker[i][1]==justColoredLocation)) {
        temp = i;
            break;
    }
}
//If there isn't node with the same location, choose the first node in U1
if (temp == 0){
    for (int i = 1; i < U1.length; i++){
        if (U1[i] > 0){
        temp = i;
            break;
        }
    }
}
// If true, then U1 is empty.  Re-Initialize
if (temp == 0){
    emptyU1 = true;

    //****Re-Inititialization****
        for (int i = 1; i < numDeployments; i++){
            dw[i] = 0;
            //Count the number of adjacent nodes to node i
            for (int j = 0; j < size; j++){
                if (edges3[j][0] == i){
```

```
                    dw[i]++;
                }
                if (edges3[j][1] == i){
                    dw[i]++;
                }
            }
            }
       break;
   }
    justColored = temp;
    colorAssignments[justColored][1] = color;
    justColoredLocation = locationTracker[justColored][1];
    colorAssignments[justColored][2] = justColoredLocation;
    counter++;
   }
}

long totalTime = (System.currentTimeMillis()-startTime)/1000;

//Generate Locations per Unit statistic
int [][] locationCounts = new int [color][locations+1];
for (int i = 0; i < color; i++){
    for (int j = 1; j < counter+1; j++){
        if (colorAssignments[j][1] == i+1){
           locationCounts[i][colorAssignments[j][2]] = 1;
        }
    }
}
int[] unitStatistics = new int [color];
double aggregateSum = 0;
double aggregateStatistic = 0;
int maxLocations = 0;
for (int i = 0; i < color; i++){
    for (int j = 0; j < locations+1; j++){
        unitStatistics[i] = unitStatistics[i]+locationCounts[i][j];
    }
    aggregateSum = aggregateSum + unitStatistics[i];
    if (unitStatistics[i] > maxLocations){
        maxLocations = unitStatistics[i];
    }
}
   aggregateStatistic = aggregateSum/color;

   //Calculate Average BOG:Dwell Ratio
   double [] ratio = new double [color+1];
   double eligibleUnits = 0;
   double sumRatios = 0;
```

```
        double bogdwellAverage = 0;
        double minRatio = 99999;
        double maxRatio = 0;
        for (int i = 1; i < color+1; i++){
            int start = -1;
            int previousEnd=0;
            int end = 0;
            int totalDwell = 0;
            double count = 0;
            int eligible = 0;
            for (int j = 0; j < numDeployments; j++){
                if (colorAssignments[j][1] == i){
                    count++;
                    previousEnd = end;
                    start = deploymentDates[j][1];
                    end = deploymentDates[j][2];
                    if (count > 1){
                        totalDwell = totalDwell + (start-previousEnd);
                        eligible = 1;
                    }
                }
            }

            if (eligible == 1){
            ratio[i] = totalDwell/((count-1)*d);
            eligibleUnits++;
            sumRatios = sumRatios + ratio[i];
            if (ratio[i] > maxRatio){
                maxRatio = ratio[i];
            }
            if (ratio[i] < minRatio){
                minRatio = ratio[i];
            }
            }
        }
        bogdwellAverage = sumRatios/eligibleUnits;

System.out.println("*****************************************************");
System.out.println("There are " + counter + " deployments and " + color
  + " colors");
System.out.println("There are " + size + " incompatible edges");
System.out.println("There is an average of "+ aggregateStatistic
+ " locations per unit");
System.out.println("The maximum number of locations for a unit is: "
+ maxLocations);
System.out.println("The average BOG:Dwell ratio is 1:" + bogdwellAverage);
System.out.println("The min BOG:Dwell ratio is 1:" + minRatio);
```

```
System.out.println("The max BOG:Dwell ratio is 1:" + maxRatio);
System.out.println("FFL Algorithm time: " + totalTime + " seconds");

LocationSwap LocationSwap = new LocationSwap(colorAssignments, edges, color,
locations, deploymentDates, d);
    }
```

Appendix L

Java Code Color Swapping

```
public class LocationSwap {

long startTime = System.currentTimeMillis();

int stakeUnit = 0;
int candidate1 = 0;
int candidate2 = 0;
int color1 = 0;
int color2 = 0;
int index = 0;
int iteration = 0;

public LocationSwap (int[][]colorAssignments, int[][]incompatibleEdges, int colors,
int locations, int[][]deploymentDates, int d){

for (int run = 0; run < 1000; run++){
 boolean terminator = true;
for (int i = 1; i < colorAssignments.length; i++){
 int location1 = 9999;
 int location2 = 9999;
 stakeUnit = colorAssignments[i][1];
 index = colorAssignments[i][0];
 color1 = colorAssignments[i][1];
 location1 = colorAssignments[i][2];
 int candidate1 = 0;

 //Find assignments for unit i at different locations
   for (int ii = 1; ii < colorAssignments.length; ii++){
     if ((colorAssignments[ii][1] == stakeUnit)&&(colorAssignments[ii][2] !=
  location1)){
        candidate1 = colorAssignments[ii][0];

 //Look at assignments for different units at the desired location
 for (int j = candidate1+1; j < colorAssignments.length; j++){
   if ((colorAssignments[j][1] != i)&&(colorAssignments[j][2] == location1)
```

```
     &&(colorAssignments[j][1] > stakeUnit)){
      candidate2 = colorAssignments[j][0];
      color2 = colorAssignments[j][1];
      location2 = colorAssignments[j][2];

 //Determine if a candidate is icompatible
 boolean compatible = true;
 for (int jj = 0; jj < incompatibleEdges.length; jj++){

  if ((incompatibleEdges[jj][0]==index)&&(incompatibleEdges[jj][1]==
candidate2)&&(candidate1 != candidate2)){
      compatible = false;
      break;
  }
     if ((incompatibleEdges[jj][0]==candidate2)&&
(incompatibleEdges[jj][1]==index)&&(candidate1 != candidate2)){
         compatible = false;
         break;
     }
}

//Check for incompatibility with previous swaps
for (int k = 1; k < colorAssignments.length; k++){
    if (colorAssignments[k][1] == color2){
        for (int kk = 0; kk < incompatibleEdges.length; kk++){
            if  ((incompatibleEdges[kk][0]==candidate1)&&
 (incompatibleEdges[kk][1]==k)&&(k!=candidate2)){
                 compatible = false;
                 break;
            }
            if  ((incompatibleEdges[kk][0]==k)&&
  (incompatibleEdges[kk][1]==candidate1)&&
(k!=candidate2)){
                 compatible = false;
                 break;
            }
        }
    }
 }
 //Check for incompatibility with previous swaps
 for (int k = 1; k < colorAssignments.length; k++){
    if (colorAssignments[k][1] == color1){
        for (int kk = 0; kk < incompatibleEdges.length; kk++){
            if  ((incompatibleEdges[kk][0]==candidate2)&&
    (incompatibleEdges[kk][1]==k)&&
(k!=candidate1)){
                 compatible = false;
```

```
                break;
            }
            if  ((incompatibleEdges[kk][0]==k)&&(incompatibleEdges[kk][1]
==candidate2)&&(k!=candidate1)){
                compatible = false;
                break;
            }
        }
    }
 }

     //Swap colors & locations
     if (compatible == true){
     colorAssignments[candidate1][1] = color2;
     colorAssignments[candidate2][1] = color1;
     terminator = false;
     break;
   }
 }}}}
 }
   iteration++;
       if (terminator == true){
           break;
      }
}

long totalTime = (System.currentTimeMillis()-startTime)/1000;

//Generate Locations per Unit statistic
int [][] locationCounts = new int [colors][locations+1];
for (int i = 0; i < colors; i++){
    for (int j = 1; j < colorAssignments.length; j++){
        if (colorAssignments[j][1] == i+1){
           locationCounts[i][colorAssignments[j][2]] = 1;
        }
    }
}
int[] unitStatistics = new int [colors];
double aggregateSum = 0;
double aggregateStatistic = 0;
int maxLocations = 0;
for (int i = 0; i < colors; i++){
    for (int j = 0; j < locations+1; j++){
        unitStatistics[i] = unitStatistics[i]+locationCounts[i][j];
    }
    aggregateSum = aggregateSum + unitStatistics[i];
    if (unitStatistics[i] > maxLocations){
```

```
        maxLocations = unitStatistics[i];
    }
}
   aggregateStatistic = aggregateSum/colors;

   //Calculate Average BOG:Dwell Ratio
   double [] ratio = new double [colors+1];
   double eligibleUnits = 0;
   double sumRatios = 0;
   double bogdwellAverage = 0;
   int minIndex = 0;
   double minRatio = 99999;
   double maxRatio = 0;
   for (int i = 1; i < colors+1; i++){
       int start = -1;
       int previousEnd=0;
       int end = 0;
       int totalDwell = 0;
       double count = 0;
       int eligible = 0;
       for (int j = 0; j < colorAssignments.length; j++){
           if (colorAssignments[j][1] == i){
               count++;
               previousEnd = end;
               start = deploymentDates[j][1];
               end = deploymentDates[j][2];
               if (count > 1){
                   totalDwell = totalDwell + (start-previousEnd);
                   eligible = 1;
               }
           }
       }

       if (eligible == 1){
       ratio[i] = totalDwell/((count-1)*d);
       eligibleUnits++;
       sumRatios = sumRatios + ratio[i];
       if (ratio[i] > maxRatio){
           maxRatio = ratio[i];
       }
       if (ratio[i] < minRatio){
           minIndex = i;
           minRatio = ratio[i];
       }
       }
   }
   bogdwellAverage = sumRatios/eligibleUnits;
```

```
System.out.println("************************************************");
System.out.println("Post-Swap ("+iteration+" iterations):
There is an average of "+ aggregateStatistic + " locations per unit");
System.out.println("The maximum number of locations for a unit is: "
+ maxLocations);
System.out.println("The average BOG:Dwell ratio is 1:" + bogdwellAverage);
System.out.println("The min BOG:Dwell ratio is 1:" + minRatio + "for color"
  + minIndex);
System.out.println("The max BOG:Dwell ratio is 1:" + maxRatio);
System.out.println("Swapping time: " + totalTime + " seconds");
}
```