# Modeling Aspects of Military Readiness

by

Jonathan Paynter

Submitted to the Sloan School of Management
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Operations Research

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2022



Author .....
Sloan School of Management
April 29, 2022

Certified by .....
Retsef Levi
J. Spencer Standish (1945) Professor of Management
Thesis Supervisor

Accepted by .....
Georgia Perakis
William F. Pounds Professor of Management Science
Co-director, Operations Research Center

**Modeling Aspects of Military Readiness**

by

Jonathan Paynter

Submitted to the Sloan School of Management
on April 29, 2022, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Operations Research

## Abstract

During peacetime, military performance assessment focuses on combat readiness. This thesis focuses on applying tools from operations research to inform and optimize strategic design decisions as well as operational decisions related to military readiness. In particular, we use a variety of optimization techniques to determine how to enhance equipping and personnel readiness and to quantify important trade-offs between personnel readiness and leader development.

Chapter 2 focuses on helicopter maintenance scheduling and is motivated by Department of Defense investment in predictive analytics for component health. We develop an index-style decision policy for integrating signal-based pre-emptive component repairs with the recurring time-based preventive maintenance tasks for the overall, multi-component system. The results highlight that the predictive model generating the component health signal must have exceptionally low false positive rates, 5% or less for use-case settings, or the pre-emptive repair decision policy will actually hurt equipment readiness. Chapter 3 models the impact of career path design policy on personnel readiness. To develop leaders for future assignments, the military implements career path design policy that restricts the sequencing and timing of an individual's assignments. Overly restrictive policy can hurt personnel readiness even when the overall system has enough personnel for every assignment. We develop a mixed integer linear programming formulation and a column-generation inspired algorithm to determine specific changes to the career path design policy that enhance readiness. For a specific U.S. Army officer career field we show how a small change in career path design policy can provide a 9% increase in personnel readiness. Chapter 4 considers the U.S. Army's recently updated assignment process that includes a matching market for the thousands of officers moving to new jobs every year. When there are more available jobs than officers, a personnel manager assesses personnel readiness to decide which jobs enter the market, and then assignments are determined by a deferred acceptance algorithm to maximize applicant satisfaction. We develop a mixed integer formulation that combines these decisions and can be used to generate a Pareto frontier between personnel readiness and applicant satisfaction. Then, we develop a tractable solution approach for finding an approximate Pareto frontier

using a local search algorithm. We use data from the U.S. Army's 2020 assignment market to show how a 2% decrease in readiness provides room for a 10-20% increase in officer assignment satisfaction.

Thesis Supervisor: Retsef Levi
Title: J. Spencer Standish (1945) Professor of Management

# Acknowledgments

I would not be successful without the wonderful support of my wife, Julie. Her ability to calmly multi-task through the day-to-day demands of raising a family, having a baby, the pandemic, and teaching her pre-calculus and calculus students leaves me in awe, every day. Thank you.

I am indebted to Retsef for his commitment as my advisor. In particular, his ability to help me get to the precise heart of a problem, and his ability to push me to explore it to the fullest. You have set the example for me on intellectual curiosity and on how to constructively push an advisee to dig deeply into a problem. Thank you.

I am grateful to the West Point Department of Mathematics for selecting me to come teach - I'm looking forward to it!

The maintenance team at the 160th SOAR provided invaluable data and countless hours of background information to help me understand the basics of aviation maintenance, their units' data goals, and their intended use of a predictive health model. In particular Glen Cassle and Jason Slusser.

I am grateful to Ike Zeitler for all of his work in Army personnel analysis that motivated the career path policy chapter; in particular his focus on the 'KD queue' and its practical importance to the Army. Additionally, I am very grateful for the data support from the Office of Economic and Manpower Analysis at West Point. Your willingness to provide data facilitated the personnel-related chapters.

To everyone who helped me - during coursework, with coding, and everything else - thank you.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

The military can't measure its success or failure on a profit and loss statement. Outside of active combat, measuring success means measuring readiness. Generally, readiness refers to the ability of the military to promptly shift from a peacetime posture to successfully executing its wartime missions. When General Milley became the Chief of Staff of the United States Army in 2015, his initial guidance to the force was "Readiness for combat is our No. 1 priority, and there is no other No. 1" [50]. It is therefore not surprising that readiness is also the first line of effort in the U.S. National Defense Strategy [57], it is a significant focus of research at the RAND Corporation, one of the military's frequent research partners (such as in [33]), and there are readiness-focused regulations, such as *Army Strategic and Operational Readiness* [59].

There are many ways to define and measure readiness. Some are more precise and quantitative and others are holistic and qualitative. The United States Army describes readiness with four functional pillars: personnel, training, equipping, and leader development [61]. Personnel readiness refers to having the right number of correctly-skilled personnel in an organization. Training readiness refers to gaining experience practicing core tasks. Equipping readiness refers to the quantity of fully operational equipment. Leader development refers

to recruiting, retaining and educating military personnel for jobs now and in the future.

Regulatory guidance in the U.S. military specifies readiness reporting requirements, including some established by Congress [1]. These reporting requirements define specific readiness measures and how those measures are gathered and consolidated from the force. For example, AR 220-1 specifies a training measurement based on the percent of the unit's core tasks that are fully or partially trained [58]. Some research exists on the use and combination of these readiness measures. For example, work on equipment availability [39], attempts to unify many measures into a single statistic [32], and introducing new timing aspects [52].

Maintaining readiness at the unit level requires constant attention on assigning personnel to jobs, scheduling maintenance, and synchronizing personnel and equipment with the training schedule. These unit-level operational decisions are shaped by strategic design decisions about leader development and equipment modernization. For example, to develop leaders, individuals progress through positions of greater responsibility, which means individuals move to new assignments every two to three years. This aspect of design means units must continuously work to find personnel that are good fits for their open jobs. Another example is equipment readiness where units work to keep their current equipment operational, but also must have maintenance teams and operators available to field new equipment as updated technology becomes available. The design decision about fielding new equipment or deploying new technology into a unit shapes the unit's ability to maintain its current equipment.

This thesis focuses on applying tools from operations research to inform and optimize strategic design decisions as well as operational decisions related to military readiness. In particular, we use a variety of optimization techniques to determine how to enhance equipping and personnel readiness and to quantify important trade-offs between personnel readiness and leader development. These provide managerial insights and specific recommendations to support military leaders making core decisions related to operational maintenance and personnel planning.

## 1.1 Thesis Overview

### The Value of a Predicted Fault in Maintenance Planning

**Background.** The Department of Defense (DoD) spends more than 200 hundred million dollars annually on military predictive analytics [9], [20]. One of the key initiatives is aircraft predictive maintenance, where machine learning models trained with large amounts of on-board sensor data provide early warning of a failing component. The early warning can drive decisions about part ordering, selecting aircraft for tasks, or pre-emptively repairing the failing part. The Army's 160th Special Operations Aviation Regiment worked with the DoD's artificial intelligence team to develop a predictive analytics model for a certain type of problem that can occur on the engines of an MH-60 helicopter. For implementation, the aviation maintainers have to integrate this new health signal generated by the predictive analytics model into their existing operational process for executing recurring, time-based preventive maintenance and repairing broken components.

**Research Questions.** Chapter 2 focuses on equipment readiness and considers a group of systems that share maintenance resources and require scheduled, system-level preventive maintenance. Maintainers have a health signal for one component on each system that is generated by a predictive analytics model. There are time efficiencies from combining a component repair with system-level preventive maintenance. The chapter addresses two research questions: (i) When should maintainers execute a pre-emptive repair based on a health signal? (ii) How good does the component health predictive model providing that signal need to be to provide real benefits from the efficiencies of combining maintenance activities?

**Methodology.** We model the setting through a Partially Observable Markov Decision Process (POMDP) that captures the stochastic nature of component failures, the shared maintenance resources and mission demands, and the partial observability of one component via a health signal. The reward function connects directly to the military's primary

equipment readiness metric which tracks the percent of systems (helicopters) that are fully operational each day. However, the POMDP is intractable as the state space is exponential in the number of systems in the group.

We leverage an approach from Multi-Armed Bandits where we allocate maintenance resources in each period to the different systems to maximize the expected number of operational systems. For tractability, we approximate the model by treating the systems in the group as identical, where the transition characteristics of each system are the same, and we apply the two group constraints in expectation. This allows us to decouple the decisions between systems and reduce the state-space under consideration. We then use a heuristic that leverages the reduced costs from a linear programming formulation of this approximated setting for a single system. This heuristic provides an index-style decision policy that is simple for maintenance teams to implement, and incorporates the health signal for a failing component. The performance characteristics of the predictive analytics model that provides the component health signal are inputs to the POMDP and heuristic solution approach. We vary these performance characteristics to numerically determine how good the predictive analytics model needs to be for there to be an operational impact.

**Data and Results.** We validate the approach with data on a fleet of MH-60 helicopters operated by the U.S. Army's 160th Special Operations Aviation Regiment that includes information on maintenance timing, equipment failures, and interviews with the maintenance team on operational practices. Our resulting policy provides insights on where signal-based pre-emptive component repairs fit into the existing time-based preventive maintenance framework. For these types of pre-emptive repairs we show that the predictive model providing the health signal must have very few false positives for their to be any improvement in readiness. For use-case parameters, the false positive rate must be below 0.05. The DoD's currently developed predictive analytics model for this specific use-case is not ready for operationally useful implementation that is focused on pre-emptive repairs.

**Contributions.** This chapter makes two contributions. First, we develop an optimization

model for a group of systems that includes partial observability of one component's health. We apply an approximation method to determine an index-style policy focused on the use of the component health signal for deciding when systems enter maintenance. Second, the solution approach allows us to numerically determine the minimum performance level of the health signal's source predictive model for there to be value from executing signal-driven pre-emptive component repairs. Crucially, this minimum quality level can vary for different system states, and this provides insight into different minimum performance thresholds for different pre-emptive repair situations. For the military, this model can assist units in developing updated maintenance practices, and can inform predictive analytics development efforts on the required quality of component-level health predictions.

## Career Path Design Policy and Military Personnel Readiness

**Background.** Personnel planners ensure that the military has the right number and composition of people to meet its workforce needs. Because all military advancement is internal, personnel planners also set policy that shapes the career path of each individual. This ensures the talent pool has the skills to meet future talent demands. Frequently, career path design policy imposes requirements where more individuals need the same type of job at the same time than there are jobs of that type. When this happens, other jobs remain unfilled. This hurts personnel readiness, which is the organization's ability to assign individuals to jobs. This tension between readiness and leader development is evident in U.S. Army officer manning, where there are more than enough officers for every job, but unfilled positions persist.

**Research Questions.** Chapter 3 considers how guidance on career path design policy can limit personnel readiness. It addresses two research questions: (i) What is the impact of career path design policy on personnel readiness? (ii) How can we add flexibility to the career path design policy to improve personnel readiness so that units are better postured to accomplish their missions?

**Methodology.** We model the operational aspects of military career management as a flow on a specially designed graph. Grouping a specialty's jobs by type, we form a time-expanded graph where there is a node for each job type and each period of time and the number of jobs of each type that need to be filled is captured through the demand at the respective node. The supply captures the personnel traveling along designated 'career' paths in the graph, thus satisfying the demand of the job nodes along the path. A job-type's readiness is the fraction of the jobs that have personnel assigned from any of the career paths. We then formulate a linear program that decides on the fraction of the total available personnel assigned to each of the career paths, limited by the number of available jobs within a certain type, to maximize personnel readiness. When the maximum personnel readiness is lower than what it could be without restricting the available career paths, we know career path design policy has decreased readiness. To address the second question, we model a process where personnel leaders adjust career path design policy and then an execution agency maximizes personnel readiness. As an exact approach, we formulate a bi-level mixed integer program where $k$ additional career paths that are not currently allowed are selected in the upper level, and then personnel readiness is maximized in the lower level using the already allowed career paths and the newly selected ones. For a more tractable approach, we develop an algorithm motivated by column generation that iteratively selects $k$ additional paths, within some allowable adjustments to professional development guidance. We then maximizes readiness using the already allowed and $k$ additional paths.

**Data and Results.** We leverage data on the assignment histories of thousands of active-duty U.S. Army officers as a use-case for applying our model, and to provide insight for Army personnel leaders. In the use-case, we describe the conditions for when career path design policy limits readiness, which heavily depend on the types of jobs available to officers, and we characterize specific career path adjustments that could improve readiness. For a specific officer specialty we consider, career path design policy limits readiness by more than 9%. The career path design policy adjustments needed to remove that 9% readiness gap adjust the personnel cohort allocation by sending 25% of officers on a new career path that includes

the same types of assignments, but with different sequencing. Computationally, the iterative algorithm scales well to use-case size. On smaller, synthetic data where the mixed integer approach is tractable, with $k = 4$, the iterative algorithm captures 90% of the readiness improvement found by the mixed integer approach more than 75% of the time.

**Contributions.** This chapter makes two key contributions. First, we develop new types of models to capture and support different decision levels in military personnel planning and operational management focused on quantifying the impact of career path guidance on personnel readiness. The key idea is connecting military career paths used in practice to network paths in our model. This includes using personnel assignment histories to produce a data-driven view of the actually-implemented career path design policy. Second, algorithmically, we develop a bi-level mixed integer formulation that augments an existing set of paths with $k$ additional paths, while allocating resources across those paths to minimize cost. We develop an iterative algorithm for finding the $k$ additional paths with an approach motivated by column-generation that avoids enumerating all possible paths.

## Bi-objective Matching with Market Composition Control

**Background.** The U.S. Army recently adopted a matching market for the thousands of officers moving to new jobs every year. When there are more available jobs than officers, a personnel manager decides which jobs enter the market, based on personnel readiness. Then after preferences are gathered, assignments are determined by a deferred acceptance algorithm to maximize applicant satisfaction. The market-based approach provides agency to moving officers, and improved officer assignment satisfaction will hopefully improve retention, which is an aspect of leader development.

**Research Questions.** In a labor market where there are more jobs than applicants, only a subset of jobs, equal in size to the number of applicants, can be filled. We consider such a market, but where the central decision-authority controls which jobs enter the market and which remain unfilled. This presents an opportunity to consider two objectives. First,

readiness based on how well the subset of jobs chosen for market inclusion fits organizational staffing needs. Second, satisfaction based on applicant preferences for jobs. No single solution maximizes both objectives, and this chapter addresses two research questions: (i) how do we tractably find a set of solutions to this market that captures trade-offs between the two goals of readiness and applicant satisfaction on or near the Pareto Frontier? (ii) to what extent do we see useful trade-offs where a small decrease in readiness provides a large increase in satisfaction?

**Methodology.** We develop a bi-objective mixed integer linear formulation that determines both the job subset and subsequent stable matching. The optimization is driven by both the readiness achieved by the selected jobs and the resulting satisfaction from the matching. This formulation ensures a stable solution between the applicants and the selected job subset. Given this formulation, we wish to generate a set of solutions that form a Pareto frontier for readiness and satisfaction. To do so we solve the formulation with a variety of values for the objective scaling parameter that captures the importance given to satisfaction versus readiness. This formulation is tractable up to a certain market-size, but is intractable for large markets. We then develop an algorithm based on a local search procedure that we call the one-swap chain that runs in polynomial time, is tractable for large market sizes, and generates an approximate Pareto frontier.

**Data and Results.** We validate these approaches with preference data from the 2020 U.S. Army officer assignment marketplace. The key insight for Army personnel leaders is that, given the preference information, the job subset decision effectively determines both objectives as the ensuing matching algorithm is fixed. By considering the preference information when making the job subset decision, there is room to improve the overall satisfaction at little expense to readiness. For example, in a use-case market a 5% decrease in readiness allows for a 40% increase in assignment satisfaction.

**Contributions.** This chapter makes two main contributions. First, we model the important use-case of a market where a decision maker controls one side of the market make-up and

has two objectives, specifically readiness and satisfaction. Additionally, we provide practical computational approaches to solve large instances. First we use a mixed integer linear programming formulation and develop an associated weighted-sum algorithm that finds a set of solutions on the Pareto frontier for the two objectives. Second, we develop a polynomial-time local-search algorithm that leverages the stability requirement of the solution to iteratively swap jobs in the selected subset while improving the resulting satisfaction objective. We store the matchings found during this process and then find a set of these solutions that approximates the Pareto frontier. This approach is very tractable even for large problems.

## Generalizations

While our driving focus is on decision policies to improve specific measures of military readiness, the models and solution approaches developed in this thesis apply more generally. The work in chapter 2 applies to the maintenance of any group of systems which have existing time-based recurring maintenance requirements where we have access to a health signal for a component on each system. This includes most companies that operate fleets of vehicles, particularly when the availability of that fleet is a key interest in their maintenance scheduling, as this aligns with the equipping readiness we consider. The work in chapter 3 on personnel readiness and career path design policy is more niche to the military, as a defining aspect of military personnel management is a focus on recruiting entry-level personnel and developing them internally. Some other hierarchical organizations, such as large police departments, share this characteristic. More generally, the approach from chapter 3 captures the allocation of resources to tasks, where the allowable sequences of tasks are predefined, there are limits on the number of resources that can be assigned to a task, and we wish to minimize the cost of uncompleted tasks. We can then consider increasing the flexibility of the allowable set of task sequences, by adding additional resource paths to further minimize cost. The algorithmic approach we develop for adding paths provides a method that eliminates the need for path enumeration. The work in chapter 4 applies to any labor market where there is some initial market composition decision on the jobs that can enter the market. The

model and solution approaches in the chapter provide a way to quantify trade-offs between competing organizational goals in a market. One example where this could apply is in school choice markets when a school district is considering expanding the capacity of some schools.

# Chapter 2

# The Value of a Predicted Fault in Maintenance Planning

## 2.1 Introduction

Many systems require recurring preventive maintenance to ensure they remain effective and safe to operate over time. Typically preventive maintenance occurs after a certain number of operational hours, or other related measures such as the number of certain activities since the last maintenance. As part of scheduled preventive maintenance, operators take the system out of use to conduct a variety of inspections and small repairs before returning the system to operation. The list of tasks is preset and based upon known preventive needs of the many individual components that make up the system.

When managing a group of systems simultaneously (e.g., a fleet of aircraft), recurring maintenance for each of the individual systems must be synchronized because operational demands and limited maintenance resources for the group are shared across systems. Both preventive maintenance and any unscheduled maintenance share the resources needed to repair broken components. Maintainers and operators coordinate each period on which systems are needed

for operations and which systems are in maintenance.

Many modern systems such as aircraft are increasingly equipped with sensors that provide very granular data on the performance of different components of the system. These data provide an opportunity to use predictive analytics tools and models to track the health of various system components, as in Figure 2-1. For example on a helicopter, a predictive analytics model could combine sensor data on engine operating temperature, hydraulic pressure, and other system aspects to predict the health of a transmission. For example, if the transmission will fail in next 10 operating hours. This predictive model could inform key operative decisions, for example: when to order replacement parts, when to conduct early inspections, or when to conduct pre-emptive component repairs. Ultimately, with sufficient sensor data from the system, data on the respective historical performance of the system, and reliable predictive analytics tools, a system's preventive maintenance could transform from being time-based to being solely based on health signals, which prevents unnecessary downtime. However, for complex multi-component systems, this would require health signals for every component.


Figure 2-1: Some systems have many sensors that provide data while the system operates. This data can be used in predictive analytics tools and models to predict the health of different components on the system. The output of a predictive analtyics model is a health signal, which is an input to the maintenance process.

In the immediate future, component health predictive analytics tools and models will become more available, but not sufficiently wide-spread to allow a shift to purely predictive, signal-based maintenance. It is expected that in the near future time-based, scheduled preventive maintenance for each system will continue, but it would potentially be possible to leverage system sensor data to create predictive analytics tools for specific components. For example, maintainers could use a prediction that a certain component is failing to trigger a pre-

emptive repair. This pre-emptive repair could occur in one of several ways. It could occur in conjunction with the already scheduled preventive maintenance, it could cause preventive maintenance to begin early to combine it with a pre-emptive repair, or the pre-emptive repair could occur as a stand-alone action.

As a specific motivating example, which will be further used subsequently to provide numerical illustration based on real data, consider a U.S. Army aviation unit that has a fleet of MH-60 helicopters and needs to meet daily operational demands for flying missions. The maintenance team has two categories of tasks. The maintainers conduct preventive maintenance for each helicopter every 40 flight-hours, and they repair or replace any broken components from the helicopter. This aviation unit is one of the first in the Army to field a machine learning-based component health predictive analytics model, which is focused on a type of problem that can occur in the helicopters' engines. When the health signal generated by the predictive analytics model indicates that a helicopter has a degraded engine, the maintainers can execute a pre-emptive repair. Depending on the helicopter's current flight hours status, the maintainers might have the opportunity to execute the pre-emptive repair in one of three ways. They could combine the pre-emptive engine repair with already scheduled system-level preventive maintenance, they could start the preventive maintenance earlier than planned and combine it with the pre-emptive engine repair, or they could execute the pre-emptive engine repair as a stand-alone action. Leveraging the predicted health status about the engine could then help the unit reduce how long helicopters are non-operational because of maintenance, particularly through creating efficiencies from combining maintenance activities. We note that for this particular type of engine degradation, there is not an increased safety risk, but it could prevent the use of the helicopter in an unplanned manner. As the component health predictive model is a new addition to the unit, there are not well understood procedures for incorporating the engine health signals or assessing how the quality of the predictive analytics model might impact the use of the signal.

To explore the integration of component-level predictive analytics with existing system-level

scheduled preventive maintenance, this chapter addresses two research questions: (1) When should maintainers execute a pre-emptive repair based on a health signal? (2) How good does the component health predictive model providing that signal need to be to provide real benefits from the efficiencies of combining maintenance activities?

To answer these questions, we consider a maintenance schedule for a group of systems where each system requires scheduled periodic preventive maintenance. Additionally, there is a single component on each system for which we have a binary health signal generated by a predictive analytics model. Maintenance planners make one of five decisions for each system, each period: operate, rest, execute preventive maintenance for the multi-component system, repair a single component, or execute the combination of preventive maintenance and a component repair. Resource constraints limit the number of systems that can be maintained during any period, and operational demands require a certain number systems to operate each per period.

We model the setting through a Partially Observable Markov Decision Process (POMDP) [45]. Four aspects of the practical problem motivate this modeling approach. First, we have a number of different systems which stochastically transition over time based on the period, state, and chosen action. Second, we have an operational demand and a maintenance capacity constraint that couple the decisions across the different systems. Third, the true health of one component from each system is only partially observable via a signal received from a specified predictive analytics model. Fourth, the performance of the group of systems is tracked by a daily status on the number of operational systems, which translates neatly to a reward function. We show that the decision to combine pre-emptive component repair with systems level preventive maintenance is monotone with respect to state elements. Specifically, these parameters are the 'time until the next scheduled preventive maintenance' and our belief probability of the true state of the component's health. However, the resulting POMDP is intractable as the state space is exponential in the number of systems in the group.

We leverage an approach based on the well-known Multi-Armed Bandits modeling framework, which considers the sequential allocation of resources to competing projects to maximize an expected reward [31]. In this setting, we allocate maintenance in each period to the different systems to maximize the expected number of operational systems. For tractability, we approximate the model by treating the systems in the group as identical, where the transition characteristics of each system are the same, and we apply the two group constraints in expectation. This allows us to decouple the decisions between systems and reduce the state-space under consideration. We then use a heuristic that leverages the reduced costs from a linear programming formulation of this approximated setting for a single system. This heuristic provides an index-style decision policy where there is a score (index) for each combination of state and action. Resources are allocated to the system-action pair with the highest score. For the use-case, this means each helicopter gets a score, based on its current state, for each possible action. If there are maintenance resources available, we select the helicopter and type of maintenance with the highest score. This index policy is simple for maintenance teams to implement, and incorporates the health signal for a failing component.

Additionally, by varying the performance characteristics of the predictive analytics model that provides the component health signal, we can then numerically determine how accurate the predictive analytics model needs to be. We apply the approach with maintenance data from a U.S. Army aviation unit. For use-case parameters, the false positive rate of the component health predictive model must be less than 0.05 for the predictive model to effectively affect the decisions. The DoD's currently developed predictive analytics model for this specific use-case does not have sufficient performance to improve readiness with signal-indicated pre-emptive component repairs.

### Contributions

This chapter has two key contributions, one with respect to each of the research questions. To answer research question (1) we develop a model that captures the key aspects of this setting, specifically the integration of system-level preventive maintenance decisions with

signal-based pre-emptive component repair. The setting includes fleet-level constraints, the stochasticity of part failure, the synchronization of system operation with both types of maintenance over multiple periods, and the partial observability of the component's health, via a noisy input signal. We then present an approximation technique that provides an index-style decision policy that is simple for real-world implementation, and leverages the predictive health signal for pre-emptive component repairs in certain settings.

To answer research question (2), the solution approach allows us to numerically determine the minimum performance level of the health signal's source predictive model for there to be value from executing signal-driven pre-emptive component repairs. Crucially, this minimum quality level can vary for different system states, and this provides insight into different minimum performance thresholds for different pre-emptive repair situations. For pre-emptive repairs to decrease fleet downtime, the quality of the predictive input signal must be exceptionally high, with very few false positives.

The model provides operational insights for maintainers and predictive health model developers. At the operational level, if a predictive health model is of high enough quality, then the decision policy developed here informs maintainers on when to act on the predictive component health signal by combining a pre-emptive repair with preventive maintenance. The model and solution approach can inform predictive analytics development efforts on the required quality of component-level health predictions.

This model is applicable for any group of systems where recurring preventive maintenance is the norm, and system sensors provide enough input data for a component health predictive model. Fleet-based examples include military organizations with a variety of vehicle-types, delivery companies, and utility companies dispatching large numbers of repair trucks. Non-vehicle settings could include manufacturing operations with many similar production lines.

**Chapter Outline**

Section 2.2 describes related aspects of the maintenance literature. Section 2.3 presents the model setting and the POMDP that incorporates the decisions around when to conduct a pre-emptive repair based on a component health signal. Section 2.4 presents the two analytic results about the monotonicity of the optimal control policy. Section 2.5 presents the solution technique from the linear programming approximation to the Multi-Armed Bandit setting for our model. Section 2.6 describes use-case numerical results based on the maintenance characteristics of a fleet of MH-60 helicopters in a U.S. Army unit. These include a description of the index-style decision policy, and the necessary quality of the health signal's source predictive analytics model. Section 2.7 provides concluding comments, and supplemental material is available in the appendices about additional model components and background on the numerical use-case.

## 2.2 Related Literature

We focus on the connections between preventive maintenance, which occurs on a regular schedule, and signal-indicated component repair, which is executed when a component fails, for a group of systems. In [19], de Jonge and Scarf provide an updated survey of recent maintenance optimization work that covers single-component systems, multi-component systems, and a variety of degradation processes; see [21] for a summary of multi-component maintenance.

Our model takes as a given the recurring preventive maintenance schedule for the system. There is a large amount of work on determining these preventive maintenance schedules, where different components have different inspection frequencies, and tasks are grouped to determine an efficient schedule; see [14], [36], and [44] for examples. When there is an economic dependency between maintenance activities, then the combined repair time is less than the sum of the two separate repair times; this is also called an efficiency. See [21] by Dekker for a more complete description of dependency types.

Each system in the group considered here has a certain component whose health is indicated by noisy signal. A repair based only on this type of input signal is sometimes referred to as conditions-based maintenance. Conditions-based maintenance includes direct sensor observation of certain parameters on a component, like operating temperature, where maintenance decisions are made by comparing a parameter against a pre-specified threshold. Conditions-based maintenance also includes the use of machine learning-based models that leverage the large amount of collected data about a component from a variety of sensors to predict component health. This is typically separate from a preventive maintenance schedule for a multi-component system, although there are versions of conditions-based maintenance that consider a multi-component system where all of the components have health signals; see [15] for an example. Kim et al. use a partially observable system to look at optimal threshold replacement rules based on sensor data in [40] and [41]. In [47], Maillart et al. use sensors and a partially observable system to determine when to test, repair, or overhaul a failing component.

Much of the maintenance scheduling literature looks at a continually operating system, such as a server farm or industrial plant, where the system is operating anytime maintenance is not conducted. Some systems operate only at certain times, which provides an additional avenue for synchronizing operations and maintenance. For example, synchronizing an aircraft's flight schedule with its maintenance schedule. This is particularly true when operational decisions impact the timing requirements for maintenance, as is the case here. [17], [42], and [49] consider synchronizing operations and preventive maintenance for aircraft, but do not consider stochastically failing components. Similar work exists in other fleet-based domains, for example high-speed trains, as in [13].

## 2.3 Model Formulation

In this section, we introduce a discrete-time model for scheduling maintenance for this group of systems, where we are integrating system-level preventive maintenance with signal-driven

component maintenance. The binary health signal for the component comes from a predictive analytics model. First, we describe the model setting and dynamics that include full knowledge of each system's tracked component. This is the special case where the predictive analytics model that generates the health signal has 100% accuracy. We then present the complete model in the partially observable setting.

### 2.3.1 Modeling the Maintenance for the Group of Systems

There are $N$ systems indexed by $i$, each with an evolving time-dependent state $s_{it}$ within $S$. The state of each system includes elements about the system and about a specific component. In each period $t$, an action $u_{it}$ can be applied to system $i$, where $u_{it}$ is within $U$. Given $s_{it}$ the state in the subsequent period evolves according to a transition matrix $T(u_{it}, s_{it})$ (i.e., a function of both the state and action taken in period $t$). Each of the $N$ systems has one component whose degradation is of specific interest for scheduling, and we have access to a health signal about that component. The health signal originates from a predictive analytics model. Figure 2-2 depicts the setting.

#### $S$: States

The state of each system in the group has four elements. The need for recurring preventive maintenance is driven by the *operating hours* of the system since it last underwent preventive maintenance. At $T_H$ operating hours, the system must enter preventive maintenance. If the system is in *preventive maintenance*, then the system has a certain number of periods remaining before it is ready for operation. Preventive maintenance takes $T_P$ periods. The *component* of interest has one of three statuses: healthy, failing, or broken. A broken status means the component is non-operable, a failing status means the component emits an identifiable signal that it is deteriorating, and a healthy status means the component is operating as intended. If the system is in *component maintenance*, then the system has a certain number of periods remaining before the system is ready for operations. Component

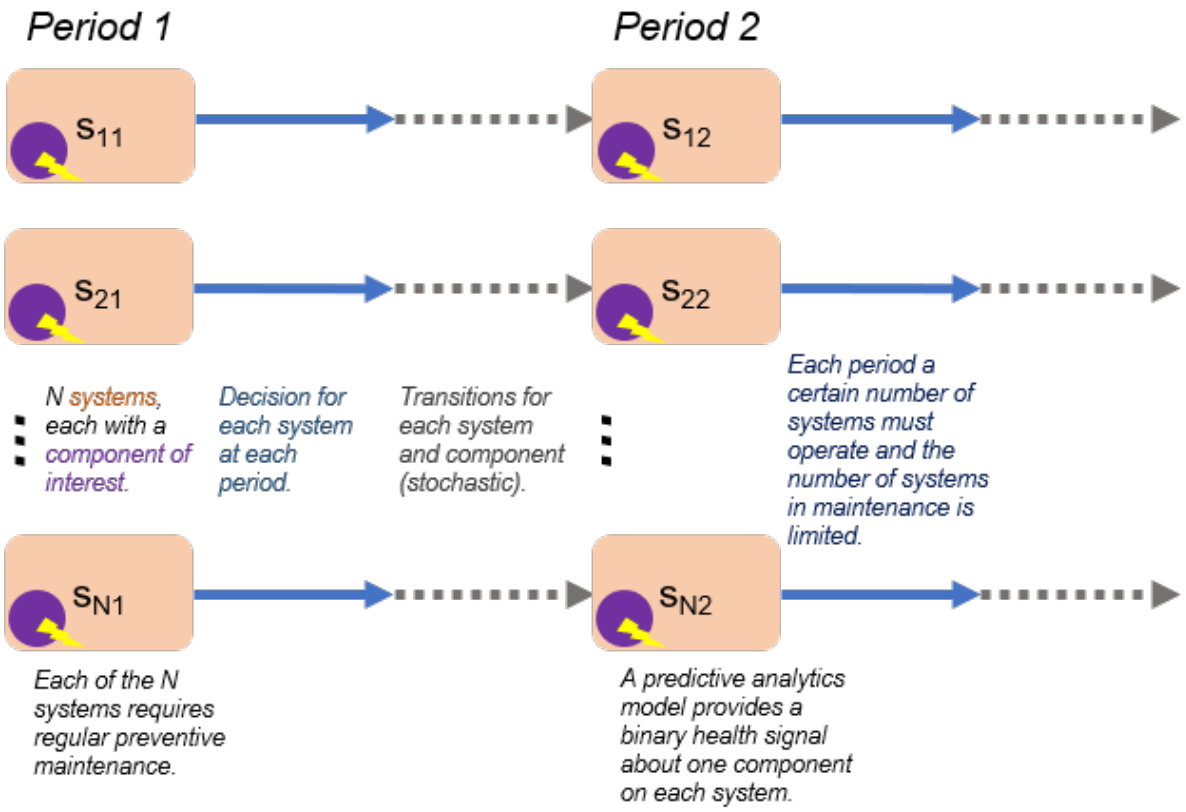


Figure 2-2: The group of $N$ systems (tan), each with a component of interest (purple circle) for 2 of the periods. Each system transitions between periods based on the prior state and action, and there are group constraints that couple the actions available across the various systems. Preventive maintenance for the system is driven by operating hours. Repairing the component of interest occurs when it breaks, or when we leverage the signal about component health and decide to pre-emptively repair it.

maintenance takes $T_C$ periods.

$$s_{it} = \begin{cases} Hours \in \{1, ..., T_H\}, & \text{Operating hours since system's last preventive maint.} \\ PerPr. \in \{0, ..., T_P + T_C\}, & \text{Periods remaining in preventive maintenance} \\ Comp \in \{H, F, B\}, & \text{Component health in one of three statuses} \\ PerCo. \in \{0, ..., T_C + T_P\}, & \text{Periods remaining in component maintenance} \end{cases}$$

### $U$: Actions

There are five possible actions for each system, each period: operate ($F$), rest ($R$, not operate), execute preventive maintenance ($M_P$), execute a repair on the component of interest ($M_C$), or execute the combination of preventive maintenance and a component repair($M_{PC}$). Which actions are available for a system depend on the state of that system.

For systems that are in preventive maintenance, with $PerPr > 0$, preventive maintenance must continue, and the only possible action is $M_P$. For systems that are in component repair, with $PerCo > 0$, the repair must continue and the only possible action is $M_C$.

For systems that are not in maintenance, preventive maintenance is due at $T_H$ operating hours and is allowed but not required up to $\delta$ hours prior to $T_H$. When a system's operating hours reach $T_H$, the only actions possible are $M_P$ and $M_{PC}$. When the system's operating hours are between $T_H - \delta$ and $T_H$, then the system can have any of the five actions. When the system's operating hours are less than $T_H - \delta$, then the possible actions are $F, R, M_C$. This applies our use-case requirements for executing system-level preventive maintenance.

Two group-level constraints couple the actions between the different systems, within each period. The limited maintenance capacity of the organization constrains how many systems can be in maintenance at any given time. So based on the number of systems in maintenance, we might have the option to begin maintenance on a certain number of additional systems, but we must always continue maintenance on any system where maintenance has already started. The daily mission demand also limits the number of systems that can rest or be in maintenance with a requirement for a certain number of systems to operate ($F$) each period. For our use-case, this models the aircraft fleet having a limited number of maintenance teams, and the need to generate a certain number of flights every day.

### $T$: Transitions

We consider the transition of a single system, $i$ as the transitions are the same for each. If the system rests, then no aspect of the state changes. When a system enters preventive

maintenance, the number of periods remaining in preventive maintenance, $PerPr$, increases to the known length of that type of maintenance, $T_P$. When a system enters component repair maintenance, the number of periods remaining in component repair, $PerCo$, increases to the known length of that type of maintenance, $T_C$. When a system enters combined maintenance ($M_{PC}$) both $PerPr$ and $PerCo$ increase. If there are no efficiencies gained from combining the system level preventive maintenance with the component repair, then both $PerPr$ and $PerCo$ increase to $T_C + T_P$. as executing the combined maintenance takes the same amount of time as executing the two actions sequentially. If there are efficiencies from combining maintenance, then $PerPr$ and $PerCo$ increase to $c(T_C+T_P)$, where $c \in [0, 1]$ is the efficiency parameter.

If a system is in preventive maintenance ($M_P$) or component repair ($M_C$), the periods remaining in maintenance decrease by one until it is again operational. When it exits preventive maintenance then its hours reset to 0. When it exits component repair its component status returns to healthy. These maintenance transitions model the aircraft maintenance practices in our use-case, including efficiencies from combining maintenance actions.

If the system operates ($F$), then its hours increase for a stochastic amount of time, and there is a stochastic transition of the component health. In Figure 2-3, we show the component health transition aspects with the true state in light blue and actions in dark blue. Transitions once an action is taken are in orange. When the component is in a healthy state there are two possible non-self transitions when the system operates. $P_{HB}$ is the probability of the component breaking, without any transition through a state that is observable. If there was always a possibility of identifying a failure in advance, then this probability would be 0. $P_{HF}$ is the probability of the component's health transitioning to a failing state, where we could identify an impending failure. It is related to the underlying reliability of the part, and we can estimate it from maintenance records on part failures. From the failing state there is one possible non-self transition when the system operates. $P_{FB}$ is the probability of breaking once a part begins to fail. This relates to the typical amount of time a part spends in the

failing state.

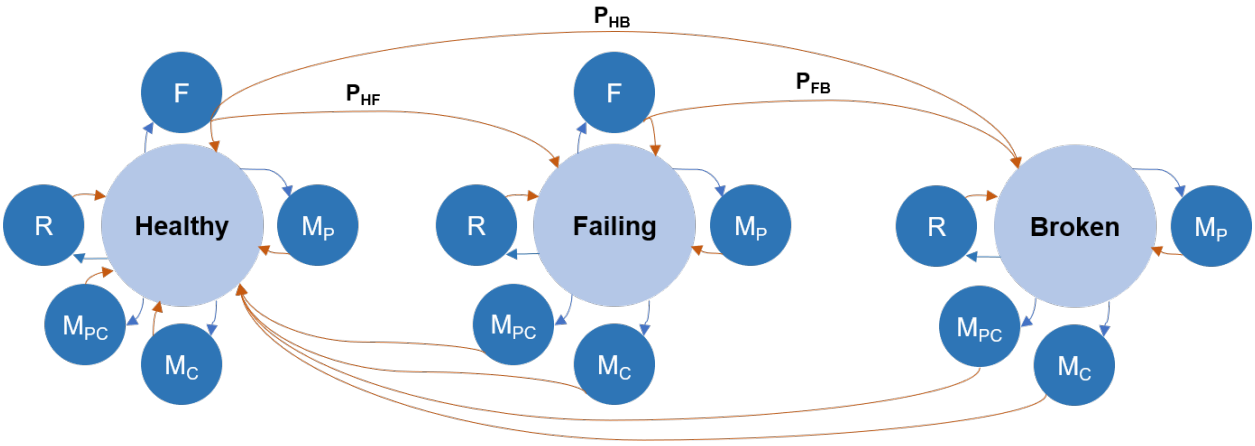


Figure 2-3: The underlying degradation process of each system's component of interest: $P_{HB}$ is the probability of transitioning from healthy to broken; $P_{HF}$ is the probability of transitioning from healthy to failing; $P_{FB}$ is the probability of transitioning from failing to broken. True states are light blue, the actions are dark blue, and transitions are orange. The identification of an impending failure is only possible once the state transitions from "Healthy" to "Failing". Actions include operate ($F$), rest ($R$), system-level preventive maintenance ($M_P$), component repair ($M_C$), and the combination of preventive maintenance and component repair ($M_{PC}$).

### $G$: Reward Function

Our reward function is based on the motivating organization's primary readiness metric, which is the proportion of systems available for operations each day. Consequently, if systems have a broken component or are in maintenance there is no reward, and the reward for the non-maintenance states is 1. We can further vary the reward function since we have knowledge of the component's health, with a value ($\delta$) for non-maintenance states with a failing component.

$$G(s) = \begin{cases} 1 & s \in \text{non-maintenance states with Healthy component} \\ \delta & s \in \text{non-maintenance states with Failing component, } \delta \in (0, 1] \\ 0 & s \in \text{maintenance states or states with Broken component} \end{cases}$$

With full knowledge of each system's component health, we can then define a Markov Deci-

sion Process (MDP) with the tuple $< S, U, T, G >$.

### 2.3.2 Group of Systems with Partially Observable Component Health

We cannot observe the true state of each system's component health unless it is broken, but we do receive a signal indicating if it is healthy or failing. $\Omega = \{Healthy, Failing\}$ is a finite set of observations we can receive about the component. $O$ is the observation function that provides, for each action and resulting state, a probability distribution over the possible observations. $O(s', u, o) = P(o|s', u)$ is the probability of observing $o$ given action $u$ and state $s'$.

This gives us the complete POMDP, $< S, U, T, G, \Omega, O >$. We now transform that into a belief MDP, $< S', U, T', G' >$ where we update the state space to track the probability of being in a certain state, and correspondingly update the transition and reward functions.

#### $S'$: State Space with Partially Observable Component Health

We now have an uncertain state space, and maintain a belief, $b$, for the probability distribution of the component health element, $\{H, F, B\}$. Breaks are fully observable, so $b(B) \in \{0, 1\}$. When the component is not broken, we have a probability distribution that captures the perceived likelihood to be in either of the states healthy, $H$, or failing, $F$. We track $b(F)$ in the state definition. For each system at each period, the resulting state, is:

$$s'_{it} = \begin{cases} Hours \in \{1, ..., T_H\}, & \text{Operating hours since system's last preventive maint.} \\ PerPr. \in \{0, ..., T_P + T_C\}, & \text{Periods remaining in preventive maintenance} \\ b(B) \in \{0, 1\}, & \text{Probability of comp being Broken} \\ b(F) \in [0, 1], & \text{Probability of comp being in a Failing status} \\ & \textit{When } b(B) = 1,\ b(F) = 0 \\ PerCo. \in \{0, ..., T_C + T_P\}, & \text{Periods remaining in component maintenance} \end{cases}$$

### $T'$: Transitions with Partially Observable Component Health

The transition function remains the same for all state elements except component health. Since the component health is only partially observable, instead of updating the true state, we update our belief of the true state based on the component health signal. To compute the updated belief states, $b'(F|o)$, we need to know the underlying component transition probabilities $p_{HH}, p_{HF}, p_{HB}, p_{FF}, p_{FB}$, (see Figure 2-3). Additionally, we need to know the observation probabilities for the health signal, $P(o = Failing|F)$, $P(o = Healthy|H)$. The component health signal comes from a predictive analytics model that uses all of the available system sensor data up to the current point in time. The current state's probability that the component is failing comes from the same underlying information, so the updated probability that the component is failing does not need to depend on both the signal and our prior belief. The updated probability that the component is failing depends only on the observation function, with $b'(F) = P(Comp = F|o, b) = P(Comp = F|o)$.

### Incorporating the Performance of the Predictive Analytics Model that Generates the Health Signal

The health signal comes from an exogenous binary predictive model, based on sensor data, that attempts to determine if a component will break in the next $J$ operating hours, which is equivalent to a binary prediction of a failing state now. So, we can connect the binary prediction's testing performance to the key input parameters we need for this model with the observation function and belief updates. We use four input parameters from the binary prediction's testing performance model for our partially observable model. See [53] for a review of prediction model performance metrics.

$$
\begin{aligned}
P(o = Failing|F) &= \alpha, \text{ sensitivity} \\
P(o = Healthy|H) &= \beta, \text{ specificity} \\
b'(F|o = Failing) &= P(Comp = F|o = Failing) = \zeta, \text{ precision} \\
b'(H|o = Healthy) &= P(Comp = H|o = Healthy) = \eta, \text{ negative predictive value}
\end{aligned}
$$

We can directly incorporate the quality of the binary prediction model that generates the health signal with these parameters. The fact that the belief state updates completely based on the health signal limits $b'(F)$ in this model to one of the three values in $\{0, \zeta, 1-\eta\}$.

### $G'$: Reward Function with Partially Observable Component Health

Our revised reward function that accounts for the belief distribution takes an expectation of $G(S)$ with respect to the uncertain component health. The total reward for a state is the sum of each system's reward. $G'(S_t) = \sum_i G'(s_{it})$.

$$G'(s_{it}) = \begin{cases} 1 - b(F)(1-\delta) & s \in \text{non-maintenance, non-broken states} \\ 0 & s \in \text{maintenance states or states with Broken component} \end{cases} \tag{2.1}$$

### The Complete Model

The complete model is the belief MDP, $< S', U, T', G' >$. This includes the revised state space to account for the uncertain component health, revised transition function to account for the observation function and the belief updates, and the revised reward function that takes an expectation over the base reward function. Our goal is to maximize the discounted expected reward, $V(s) = G(s) + \max_u \gamma \sum_{s' \in S} T(s, u, s')V(s')$.

In some systems, the probability of a component failure might vary based on the age of the component. We address incorporating this aspect into the model in Appendix A.1. The component degradation process on each system naturally extends to additional degradation states for the component. However, for the remainder we stay with three component health states, on the premise that we can estimate the transition probabilities from the basic part reliability history, and the assumption that existing component health signals provide good delineation between the true component health statuses of healthy and failing. Another modeling approach could consider the signal as a function of the true state of the component's health plus noise. We maintain the completely exogenous signal as it aligns with the

motivating application, where the health signal is provided by a component health prediction model.

## 2.4 Structural Results

In this section, the focus is on using the component health signal to decide on a pre-emptive repair, which can be valuable for decreasing system downtime when we can combine it with preventive maintenance. To better understand when it is optimal to combine maintenance actions, we want to establish that for any system in the group, the decision to combine maintenance exhibits a threshold behavior with respect to two key elements in the state vector. That is, if it is optimal to combine system-level preventive maintenance and a pre-emptive component repair on a system with a state element of a certain value, then if that value increases, it remains optimal to combine maintenance actions.

First, we establish this threshold behavior for the $Hours$ element of the state vector. In the use-case, this would mean that if it was optimal to combine maintenance for an aircraft that is 5 hours from being due for preventive maintenance, then for an aircraft with less than 5 hours until preventive maintenance is due, it is still optimal to combine maintenance, all other elements held constant.

**Proposition 2.4.1.** *For state $S_1$ where system $j$ has $Hours = h^*$, if $u_{jt} = M_{PC}$ is optimal, then for any $S_2$ which is identical to $S_1$, but with $Hours > h^*$ for system $j$, $u_{jt} = M_{PC}$ remains optimal.*

Next, we establish threshold behavior for the element of the state vector of the probability of being in a Failing state, $b(F)$. If it is optimal to combine a pre-emptive repair and preventive maintenance for a certain system, and, if our belief of failure for that system's component increases while all other elements remain constant, then it is still optimal to combine maintenance for that system.

**Proposition 2.4.2.** *For state $S_1$ where system $j$ has $b_1(F) = b^*$, if $u_{jt} = M_{PC}$ is optimal,*

*then for any $S_2$ which is identical to $S_1$, but with $b_2 > b^*$ for system $j$, $u_{jt} = M_{PC}$ remains optimal.*

An extension of Proposition 2.4.2 relates to the quality of the binary prediction model that generates the health signal. This final result indicates that if in a certain situation it is optimal to combine preventive maintenance with a component repair, then any improvements to the quality of the predictive model that provides the health signal don't change that decision. Here, quality is the precision, $\zeta$, of the binary prediction model.

**Corollary 2.4.2.1.** *When the component health signal comes from an exogenous prediction model with $P(comp = F|o = \text{“}Failing\text{”}) = \zeta^*$, if $u_{it} = M_{PC}$ is optimal for a system with state $s_{it}$, then for any $\zeta > \zeta^*$, $u_{it} = M_{PC}$ remains optimal.*

These threshold behavior results require one condition about the maintenance inputs and two conditions on the transition function.

**Condition 1:** For there to be value from combined maintenance we must have $T_P > 1$, $T_C > 1$ and $c < 1$. When $c = 1$, there is no difference between combined maintenance and sequentially executing the two maintenance actions. For there to be value from combining maintenance actions the length of each separate maintenance action must be at least one period, otherwise the maintenance can be completed in less than one period and the action does not impact the reward function.

**Condition 2:** Let $b$ be the belief of failure and $b'$ the belief of failure after operating, such that $b'_1 = b'(F)|b(F) = b_1, u = Operate$ and $b'_2 = b'(F)|b(F) = b_2, u = Operate$. If $b_1 > b_2$, then $b'_1$ is stochastically larger than $b'_2$: $E[b'_1] \geq E[b'_2]$. For two systems, if one belief of failure is higher than another, then after operating, the expectation of the previously higher belief is at least as large as the expectation of the other.

**Condition 3:** Let $s_{Hours}$ denote the hours element of one system in the group, and $f$ the probability distribution of the operating lengths for $u = Operate$. $f(.|s_{Hours}) = f(.) \quad \forall s_{Hours}$.

The mission lengths assigned to a system do not depend on the system's operating hours. This means that we expect the operating hours to evolve similarly over time for different systems.

All proofs are in Appendix A.2.

## 2.5 Solution Approach

Unfortunately, the complete model is intractable. The dimensionality of the full state space is $|S| = |s_{it}|^N$, where $s_{it}$ is the state space for a single system from the $N$ total. In our aviation use-case, $|S| \approx 8000^{25}$. To develop usable algorithms to inform the dynamic maintenance decisions we apply a heuristic first developed in Multi-Armed Bandit problems. This applies well, since our setting consists of a group of like-items where we allocate maintenance resources sequentially over time.

### The Multi-armed Bandit Problem

A Multi-Armed Bandit (MAB) problem considers allocating resources between a number of competing projects. In the canonical version, there are $N$ projects and we must decide, at each time period, to operate one project while the others remain passive. We receive a reward for the project we operate, based on the state of the project. After the reward, the operated project transitions to a new state. In a MAB Superprocess, there are multiple ways in which to operate a project and both the project and the control-type must be selected. In the Restless MAB, projects that are not selected can transition states, and instead of selecting a single project, we are allowed to operate $m$ of $N$ projects.

In an index-form solution, each project is assigned an index value based on its state (and not the states of the other projects), and we operate the project with the highest index value. See [31] for more details. This index-type solution provides a simple rule for implementation when considering resources between projects, and the computation is significantly simplified since each project can be considered independently. An index-form solution is optimal for

the canonical MAB. Whittle extends the Gittins index policy to the restless case in [65], but it no longer has an optimality guarantee. It does however provide a simple, interpretable policy that is based on a calculation for each project, independently. The key idea in the heuristic comes from relaxing the $m$ of $N$ requirement for every time period, and instead requiring it in expectation over time. Berstimas and Nino-Mora developed a heuristic for solving the Restless Multi-Armed Bandit problem in [11] that relies on taking a sequence of LP relaxations as approximations of the restless MAB. They then leverage a primal-dual heuristic based on the reduced cost difference between operating and not operating a project. This heuristic simplifies to an index-type policy when there are no isolated states.

In [16], Cho et al. leverage this heuristic for a restless MAB superprocess that models a specific type of new aircraft maintenance related to the stealth coating on US Air Force planes. In [2], Abbou et al. use the Bertismas and Nino-Mora approach as a baseline for a new heuristic in a maintenance problem, and they also leverage system sensor observations.

### 2.5.1 An LP Approximation

We build on the work from [11] and [16] and consider our model as a MAB superprocess; each system is identical and make up the $N$ projects in the MAB description above. Our index policy is based on the reduced costs for each decision variable in a linear program which is an approximation of the full model. In the linear program, each decision variable is a combination of state and action. This means that for an action, like component repair, we can compare different states based on the reduced costs to determine the state where this action has the largest impact on the objective function.

To setup the LP, we first create an indicator for each system, state, action, and time combination. $I^a_{su}(t) = 1$ if system $a$ at time $t$ is in state $s$ with action $u$; 0 otherwise. The fundamental idea for approximating the system is to consider the system in expectation, including the constraints.

When we relax the maintenance capacity and mission demand constraints at every time pe-

riod, and instead consider them in expectation, we de-couple the systems from each other. The decision to enter one system into maintenance is no longer constrained by what we do with the other systems. We now constrain the expected number of systems in maintenance, which constrains each individual system's expected amount of time to a corresponding fraction. This enables us to reduce the dimension of our state-space from $|S| = |s_{it}|^N$ to $|S| = |s_{it}|$ as we develop the LP approximation, with a number of variables equal to $|S||u|$.

We then define the decision variables as the total expected discounted amount of time that a system spends in state and action combinations:

$$x_s^u = E \sum_{t=0}^{\infty} \sum_{a=1}^{N} \frac{I_{su}^a(t)\gamma^t}{N}$$

We want to maximize the expected reward by choosing the best combination of state-actions, given the mission constraint, maintenance constraint, and flow balance constraints between states, for a single system.

$$\max_{x_s^u} \sum_{s \in S} \sum_{u \in U} G'(s) x_s^u \tag{2.2}$$

$$\sum_{u \in U(s)} x_s^u = x_{s,init} + \sum_{s' \in S} \sum_{u \in U(s')} \gamma T'(s', u, s) x_{s'}^u \quad \forall s \in S \tag{2.3}$$

$$\sum_{s \in S_M} \sum_{u \in U(s)} x_s^u \leq \frac{K}{1-\gamma} \tag{2.4}$$

$$\sum_{s \in S} \sum_{u \in Operate} x_s^u \geq \frac{D}{1-\gamma} \tag{2.5}$$

$$x_s^u \geq 0 \quad \forall u, s \tag{2.6}$$

In this linear program, the objective, equation 2.2, maximizes the reward by keeping the

system in the best states possible for the expected discounted time. Constraint 2.3 accounts for the state transition dynamics of the system where $T'(s', u, s)$ is the transition matrix when taking action $u$. Constraint 2.4 limits the average number of system in maintenance at any point in time by restricting each system's expected discounted amount of time in maintenance to $K$, which is the maintenance capacity divided by the total number of systems. Constraint 2.5 ensures that on average there are enough operable system to meet all the missions, where $D$ is the average number of required missions divided by the total number of systems. The discount factor is $\gamma$, with values closer to 1 emphasizing long-term rewards, and smaller values focused on short-term rewards.

### 2.5.2 Computing an Index Value for a State

The goal is to have an index value for every state so that we can easily compare index values between different systems. To find the index values, we solve the LP, and compute the reduced cost for each decision variable. For a given state, we can then compare the reduced costs for the different actions in that state, and determine the action which has the largest objective function benefit. Similarly, given an available action, we can determine the state with the largest objective function benefit for that action.

In our maximization LP, reduced costs are negative for most non-basic variables and zero for the variables in the basic solution. We first develop the maintenance index, $M_s$, with reduced costs of $\psi_s^u$ for each decision variable.

$$M_s = \psi_s^{Rest} - \min_{u \in \{M_P, M_C, M_{PC}\}} \psi_s^u \quad \forall s \in S_{Maintenance}$$

So for every state, $s$, we find the action, $u$, where switching from that action to rest causes the largest decrease to our objective function (most negative number, so minimizing in the definition). In other words, the maintenance index summarizes the penalty we incur if we do not take action $u$ and instead, rest.

If, at some point in time we have unused maintenance capacity, we can then quickly deter-

mine which system has the largest value of $M_s$. The index value can be computed offline, in advance, and it provides an immediate heuristic for implementation by maintenance coordinators.

Because the system has degenerate solutions, in many cases the reduced costs for all of the maintenance actions are 0. We can solve for $M_s$ with the same logic, but we know to only consider the action, $u$, that corresponds to a positive primal variable. This is in-line with the original primal-dual heuristic, where a positive primal variable is the initial screening criteria.

To develop the operating index, $F_s$, we proceed similarly. If we need a system for an operation, then we know that the system with the largest operating index is the best choice.

$$F_s = \psi_s^{Rest} - \psi_s^{Operate} \quad \forall s \in S_{Operate}$$

We focus our numerical analysis on the maintenance index, $M_s$.

### 2.5.3 Impact of the Performance of the Predictive Analytics Model that Generates the Health Signal

In the predictive analytics model that is an input to this work, the health signal for the system's component is binary and indicates if the component is failing or healthy (breaks are fully observable). Recall that four performance measures of the predictive analytics model that generates the health signal are inputs to our model. These are the precision ($\zeta$), sensitivity ($\alpha$), specificity ($\beta$), and negative predictive value ($\eta$). Knowledge of three of the parameters is sufficient to determine the fourth. These input parameters impact our belief that the component is in a failing state, as two of the possible values are $\zeta$ and $1 - \eta$. The input parameters $\alpha$ and $\beta$ impact transition function via the observation function.

Our second overall goal for the chapter is to determine a sufficient performance level of the predictive analytics model so that there is value from making signal-based pre-emptive

repairs. More precisely, we wish to understand how the maintenance index, $M_s$, is impacted by changes in the precision, sensitivity, and specificity of the binary predictive analytics model that generates the health signal.

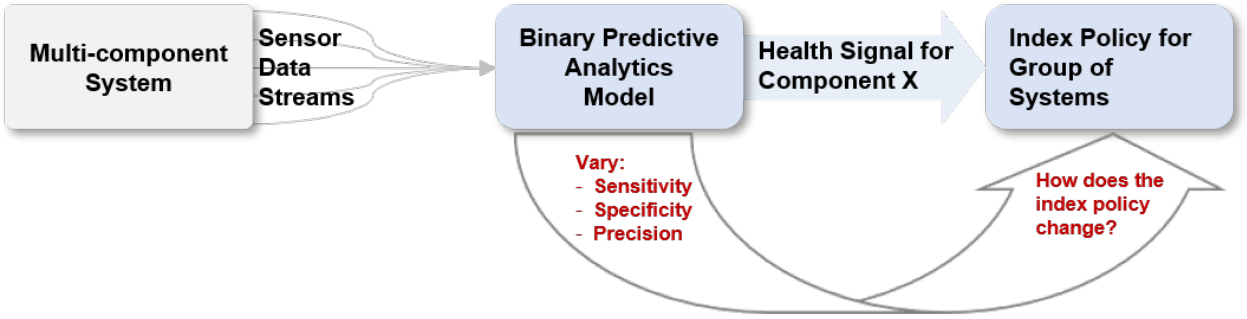


Figure 2-4: The precision, specificity, and sensitivity of the binary predictive analytics model impact the index policy. We explore the impact numerically by varying three input parameters to the LP approximation and solving for $M_s$.

Consider the maintenance index as a function of these three parameters: $M_s(\alpha, \beta, \zeta)$. For any state $s$, we can then determine the range of parameters where the index matches the fully observable case. When $M_s(\alpha, \beta, \zeta) = M_s(1, 1, 1)$ we know that the predictive analytics model providing the health signal has sufficient performance for us to act as though we have complete knowledge of the true component health.

## 2.6 Illustrative Example Based on Field Data

To illustrate how the model can be used in practice, we use maintenance data from a U.S. Army aviation unit flying a variant of the H-60 helicopter. In implementation, the maintenance index provides a simple tool for determining which aircraft should next enter maintenance. First, we characterize the index values in the fully observable special case to better understand which types of maintenance get prioritized over other types. Then, we analyze variation in the quality of the predictive analytics model that generates the health signal to better understand when the component health predictions impact the maintenance decision. Third, we look at the specific component of interest in the use-case.

The use-case data included records for three years of maintenance on a fleet of 26 MH-60 helicopters. This included information on the frequency and duration of aircraft-level preventive maintenance, and information about component breaks and repairs. From this maintenance data we determine maintenance length parameters, and ranges of component reliability to determine parameters in the transition matrix. Additional records included flight usage information from which we estimated the typical mission demand. Interviews with the maintenance team provided information on scheduling procedures, maintenance re-sourcing, and the unit's intended use for the predictive analytics. For additional background on the use-case, see Appendix A.3.

### 2.6.1 Decision Policy

Currently, the maintenance team has a set of existing tasks that constitute the status quo. Key among them is the aircraft-level preventive maintenance required every 40 flight hours. If there are no broken components, then maintainers execute the preventive maintenance on schedule. If there is a broken component, then maintainers adjust the schedule and repair the broken component. If the aircraft-level preventive maintenance is due or the aircraft is less than 5 flight hours away from required preventive maintenance, then the maintainers can combine the component repair with the aircraft-level preventive maintenance.

When the predictive analytics for component health are available, then the maintainers would have a health signal for one component, such as an engine, on each aircraft. We depict the signal in Figure 2-5 as green (healthy), yellow (failing), or red (broken). When the signal indicates healthy or the component is broken, then the maintainers can continue with the current policy.

When the health signal indicates that the component is failing (yellow), then the maintainers have the opportunity for a pre-emptive component repair. How they execute this repair then depends on the flight hours status of the aircraft, seen in Figure 2-5 with the three arrows starting at the failing signal. If the aircraft is at 40 flight-hours and preventive maintenance

is due, then the pre-emptive repair could be combined with the already scheduled preventive maintenance as an opportunistic component repair. If the aircraft is at 35 to 39 flight hours, then the maintainers could start preventive maintenance early and combine it with the component repair as opportunistic preventive maintenance. If the aircraft has less than 35 flight hours since its last preventive maintenance, then the pre-emtpive component repair could be a stand-alone action.

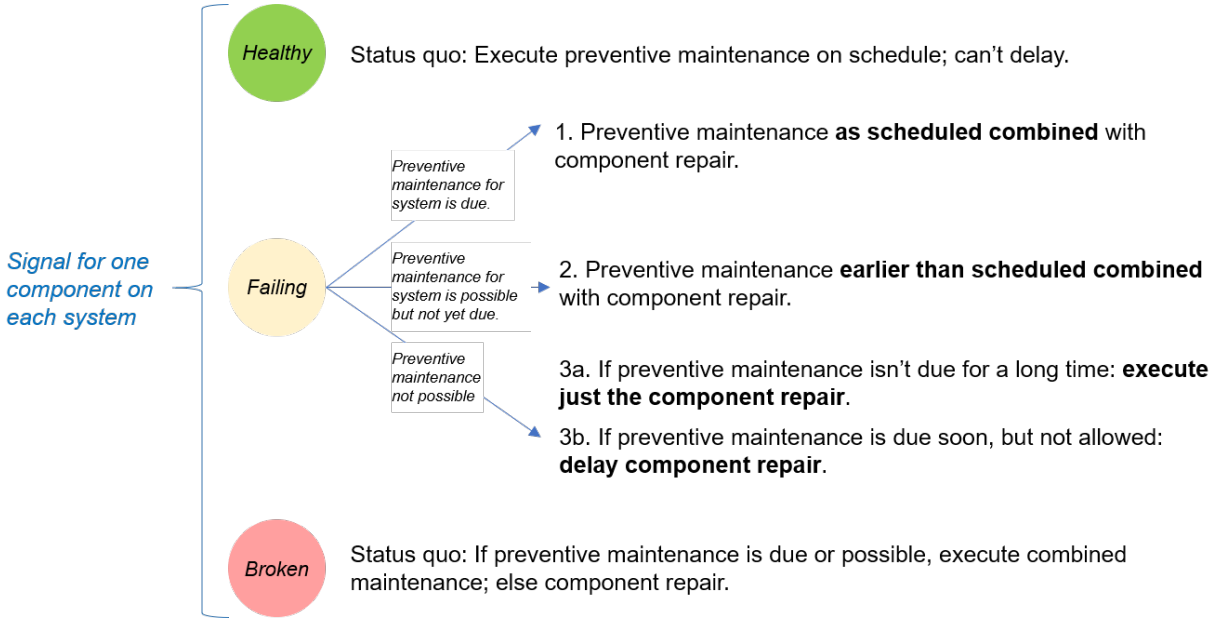


Figure 2-5: The health signal for one component on an aircraft. When the signal indicates healthy or the component is broken, the maintenance procedures are the same as the status quo. When the health signal indicates failing, then there are three possibilities based on the state of the aircraft.

### 2.6.2 General Index-Policy Description

To characterize the decision policy, we look at the maintenance index values for aircraft not already undergoing maintenance. For fully observable component health ($\alpha = \beta = \zeta = 1$), there are two elements in the system's state: the component heath and the number of hours remaining until preventive maintenance is required, the residual. Figure 2-6 depicts a representative example of the maintenance index for the use-case, with each panel representing a different underlying component health, and the horizontal axis as the residual time until re-

quired system-level preventive maintenance. This allows us to capture the state-dependency of the index value, which is on the vertical axis. Given maintenance capacity, we execute maintenance (of the correct type) for the aircraft with the largest non-zero index. Associated maintenance types are color coded. This provides the following seven-tier prioritization of aircraft for available maintenance resources.

**When there is available maintenance capacity, select the aircraft from the highest tier:**

1. For aircraft with a broken component at 40 flight hours since the last preventive maintenance, execute the combined component repair and aircraft-level preventive maintenance.

2. For aircraft with a broken component and 0 - 35 flight hours since last preventive maintenance, execute the component repair.

3. For aircraft with a broken component and 35 - 40 flight hours, execute the combined component repair and aircraft-level preventive maintenance.

4. For aircraft with a component signal of failing, and at 40 flight hours since the last preventive maintenance, execute the combined component repair and aircraft-level preventive maintenance.

5. For aircraft with a component signal of failing and 35 - 40 flight hours since last preventive maintenance, execute the combined component repair and aircraft-level preventive maintenance.

6. For aircraft with a component signal of failing and 0 - 20 flight hours, execute a component repair.
   *If 20-35 flight hours, do not act on the failing signal.*

7. For aircraft with a component signal of healthy, execute aircraft-level preventive main-

tenance as scheduled.

*These tiers apply with a perfect underlying predictive analytics model*

Tiers 1,2,3, and 7 represent the status quo. Tiers 4, 5, and 6 are actions based on the health signal. This policy prioritizes those actions in the context of the status quo actions, and highlights that in some cases, we do not execute a pre-emptive repair even when the signal indicates the component is failing.

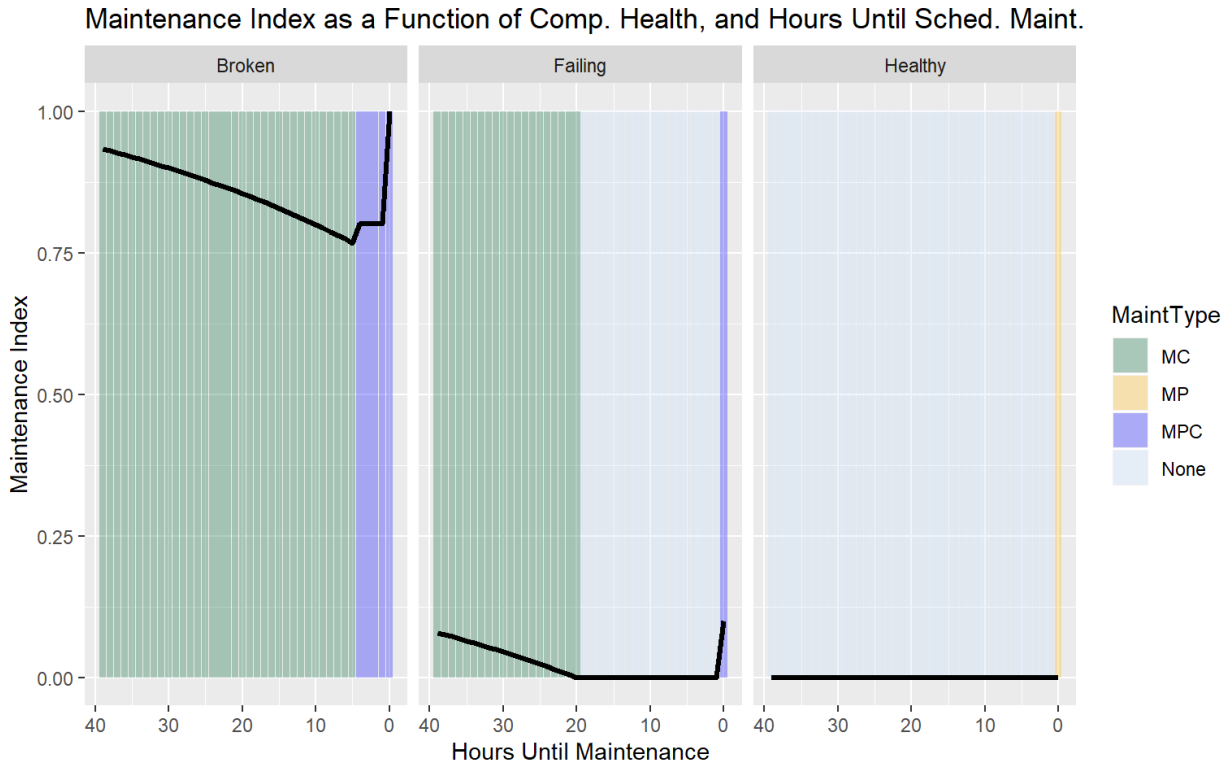


Figure 2-6: A depiction of the maintenance index (black line) for representative parameters. The horizontal axis is the number of hours remaining until required preventive maintenance, and the three panels are the three possible statuses of the health of the component. The type of maintenance selected is color-coded.

### 2.6.3 Necessary Performance Level of the Underlying Predictive Analytics Model

In the partially observable case, we are no longer certain about the state of the component's health. Knowledge about the component's health comes from a health signal, and variations to the solution of the partially observable model come from varying the four performance metrics associated with the health signal's source predictive analytics model: sensitivity ($\alpha$), specificity ($\beta$), precision ($\zeta$), and negative predictive value ($\eta$). We need to know three of the parameters to define the fourth. We can then compare the decision policy with varying levels of uncertainty in the health signal to the decision policy where we have certain knowledge of component health. This impacts three specific tiers in the policy characterization above.

**Tier 4:** When we reach the threshold for executing preventive maintenance, what is the necessary performance of the health signal's source predictive analytics model for us to execute a pre-emptive component repair at the same time as aircraft-level preventive maintenance? In Figure 2-6, we see this when the decision at "0 hours remaining" is to conduct combined maintenance (blue), instead of just preventive maintenance (orange).

**Tier 5:** When we are in the tolerance window where we can start preventive maintenance early if we choose to do so (35 - 40 flight hours), what is the necessary performance of the health signal's source predictive analytics model for us to decide to start preventive maintenance early and the combine it with a pre-emptive component repair? In Figure 2-6, we see this with combined maintenance (blue) occurring prior to the "0 hours remaining" position on the horizontal axis.

**Tier 6:** When we aren't near a preventive maintenance inspection, what is the necessary performance of the health signal's source predictive analytics model for us to decide to repair a possibly failing component? When might we deliberately skip the possible repair? In Figure 2-6, we see this with a positive maintenance index for a failing component.

For each of these three tiers, we can then determine sufficient performance of the predictive

analytics model by numerically determining values that produce the same maintenance index as the fully observable case. Figure 2-7 shows a representative example for tier 4 in a manner based on a true positive rate (TPR) / false positive rate (FPR $= 1 - \beta$) plot. This plot allows us to see the region of values for $\alpha$ and $\beta$ where the decision policy in the partially observable case matches the fully observable case, with a predictive analytics model precision ($\zeta$) of 0.91. Only if the associated sensitivity and specificity are in the blue region do we act on our belief that the component is failing and combine the pre-emptive repair with preventive maintenance when the 40-hour preventive maintenance is due. If the performance of the underlying model is not in the blue region, then tier 4 is removed from the decision policy.

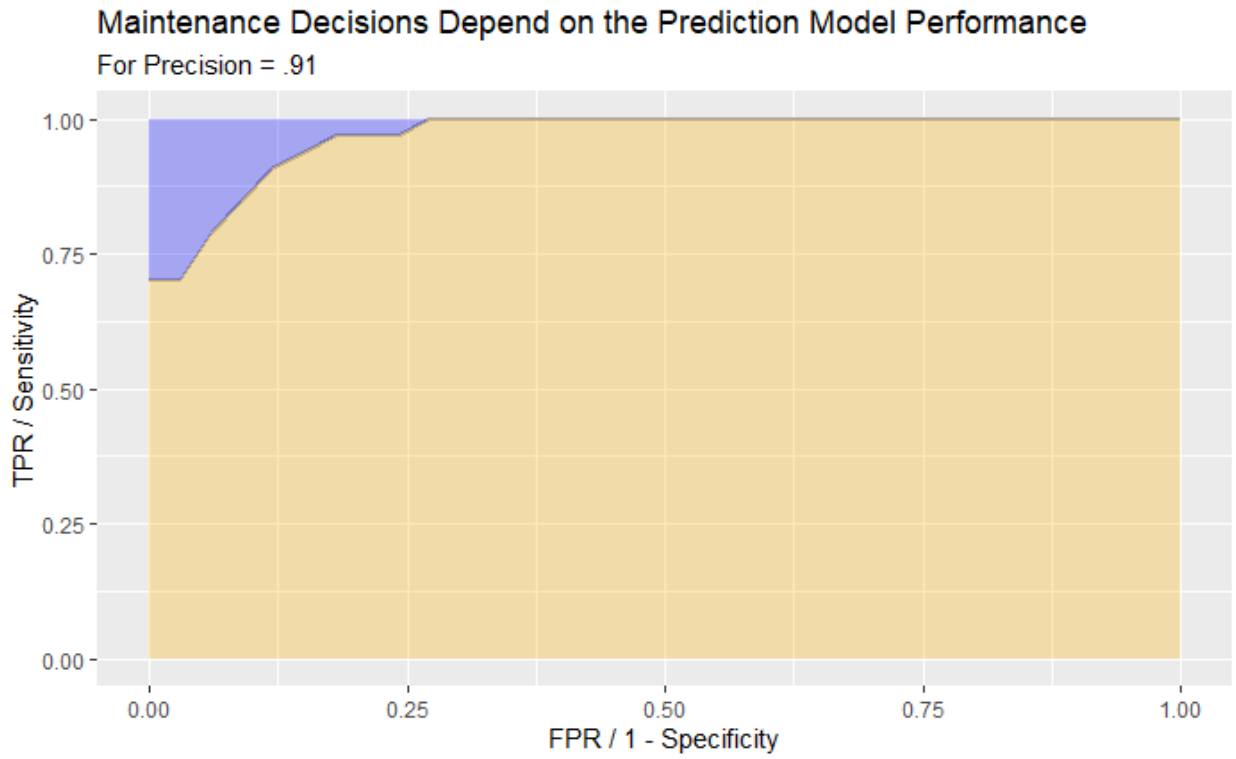


Figure 2-7: A depiction of the performance region where the partially observable model yields the same policy for a failing component as the fully observable case where knowledge of the failing component is certain. Only in the blue region are we confident enough in the underlying prediction model to change our action when we reach the operating hours that requires preventive maintenance to also include a pre-emptive repair.

We now consider tiers 4,5, and 6, and instead of fixing a precision and plotting the FPR

/ TPR, we fix the underlying fault prevalence. This is known from reliability data, and represents the probability that a randomly selected component is in a failing health status. Figure 2-8 depicts the threshold curves for the three tiers. In this case, for tier 4, the underlying component health prediction model must have a FPR below 5%, and as the sensitivity of the model decreases, the FPR must be even lower. For tier 5, we see that the health signal's source prediction model must be perfect - only when we know that a component is failing is it worth while to initiate early preventive maintenance combined with a pre-emptive repair based on a health signal. In tier 6 we see that the specificity must be very close to 1, and the sensitivity must be above 0.8. The status quo representation of the decision policy is one that includes tiers 1, 2, 3, and 7. Tiers 4, 5, and 6 then enter the decision policy for different performance thresholds of the underlying predictive analytics model.

The characterization of these thresholds, for this use-case, highlight important cautionary notes for the implementation of component-level fault prediction models. In each of the three tiers, the health signal's source prediction model must have exceptionally high performance before it provides any utility for updating the unit's maintenance execution. For tiers 5 and 6, the specificity must be extremely close to 1. In other words, the downside of false positives is so high, that if a model produces any, it is not worth using unless the signal comes at the same time that aircraft-level preventive maintenance is required.

### 2.6.4 Predictive Analytics Model Evaluation

The military's first predictive analytics maintenance model for the H-60 helicopter based on system sensor data was funded as a test-case to learn about the end-to-end process of building and implementing a machine learning model for predicting component health; additional details are in Appendix A.3. For implementation, there were two discussed uses. First, if the unit was deploying helicopters forward from a base to a combat zone, the fault prediction model for engine health could inform the unit on which helicopters were suitable. We do not evaluate the impact of this use here. Second, the predictive model could inform

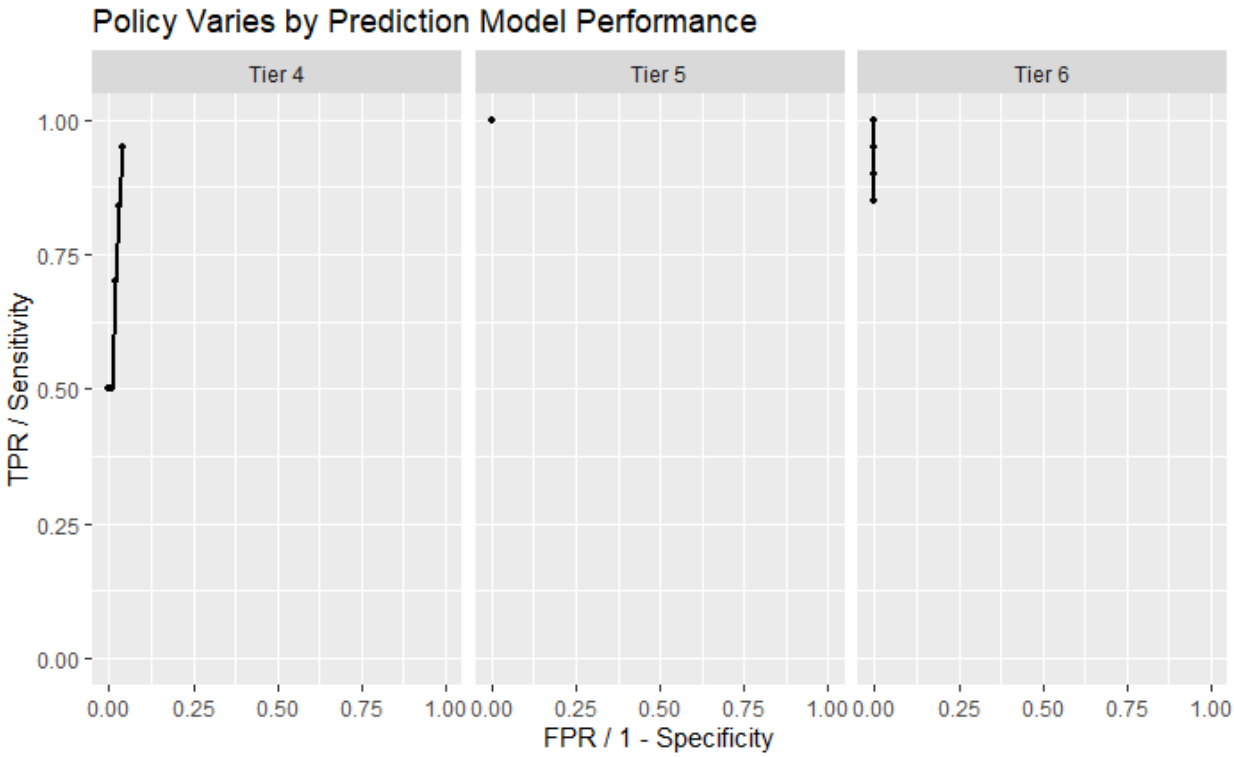


Figure 2-8: Required performance of the health signal's source predictive analytics model before the policy begins to incorporate knowledge of a failing part, for the decision policy tiers impacted by a failing health signal (panels). Only for performance levels above and to the left of the black line are we confident enough in the health signal's source predictive analytics model to execute a pre-emptive repair.

the unit about maintenance planning, one aspect of which, pre-emptive repairs, we can evaluate.

In terms of achievable predictive success, this test-case model is not close to the necessary threshold to use in maintenance planning. We use the same example of the numerical threshold previously shown, where the prediction model performance must be in the blue region for the policy to leverage any information about a failing component. We overlay on that graph the approximate ROC curve from the engine health model's predictions for a high-end precision estimate. (Full performance metrics are not releasable.) This shows the health signals from this predictive analytics model for engine health should not be used for scheduling pre-emptive repairs. If the unit adopts a myopic use of the health signal,

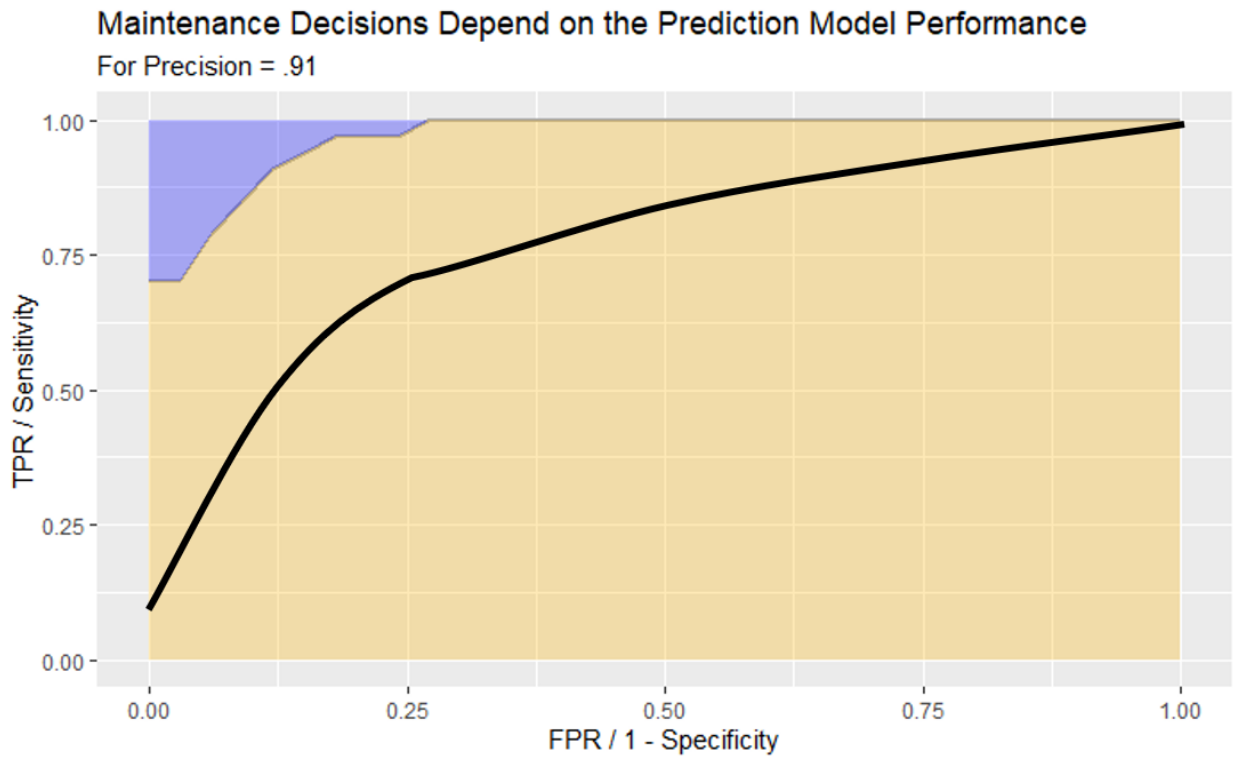


Figure 2-9: The approximate ROC curve for the engine health model for the H-60 (black) compared to the region where the model performance needs to be for scheduling use in tier 4.

where anytime the signal indicates a component is failing and there is maintenance capacity, it could decrease the unit's readiness. Simulation details comparing this index policy to a myopic policy are in Appendix A.3.

## 2.7 Conclusions

For a group of systems, each with a health signal for one component generated by a predictive analytics model, we are interested in synchronizing system preventive maintenance with repairs of that component. Four aspects collectively distinguish this problem from others in the maintenance scheduling literature. They are: combining signal-based component pre-emptive repairs with a time-based schedule, stochastic part failure, fleet-level constraints, and an objective that is not based on traditional costs. These aspects drive our modeling

approach, which uses a POMDP to capture the group of systems and its characteristics. We develop an approximate decision policy for this model using an LP-driven heuristic first developed for multi-armed bandit superprocesses. This solution provides an index-style policy, which makes for simple maintenance team implementation that in the use case characterizes into 7 priority tiers for which aircraft should next enter maintenance. When there is maintenance capacity, the aircraft in the highest tier enters maintenance.

We then determine how the performance of the health signal's source predictive analytics model impacts the inclusion of pre-emptive repairs in the decision policy. The minimum performance level varies based on input parameters, but in general the results show that a health signal must have an extremely low false positive rate before it can usefully influence scheduling decisions in an organization. For the U.S. Army aviation use-case, this result indicates that existing component health prediction models are not ready for operational use triggering pre-emptive repairs. The downtime incurred by executing component repairs on false positive health signals is too high.

This key takeaway can influence organizational decisions about when a model is good enough to move from development to operations. This methodology applies to maintenance domains where multi-component systems are managed as a group, and we attempt to incorporate a component health signal into our existing preventive maintenance framework.

# Chapter 3

# Career Path Design Policy and Military Personnel Readiness

## 3.1 Introduction

The military needs the right number and composition of qualified people to meet the workforce demands of its many hundreds of subordinate organizations. This is a critical aspect for ensuring that the organizations can fulfill their missions. There is a wide variety of necessary skills and levels of experience, from new ordnance maintainers to seasoned infantry officers. Many requisite skills and experiences are only achieved from training and work experience within the military, so hiring is almost exclusively for entry-level personnel, who then advance to positions of greater responsibility. A typical individual in the military advances along a career path through multiple types of jobs for a number of years.

Personnel planners in the military work to ensure the size and composition of the force, and focus on four key decision areas, depicted in Figure 3-1. First, given the cumulative workforce demands from the overall force design, planners determine the target workforce composition based on the military's current method of categorizing individuals by job spe-

cialty and rank [30]. Second, planners design the workforce supply to meet these targets by acquiring entry level personnel, retaining personnel, transitioning personnel between job specialties, and promoting personnel from one rank to the next so they can assume more advanced roles. These decisions lead to recruiting goals, retention incentives, job specialty transfer allowances, and promotion targets. Third, leaders in the various job specialties make strategic decisions about the types of experiences that are necessary for individuals at each rank so that they are prepared for future jobs with greater responsibility. This professional development guidance includes details on required job types, assignment durations, pre-requisite assignments, and required schooling. Taken collectively, this guidance sets the career path design policy that determines possible individual assignment sequences. Fourth, a single execution agency focuses on operational decisions that ensure each of the many geographically-dispersed organizations in the force has the right personnel assigned for its needs. When all of the jobs in an organization have correctly-skilled personnel assigned to them, the organization has maximized personnel readiness.

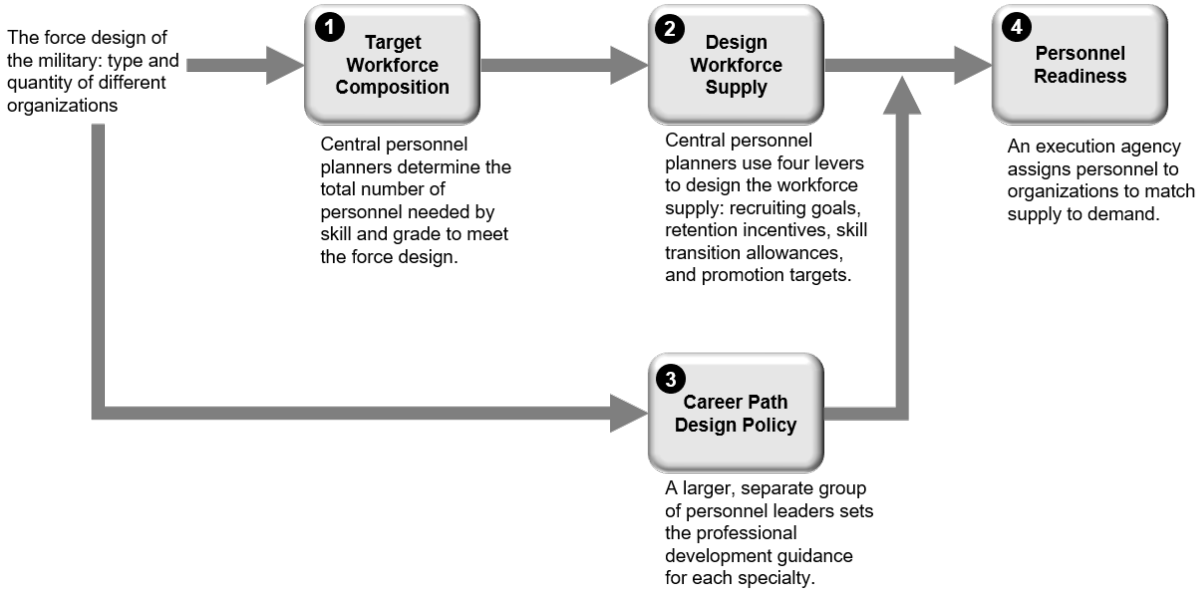


Figure 3-1: Four key decision areas for military personnel planners. Central personnel managers 1) decide on the target workforce composition and 2) design the workforce supply. 3) A different group of personnel leaders from each specialty set the career path design policy. 4) A human resources execution agency maximizes personnel readiness.

Designing the military personnel supply to meet the workforce targets is a large analytic effort that occurs on a multi-year horizon. Career path design policy updates infrequently, and the policy is the cumulative result of many leaders' decisions on aspects of professional development guidance. The execution agency that assigns people to roles within organizations operates on a shorter time-horizon as its primary goal is personnel readiness now and six months in the future. This agency takes the current workforce supply and the constraints established by career path design policy, and works to assign individuals to organizations across the force to maximize personnel readiness. Frequently, career path design policy imposes requirements and constraints where more individuals need the same type of job at the same time than there are jobs of that type. In other situations assignment duration minimums limit the number of personnel who complete a key assignment type each year which restricts the number of personnel available for jobs that require the key assignment as a prerequisite. This creates different types of bottlenecks in the military's personnel flow, which can lead to a shortage of personnel for important jobs. Personnel readiness is measured with the proportion of jobs filled by qualified individuals. The details of the execution agency's operations that are focused on readiness, including the constraints that career path design policy place on individuals, are absent or not considered as part of the multi-year personnel design process that is focused on ensuring the workforce supply. In fact, an implicit assumption in current workforce design models is that the right number and composition of people ensures high personnel readiness.

A specific example, which we return to as a numerical use-case, relates to U.S. Army captains who are required to hold a "key and developmental" (KD) job as a prerequisite to more advanced positions and command roles. Examples of KD jobs include commanding a company of soldiers and staff work as a battalion's primary intelligence officer or primary logistician. In particular, if the number of available KD jobs is insufficient for the number of captains that need them, a backlog occurs consisting of captains waiting for the KD jobs. So even when this specialty of captains is correctly sized and there are enough captains to fill every required job, this backlog can cause post-KD jobs to remain unfilled. In this case, career

path design policy, while useful for future talent development, has hurt military personnel readiness, which we describe in detail in Section 3.3.

This chapter addresses two research questions: (i) What is the impact of career path design policy on personnel readiness? (ii) How can we add flexibility to the career path design policy to improve personnel readiness?

To address the first question, we model the operational aspects of military career management as a flow on a specially designed graph. Grouping a specialty's jobs by type, we form a time-expanded graph where there is a node for each job type and each period of time and the number of jobs of each type that need to be filled is captured through the demand at the respective node. The supply captures the personnel travel along designated 'career' paths in the graph, thus satisfying the demand of the job nodes along the path. A job-type's readiness is the fraction of the jobs that have personnel assigned from any of the career paths. The overall readiness shortfall is the difference between demand and supply throughout the nodes in the graph. We then formulate a linear program, **MPRP**, that considers personnel planners allocating a fraction of the total available personnel to each of the career paths, constrained by the number of available jobs within a certain type, with the goal of maximizing personnel readiness. When the maximum personnel readiness is lower than what it could be without restricting the available career paths, we know career path design policy has decreased readiness.

To address the second question, we model a process where personnel leaders change career path design policy and then an execution agency maximizes personnel readiness. As an exact approach, we formulate a bi-level mixed integer program,$\mathbf{MPRPF_k}$, in which leaders select $k$ additional career paths that are not currently allowed, and then the personnel planners maximize readiness using the already allowed career paths and the newly selected ones. For a more tractable approach, we develop an algorithm, $\mathbf{GFA_k}$, that iteratively selects $k$ additional paths, within some allowable adjustments to professional development guidance, using a column-generation inspired approach. Personnel leaders then maximize readiness

using the already allowed and $k$ additional paths.

### Contributions

This chapter makes two key contributions. First, we develop new types of models to capture and support different decision levels in military personnel planning and operational management focused on quantifying the impact of career path guidance on personnel readiness. The key idea is connecting military career paths used in practice to network paths in our model. This includes using personnel assignment histories to produce a data-driven view of the actually-implemented career path design policy.

Second, algorithmically, we develop a bi-level mixed integer formulation that augments an existing set of paths with a $k$-sized subset while allocating resources across those paths to minimize cost. We develop a computationally attractive iterative algorithm for finding the $k$-sized augmenting subset with an approach motivated by column-generation that avoids enumerating all possible paths.

Finally, we validate the approach with personnel data from the U.S. Army where we measure the readiness impact of career path design policy on a specific officer specialty, and determine feasible adjustments to professional development guidance that best enable an increase in readiness. This approach can also assist planners during times of large organizational change. The models here can help planners understand which specialties might need career path guidance updates following changes to the number and types of jobs for that specialty.

### Generalizations

This approach applies to other enterprise personnel planning efforts where an individual's possible sequence of assignments is restricted to predefined sequences. Examples include academia, large police departments, and large consulting firms. More generally, the approach captures the allocation of resources to tasks, where the sequence of task completion matters, there are limits on the number of resources that can be assigned to a task, and we wish to

minimize the cost of uncompleted tasks.

We can then consider increasing the flexibility by expanding the allowable set of task sequences, by adding additional resource paths, such that we further minimize cost. The algorithmic approach to adding paths, $\mathbf{GFA_k}$, provides a method for considering reasonable adjustments that does not require the enumeration of all possible paths. An example application area where multiple overlapping regulatory agencies limit the allowable paths and we might want to consider additional paths is routing hazardous cargo.

### Chapter Outline

In Section 3.2, we summarize existing personnel planning literature and other related works. In Section 3.3, we define personnel readiness, present the model of personnel flow though job-types, and the optimization formulation, **MPRP**, for allocating personnel to career paths. Section 3.4 presents the two methods for considering changes to career path guidance, with the bilevel mixed integer formation, $\mathbf{MPRPF_k}$, and the iterative algorithm, $\mathbf{GFA_k}$. Section 3.5 describes the connection between personnel readiness constrained by career path design policy described in this chapter, and readiness during the workforce design process in practice. It also includes a computational comparison of the two flexibility approaches. Section 3.6 presents a numerical use-case based on assignment data from thousands of U.S. Army officers, including the impact of specific career path design policy, and adjustments that could increases readiness. Section 3.7 provides concluding comments. Supplemental material is available in the appendices about additional model aspects.

## 3.2 Related Literature

Much of the existing personnel planning literature addresses workforce design, where the goal is to make decisions that match the size of the workforce in a certain specialty to the target for that workforce as closely as possible using accessions, retention, transfers, and promotion. In their recent survey, [56], Bastian and Hall summarize the main methodological categories

of manpower modeling and discuss two methods for workforce design: goal programming and Markov decision models. Their review builds on the foundational manpower modeling overview from Gass in [30]. An example of research on workforce design is Horn et al. in [37] who develop a mixed integer goal programming formulation to determine accession cohort size while accounting for skill-level and training resources. Another recent example is [67] by Zais and Zhang, which uses a Markov decision model to incorporate uncertainty in a sequential decision process focused on determining the right force design actions. These representative works focus on ensuing that the workforce is correctly sized, but do not consider the career path guidance that restricts the assignments of individual service members.

There is limited research on personnel models that include aspects of career path guidance. Dabkowski et al. address individual service member talents including the impact of attrition, but not career path guidance, in [18]. Hall and Fu in [35] combine rank and cohorts to model levels of seniority within a grade which accounts for how long an individual has been in the system, but not career paths. Abdessameud et al. address that certain jobs require certain experiences by adding competencies to an existing view of ranks in [3]. They use this approach to address accessions and promotions when considering jobs that require a combination of skills. The closest work found by the author that incorporates career path guidance is [10] by Baumgarten, which addresses United States Marine Corps accession targets with some aspects of career path guidance, but does not account for readiness.

Methodologically, the exact formulation for determining the augmenting $k$-sized subset builds from bi-level mixed integer programming. A representative work of this type that uses a path-based formulation with two decision authorities operating in succession is [64] by Verter and Kara. For the iterative approach to flexibility, we use an aspect of column generation. Lubbecke and Desrosiers provide an overview of column generation and its many implementations in [46]. This setting shares some common attributes with scheduling with precedence constraints. In a typical approach to the scheduling problem, tasks are depicted as nodes and task dependencies are edges between the tasks. See [66] for a review

of workflow scheduling. Two aspects motivate using a different approach. First, we want to leverage the complete histories of people's assignments as the career path policy is not strictly a function of immediate dependencies. Second, our goal is not minimizing an overall schedule length, but maximizing the proportion of the job-types (tasks) that are in use.

## 3.3 Modeling Personnel Readiness

This section describes the model and presents the optimization formulation that determines the readiness-maximizing allocation of personnel to paths in a given set.

### 3.3.1 Model

We generate the directed time-dependent graph $\mathcal{G} = (V, E)$ over a finite horizon and for personnel in a specific specialty. $\mathcal{G}$ has nodes $v \in V$ for each combination of time-period $t$, and job type $s \in S$, with edges that fully connect nodes in successive time periods, as in Figure 3-2. The number of jobs of each type is the aggregated demand for the nodes of that type (e.g., the demand for red jobs can be met at any red node). A source node, $b_0$ originates the personnel flow.

The key modeling idea is to connect the career path design policy to $\mathcal{G}$ by directly relating career paths in practice to allowable network paths in the constructed graph. Career path design policy determines a known set of paths for personnel, $P_A$.

#### Personnel Flow

A volume of personnel, $b$, flows through the graph $\mathcal{G}$ using only the allowable career paths. In each time period, a fraction of the personnel volume on each path leave this job specialty, and only $\beta_p^t$ of the starting amount remain. The number of jobs of a certain type, $n_s$, constrains the total volume of personnel on all paths that pass through any nodes of that job type, $s$. In the military, a new group of personnel start their career paths every year, and the time-expanded nature of the graph allows us to capture the path-use decision as one that repeats

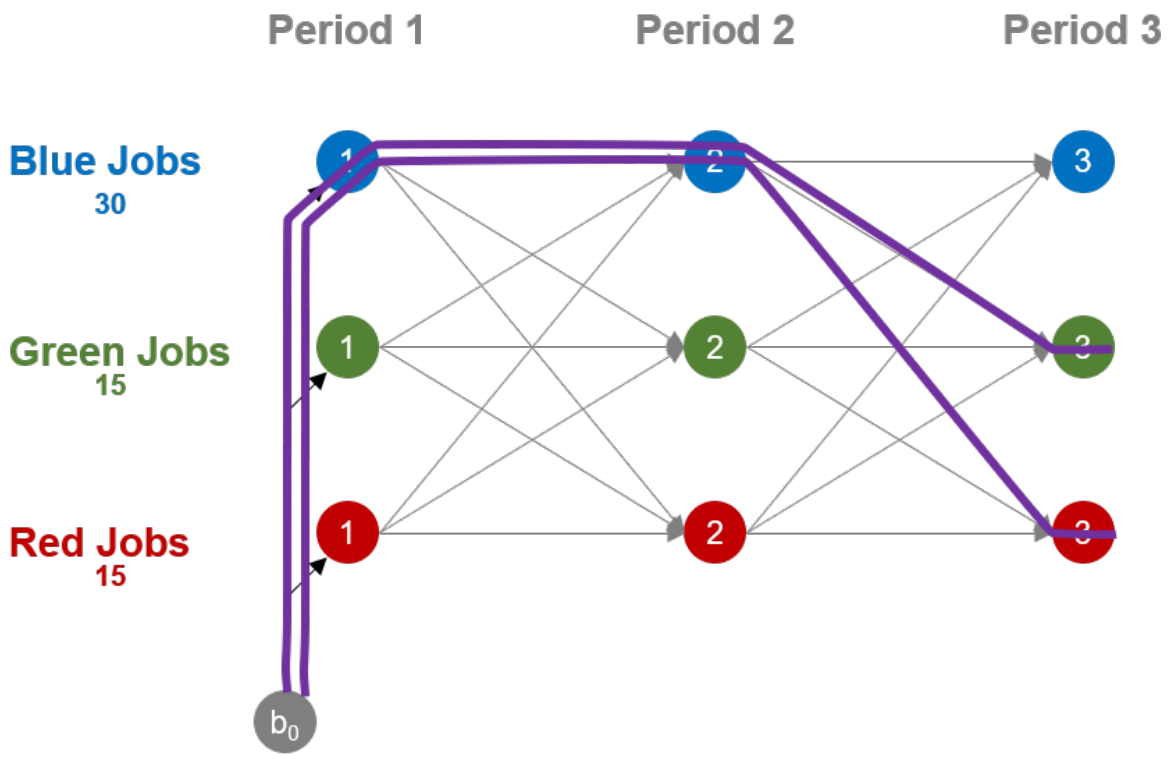


Figure 3-2: An example graph depicting the combination of job structure and career path design policy with 3 time periods, $T = 3$, and 3 job categories, $S = \{Blue, Green, Red\}$. The number of jobs for each job type is the demand, and two sample paths are shown in purple.

each year. For example in Figure 3-2, consider 15 personnel entering the blue-blue-red path each year, and all 15 stay for all three periods (no loss). Then, the demand for blue jobs (30) is met by the every-period decision for 15 personnel to enter the blue-blue-red path. This is equivalent to viewing a period, such as $t = 2$, where we would see 15 personnel in blue jobs who started on the blue-blue-red path in $t = 1$, and 15 more personnel in blue jobs who started on the blue-blue-red path in $t = 2$.

### Readiness

The military's primary view of manpower readiness is the proportion of jobs that have an assigned worker, assuming each job is filled by a qualified worker.

**Definition 3.3.1** (Readiness, $R$)**.** For a certain job specialty, the proportion of total jobs,

$N$, that are filled, where $D$ is the number of unfilled jobs. $R = \frac{N-D}{N}$.

In this setting, we consider jobs aggregated by type, and can measure the number of filled jobs in each category. Readiness for a job type is then the ratio between the volume of personnel on all paths that pass through a certain job type, and the number of those jobs. If 15 personnel enter and remain on the blue-blue-red path in Figure 3-2, then the readiness for blue jobs is $\frac{2(15)}{30} = 1$. In the optimization below, we consider the readiness shortfall for a job type as the difference between the number of filled jobs and the number of jobs, possibly scaled. The overall readiness shortfall is the weighted sum of the job-type readiness shortfalls.

### 3.3.2 An Illustrative Example of Career Path Design Policy Restricting Readiness

Consider the example in Figure 3-2 with 20 available personnel, and no loss of personnel over time. The total demand is 60 jobs and the 20 personnel are all available for three periods. We therefore expect the readiness to be 1. If we consider any of the 27 possible paths in the graph, one allocation that achieves a readiness of 1 is 10 personnel on blue-blue-blue, 5 on green-green-green, and 5 on red-red-red. This allocation meets all of the demand.

Now, consider career path design policy restricting the allowable paths to only the two purple paths shown: blue-blue-red and blue-blue-green. This represents two prerequisite requirements, blue before red and blue before green, and an assignment duration requirement, blue for two periods. This is similar to the motivating case with U.S. Army captains, where a KD job is a prerequisite for other types of jobs, and the KD job has a minimum assignment duration.

In this case, one allocation is 8 personnel to blue-blue-green and 7 personnel to blue-blue-red. This achieves a readiness of $\frac{2(8+7)}{30} = 1$ for blue jobs, but only a readiness of $\frac{8}{15} = 0.53$ for green jobs and $\frac{8}{15} = 0.47$ for red jobs. It also leaves 5 personnel unassigned to a career path,

who in practice, might have to wait for an available assignment. The career path design policy captured by only allowing the two depicted paths hurts the achievable readiness.

### 3.3.3 Military Personnel Readiness Problem (MPRP)

A readiness maximizing solution for the illustrative example is simple to find. In practice, there are more job types and time periods, the career path design policy includes other restrictions, and we need to account for personnel loss. **MPRP** is a linear program that determines the allocation of personnel to paths in a given set driven to minimize any readiness shortfalls.

#### Sets and Parameters

We use the following sets and parameters.

| | | |
|---|---|---|
| $T$ | : | The time horizon, in periods, $t$. |
| $S$ | : | The set of job types. |
| $\mathcal{G}$ | : | The graph of job types, $S$, over periods $\{1, ..., T\}$. |
| $P$ | : | The set of all possible paths, $p$, in $\mathcal{G}$. |
| $P_A$ | : | The set of paths allowed by career path design policy, $P_A \subseteq P$. |
| $n_s \geq 0$ | : | The number of jobs in assignment category $s \in S$. |
| $N$ | : | The total number of jobs, $\sum_{s \in S} n_s$. |
| $r_s \in [0, 1]$ | : | The readiness factor for assignment category $s \in S$. |
| $c_s \in [0, 1]$ | : | The readiness cost for shortfalls in assignment category $s \in S$. |
| $\beta_p^t \in [0, 1]$ | : | The proportion of the initial amount on path $p$ remaining at $t$. |
| $\delta_s^t(p) \in \{0, 1\}$ | : | Indicator variable that path $p$ is on node $s$ at $t$. |
| $b \geq 0$ | : | The maximum starting amount of personnel. |

The central personnel managers' goal is to determine the personnel volume on each career path to minimize any readiness deviations. Let $f_p$ be the volume of personnel assigned to path $p \in P_A$, and $d_s$ is the readiness shortfall in assignment category $s$. Each assignment category, $s$, has a cost, $c_s \in [0, 1]$, for a readiness shortfall. With a minimum cost objective, Formulation 3.1 seeks decisions where personnel flow meets as much of the required demand as possible. This ties the objective to maximizing readiness, where the total number of unfilled jobs, $D$, in Definition 3.3.1, connects to the readiness shortfalls, $d_s$.

$\mathbf{MPRP(P_A)}$ :

$$\min_{f,d} \quad \sum_{s \in S} c_s d_s$$

$$s.t. \quad \sum_{t=1}^{T} \sum_{p \in P_A} \delta_s^t(p) \beta_p^t f_p + d_s \geq r_s n_s \qquad \forall s \in S \qquad (3.1a)$$

$$\sum_{t=1}^{T} \sum_{p \in P_A} \delta_s^t(p) \beta_p^t f_p \leq n_s \qquad \forall s \in S \qquad (3.1b)$$

$$\sum_{p \in P_A} f_p \leq b \qquad (3.1c)$$

$$f_p \geq 0 \qquad \forall p \in P_A$$

$$d_s \geq 0 \qquad \forall s \in S$$

Constraints (1a) and (1b) account for the dynamics of the system. Constraint (1a) uses the readiness adjusted number of jobs as a lower bound that determines the shortfall penalty, $d_s$. If the personnel flow fails to meet the lower bound, a penalty makes up the difference. Constraint (1b) is an upper bound on the flow through a job type. In these two constraints, we adjust the volume of flow on each path for loss over time, such that only a fraction of the volume that begins on path $p$, remains at time $t$, $\beta_p^t \in [0, 1]$. So if a path $p$ passes through node $s$ in period 5, then that path provides $\beta_p^5 f_p$ towards the readiness demand at node $s$. In this way, a path $q$ that went through the blue node in period 1 and period 2 would contribute $\beta_p^1 f_q + \beta_p^2 f_q$ towards the demand for blue jobs. This captures loss as part of the expected

'person-years' that a service member would spend in that job-type. Time-based attrition is the primary driver in our use-case, but this formulation also allows for path-specific attrition. Constraint (1c) is a cap on the initial personnel volume.

The main driver of the formulation size is the number of paths, and a complete set of possible paths in $\mathcal{G}$ is exponential in the number of periods. Formulation $\mathbf{MPRP}(\mathbf{P_A})$ has $|P_A|$ continuous variables and $2|S| + 1$ constraints. We restrict our attention to the set of paths currently allowed, $P_A$, to capture the career path design policy. In practice $P_A$ is significantly smaller than the full set of possible paths in $\mathcal{G}$.

**Model Limitations**

This models a steady-state system, where we ignore tactical adjustments and stochasticity to isolate the impact of career path design policy. Tactical adjustments include one-off career path exceptions that are made during the execution agency's process for assigning personnel to organizations. The system in practice does not assign individuals to pre-determined career paths, but instead consists of a number of sequential decisions over time as people's careers unfold. In this steady-state, non-stochastic setting, there is no distinction between deciding the allocation in advance, and deciding each period. This model's use as a planning tool does not require a shift to pre-determined career paths for newly hired personnel. Its focus is on understanding the fraction of a group of personnel that need to take certain paths. Hence, there is no loss to maintaining $f_p$ as continuous. A traditional edge-based network flow formulation cannot capture assignment histories, which are necessary for capturing the career path design policy. Appendix B.1 has additional details and examples on the motivations for the path-based formulation.

## 3.4 Adding Career Path Flexibility

When career path design policy limits readiness, the addition of career paths to the system provides flexibility for the flow of personnel through $\mathcal{G}$, which can improve readiness. This

section looks at modeling approaches for adding flexibility to the system, with the addition of a limited number of paths.

### 3.4.1 Military Personnel Readiness Problem with Flexibility

**($\mathbf{MPRPF_k}$)**

We formulate an extension of **MPRP** that considers adding $k$ additional paths to the existing set of allowable paths, $P_A$, and then minimizes personnel readiness shortfalls using the original and new paths. The number of $k$-sized subsets is combinatorial, so instead of an enumeration approach, we use a bilevel formulation.

| | | |
|---|---|---|
| $P'$ | : | The set of additional career paths under consideration. $P_A \cup P' \subseteq P$ |
| $k \in Z_+$ | : | The maximum number of additional career paths. |

The upper level decision is for the $k$ additional paths, where $z_p = 1$ for each selected path in $P'$. Then, the readiness shortfall is minimized as in **MPRP**, but using the original allowable paths, $P_A$, and the newly-selected $k$ paths. The complete $\mathbf{MPRPF_k}$ is shown in Formulation

3.2.

$\mathbf{MPRPF_k(P_A, P')}$ :

$$\min_{z} \quad \min_{f,d} \quad \sum_{s \in S} c_s d_s$$

$$s.t. \quad \sum_{t=1}^{T} \sum_{p \in P_A \cup P'} \delta_s^t(p) \beta_p^t f_p + d_s \geq r_s n_s \qquad \forall s \in S \qquad (3.2a)$$

$$\sum_{t=1}^{T} \sum_{p \in P_A \cup P'} \delta_s^t(p) \beta_p^t f_p \leq n_s \qquad \forall s \in S \qquad (3.2b)$$

$$\sum_{p \in P_A \cup P'} f_p \leq b \qquad (3.2c)$$

$$f_p \leq b z_p \qquad \forall p \in P' \qquad (3.2d)$$

$$\sum_{p \in P'} z_p = k \qquad (3.2e)$$

$$f_p \geq 0 \qquad \forall p \in P_A \cup P'$$

$$d_s \geq 0 \qquad \forall s \in S$$

$$z_p \in \{0, 1\} \qquad \forall p \in P'$$

Constraints (2a-c) mirror the **MPRP**, with the additional paths from $P'$ added to consideration. Constraint (2d) ensures that from $P'$, only paths selected for inclusion with $z_p = 1$ can have non-zero personnel volume. Constraint (2e) is the cap on the number of additional paths. The main drivers of the formulation size are the number of paths allowed and the number of paths under consideration. Formulation $\mathbf{MPRPF_k(P_A, P')}$ has $|P_A| + |P'|$ continuous variables, $|P'|$ binary variables, and $|P'| + 2|S| + 2$ constraints.

### 3.4.2 Greedy Flexibility Augmentation ($\mathbf{GFA_k}$)

When $P'$ is large, $\mathbf{MPRPF_k}$ may not be tractable. Another approach is to iteratively find the $k$ paths. This approach has no guarantee of finding the best $k$ subset, but has computational benefits described in Section 3.5.

**Column Generation-Inspired Approach**

Finding the best single additional path from $P'$ to add to $P_A$ is a search for the new path that best improves the readiness in $\mathbf{MPRP}(\mathbf{P_A})$: $\min_{p \in P'} \ \mathbf{MPRP}(\mathbf{P_A} \cup \{\mathbf{p}\})$. One method of approaching this is to find the path $p \in P'$ that has the most negative reduced cost in $\mathbf{MPRP}(\mathbf{P_A})$. We can find this path $p$ by developing a pricing problem, as in column generation.

We first consider all of the possible paths in $\mathcal{G}$, where $P' = P \setminus P_A$. This allows us to establish a baseline pricing problem, which we then modify to address cases where $P' \subset P \setminus P_A$. We can find the optimal dual variables from $\mathbf{MPRP}(\mathbf{P_A})$, and the coefficient column for an entering variable is dependent on how the path evolves through assignment categories, $s$, over time, $t$. We introduce the following notation to then define the Pricing Problem, Unconstrained (**PPU**).

$h_s$ : Dual variables from $\mathbf{MPRP}(\mathbf{P_A})$ for the lower bound constraints.

$q_s$ : Dual variables from $\mathbf{MPRP}(\mathbf{P_A})$ for the upper bound constraints.

$\lambda$ : Dual variable from $\mathbf{MPRP}(\mathbf{P_A})$ for the cohort size constraint.

$\bar{\beta}^t$ : The loss parameter for new paths; time-based average of known loss.

$y_{st}$ : Decision variable. Indicator for if the new path is on node $s$ at time $t$.

$\mathbf{PPU}(\mathbf{h}, \mathbf{q})$ :

$$\max_{y} \quad \sum_{t} \sum_{s} \bar{\beta}^t (h_s + q_s) y_{st}$$

$$s.t. \quad \sum_{s} y_{st} = 1 \qquad \forall t \qquad (3.3a)$$

$$y_{st} \in \{0, 1\} \qquad \forall s, t$$

The decision in **PPU** is $y_{st}$, which is the indicator that the newly created path is on node

$s$ at time $t$. The objective in **PPU** is to find the largest (most negative) reduced cost. For known paths, this would incorporate the loss parameters, $\beta_p^t$, but if we now consider paths for which we have no history for loss estimation, we can use the time average loss, $\bar{\beta}^t$. The constraint ensures that we select one assignment category for each time period; we can convert this new pricing problem into a shortest-path problem.

**PPU** does not address the fact that military leaders are interested only in a subset of paths. Practically, while there is room for flexibility adjustments to career path design policy, military leaders need to maintain some professional development restrictions. We need a method for adjusting **PPU** that allows us to capture only the 'reasonable' paths that leaders might allow.

To constrain ourselves to only $P' \subset P \setminus P_A$ paths that are 'reasonable' we constrain the feasible region by capturing the different aspects of professional development guidance that make up career path design policy. We introduce the following notation to define the Pricing Problem, Constrained (**PPC**), based on common considerations in military career path guidance. This allows us to incorporate guidance that mandates certain positions, required sequencing, assignment blocks, and timing.

$S_{min}$ : The set of assignment categories with period minimums. $S_{min} \subseteq S$

$S_{max}$ : The set of assignment categories with period maximums. $S_{max} \subseteq S$

$S_{prec}$ : The set of assignment categories with precedent constraints. $S_{prec} \subseteq S$

$S_{prec}^w$ : The set of assignments, $i$, that must precede $w$. $S_{prec}^w \subseteq S$

$S_{follow}$ : The set of assignment categories with follow-on constraints. $S_{follow} \subseteq S$

$S_{follow}^w$ : The set of assignments, $i$, that prevent a following $w$. $S_{follow}^w \subseteq S$

$S_{block}$ : The set of assignment categories that must occur in a block. $S_{block} \subseteq S$

$m_i$ : The minimum number of periods for assignment category $i \in S_{min}$.

$M_i$ : The maximum number of periods for assignment category $i \in S_{max}$.

$\mathbf{PPC}(\mathbf{h}, \mathbf{q})$ :

$$\max_{y,x} \quad \sum_t \sum_s \bar{\beta}^t (h_s + q_s) y_{st}$$

$$s.t. \quad \sum_s y_{st} = 1 \qquad \forall t \qquad (3.4a)$$

$$\sum_t y_{it} \geq m_i \qquad \forall i \in S_{min} \qquad (3.4b)$$

$$\sum_t y_{it} \leq M_i \qquad \forall i \in S_{max} \qquad (3.4c)$$

$$\sum_{j<t} y_{ij} \geq y_{wt} \qquad \forall t > 1, w \in S_{prec}, i \in S^w_{prec} \qquad (3.4d)$$

$$y_{wt} \leq 1 - \frac{1}{t} \sum_{j<t} y_{ij} \qquad \forall t > 1, w \in S_{follow}, i \in S^w_{follow} \qquad (3.4e)$$

$$x_{sij} \leq y_{si} \qquad \forall s \in S_{block}, i, j \qquad (3.4f)$$

$$x_{sij} \leq y_{sj} \qquad \forall s \in S_{block}, i, j \qquad (3.4g)$$

$$x_{sij} \geq y_{si} + y_{sj} - 1 \qquad \forall s \in S_{block}, i, j \qquad (3.4h)$$

$$y_{st} \geq x_{si,t+1} \qquad \forall i < T-1, t \in \{i+1, ..., T-1\} \qquad (3.4i)$$

$$y_{st} \in \{0, 1\} \qquad \forall s, t$$

$$x_{sij} \in \{0, 1\} \quad \forall s \in S_{block}, \quad i \in \{1, ..., T-2\}, j \in \{i+1, ..., T\}$$

(3.4j)

As in **PPU**, the primary decision variable is $y_{st}$, the indicator that the newly created path is on node $s$ at time $t$. Constraints (4b-c) ensure compliance for assignment categories with a minimum or maximum number of periods. Constraint (4d) ensures compliance if there is a prerequisite, $i$, for assignment $w$. Constraint (4e) ensures compliance if assignment $w$ cannot follow $i$. Ensuring that all assignments of a certain type occur together, in a block, is somewhat more involved. We introduce binary decision variable $x_{sij}$ as the linearization of $y_{si}y_{sj}$, with three constraints (4f-h). Constraint (4i) ensures that if the assignments at $i$ and $t+1$ are the same type, then the assignment at $t$ must be as well. If a certain assignment type is not allowed at a certain period, we can remove the variable. Formulation

**PPU** has $|S|T$ binary variables and $T$ constraints. A typical instance for our use-case has 160 binary variables and 16 constraints. Formulation **PPC**, particularly with assignment block constraints, grows substantially. It has $|S|T + |S_{block}| \sum_{l=1}^{T-2} l$ binary variables and $T + |S_{min}| + |S_{max}| + (T-1) \sum_{w \in S_{prec}} |S_{prec}^{w} + 4|S_{block}| \sum_{l=1}^{T-2} l$ constraints. A typical instance for our use-case has 265 binary variables and 499 constraints.

### Iterative Procedure

We now consider iterating this procedure for $k > 1$. We start by solving $\mathbf{MPRP}(\mathbf{P_A})$. If the solution is 0, then we know there is no additional benefit from adding a career path. If the solution is greater than 0, then we know that there could be some additional benefit to increased flexibility in career path design policy. We then iteratively add variables to **MPRP** through a column generation procedure using **PPC** as the sub-problem. We stop the procedure when there is no possible benefit to adding a path, or we have added $k$ paths. This is depicted in Figure 3-3, and specified in Algorithm 1.

**Algorithm 1:** Greedy Flexibility Augmentation ($\mathbf{GFA_k}$)

**Result:** $\mathcal{Z}^*_{\mathbf{GFA(P_A)}}$; Augmented paths, $\tilde{P}_A$
**Input :** $S$, $T$, $\mathcal{G}$, $n$, $c$, $r$, $b$, $P_A$, **MPRP**, $k$
Solve $\mathbf{MPRP}(\mathbf{P_A})$
**if** $\mathcal{Z}^*_{\mathbf{MPRP(P_A)}} = 0$ **then**
  Complete
**else**
  Counter = 0
  $\tilde{P}_A = P_A$
  Solve $\mathbf{PPC}(\mathbf{h}, \mathbf{q})$
  **while** *counter* $< k$ & $\mathcal{Z}^*_{\mathbf{PPC}} > 0$ **do**
    Construct $p$
    $\tilde{P}_A = \tilde{P}_A \cup \{p\}$
    counter++
    Solve $\mathbf{MPRP}(\tilde{\mathbf{P}}_\mathbf{A})$
    Update dual values, $h, q$
    Solve $\mathbf{PPC}(\mathbf{h}, \mathbf{q})$
  **end**
**end**

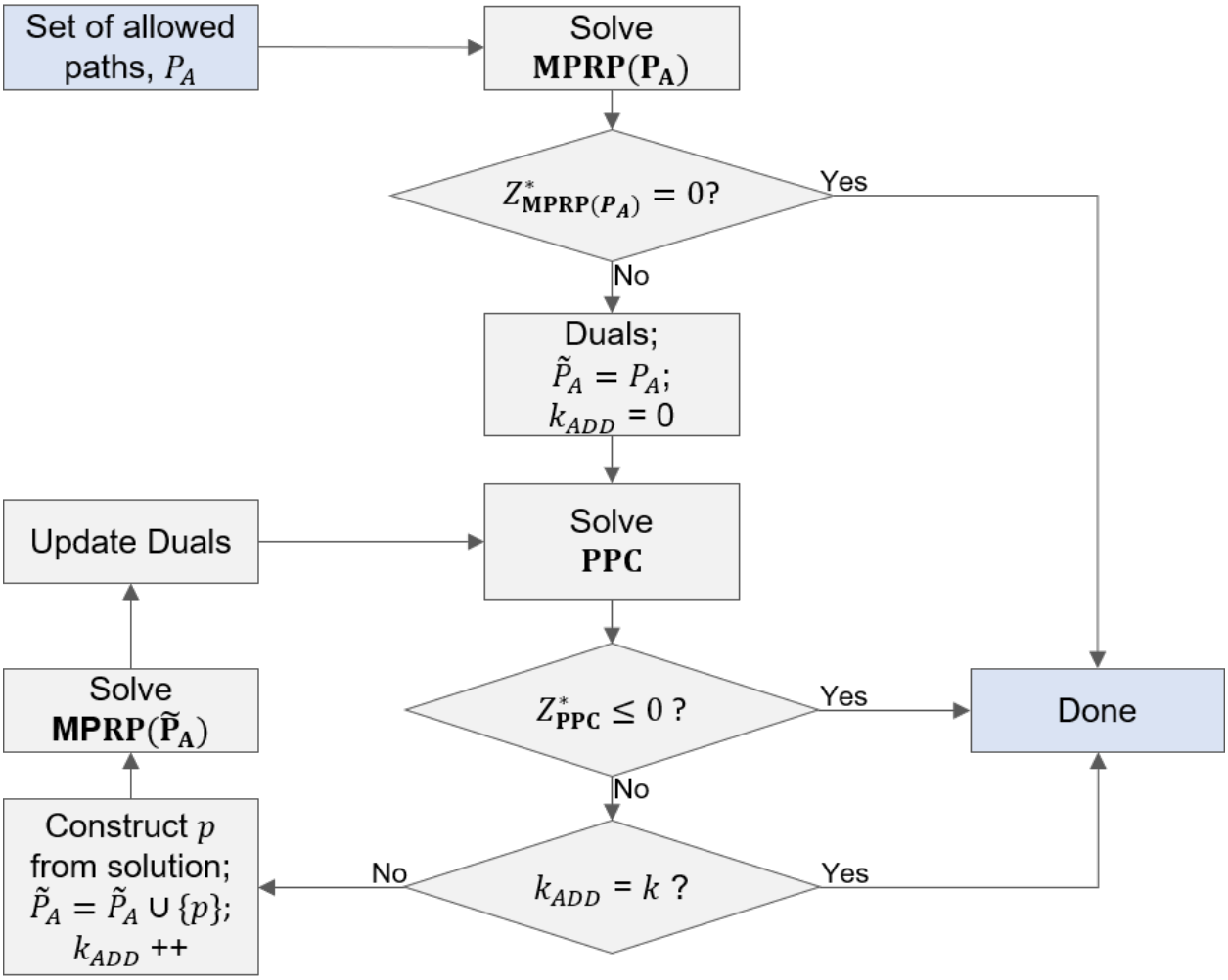


Figure 3-3: The Greedy Flexibility Augmentation Algorithm, which iteratively finds $k$ paths to add to the already allowed paths, $P_A$.

### 3.4.3 Connections Between the Two Flexibility Approaches

We use $\mathcal{Z}^*_F$ for the optimal objective value of formulation $F$, or of algorithm $F$ whose steps include solving to optimality.

**Proposition 3.4.1.** *$\mathcal{Z}^*_{\mathbf{MPRPF_k(P_A,P')}}$ and $\mathcal{Z}^*_{\mathbf{GFA_k(P_A)}}$ are monotonically decreasing in $k$: $\mathcal{Z}^*_{\mathbf{MPRPF_{k+1}(P_A,P')}} \leq \mathcal{Z}^*_{\mathbf{MPRPF_k(P_A,P')}}$ and $\mathcal{Z}^*_{\mathbf{GFA_{k+1}(P_A)}} \leq \mathcal{Z}^*_{\mathbf{GFA_k(P_A)}}$.*

Adding paths cannot hurt readiness, but it does not necessarily improve readiness. Our greedy algorithm, $\mathbf{GFA_k}$, shares this trait with the exact solution from $\mathbf{MPRPF_k}$.

**Proposition 3.4.2.** *For a fixed value of $k$, $P' \subseteq P \setminus P_A$, and a set of constraints in* **PPC** *as part of* $\mathbf{GFA_k(P_A)}$ *that produce a feasible region equal that generates a set of paths equal to $P'$:* $0 \leq \mathcal{Z}^*_{\mathbf{MPRP(P)}} \leq \mathcal{Z}^*_{\mathbf{MPRPF_k(P_A,P\setminus P_A)}} \leq \mathcal{Z}^*_{\mathbf{MPRPF_k(P_A,P')}} \leq \mathcal{Z}^*_{\mathbf{GFA_k(P_A)}} \leq \mathcal{Z}^*_{\mathbf{MPRP(P_A)}}$.

Proposition 3.4.2 connects the model from Section 3.3 with the flexibility approaches in Section 3.4. As we are minimizing, we see that the best possible readiness solution comes from using every possible path, set $P$, and is bounded by 0. We can improve on the readiness solution given our currently allowed paths, $P_A$, by adding paths. We have the possibility of more improvement when using the exact bi-level integer program $\mathbf{MPRPF_k}$ than the greedy algorithm $\mathbf{GFA_k}$. When using the exact formulation, we have the possibility of more improvement if we consider any additional path, $p \in P \setminus P_A$, not just those constrained in $P'$.Proofs of Propositions 3.4.1 and 3.4.2 are in Appendix B.3.

## 3.5 Structural and Computational Insights

This section ties together personnel readiness, the readiness maximizing formulation that accounts for carer path design policy, **MPRP**, and readiness as it appears in practice when personnel planners design the workforce supply. It highlights that in this model setting, **MPRP** provides an upper bound on the achievable readiness. The section also includes a computational comparison of the two flexibility approaches that shows the tractability and utility of the iterative algorithm $\mathbf{GFA_k(P_A)}$.

### 3.5.1 A Structural Insight into Personnel Readiness

We wish to understand how career path design policy limits readiness, and how that limitation is not visible at the workforce design level. Recall from Figure 3-1 that central personnel planners design the workforce supply based on targets, and without consideration for career path design policy. The execution agency maximizes personnel readiness with the supply of personnel set by the personnel planners, and the execution agency's decisions are constrained

by career path design policy.

Personnel planners design the workforce supply for a given career field based on the total number of positions in the force for that career field, and an estimate of that career field's personnel retention. We can interpret $\sum_{t=1}^{T} \beta^t$ as the expected amount of time that a service member will be available during the time considered by the model, and when multiplied by $b$ we have the expected 'person-years' available from a specific cohort. For example, if for a certain career field $\sum_{t=1}^{T} \beta^t = 5$ and $b = 20$, then personnel planners expect that 100 positions of all types for that career field can be filled. Then, the goal of the workforce designers is to use accessions, retention, transfers and promotion to ensure that $b = \frac{N}{\sum_{t=1}^{T} \beta^t}$.

When there are no restrictions on the allowable paths from career path design policy, then the personnel readiness maximizing solution to **MPRP** matches the aggregate view used by personnel planners in workforce design, where if $b = \frac{N}{\sum_{t=1}^{T} \beta^t}$, then the readiness shortfall is 0.

**Lemma 3.5.1.** *When there is no path-specific loss,* $\beta_p^t = \beta^t \quad \forall p, t$, *and job assignment categories are weighted equally,* $r_s = r \quad \forall s \in S$, $c_s = c \quad \forall s \in S$, *then:* $\mathcal{Z}^*_{\mathbf{MPRP(P)}} = max\big(0, c(rN - b\sum_{t=1}^{T} \beta^t)\big)$.

Lemma 3.5.1 connects **MPRP** to a view of readiness that is not constrained by career path design policy. To connect the career path constrained model to readiness in practice, Definition 3.3.1, we introduce notation for two variants of readiness, particularly applicable with job-types weighted equally and with no path-based loss.

The workforce designers view of readiness is on the availability of the personnel for jobs. Let $R_A = \frac{b\sum_{t=1}^{T} \beta^t}{N}$ be the *available readiness.* Let $R_{\mathbf{MPRP(P_A)}} = \frac{N - \mathcal{Z}^*_{\mathbf{MPRP(P_A)}}}{N}$ be the *career-path constrained readiness.* We can then connect these readiness variants.

**Proposition 3.5.2. MPRP** *provides an upper bound on the achievable readiness. In a system with no path-based loss, where* $S' = S \setminus schooling$, $c_s = r_s = 1 \quad \forall s \in S', c_{schooling} =$

$r_{schooling} = 0$, *and* $N = \sum_{s \in S'} n_s$*:* $R \leq R_{\mathbf{MPRP(P_A)}} \leq R_A$.

This is the key connection for understanding that path restrictions limit readiness. The result of the readiness associated with the optimization formulation, $R_{\mathbf{MPRP(P_A)}}$, provides an upper bound on the achievable readiness in practice. And, when $R_{\mathbf{MPRP(P_A)}} < R_A$, we know that career path design policy has limited achievable readiness. This is precisely the situation we are interested in exploring numerically. A better understanding of this situation by personnel leaders can help elucidate the connection between setting career path design policy and the eventually realized readiness after individuals are assigned to jobs. We address additional analytic aspects related to the optimization as a set function in Appendix B.2. Proofs for Lemma 3.5.1 and Proposition 3.5.2 are in Appendix B.3.

### 3.5.2 Computational Comparison of the Flexibility Approaches

We evaluate the difference between the greedy algorithm, $\mathbf{GFA_k(P_A)}$ and the bilevel mixed integer program $\mathbf{MPRPF_k(P_A, P')}$ for run-time and performance.

#### Run-times

Our goal horizon for the use-case is $T = 16$, which leads to $|P'| \approx 1e6$ additional paths for consideration. Strictly enumerating all options requires iterating through many instances of a problem comparable in size to $\mathbf{MPRP(P_A)}$. Even with run-times at a fraction of a second, this is undesirable, particularly for combinatorial challenges with $k > 1$. $\mathbf{MPRPF_k(P_A, P')}$ includes binary variables for every additional path under consideration, and is intractable for $T = 16$. In Table 3.1, we show run-times for 100 realistic path-size instances at $T = 8$ (with $|P| \approx 100$ and $|P'| \approx 5000$), and randomly generated $n_s$. The run time of $\mathbf{GFA_k}$ scales linearly with the run-time of **MPRP**, as expected. While at $T = 8$, $\mathbf{MPRPF_k(P_A, P')}$ is solvable for most instances, the parameters from the random instances of $n_s$ that are hardest to solve are those that are of most interest to us – places where the existing career path design policy limits the readiness, and there is room to improve with flexibility. In Table 3.1, we see this at $k = 3$ where 14% of instances timed out with no improvement in the optimality

gap. Computations were performed with Julia 1.2.0 using JuMP 0.21.0 and Gurobi 0.9.11 [12], [22].

| $k$ | **MPRP** | $\mathbf{MPRPF_k}$ | $\mathbf{GFA_k}$ |
|---|---|---|---|
| 0 | 0.005 | - | - |
| 1 | - | 0.55 | 0.08 |
| 2 | - | 2.43 | 0.16 |
| 3 | - | 26.1 (14%) | 0.24 |
| 4 | - | 36.2 (9%) | 0.31 |

Table 3.1: Computational times, in seconds, for 100 use-case instances with varying $n$. For $\mathbf{MPRPF_k}$, the number in parenthesis is the percent of iterations that failed to solve with an optimization solver time limit of 10 minutes.

**Greedy Algorithm Performance**

We want to understand the quality of solutions from $\mathbf{GFA_k}$ compared to the exact solution found with $\mathbf{MPRPF_k}$. We can solve instances with thousands of additional paths (but not hundreds of thousands), and so we baseline the performance comparison between models with $T = 8$, and realistic use-case parameters for $P_A$, $P'$, $r$, and $c$. For 100 randomly generated instances of $n$, we see the performance difference between methods depicted in Figure 3-4. Our comparison metric is the proportion of the flexibility improvement that is possible, but not found with $\mathbf{GFA_k}$.

We see that $\mathbf{GFA_k}(\mathbf{P_A})$ provides much of the benefit that we find in the optimal solution to $\mathbf{MPRPF_k}$, with results closer to 0, particularly as $k$ increases. We proceed to scale $\mathbf{GFA_k}(\mathbf{P_A})$ for our use-case.

## 3.6 Illustrative Example with U.S. Army Captains

Our use-case is motivated by a persistent gap between observed readiness and the available readiness for a subset of personnel in the U.S. Army. Figure 3-5 depicts this gap in readiness, over time, for active duty U.S. Army officers. Clearly there are enough officers for every job,

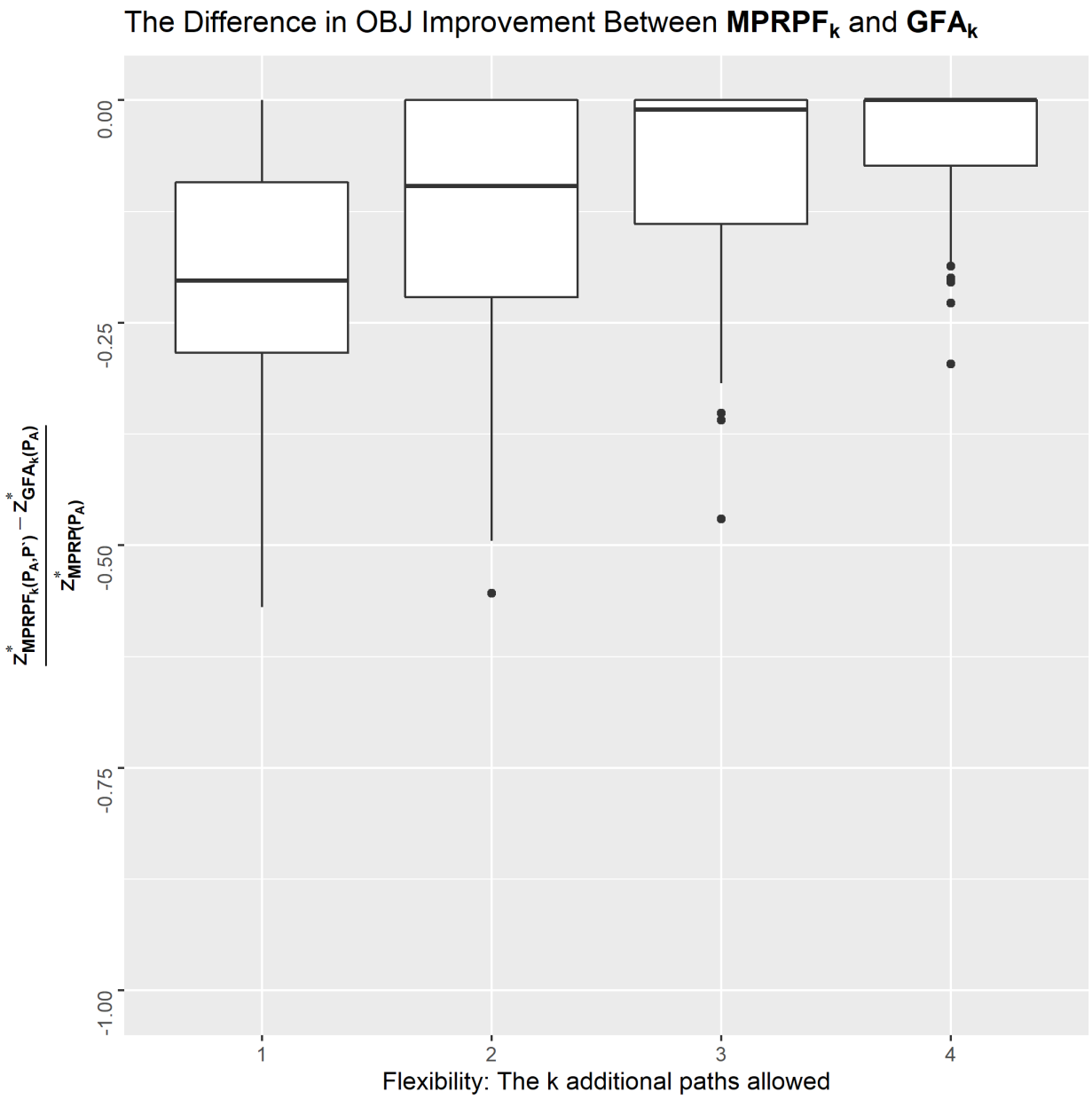


Figure 3-4: For 100 problem instances, we take the difference between the optimal solution for $\mathbf{MPRPF_k}$ and the solution to $\mathbf{GFA_k(P_A)}$. We normalize based on the objective value with no flexibility.

but unfilled jobs persist, and this picture is similar within each specialty. This use-case shows that career path design policy limits readiness, per Proposition 3.5.2. We outline the use-case and describe three areas of insight for personnel leaders: the impact of career path

design policy on readiness, the impacts of structure change without associated career path design policy changes, and the readiness benefit of career path flexibility.

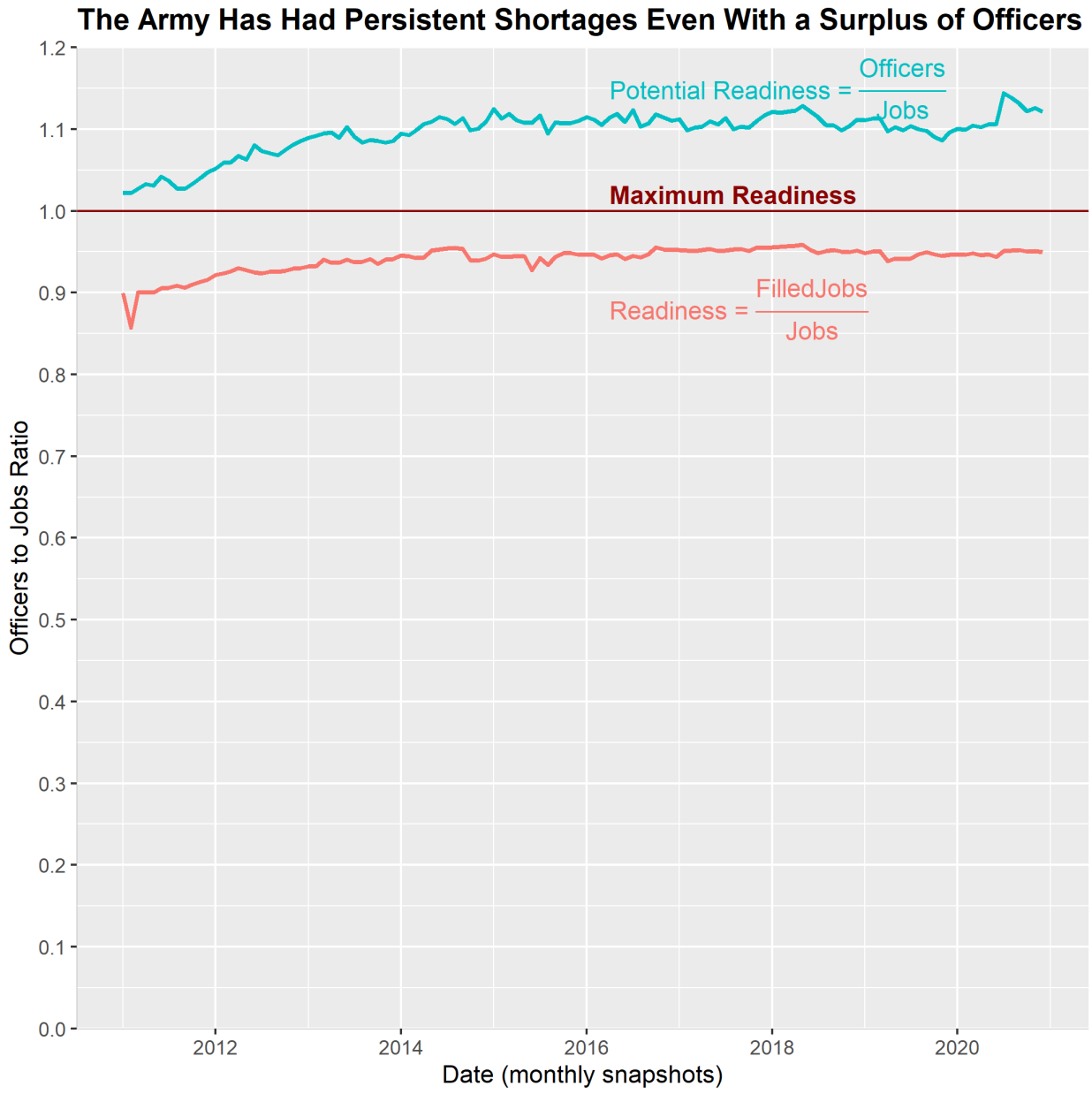


Figure 3-5: A view of Army officer manpower over time, where we see that there are more than enough officers for every job (teal above 1), but that unfilled jobs persist. This is true for various subset views as well (by rank, by specialty, etc). Data from Total Army Personnel Database Snapshots.

### 3.6.1 Army Use-case Details

Officers enter the Army with a service obligation that commits them to a certain number of years, typically three to five. Officers commission in the Army as second lieutenants, and typically spend four years as a lieutenant before promoting to captain (if selected), where they serve for an additional six years before promotion to major (if selected). For most officers, their initial time commitment to the army expires early in their time as a captain, which means that attrition is a key aspect of modeling officer personnel strength. Career path design policy for captains includes the requirement to attend a branch-specific Captains' Career Course (CCC), required experiences called 'Key and Developmental' (KD), prerequisite experiences for some assignments, and assignment length minimums. We look at officers between the 3rd and 10th year of service to fully capture the career path design policy around captains. A typical annual cohort from this time-frame has 5500 officers in 16 different branches.

We leverage data on officer assignments Total Army Personnel Database snapshots courtesy of the Army's Office of Economic and Manpower Analysis. The assignment data includes de-identified information about all active duty commissioned officers who served between 2002 and 2018. It includes an annual snapshot of various attributes, and a monthly snapshot of each officer's unit assignment. This latter information forms the basis for our development of the historical path set. Additionally, we used information to estimate the model loss parameters, $\beta$, based on the proportion of officers leaving the service each time period. For job categorization and the demand in the model, we use current Army structure documents and recent monthly unit personnel readiness snapshots from the Army's Human Resources Command.

## Path and Parameter Generation

The critical input to the formulation is $P_A$, the set of currently allowed paths, based on the professional development guidance published periodically by the different military services,

for example in *Officer Professional Development and Career Management* for Army officers [60]. This guidance has many different elements that taken collectively, define the allowable paths. Consequently, we estimate the true set of allowed paths with historical data on personnel assignment types and timing. This provides a realistic look at what actually occurs when the different portions of professional development guidance are collectively applied. Additional information on the types of career path design policy guidance and the estimation approach are in Appendix B.1.

For this use-case, we focus on a single specialty, across multiple years, and estimate the allowed paths from the assignment histories of 4570 officers. From this we also estimate $P_A$, the typical cohort size $b$, and the retention parameters, $\beta_p^t$. Of note, we found no evidence of path-specific loss, and use only time-based loss for the use-case analysis, where $\beta_p^t = \beta^t \quad \forall t \in \{1, ..., T\}$. We use an actual time horizon of eight years, and consider 6 month period lengths, for $T = 16$. We develop the assignment categories based on domain knowledge of the branch, and then develop the job count, $n_s$, from the Army's master job listing, with the same mapping for $S$. We mirror the Army's readiness measures by weighting all jobs, except for schooling, equally.

### 3.6.2 Readiness and Professional Development Guidance

#### Use-case Readiness

A recent snapshot from early 2021 of use-case branch officer strength shows available readiness of $R_A = 1.01$ and actual readiness of $R = 0.92$. This measure includes some elements that are not part of the model here, such as tactical adjustments to assignment timing and slight career path adjustments for some officers. It is in-line with our modeled readiness $R_{\mathbf{MPRP}(\mathbf{P_A})} = 0.91$. The unfilled positions, $D$, in practice are not uniformly spread across the different assignment types, but concentrated in post-KD assignments. Even after the tactical adjustments made in the real, not steady-state system, we see $R^{\text{Pre-KD}} = 0.98$ and $R^{\text{Post-KD}} = 0.82$.

A number of factors impact the gap between $R$ and $R_{\mathbf{MPRP(P_A)}}$ that are not modeled in the steady-state system, including variation in cohort size, variation in retention, local commander decisions on timing, and a different career path progression than what we see in the optimal solution. In general, the career paths from the optimal solution to **MPRP** maximize the officer-years for post-KD assignments. Officers move as quickly as possible into the required course, CCC, without leaving a readiness gap in the LT assignments. Officers leave the CCC and move through the KD assignments with the minimal amount of required time, and on to post-KD assignments. Deviations from this, even ones that are locally beneficial – such as extended command times and delayed schooling – decrease personnel readiness over time for the entire system.

### Readiness as Personnel Volume Increases

When $R < 1$, which commanders across the Army see whenever they have an unfilled job, a frequent response is the claim that the branch or specialty needs additional officers. However, if $R_{\mathbf{MPRP(P_A)}} < R_A$, increasing the number of available officers can have little effect when career path design policy is still in place. Figure 3-6 depicts the readiness of a specialty as a function of the annual cohort size for our use-case.

For the use-case branch and typical cohort sizes entering the third year of service, we see shortages of 9%, $R_{\mathbf{MPRP(P_A)}} = 0.91$, even when there are a surplus of officers, $R_A > 1$. The structural limitations of career path design policy, if kept in place, prevent an increase in readiness despite an increase in cohort size. Additionally, a cohort size increase when there is no adjustment in the career path design policy leads to a surplus of officers in jobs upstream of the bottleneck, which could lead to increased officer dissatisfaction and attrition (not modeled here).

### Readiness Based on the Bottleneck Capacity

The primary driver of the readiness gap, $R_A - R_{\mathbf{MPRP(P_A)}}$ is the number of KD assignments, when they are a prerequisite for other assignments. Other factors include school throughput,

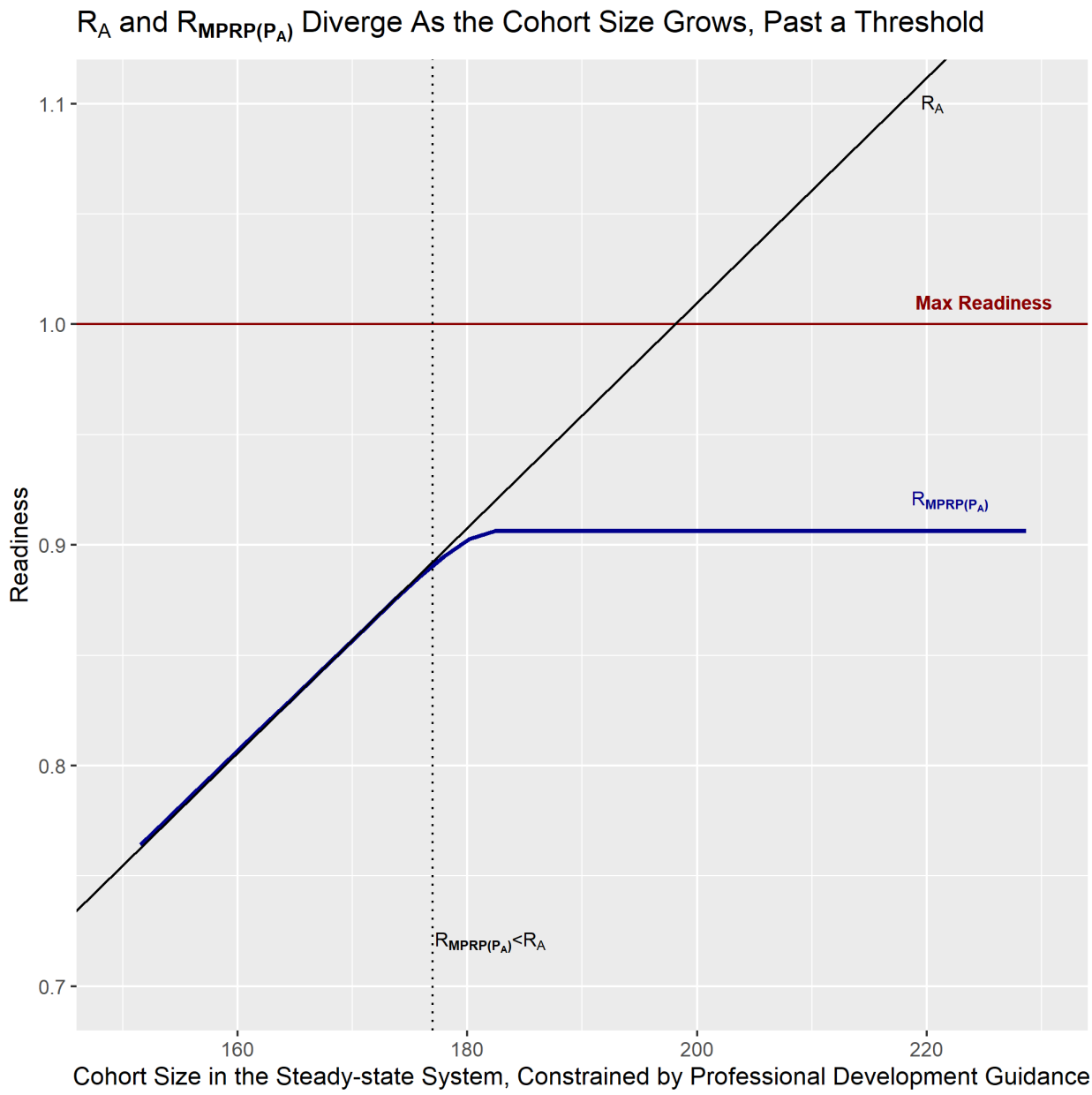


Figure 3-6: We see for a fixed number of jobs where the optimal constrained readiness diverges from the available readiness: $R_{\mathbf{MPRP(P_A)}} < R_A$. Below a certain threshold of cohort size (here at a cohort of 178 officers every 6 months), the available readiness matches the optimal constrained readiness. Above that threshold, the readiness measures diverge as the allowed set of career paths limits the flow of officers creating a surplus upstream and shortages downstream. Above a certain point (here at 198), the available readiness shows a surplus of officers, even though the readiness, $R$, cannot increase.

and the amount of time officers are available for post-KD assignments. We fix model parameters except for the number of KD assignments, and view the expected shortfall as a function of the proportion of KD assignments out of the total number of assignments. In Figure 3-7, there are a constant number of jobs and a constant number of officers, but the proportion of prerequisite satisfying jobs is increasing, thus increasing the possible throughput in the system until readiness is maximized.

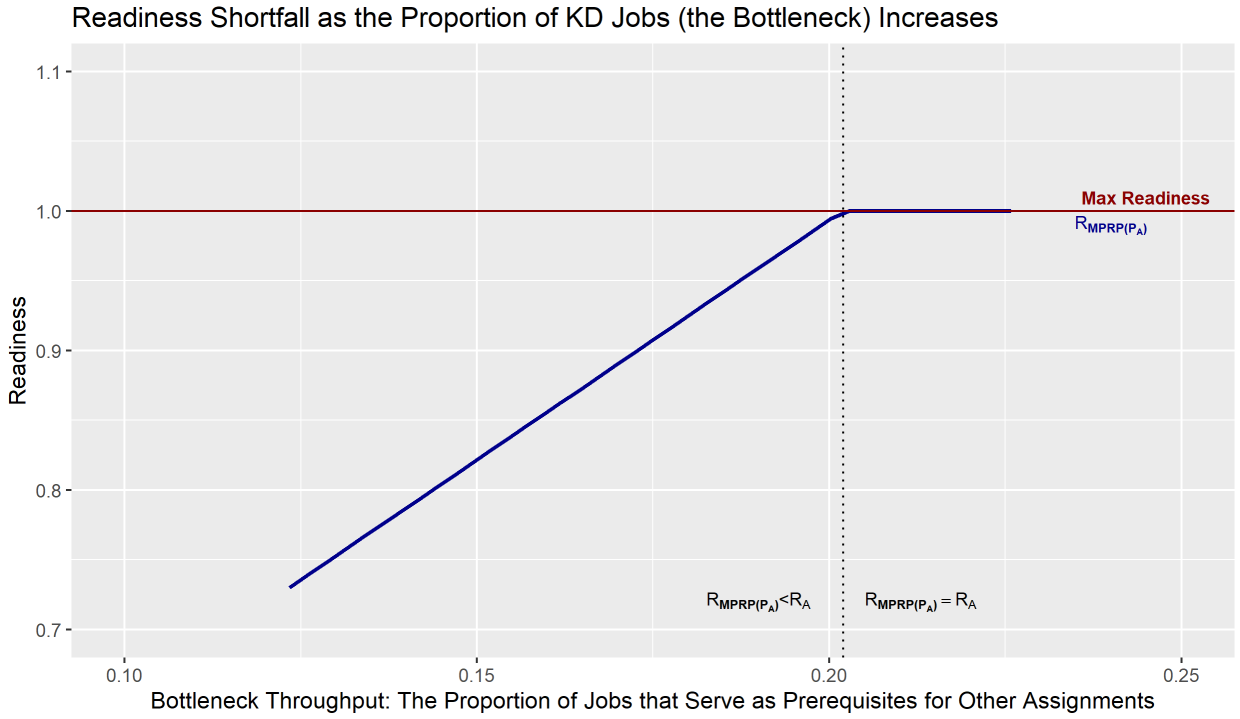


Figure 3-7: We see that for a fixed number of jobs, with $R_A = 1$, the proportion of jobs that serve as prerequisites drives the readiness shortfall, given the allowed career paths. This is a measure of throughput capacity in the system.

This emphasizes that to understand when $R_{\mathbf{MPRP(P_A)}} < R_A$, we have to include the structure with career path design policy, not the policy in isolation. Another branch with similar guidance to the use-case branch, but with a different mix of assignment types, could have no bottleneck.

### 3.6.3 Readiness in Force Design Analysis

As described in Section 3.2, the implicit assumption of force shaping models is that when $R_A = 1$, then $R = 1$. When the Army downsizes, and the number of jobs for each specialty decreases, available readiness appears to increase: as $N$ decreases, $R_A$ increases. But, when we view readiness through career path design policy in $R_{\mathbf{MPRP(P_A)}}$, this is not the case. Sometimes a decrease in structure that has an assumed increase in available readiness, actually hurts readiness.

We can view this by taking our use-case with set career path design policy, and consider what would happen with 10% fewer jobs. If $N$ decreases by 10%, then $R_A = 1.1$, and the manpower shaping process looks to decrease the number of officers accordingly.

We take the 10% decrease in structure and apply it to the parameters of $\mathbf{MPRP(P_A)}$ in five ways. We proportionally allocate the change across assignment types, and we allocate all of the change to one of four assignment types, in turn: lieutenant positions, which are at the beginning of the career paths (LT); pre-bottleneck captain positions (Pre-KD); bottleneck positions (KD), and positions that require the bottleneck as a prerequisite (Post-KD). The results on $R_{\mathbf{MPRP(P_A)}}$ are shown in Figure 3-8.

When we decrease structure, the increase in $R_A$ does not manifest with $R_{\mathbf{MPRP(P_A)}}$. A decrease in bottleneck positions hurts readiness, as the throughput capacity of the system is lower. These final points motivate the inclusion of this model's view of readiness in Army headquarters discussions about future structure composition. For many specialties, these bottleneck positions occur in the types of organizations that might be considered in a force structure decrease, and without some type of career path design policy adjustment, readiness could suffer.

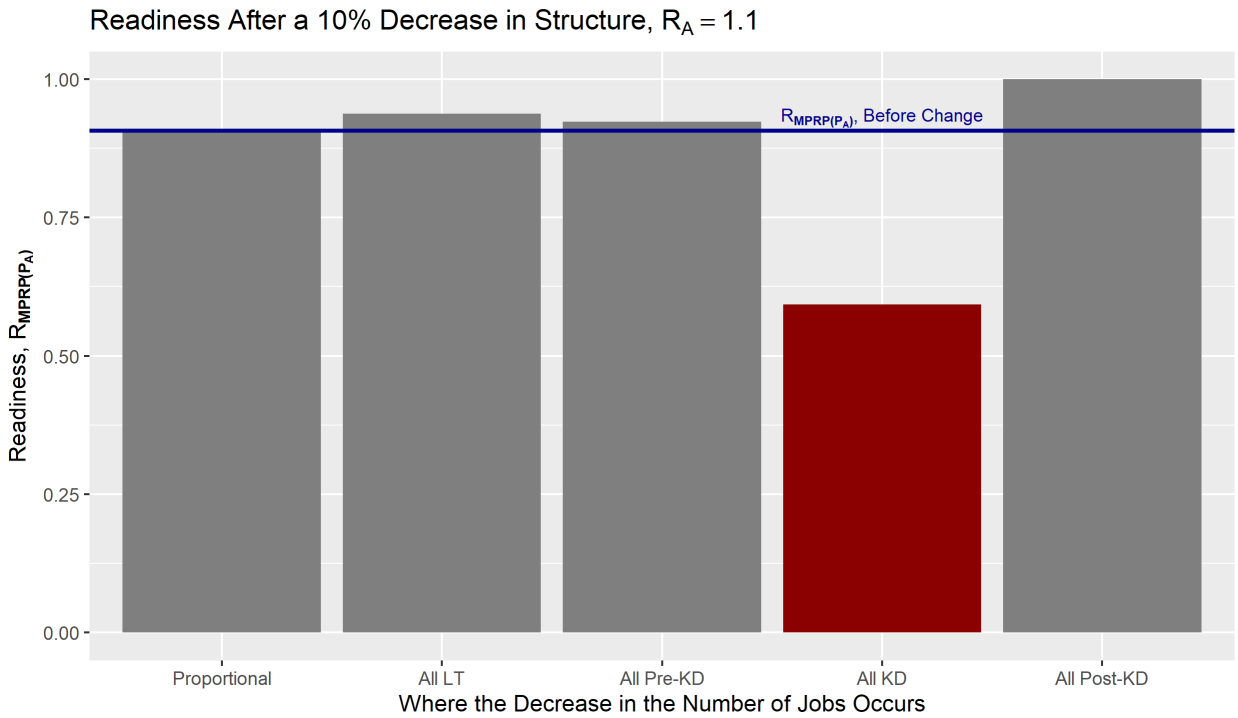


Figure 3-8: The impact of decreasing structure on readiness, with continued restrictions from career path design policy $R_{\mathbf{MPRP(P_A)}}$. Fixing all parameters except for $n_s$, we change $N$ by 10%, with the change occurring in one of five ways (colors). With the same number of officers, after a decrease, $R_A = 1.1$. The magnitude of the change in $R_{\mathbf{MPRP(P_A)}}$ does not match what is expected with $R_A$, and if the job-type at the bottleneck decreases, then readiness goes down, even when the available readiness appears to rise.

### 3.6.4 Readiness and the Benefit of Flexibility

Given a situation where $R_{\mathbf{MPRP(P_A)}} < R_A$, we can improve readiness by changing the career path design policy. Section 3.4 outlines an approach where increasing flexibility is modeled by the addition of career paths, with $\mathbf{GFA_k(P_A)}$. For our use-case branch, we focus on a single version of the pricing problem that generates additional paths, **PPC**. We relax the constraints on the timing of different assignments, and, we allow certain assignments that previously had to follow the KD assignments, to precede KD assignments. Figure 3-9 shows the results.

The best solution to $\mathbf{GFA_k(P_A)}$ that includes a single additional career path removes the difference between $R_{\mathbf{MPRP(P_A)}}$ and $R_A$. This solution uses the same types of career paths

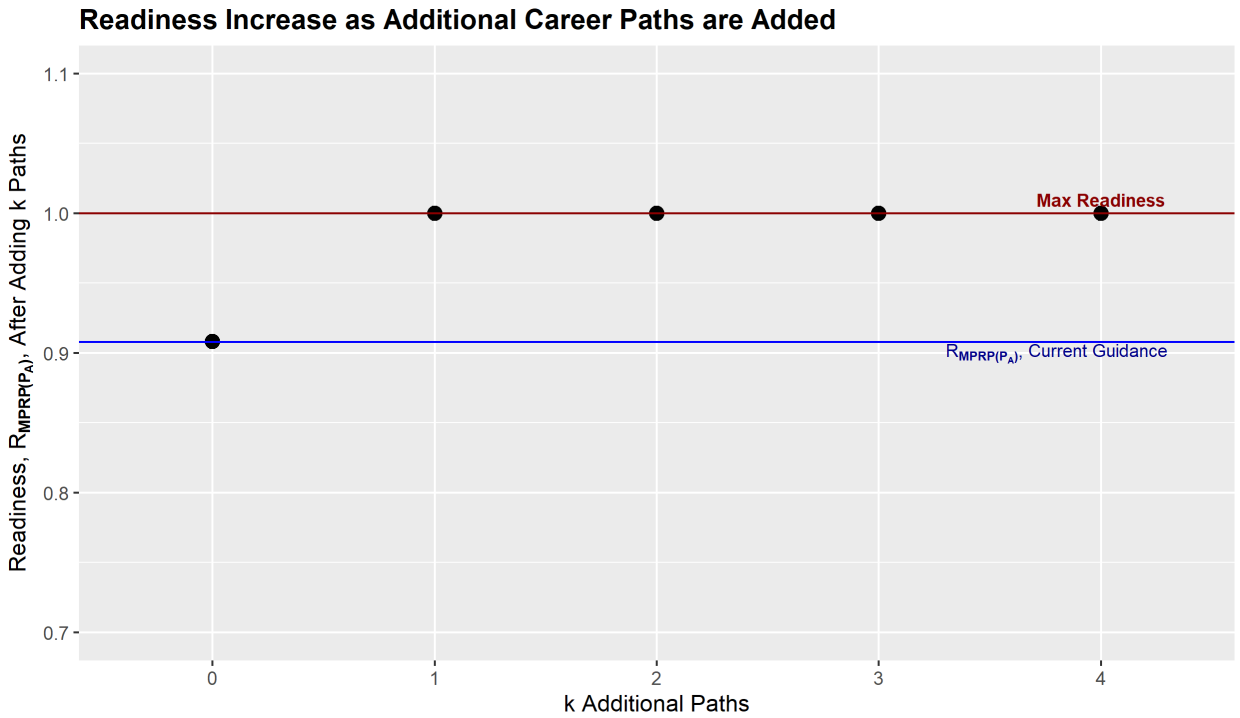


Figure 3-9: We maximize $R_{\mathbf{MPRP(P_A)}}$ with the addition of a single career path that removes the 'KD' prerequisite from certain assignments, which we find using $\mathbf{GFA_k(P_A)}$.

that are currently in practice, and adds a path where officers delay the KD assignment by taking another type of assignment between schooling and KD.

This solution does not require most officers to adjust from the traditional career paths, and it does not require that all traditionally post-KD assignments be eligible for pre-KD officers. We close the readiness gap with a solution that requires 25% of officers to take a traditionally post-KD assignment before KD, but we only need to designate 34% of the assignments for this purpose. Currently 14% of officers take this type of assignment, even though it is not part of the traditional career progression. It happens as a tactical, ad-hoc, adjustment when leaders recognize a particularly large gap between $R$ and $R_A$, and then make in-the-moment adjustments to some officers' upcoming assignments. These adjustments are treated as short-term for the officers in question, and the officers move into KD jobs as quickly as possible following an assignment of minimum length. The result here motivates the idea that this should be a deliberate practice, for more officers, and that it can address the structural

readiness shortages.

Other options, based on different constraints in **PPC**, result in different additional paths that reduce the readiness price to different degrees. Only relaxing the timing, but not the assignment sequencing, provides a small readiness benefit. Requiring less time at the bottleneck which increases the throughput, completely removes the gap. Practically, leaders may consider combinations of these options to remove the readiness shortfall caused by career path design policy.

## 3.7 Conclusions

In this chapter we propose new personnel planning models that help the military understand the impact of career path design policy on personnel readiness. We determine the highest achievable readiness in a non-stochastic setting given that personnel must follow a set of defined career paths. We then consider how the military could add flexibility to the career path guidance to increase readiness. We propose two methods for finding the best $k$-sized subset of paths to add to the existing set of allowed career paths.

Extensions to this model fall into two main categories. First, the current work is entirely deterministic, and there could be value in understanding how uncertainty in key parameters, particularly retention estimates, impacts the solution. A robust optimization approach that considers an uncertainty set of possible retention parameters would be one avenue. Second, there are connections between various job specialties. Sometimes there are job types that could be filled by an individual in more than one specialty, and there are also individuals transferring between specialties. Both of these motivate a larger framework that considers multiple job specialties, where the officer flow within each specialty is based the models here.

This approach provides insight for personnel leaders on the impact of career path design policy on personnel readiness. This impact depends heavily on structure in the presence of policy that creates a bottleneck, and also depends on required courses and timing. The

impact does not fall evenly across all assignment types, but heavily on those jobs that require a certain assignment as a prerequisite. We show why a version of readiness that includes career path design policy is an important, missing, aspect in force design and can highlight when adjustments are needed in the policy. We then describe how to find such an adjustment in career path design policy by adding flexibility to the set of allowed career paths. We characterize, for a use-case, specific career path adjustments that could lead to this improvement.

We include three appendices: one with additional details on the model, one that considers $\mathbf{MPRP}(\mathbf{P_A})$ as a set function, and the proofs for Lemma 3.5.1 and Propositions 3.4.1, 3.4.2, and 3.5.2.

# Chapter 4

# Bi-objective Matching with Market Composition Control

## 4.1 Introduction

Matching applicants to jobs using participant preferences occurs in a variety of labor market settings, including school choice and medical residency. A frequent requirement is that the outcomes must be stable, which means that no pair of applicant and job prefer each other over their assigned matches. When there are multiple stable solutions, another goal is that the matching is optimal for applicants; that is, the applicants prefer the selected matching over any other stable matching [29]. In the classic model, there are an equal number of applicants and jobs who each submit ordinal preferences. These preference lists are used to determine the matching between applicants and jobs. The deferred acceptance algorithm [29] provides a fast method for finding the stable matching most preferable to applicants (or jobs depending on the way the algorithm is executed), and the preference-based matching literature has studied a range of extensions and additions to this classic setup.

The U.S. Army recently adopted a matching market mechanism to determine the assignments

for the thousands of officers who move to new jobs every year. Twice a year officers who will move during the next twelve months enter a marketplace to preference the available jobs. The units with the available jobs similarly submit their preferences for the moving officers. An officer-proposing deferred acceptance algorithm determines the resulting match based on those preferences. A distinguishing feature of this market is that frequently, before the marketplace opens, the Army's central personnel managers must decide, based on readiness, which jobs will enter the market, as there are more job openings than moving officers.

Motivated by this military application, in this chapter we consider a market where there are more jobs than applicants, and a central decision-maker selects a subset of jobs, equal in size to the number of applicants, to enter the market. Once the subset of jobs is identified, the market proceeds in the classic manner using an applicant-proposing deferred acceptance algorithm. For this market, the central decision-maker has two goals: readiness and overall applicant satisfaction. Readiness is determined solely by the selected subset of jobs and to what extent it fits the organization's staffing needs, and applicant satisfaction is determined by the matching. The current process sequentially optimizes, first for readiness, and then for applicant satisfaction. Jobs are grouped by unit, and the readiness objective measures the number of jobs in each unit that will be filled by an applicant, via the market. Satisfaction is measured with the average applicant preference for their matched jobs.

Some jobs are critical or important enough to readiness that they require inclusion in the selected subset of jobs that enters the market. In other cases, there may be many different combinations of selected jobs that produce similar overall readiness for the decision-maker. If we decide on the job subset and the matching simultaneously, instead of sequentially, we can optimize for a range of trade-offs between readiness and satisfaction outcomes. This would enable a decision-maker to assess different solutions that each produce a readiness and satisfaction outcome where no other solution is improves on one outcome without a loss in the other outcome: a Pareto frontier. The decision-maker could still prioritize readiness. However, if the decision-maker wanted to increase applicant satisfaction to help increase

retention, then he could select another solution with slightly lower readiness but higher applicant satisfaction.

This chapter focuses on the military process described above that motivates the model, and we validate the approach using data from this use-case market. Our central research questions are: (i) how do we tractably find a set of solutions to this market that captures trade-offs between the two goals of readiness and applicant satisfaction on or near the Pareto Frontier? (ii) in the use-case, to what extent do we see realistic trade-offs where a small decrease in readiness provides a large increase in satisfaction?

The military's current two-stage decision-making process for this market, leads to the following. In the first stage, we select the job subset that maximizes readiness given information about how jobs map to units, and the current personnel status of those units. In the second stage, we take the set of applicants and the selected subset of jobs, along with their preferences, and find the applicant-optimal matching with the deferred acceptance algorithm.

In contrast to the existing process, the proposed approach considers a bi-objective optimization that simultaneously decides on the job subset and subsequent stable matching, and is driven by both the readiness achieved by the selected jobs and the resulting satisfaction from the matching. This linear mixed integer program (**OAT**) ensures a stable solution between the applicants and the selected job subset, and a scaling parameter in the objective function captures the importance given to satisfaction versus readiness. We solve this formulation for a variety of values for the objective function scaling parameter to generate a set of non-dominated solutions that define a Pareto frontier. This approach works well computationally up to a certain market-size, but is intractable for large markets.

We then develop an algorithm based on a local search procedure that we call the *one-swap chain*. The algorithm starts with the solution from the current process, and then leverages the stability criteria to iteratively swap one job from the selected subset for one not selected in a manner that improves overall satisfaction. From the set of solutions found during this

process, we select a subset that approximates the true Pareto frontier. This algorithm runs in polynomial time and is tractable for very large market sizes.

We apply the approach to data from the 2020 U.S. Army officer assignment marketplace. The key insight for Army personnel leaders is that, given the preference information, the job subset decision effectively determines both objectives as the ensuing matching algorithm is fixed. By considering the preference information when making the job subset decision, there is room to improve the overall satisfaction at little expense to readiness. Moreover, the methods here do not provide a single outcome, but a range of alternatives for a decision maker.

### Contributions

This chapter makes two main contributions. First, we model the important use-case of a market where a decision maker controls one side of the market make-up and has two objectives, specifically readiness and satisfaction. Additionally, we provide practical computational approaches to solve large instances. First we use a mixed integer linear programming formulation and develop an associated weighted-sum algorithm that finds a set of solutions on the Pareto frontier for the two objectives. Second, we develop a polynomial-time local-search algorithm that leverages the stability requirement of the solution to iteratively swap jobs in the selected subset while improving the resulting satisfaction objective. We store all the matchings found during this process and then find a set of these solutions that approximates the Pareto frontier. This approach is very tractable even for large problems.

The analytical models and algorithms are then used to study the related tradeoffs in the context of an important use-case of a military process to assign personnel to jobs. Moreover, we apply this approach to real data from the military to obtain important managerial insights. In particular, applying the approach to U.S. Army officer assignments highlights opportunities for personnel leaders to consider allowing some flexibility in the chosen subset of jobs that could lead to slightly lower readiness, but in return will significantly increase

officer professional satisfaction, which could greatly benefit the Army long-term.

### Chapter Outline

Section 4.2 highlights related literature. In Section 4.3 we develops our matching model with job subset control, **OAT**. Section 4.4 presents the exact solution approach for defining a portion of the Pareto frontier. It also develops the polynomial-time local search algorithm that provides a set of solutions that approximate the one found with the exact solution method. Section 4.5 presents computational results, including a use-case based on preference data from the assignment marketplace of thousands of U.S. Army officers and hundreds of units. Section 4.6 provides concluding comments.

## 4.2 Related Literature

There is a rich body of literature on preference-based matching including work on medical residency [54], college admissions [29], and school choice [4]; see [48] for an extensive overview. Military applications include recent work on initial specialty selection for newly commissioned officers [23], [34]. The general matching problem has many variants, and extensive work exists on how various constraints impact existing properties (for example: [38], [8]).

One of the objectives that motivates this work is a measure of organizational readiness, which is very important in the military context. This objective is most closely tied in the current matching market literature to situations where applicants apply to organizations for jobs, and those organizations have a limit on the number of available jobs, as an upper quota, for example in a school or hospital residency. More infrequently, a lower quota might also exist as a threshold for executing a program, for example the minimum number of students needed for a certain graduate program [6]. Lower quotas complicate traditional market clearing algorithms. Work in [24] and [28] addresses questions associated with ensuring demographic minimums, such as racial goals for school composition. In this context, a hard lower quota is when a certain number of applicants must match to jobs for the matching outcome to be valid.

With a soft lower quota, like that in [24], every organization prioritizes meeting its lower quota, but not at the expense of applicant preferences. The lower quota is transformed into a part of an organization's preferences as opposed to a separate criteria. In this chapter, we maintain a distinction between the preferences jobs have for applicants and the readiness of the organizations. But, similar to the soft lower quota, we treat the organization's readiness requirements as goals and not necessities. This is further described with the model in Section 4.3. The view of mapping jobs to units that we take allows additional hierarchical groupings where units are in super-units such that across the grouping layers we have one lower goal and multiple upper goals, or multiple lower goals and one lower goal.

Many matching schemes can also be modeled with optimization formulations. The integer formulation for the canonical one-to-one matching with stability constraints can be solved as a linear program, as the feasible region is an integral polytope [63], [55]. Additional decision criteria and constraints often break the integrality property of the polytope and require a mixed integer programming formulation. There is recent work employing integer programming formulations in [6], [7], and [5].

Algorithmically, our solution approach using one-swap chains, developed in Section 4.4, shares some characteristics with work by Erdil and Ergin in [26] and [27]. They consider the structure of ties in preference lists, and how applying tie-breaking rules before using existing matching algorithms can add artificial stability constraints. They develop methods for pareto-improving applicant results from an initial stable matching by looking for trades in the matching, where applicants exchange assigned jobs. This can help when these trades improve the outcome for the applicants while maintaining stability if each job is indifferent between the original assigned applicant and the new applicant. The building block of our method is a swap in the matching, where one applicant exchanges his current match for a new job that is currently unmatched. These swaps chain together to create an exchange similar in style to the one in Erdil and Ergin. The primary difference is that our swaps are initiated by a new, unmatched job that replaces a currently matched job, and not because

of preference indifference. In our case, the unmatched job has not previously been part of the stability determination or the applicant satisfaction.

## 4.3 Model

This section presents the optimization formulation for the bi-objective stable matching that includes the job subset selection, **OAT**, or *Officer Assignment Trade-offs*. We first describe the model setting, and then the formulation that captures both the job subset and matching decisions. We conclude with an explanation of how this formulation relates to the current process from the motivating use-case of U.S. Army officer assignments.

### 4.3.1 Model Setting

We have a set of applicants (officers), $O$ and jobs, $J$, where jobs are grouped into units, $u \in U$. In Figure 4-1 below, each unit's to-be-assigned jobs appear in a different color. Each unit $u$ has a readiness target band, $[\underline{g}, \overline{g}]$, that specifies the number of jobs that need to enter the market for that unit to meet its readiness goal. Each applicant submits an ordinal preference list for all of the jobs, $P_o$, and each job submits an ordinal preference list for all applicants, $P_j$. These preference lists define a strict preference relation. There are two decisions that in the current process are considered independently of each other. First, the decision-maker selects a subset of jobs, $J' \subset J$, to enter the market. Note that $J'$ is equal in size to the number of applicants, $|O|$. The objective driving this decision is to get each unit as close as possible to its readiness target range. Each unit has associated shortfall and overage penalties if it does not meet its readiness target range. In the Army officer use-case, this decision is made by personnel managers at the central human resources organization for the Army. Then, the decision-maker uses the preferences to match jobs to applicants to maximize applicant satisfaction while maintaining a stable match. Currently, the mechanism for deciding the matching is known in advance, and it is done with an applicant-proposing deferred acceptance algorithm. A *matching*, $\mu : O \to J$ maps applicants to jobs, and in this

setting no applicant remains unmatched. Satisfaction is higher when applicants receive more preferred jobs. We measure applicant $i's$ job satisfaction with the ordinal preference of his assigned job, $P_{o_i}(\mu(o_i))$. The satisfaction objective is then the average of this measure for all applicants, which we minimize. A *stable matching* is one where no applicant and job prefer to be matched to each other over their currently assigned matches in $\mu$ [29].

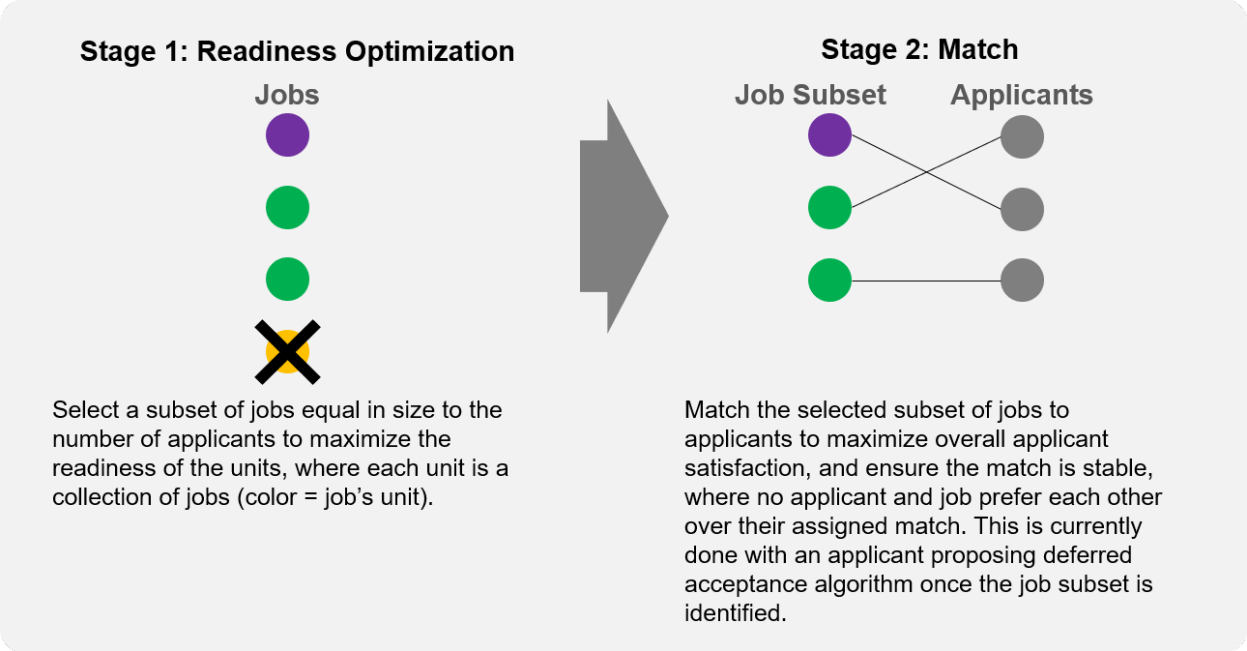


Figure 4-1: The current market of interest. In the current process, these two stages occur sequentially, where first a job subset equal in size to the number of applicants is selected for the market, driven by readiness, and then the stable matching is determined by an applicant-proposing deferred acceptance algorithm. In this section's optimization formulation, these two stages occur simultaneously. Jobs are grouped by unit, shown by color.

### Sets and Parameters

Sets and parameters used in the model and solution approaches are below. We use $a \succ_j b$ to denote that job $j$ prefers applicant $a$ to applicant $b$ and $c \succ_o d$ to denote that applicant

$o$ prefers job $c$ to job $d$.

$U = \{u_1, u_2, ..., u_q\}$, the set of units.

$J_u =$ The set of jobs for each unit, $u$.

$J = \cup_{u \in U} J_u = \{j_1, j_2, ..., j_n\}$, the set of jobs.

$n = |J|$, the number of jobs.

$\tilde{r}_u \in [0, 1]$, the projected readiness of unit $u$, if no jobs from $J$ enter the market.

$\underline{g}_u \in [0, 1]$, the lower readiness goal for unit $u$.

$\overline{g}_u \in [0, 1]$, the upper readiness goal for unit $u$.

$\gamma_u^+ =$ The overage penalty for unit $u$ if the number of jobs exceeds the target range.

$\gamma_u^- =$ The shortfall penalty for unit $u$ if the number of jobs falls short of the target range.

$N_u =$ The number of total jobs in unit $u$.

$O = \{o_1, o_2, ..., o_m\}$, the set of applicants.

$m = |O|$, the number of applicants. $m < n$.

$P_j =$ Strict preferences of job $j \in J$ over applicants, $o \in O$.

$P_o =$ Strict preferences of applicant $o \in O$ over jobs $j \in J$.

$P = \{P_{j_1}, ..., P_{j_n}, P_{o_1}, ..., P_{o_m}\}$, the list of preferences.

$p_{oj} =$ Preference ranking of job $j$ by applicant $o$.

### 4.3.2 Model Formulation, OAT

The bi-objective formulation that combines the readiness and matching stages depicted in Figure 4-1 is **OAT**. The key elements are connecting the job subset decision to the matching, and maintaining the stability of the job subset and applicant match without the preferences from the non-selected jobs impacting the feasible region.

**OAT** :

$$\min_{d,r,y,z} \quad \lambda \sum_{u \in U} d_u + (1-\lambda)\frac{1}{m} \sum_{o \in O} \sum_{j \in J} p_{jo} z_{oj}$$

$$s.t. \quad r_u = \tilde{r}_u + \frac{1}{N_u} \sum_{j \in J_u} y_j \qquad \forall u \in U \qquad (4.1a)$$

$$d_u \geq \gamma_u^-(\underline{g}_u - r_u) \qquad \forall u \in U \qquad (4.1b)$$

$$d_u \geq \gamma_u^+(-\overline{g}_u + r_u) \qquad \forall u \in U \qquad (4.1c)$$

$$\sum_{j \in J} y_j = m \qquad (4.1d)$$

$$\sum_{o \in O} z_{oj} = y_j \qquad \forall j \in J \qquad (4.1e)$$

$$\sum_{j \in J} z_{oj} = 1 \qquad \forall o \in O \qquad (4.1f)$$

$$z_{oj} + \sum_{k : o \succ_j k} z_{kj} + \sum_{k : j \succ_o k} z_{ok} \leq 1 \qquad \forall o \in O, j \in J \qquad (4.1g)$$

$$d_u \geq 0 \qquad \forall u \in U \qquad (4.1h)$$

$$y_j \in \{0,1\} \qquad \forall j \in J \qquad (4.1i)$$

$$z_{oj} \in \{0,1\} \qquad \forall o \in O, j \in J \qquad (4.1j)$$

In formulation 4.1, the primary decisions variables are $y_j$, a binary selector for including job $j$ in the selected subset, $J'$, and binary indicator $z_{oj}$, for when applicant $o$ matches to job $j$. Constraints 4.1a-4.1d account for readiness. Constraint 4.1a determines the projected readiness for each unit $u$ based on the current personnel status of the unit, $\tilde{r}$, and the increase provided by each job selected for the market. The readiness portion of the objective, the left-hand term, is determined by constraints 4.1b and 4.1c which find the deviation of the projected readiness from the unit's target readiness. When the number of jobs selected for the market results in a readiness, $r_u$, that falls short of the lower end of the target range, $\underline{g}_u$,

then the unit incurs a penalty equal to that deviation times a shortfall weight, $\gamma_u^-$. When the number of jobs selected for the market results in a readiness, $r_u$, that exceeds the upper end of the target range, $\overline{g}_u$, then the unit incurs a penalty equal to that deviation times a overage weight, $\gamma_u^+$. The readiness objective is the sum of the weighted readiness deviations and is weighted by $\lambda$. Constraint 4.1d ensures that enough jobs are selected to balance the market, where $m$ is the number of applicants.

Constraints 4.1e-4.1g account for the matching. Constraints 4.1e and 4.1f ensure a perfect matching between $O$ and the selected subset of jobs, $J'$. Constraint 4.1e couples the readiness and matching aspects together, where jobs can only match if they are selected for the market. Constraint 4.1g ensures the stability of the match between applicants and the job subset. When $z_{oj} = 1$, constraint 4.1g is not limiting as the perfect matching constraints ensure that the second and third terms in constraint 4.1g equal 0. When $z_{oj} = 0$ in constraint 4.1g, the second term's summation includes the matching indicators for all of the applicants that $j$ prefers less than $o$. The third term's summation includes the matching indicators for all of the jobs that $o$ prefers less than $j$. The constraint prevents both the second and third term equaling one at the same time, which ensures that the applicant and job don't both prefer each other over their assigned match. We generate the constraint for all combinations of applicants and jobs, even those jobs not selected for the market. This is possible without the preferences of $j \notin J'$ impacting the stability determination since in constraint 4.1g for $j \notin J'$, the first and second terms have to be 0. The satisfaction objective is the average job preference received by applicants. We find each applicant's preference $p_{jo}$ for his matched job. We then take the average and weight it by $(1-\lambda)$. A standard definition of an applicant-optimal stable solution is a stable matching that every applicant likes as well as any other stable matching (see, for example, Theorem 2 in the original stable matching paper from Gale and Shapley [29]). We focus here on the summarized measure of this with average preference received.

**Using OAT to Describe the Current Process**

**OAT** solves for the optimal job subset and matching simultaneously. The current process solves these sequentially, by first finding a readiness maximizing solution (left side of Figure 4-1) and then applying the applicant-proposing deferred acceptance algorithm (right side of Figure 4-1). If we set $\lambda = 1$ in **OAT** then the resulting job subset determined by $y_j$ finds a readiness maximizing (deviation minimizing) solution. We can discard the matching associated with $z$, which is stable but not necessarily applicant-optimal, and apply the applicant-proposing deferred acceptance algorithm. Thus, **OAT** with a known follow-on matching algorithm captures the current process. We highlight that solving for those two decisions sequentially means not accounting for the matching at all in the job subset determination. When there are multiple readiness-maximizing solutions, this sequential process is not guaranteed to find the readiness-maximizing solution with the best satisfaction (lowest average applicant job-preference received).

## 4.4 Solution Approaches

We present definitions and background on finding a Pareto frontier for the bi-objective problem. We then present a version of the weighted-sum method using **OAT** that solves for a set of solutions that are on the Pareto frontier. Then, we present a search algorithm that finds a set of solutions that approximate the Pareto frontier. The proofs for the lemma and propositions in this section are in Appendix C.2.

### 4.4.1 Non-dominated Solutions

Every solution to **OAT** maps to a point in the objective function value space specified by the readiness objective value and the satisfaction objective value. The solution using the current process, described in Section 4.3.2, finds a single one of these points that maximizes the readiness value, and then finds the matching which maximizes satisfaction given the already selected subset of jobs. If we normalize any other solution's objective values as the

change from the objective values found with the current solution, then we can depict the objective space with a horizontal axis as the percent change in the readiness objective, and the vertical axis as the percent change in the satisfaction objective. An illustrative example is provided in Figure 4-2, where each point represents the (readiness, satisfaction) values mapping of a different solution. As mentioned above, there could be multiple job subsets that map to the maximum readiness value. There is no guarantee that the current process finds the satisfaction-maximizing job subset among the readiness-maximizing subsets. When there is a different job subset with the same readiness and higher satisfaction, we would see a point on the Pareto frontier with a readiness change of 0, and a positive satisfaction change.

We define the following terms, adopting terminology from Przybylski et. al in [51]. For a good overview of multi-objective combinatorial optimization, see [25]. A feasible solution $x^*$ is *efficient* if no other solution $x \in X$ exists that improves on one objective without a decrease in the other objective. The image of an efficient solution in the objective space, $z(x^*)$, is a *non-dominated point* (non-gray points in Figure 4-2). We define the set of efficient solutions as $X_E$ and the set of non-dominated points as $Z_N$ which form the *Pareto frontier*.

We can partition $Z_N$ into two subsets, $Z_N = Z_{NS} \cup Z_{NN}$. The set of points that define the convex hull of $Z_N$ are *supported non-dominated* (green points in Figure 4-2), and part of $Z_{NS}$. The set of points on the Pareto frontier but in the interior of the convex hull are *non-supported non-dominated* (tan points), and part of $Z_{NN}$. We can further partition $Z_{NS}$ into points that are on the vertices of the convex hull, the extreme supported non-dominated points, $Z_{NS1}$, and those that are not on the vertices of the convex hull, $Z_{NS2}$.

### 4.4.2 Exact Solution Approach for Finding the Pareto Frontier

**OAT** has an objective function that is a convex combination of the readiness and satisfaction objectives. The weighted sum method is an algorithm for iterating through different objective function weights to find a Pareto frontier, [25]. With a mixed integer formulation, the weighted sum method finds the convex hull of the Pareto Frontier, $Z_{NS1}$ (green points in

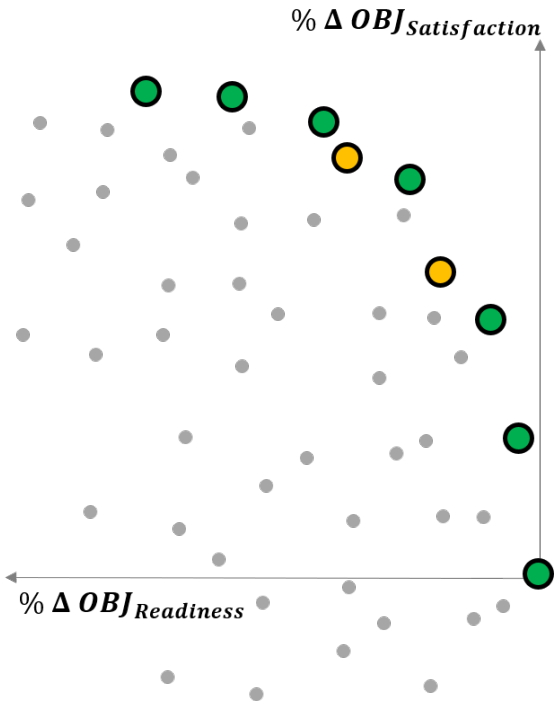


Figure 4-2: This is an example graph of feasible solutions plotted in the objective space (readiness and satisfaction, scaled in terms of percent change from the objective space mapping of the current process solution). The set of points on the Pareto frontier, $Z_N$ have a black outline. $Z_N = Z_{NS1} \cup Z_{NS2} \cup Z_{NN}$. Extreme supported non-dominated points, $Z_{NS1}$ are green and on the vertices of the convex hull of $Z_N$. Non-extreme supported non-dominated points, $Z_{NS2}$ could exist on the line segments connecting the vertices of the convex hull of $Z_N$ (not depicted). Non-supported non-dominated points, $Z_{NN}$ are orange and on the Pareto frontier, but the interior of the convex hull of $Z_N$.

Figure 4-2).

### Properties of OAT for Algorithm Implementation

To find the exact Pareto frontier we need to solve **OAT** many times, and we describe a computationally faster version of the formulation before detailing the algorithm.

**Proposition 4.4.1. OAT** *always has a feasible solution that is stable between the set of applicants, $O$, and the selected subset of jobs, $J'$.*

This follows from the original constructive proof of Gale and Shapley in [29] that established

the existence of a stable matching. In this case, the readiness portion of **OAT** does not impact the construction. We can re-cast the mixed integer linear formulation, **OAT**, using a min-cost network flow with side constraints, $\mathbf{OAT}_F$, described in detail in Appendix C.1.2.

**Lemma 4.4.2.** $\mathbf{OAT} = \mathbf{OAT}_F$.

Then, we can relax the integrality constraint on many of the variables. Consider a version of $\mathbf{OAT_F}$ from Formulation C.2 in Appendix C.1.2 where in C.2d we relax the integrality of $x_{ij}\ \forall (i,j) \in A : i \notin J$, denoted $\mathbf{OAT_{F-PR}}$.

**Proposition 4.4.3.** $\mathbf{OAT}_F = \mathbf{OAT}_{F-PR}$ *with integral data inputs.*

This allows us to solve **OAT** with $|J|$ binary variables and $|O||J|+4|U|$ continuous variables instead of $|O||J| + |J|$ binary variables and $4|U|$ integer variables.

### Weighted Sum Algorithm

Algorithm 6 takes $\mathbf{OAT}_{F-PR}$, and iteratively finds points along the convex hull of the Pareto frontier. The algorithm is depicted in Figure 4-3.

The algorithm initializes by finding the left and right end points of the Pareto frontier (points 1 and 2 in Figure 4-3). Of the possible readiness maximizing solutions, it finds the one that has the best satisfaction. Then, of the possible satisfaction maximizing solutions, it finds the one with the best readiness. Of note this is not guaranteed with $\lambda \in \{0, 1\}$, and we instead use a value close to 0 and a value close to 1. We can see this in Figure 4-3 where the point to the left of point 2 has the same satisfaction, but worse readiness.

Then, the algorithm uses those two objective space points to determine a value of $\lambda$ such that the objective function is parallel to the line connecting those two points, and solves **OAT**. This is depicted with the red minimization function shown in Figure 4-3. If there is another point on the convex hull of the Pareto frontier between points 1 and 2, this minimization will find it, which in the Figure would be point 3. The algorithm then selects the two adjacent

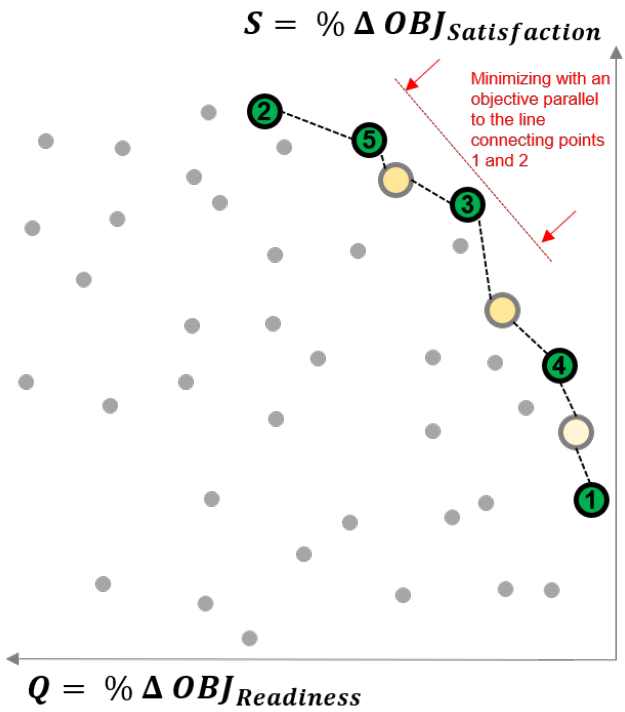


Figure 4-3: This depicts the weighted sum method which finds the convex hull of the Pareto frontier. It sets the objective scaling parameter so that the objective is parallel to the line connecting two known points in the objective space. If there is a solution with an objective above this line, then the optimization will find it as it minimizes.

points on the frontier that are furthest apart, and uses them to find a new value of $\lambda$ such that the new objective function is parallel to the line connecting those points. In Figure 4-3, the algorithm would select points 1 and 3 (from the available 1,2, and 3), and the next solution to $\mathbf{OAT}_\lambda$ would find point 4. The algorithm continues in this manner until there are no additional points to be found, or the maximum distance between adjacent Pareto frontier points is less than the distance tolerance in the algorithm, $\epsilon$.

**Proposition 4.4.4.** *Algorithm 6 terminates in finite steps, and when $\epsilon = 0$, Algorithm 6 finds all of the extreme supported non-dominated points, $Z_{NS1} \subset \tilde{Z}_{NS}$.*

Algorithm 6 induces no cycles as once the maximum distance between adjacent objective-space points is less than the threshold $\epsilon$, the algorithm terminates. When the threshold is 0, the algorithm continues to find solutions for different values of $\lambda$ until there are no

more solutions to be found. This means it finds the complete set of extreme supported non-dominated points, and possibly some non-extreme supported non-dominated points on the line segments connecting the vertices of the convex hull of the Pareto frontier. If we wanted additional points on the Pareto frontier, those from $Z_{NN}$, we could then apply approaches such as the two-phase method described in [51], which are not addressed here.

### 4.4.3 One-Swap Chain Algorithm

The weighted sum method requires solving a large number of optimization problems, and when each problem takes a long time to solve or is intractable, we need a different approach. A typical market in our use-case setting has 300 jobs and 250 applicants, and the weighted sum algorithm takes more than two hours. For a market with 500 jobs, the weighted sum algorithm with $\epsilon = 0$ does not terminate after 18 hours. Detailed computational timing results are in Section 4.5.

The one-swap chain algorithm uses a local search method that starts with a stable solution and generates additional solutions by iteratively exchanging one job not in the selected subset for one that is in the selected subset, in a particular way. This approach allows us to quickly generate a large number of additional solutions, with certain properties. Because we want a set of trade-offs for decision makers, we explicitly want to generate many different solutions and we store all of the solutions we iterate through during the algorithm. We initiate the algorithm on a readiness maximizing solution, and generate additional solutions generally from right to left in the objective space described above. We can then find the non-dominated points in this generated set, which while not the Pareto frontier of the **OAT**, do provide a useful set of trade-offs as an approximation of the Pareto frontier.

#### One-Swap Chains (Local Search Step)

Consider a selected subset of jobs, $J'$, and an associated matching, $\mu$. We can update the solution by bringing in one of the jobs that is not currently in the selected subset and replacing one of the selected jobs. We update the matching by taking the applicant matched

to the exiting job and matching him to the entering job. Sometimes, it is possible to swap a job for a new one in a way that improves satisfaction and maintains stability which we call a *one-swap*. In matching $\mu$, $\mu(o)$ is applicant $o$'s match, and job $j$'s match is denoted $o : \mu(o) = j$.

**Definition 4.4.1** (One-Swap)**.** Given a subset of jobs, $J'$, and an associated matching, $\mu$, one job not in the subset, $j_{in} \in J \setminus J'$, replaces one job in the subset, $j_{out} \in J'$, and the matching $\mu$ updates for the single applicant, $\tilde{o}$, previously matched to $j_{out}$ who is now matched to $j_{in}$. Jobs $j_{in}$ and $j_{out}$ must meet two criteria: (i) Applicant $\tilde{o}$ benefits from the swap: $j_{in} \succ_{\tilde{o}} j_{out}$ and (ii) All applicants that prefer $j_{in}$ to their current match are less preferred by $j_{in}$ than $\tilde{o}$: $\forall o_k \in O : j_{in} \succ_{o_k} \mu(o_k), \quad \tilde{o} \succ_{j_{in}} o_k$.

Figure 4-4 depicts an example one-swap with preferences, $P$, in the caption. We consider a given initial job subset, $J'_0 = \{j_1, j_3, j_5\}$, which results in the matching shown with red lines. We consider one job not in the selected subset, $j_2$. Every applicant could benefit from $j_2$ (criteria (i)), and of those, $j_2$ prefers $o_2$ (criteria (ii) means $\tilde{o}$ must be $o_2$). So, $j_2$ replaces $j_3$ in the selected subset and in the match for $o_2$, who was previously matched to $j_3$. This improves the resulting satisfaction objective, and the match is still stable.

**Proposition 4.4.5.** *Given a stable matching, a one-swap always results in another stable matching with improved satisfaction.*

By construction, the one-swap maintains stability and only exists when satisfaction improves. After a one-swap, we can take the exiting job, $j_{out}$, and see if a one-swap exists for it as the entering job, and iteratively continue this process with the exiting job from each one-swap until no more one-swaps are possible. This iterative process is a *one-swap chain*, and is depicted in Figure 4-4. In this example, $j_3$ exits the first one-swap, and we then consider it as $j_{in}$ and evaluate the two criteria for another a one-swap. As $o_3$ benefits from a swap from his current match but no other applicants do, $j_3$ replaces $j_5$ in the selected subset and as the match for applicant $o_3$. No applicant benefits from a swap to $j_5$, and so the one-swap chain

ends.

**Definition 4.4.2** (One-Swap Chain)**.** A sequence of one-swaps where initially a job $j_{in} \in J \backslash J'$ replaces $j_{out} \in J'$, and then $j_{out}$ can re-enter and replace a job in another one-swap. The process continues as long as a one-swap exists, which is as long as there is strict improvement in the preference-based objective while maintaining stability.

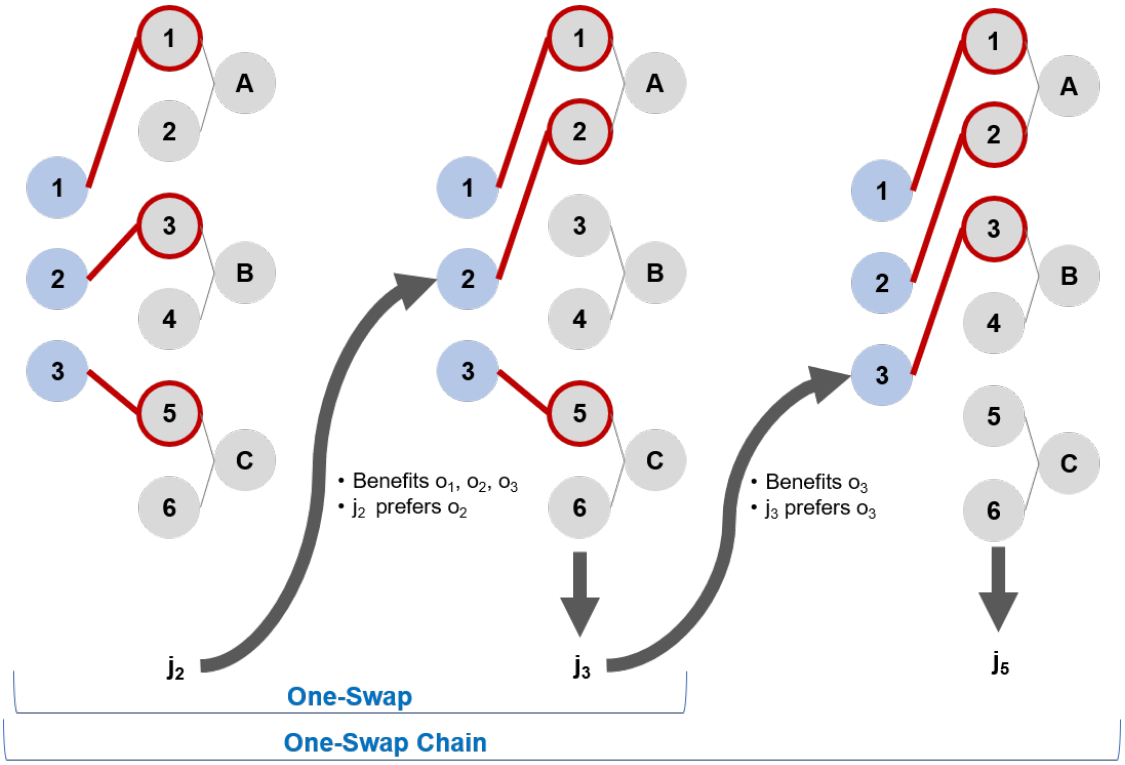


Figure 4-4: A one-swap and one-swap chain with 3 applicants and 6 jobs grouped into 3 units with the following preferences: $P_O = \{\{4,2,1,3,5,6\},\{6,2,4,3,5,1\}, \{2,6,3,5,4,1\}\}$; $P_J = \{\{3,2,1\},\{2,1,3\},\{1,2,3\},\{1,2,3\},\{1,2,3\},\{3,2,1\}\}$. The initially selected job subset is $\{1,3,5\}$, depicted with red circles, and matching are shown with red lines. Job $j_2$ is allowed to enter. It initially replaces $j_3$ as the match for $o_2$ as all applicants prefer $j_2$ to their current match, and $j_2$ prefers $o_2$. Job $j_3$ exits. Then $j_3$ re-enters and replaces $j_5$ as the match for $o_3$ since applicant $o_3$ prefers $j_3$ to his current match, $j_5$. No applicant benefits from swapping to $j_5$ so the one-swap chain is complete. There were two stable matchings found during the one-swap-chain, and we store both along with the initial matching, not just the final version, as they might have different readiness objectives.

During a one-swap chain, each one-swap results in a satisfaction-improving stable matching. The Jaccard index, $\mathcal{J}$, measures the similarity of sets, $\mathcal{J}(A,B) = \frac{A \cap B}{A \cup B}$. We denote the

initial job subset for the chain as $J'_0$ and index each subsequent subset with $i$. Once a one-swap chain initiates, the new job, $j_b \in J \setminus J'_0$ always remains in the subset, and so every subsequent subset in the chain, $J'_i$, shares all but one job with the starting subset, $J'_0$: $\mathcal{J}(J'_i, J'_0) = \frac{m-1}{m} \quad \forall i$, where $m$ is the number of applicants. Because different jobs map to different units, which jobs are in or out of the selected subset change the possible resulting readiness value, but the majority of the subset remains the same which keeps the change in the readiness value small. So while satisfaction is monotonically improving during each swap in a chain, the readiness does not change in such a manner. The solution from each one-swap in the chain could provide a useful trade-off between readiness and satisfaction, so we track all of the solutions during the chain.

**Proposition 4.4.6.** *The one-swap chain, when starting with an applicant-optimal matching, includes at most $m$ one-swaps.*

After each applicant changes a match once, any subsequent change results in a matching that has a subset of jobs already considered at some point in the chain. Since the one-swaps are always satisfaction improving, another swap cannot improve the matching over one already considered if we start with an applicant-optimal matching.

### The One-Swap Chain Algorithm to Approximate the Pareto Frontier

Our goal is generate a set of solutions that provide good trade-offs for decision-makers. Ideally, this set of solutions is on the Pareto frontier of **OAT**, and if not, it well approximates the Pareto frontier. Three ingredients provide the basis for the algorithm. First, we know that the current process finds a solution that maximizes readiness, and this provides a starting point for a local search method that improves in satisfaction at some expense in readiness. Second, a one-swap chain, when possible, generates a set of satisfaction-improving stable solutions. Third, from criteria (ii) in Definition 4.4.1, there can be at most a single one-swap for each job not currently in the selected subset, and therefore a single one-swap chain for each job not currently in the selected subset.

Figure 4-5 depicts the algorithm. The green points form the true Pareto frontier, which is unknown to us. We fix a depth parameter, $\tau$, which is the number of jobs to consider replacing in $J$, $\tau \in \{1, ..., n-m\}$. In this example, $\tau = 3$. First, we initialize a solution using the current process, depicted in Figure 4-5 with the black point numbered 0 at $(0, 0)$. Second, we take every job not in the selected subset, $j \notin J'$, and compute the one-swap chain for each, if it exists, storing all of the interim stable matches found during each one-swap chain. This results in a set of solutions with objective space points depicted in light blue, each with only one job different from the solution to the current process, but with an improved satisfaction objective. Then, for $\tau > 1$, we take additional steps, and we choose one of these new job subsets and matchings as the new starting point. In Figure 4-5 for the next depth step, this is the gray point numbered 1, selected because it was the satisfaction maximizing solution out of all one-swap chain generated solutions. With point number 1 as the new starting point, we again take every job not in the selected subset, compute the one-swap chain for each if it exists, and store the resulting solutions with objective space points depicted here in orange. For the third depth step, we choose one subset, here point 2, as the new starting point and repeat the process generating the dark blue points. From all of these generated points we find the non-dominating points that form the approximation to the Pareto frontier, depicted with red outlines in Figure 4-5. The details are specified in Algorithm 2.

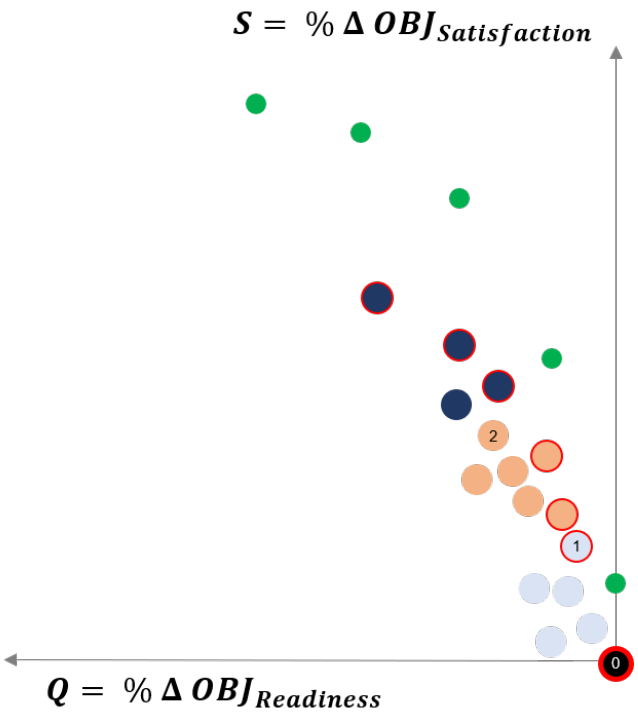


Figure 4-5: A depiction of the one-swap chain algorithm for $\tau = 3$.

---

**Algorithm 2:** Generate solutions that provide an approximation, $\hat{Z}_N$, of the Pareto frontier, $Z_N$. Each point on the frontier has two components, $(Q, S)$.

---

**Result:** $\hat{Z}_N$

**Input :** $\mathbf{OAT}_{\lambda=1}, \tau$

Set selected job subset, $J'_C$, from solution to $\mathbf{OAT}_{\lambda=1}$

Set matching, $\mu(J'_C)$, from applicant proposing deferred acceptance algorithm

$Z_{out} = \emptyset$

**for** $t \in 1...\tau$ **do**

  **for** $j \notin J'_C$ **do**

    Execute one-swap chain for $j$, if possible

    Store matching and objective values from each one-swap in $Z_{out}$

  **end**

  Set $x_{new}$ as satisfaction-maximizing solution in $Z_{out}$

  Set $J'_C$ and $\mu(J'_C)$ from $x_{new}$

**end**

Set $\hat{Z}_N$ as the non-dominated points in $Z_{out}$

---

Computationally, the one-swap chain algorithm provides a fast method for generating an approximate Pareto frontier.

**Proposition 4.4.7.** *The one-swap chain algorithm runs in polynomial time.*

The one-swap chain algorithm initiates with a solution using the current process which requires solving a linear programming problem and then using the deferred acceptance algorithm. The one-swap chain algorithm then computes $\sum_{i=1}^{\tau} n - m - i + 1$ one-swap chains ($< \tau(n-m)$). Each one-swap chain consists of at most $m$ one-swaps, and each one-swap requires one pass through the preferences of each applicant. After solving the min-cost flow problem that determines the readiness-maximizing job subset, this is $\approx O(m^3)$.

We note that the nature of the stability that is ensured in the algorithm means that entering jobs will always match and remain with their highest preferred applicant during a one-swap, when a one-swap is possible. Thus, given a subset $J'$ at some point during the one-swap chain, there is no guarantee that it is the applicant optimal matching for that subset, just that it is a stable matching for that subset. Additionally, we note that given the depiction of the frontier in Figure 4-5, this algorithm finds the bottom right point and then searches up the frontier for additional points, generally moving from right to left. We can find a satisfaction-maximizing solution by solving **OAT** with $\lambda = 0$. This would give us a starting point in the upper left of the objective-space we consider (but not necessarily on the Pareto frontier). However, we do not have an efficient procedure for trading a decrease in satisfaction for an increase in readiness in a manner that maintains stability like we do with the one-swap. This prevents a similar fast operation that moves down the frontier, generally from left to right.

A natural extension to this is to consider an *n-swap* where we take a number of jobs not in the selected subset $J'$, and bring them all into the selected subset at the same time, in a manner that improves satisfaction and maintains stability. To move to an *n-swap* from a *one-swap* we update the definition to include that if multiple jobs benefit the same applicant

and that applicant is preferred by multiple jobs, then the entering job that replaces that applicant's current match is his most preferred. In this bi-objective case, we want a set of solutions, and specifically do not want to move, without interim steps, directly to a single satisfaction-improved solution. We therefore maintain one-swaps, applied in sequence as desired with $\tau$, instead of *n-swaps*.

## 4.5 Illustrative Results for a U.S. Army Officer Assignment Market

### 4.5.1 Use-case Overview

Our motivation for considering the combination of organizational readiness and applicant satisfaction is the marketplace used to match U.S. Army officers to their next jobs. In a typical year, thousands of officers from dozens of different specialties move as part of their professional development. Typically, each specialty has its own market of assignments twice a year, and these vary in size based on the number of officers in that specialty. When there are more job vacancies than the expected number of moving officers, the Army's central personnel managers select a subset of the available jobs to enter the market. In the summer of 2020, more than 30 thousand officers participated in the first full iteration of the talent marketplace in many separate markets. For example, around 250 majors with an intelligence specialty participated in a market that had more than 300 jobs pre-market, but a readiness-driven 250 selected to be in the market. Officers and units submit ordinal preferences, and any missing preferences are randomly imputed by the personnel managers.

We have access to assignment preference data for the thousands of officers and units that participated in this market. For each officer, there are ordinal preference values for some or all of the jobs that were in the market. For each job, there are ordinal preference values for some or all of the officers in the market. We compile and clean the associated preference data in the same manner as the Army personnel managers, where any missing preferences

are randomly imputed. For readiness data, we leverage snapshots of Army organizational strength, and use randomly generated weights to test a variety of parameter settings.

The current assignment market only collects preference data on the jobs that are selected for the market. This leaves a gap, as we want to consider preferences from officers for all of the possible jobs, not just the selected subset. We want to avoid imputing preferences for jobs that officers never saw in the market, and for jobs that did not participate in the market at all. To account for this in a manner that leverages the actual data as much as possible, we use only the jobs that were selected, but we sample a set of officers such that the proportion of jobs that can be filled is similar to what would have happened if every job had entered the market for all of the officers. For example, if we have a specialty with 250 jobs and 200 officers, where only 200 jobs entered the market and we have no preference information on the other 50 jobs, then we use the preferences from the 200 jobs where we have data and sample 160 officers' worth of preference data. This allows us to leverage actual preference data for the use-case as much as possible, at the expense of artificially shrinking the market.

### 4.5.2 Computational Timing

This use-case motivates our computational considerations, with the majority of the specialty markets for the officers moving in 2020 having fewer than 300 jobs. For the exact solution method, the stability constraints in **OAT** make an extremely dense constraint matrix, and the memory requirements grow quickly as the number of jobs and officers increase. Table 4.1 shows the solution times for solving a single instance of $\mathbf{OAT}_{F-PR}$ at varying sizes. Computational evaluation was done on a cluster using 4 CPUs and up to 128 GB of memory with Julia 1.2.0 [12], JuMP 0.21.0 [22], and Gurobi 0.9.11.

#### Computational Evaluation of the the Local Search Approach

To find a set of solutions, we need to solve many optimization problems or use the one-swap chain algorithm. Table 4.2 shows the solution times for various problem sizes using the weighted-sum method, Algorithm 6 and the one-swap chain algorithm, Algorithm 2. As the

| Jobs | Binary Variables | Cont. Variables | Constraints | Time [min] |
|---|---|---|---|---|
| 100 | 100 | 8e3 | 1.6e4 | 0.03 |
| 200 | 200 | 3.2e4 | 6.4e4 | 0.47 |
| 300 | 300 | 7.2e4 | 1.4e5 | 1.62 |
| 400 | 400 | 1.3e5 | 2.6e5 | 3.49 |
| 500 | 500 | 2.0e5 | 4.0e5 | 11.7 |
| 1000 | 1000 | 8.0e5 | 1.6e6 | 116.0 |

Table 4.1: Formulation size of $\mathbf{OAT}_{F-PR}$ and average computational times, in minutes, for 100 use-case instances with 10 units, a number of officers equal to 80% of the jobs, and randomly generated parameters.

weighted-sum method time increases dramatically, to over 3 hours for 400 jobs, the one-swap chain algorithm continues to solve for an approximate frontier in less than one minute.

| Jobs | Weighted-sum Method Time [min] | One-Swap Chain Algorithm Time [min] |
|---|---|---|
| 100 | 0.63 | 0.01 |
| 200 | 16 | 0.04 |
| 300 | 122 | 0.24 |
| 400 | 185 | 0.32 |
| 500 | - | 5.7 |
| 600 | - | 50.7 |
| 700 | - | 17.8 |
| 800 | - | 119 |
| 900 | - | 244 |
| 1000 | - | 355 |

Table 4.2: Computational time, in minutes, for the two different frontier-generating methods. The number of officers equal to 80% of the jobs, with a depth $\tau = \frac{1}{3}(n-m)$, and randomly generated parameters. In the weighted-sum method, $\epsilon = 0$.

### 4.5.3 Use-case Evaluation

We validate our sampling method in the preference data by taking a single officer specialty's market and computing the exact Pareto frontier with many different samples. Figure 4-6 shows the results. There is variability from the differences in individual preferences, but the density plots show that the variability is not large and a single sample is typically representative.

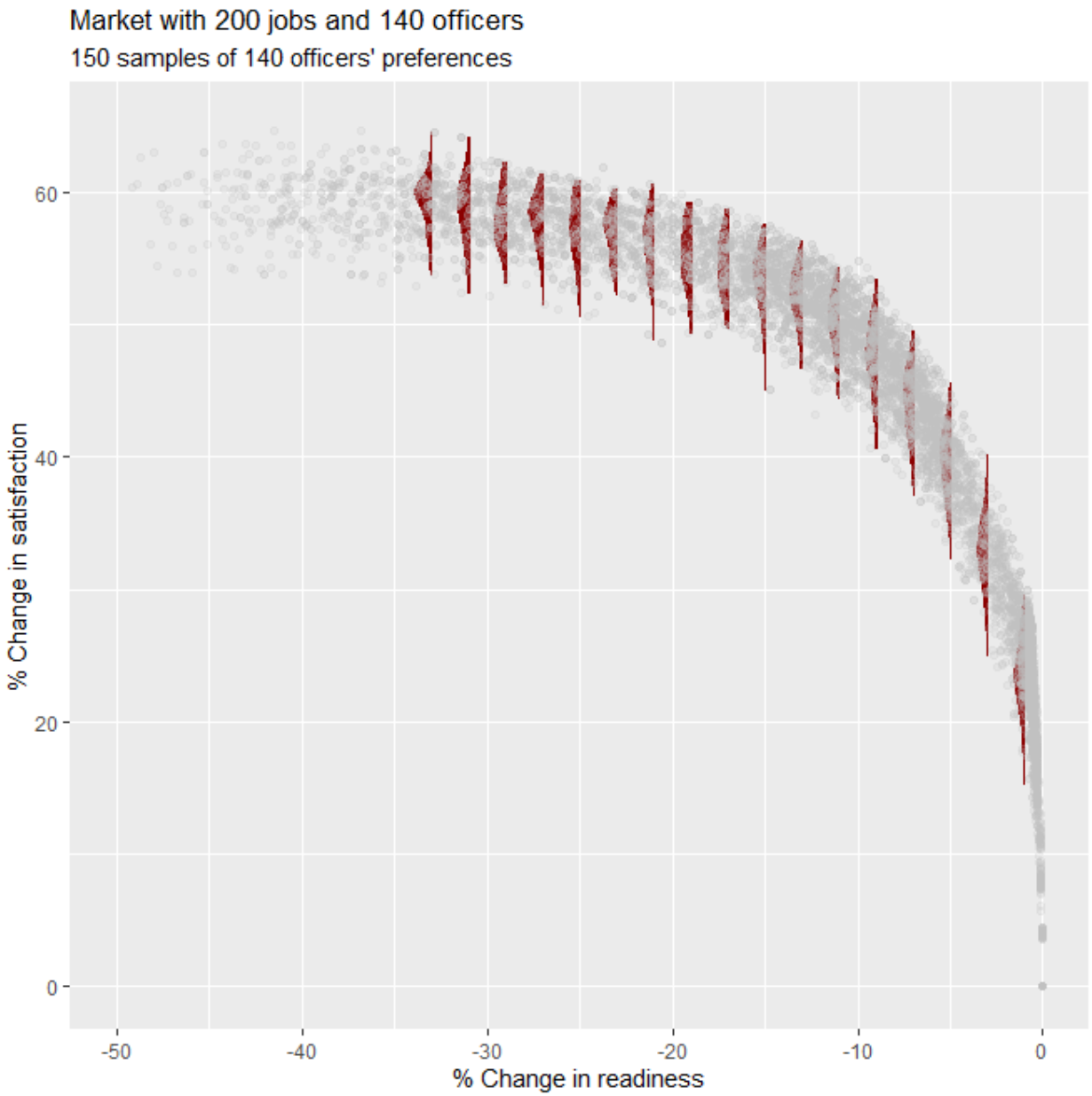


Figure 4-6: Many Pareto frontiers for a single specialty where each is computed for the preferences associated with all of the jobs and a different, sampled, group of officers that allows us to leverage the preference data in an $m < n$ setting.

To evaluate our methods in the use-case, we consider a single specialty market and generate the associated frontiers with Algorithm 2 and the weighted-sum method with Algorithm 6, for a representative sampling of officer preferences.

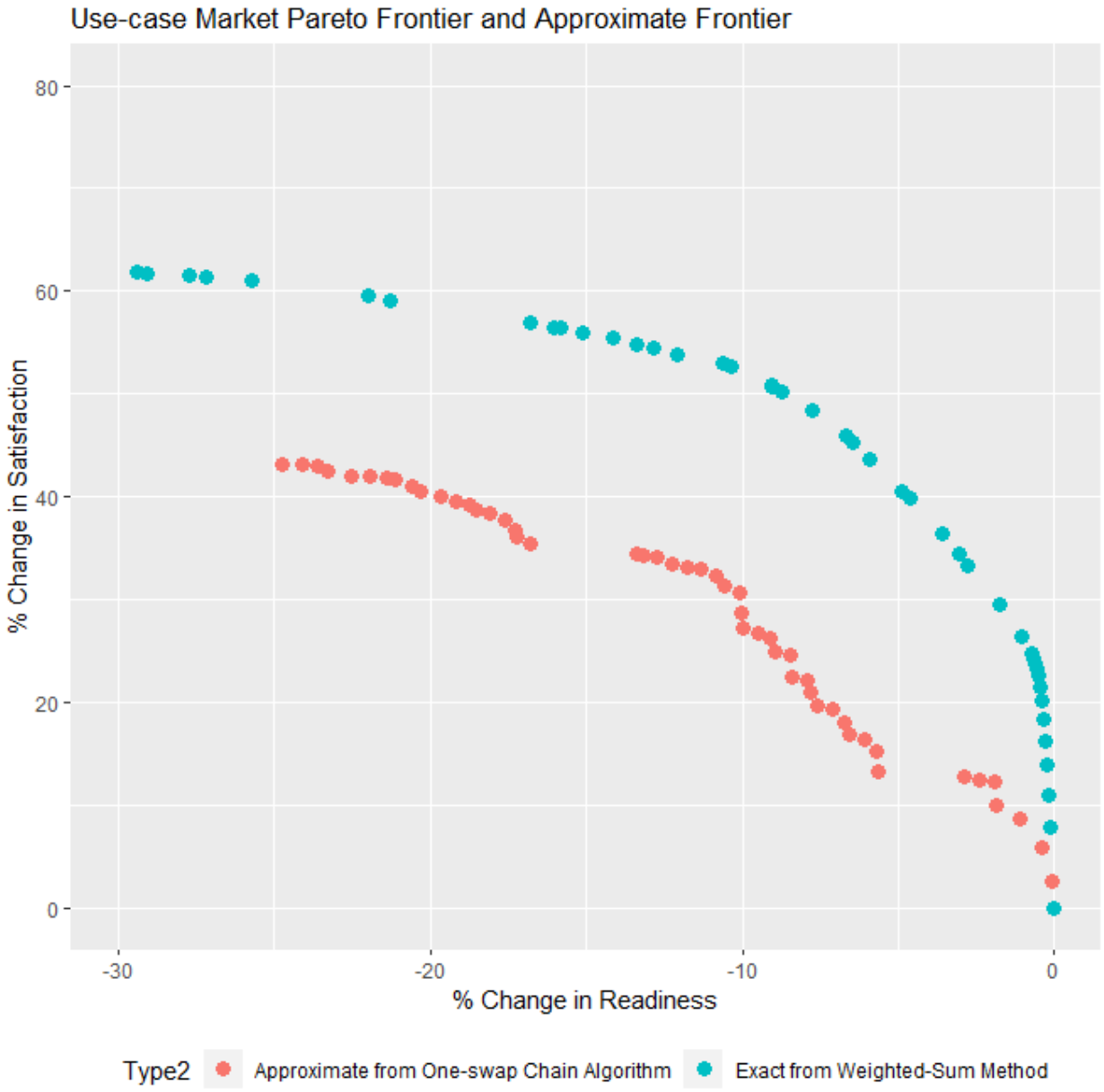


Figure 4-7: The exact Pareto frontier (blue) and approximate frontier (orange) for a use-case market using U.S. Army assignment marketplace data for preferences, and randomly drawn readiness weights.

These results demonstrate a number of key points. First, the current method used by the Army does not necessarily find the subset of jobs that maximizes readiness and dominates other solutions' satisfaction. We can see this when points with a positive change in satis-

faction dominate the point $(0, 0)$. Second, the initial steepness of the blue curve in Figure 4-7 shows that there is a large amount of possible satisfaction improvement for very little decrease in readiness. In this case, that includes a 20% improvement in satisfaction for less than a 2 % drop in readiness. Third, the one-swap chain algorithm, while sub-optimal, provides a fast solution method for finding a set of trade-offs. Additionally, its trade-off curve has an intuitive explanation for decision-makers, as the difference between the selected sets is small, with a Jaccard index of at most $\mathcal{J}(J'_i, J'_j) \leq \frac{m-\tau}{m} \quad \forall i, j$. This is not true of the exact solution, where adjacent points on the frontier might have significantly varying sets. in application, this allows decision makers to familiarize themselves with the details of the initial solution. Then for any other solution on the approximate frontier, they can consider the jobs that are different, which will be small in number and similar for adjacent points.

## 4.6 Conclusions

In this chapter we propose methods for finding a set of solutions to a bi-objective matching problem with more jobs than applicants, where a decision-authority has control over the subset of jobs that enter the market, and a stable solution is required. These methods provide an opportunity for the decision maker to consider trade-offs between the two objectives with a variety of solutions. We present an exact approach that finds the convex hull of the Pareto frontier using the weighted-sum method implemented on a bi-objective mixed integer linear program. We present an algorithm that finds a set of solutions as an approximation to the Pareto frontier that runs quickly even for large instances. This algorithm finds an initial solution using the method for the current process which maximizes readiness and satisfaction sequentially, and then conducts a series of job exchanges we call a *one-swap chain*. These chains, when possible, strictly improve the satisfaction outcome while maintaining stability.

For the motivating use-case, this approach provides an opportunity for military personnel leaders to consider trade-offs between organizational readiness and service member satisfaction. We validate the approach with data from U.S. Army officer assignment marketplace,

but other services are also implementing market-based assignment processes where the approach is relevant. The key insight is that a much better solution in terms of satisfaction could be possible with only a slight decrease in readiness. One extension of this work is a school district expanding its capacity, and considering the decision of which schools to expand. When the school district uses a preference-based market for matching students to schools, then the model and approaches here could provide the district the ability to consider which schools should have additional capacity while considering preference information from families and a separate district staffing goal.

We include an appendix with the details of formulation variants and the weighted-sum algorithm, and one with the proofs for the propositions in Section 4.4.

# Chapter 5

# Conclusion

This thesis develops methods for improving military equipment and personnel readiness using tools from operations research. We consider novel modeling approaches for operational maintenance decisions and for personnel policy at both the strategic design level and operational level. Methodologically, this thesis applies techniques from linear programming, integer programming, and Markov Decision Processes to formulate models, and looks at both exact and heuristic solutions. Each portion includes a numerical use-case that leverages data from a U.S. Army organization.

The decision policy developed in chapter 2 for integrating a component health predictive signal into a larger maintenance framework can help maintainers with day-to-day operational decision. The insights on the necessary quality of the underlying predictive model can inform predictive analytic development efforts at the Department of Defense on when component health predictive models are ready for operational use. The insights in chapter 3 about career path design policy can help military personnel analysts understand factors that drive personnel assignment bottlenecks. The work on increasing the flexibility of the career path design policy can help personnel leaders make strategic decisions about how individuals progress through assignments. The work on the assignment marketplace in chapter 4 provides

methods for personnel leaders to consider matching officers to jobs is a preference-based, stable manner where a slightly lower readiness result provides room for large gains in officer assignment satisfaction.

# Appendix A

# Supplement for Chapter 2

We include three appendices: one that details model variations, one for the proofs associated with the two analytic results, and one that has additional use-case information including background, a sensitivity analysis, and a simulation.

## A.1 Additional Model Notes

### Varying Component Lifetime

The model extends to a component of interest that has a lifetime replacement. We add an additional element, $Lifetime$, to each system's state, $s_{it}$. This tracks the total number of operating hours for the component. To refine the transition function (and later the observation function), we segment $Lifetime$ with segments $l \in \{1, ..., V\}$, $V \leq L$. We then specify parameters based on different segments, $l$. If the part has a mandated age-based replacement policy, that specifies $L$. If the part has no mandated age-based replacement policy, then we only need to track $L$ such that it allows us to identify the segment, $l$. So, the maximum lifetime we need to track in the state space is the value of $Lifetime$ that specifies the beginning of $l = V$, the final segment. This is because we gain no additional information tracking $Lifetime$ once we transition into the final segment, $l = V$.

Two specific examples illustrate the utility of the lifetime varying approach, when there is sufficient data to model the differences. In the first, consider a scenario where we have enough data to evaluate the performance of the health signal's source prediction model for $V = 2$, with an early part life and a later part life. If we are more confident in the health signal when the part is older because that signal comes from a source with better performance metrics, then the parameters from the predictive model that are inputs to the POMDP for $l = V$ will be higher than for $l = 1$.

In the second scenario, consider a part that is approaching its projected part lifetime, $L$, and the health signal indicates the component is healthy. If the health signal quality later in a part's life is not as good, then we could maintain the existing age-based replacement policy for the component at $Lifetime = L$. If our confidence in the health signal is high, then we have the opportunity to explore extending the life of the part. In our use-case, the latter is unlikely without much more testing because of the fear of crashes and the strict air-worthiness standards for parts.

## A.2 Proofs

**Proof of Proposition 2.4.1**

*Proof.* In $S_1$, we have a system $j$ with $Hours = h^*$; preventive maintenance is possible, but not yet due (time until maintenance is $T_H - Hours$); system $j$'s component is not broken and has a certain probability of being in a failed state, $b(F)$; system $j$ is in neither type of maintenance, and the optimal decision is to combine preventive maintenance and component repair, $u_{jt} = M_{PC}$.

We know the basic structure of the optimization problem from Section 2.3. Since combining maintenance is optimal for system $j$ in $S_1$, we know that $V(S_1; u_{jt} = M_{PC}) > V(S_1; u_{jt} \in \{M_P, M_C, Operate, Rest\})$, where $M_{PC}$ is combined maintenance, $M_P$ is preventive maintenance only, and $M_C$ is component repair only.

In state $S_2$, we hold the other systems constant, and only adjust the $Hours$ for system $j$. This means the operating constraint is still met, and that maintenance capacity exists for $S_2$, as it did for $S_1$. We wish to show that $V(S_2; u_{jt} = M_{PC}) > V(S_2; u_{jt} \in \{M_P, M_C, Operate, Rest\})$, and we consider each decision in turn.

Combined maintenance resets both the component and the hours, which resets $S_1$ and $S_2$ into identical states, and since the two states have identical current rewards, $V(S_1; u_{jt} = M_{PC}) = V(S_2; u_{jt} = M_{PC})$.

Preventive maintenance resets the hours, and since that is the only difference between the two states, $V(S_1; u_{jt} = M_P) = V(S_2; u_{jt} = M_P)$.

Consider, for $S_2$, executing a component repair and then following the optimal policy, as in equation A.1, where $S_2' = S'|S = S_2, u_{jt} = M_C$.

$$V(S_2, u_{jt} = M_C) = G(S_2) + \gamma E[G(S_2')] + \gamma^2 E[V(S_2'')] \tag{A.1}$$

If $S_1$ follows an identical policy to the one in equation A.1, then it will have an identical value. This is because the only difference in the two states is the hours, and the reward function depends only on $b(F)$. So for any action in equation A.1's policy, the expected result for the reward is identical between the two states. Because the hours are higher for $S_2$, the constraint that forces a system into preventive maintenance occurs sooner, in expectation, than the same constraint for $S_1$, and so this constraint does not limit the ability of $S_1$ to follow the policy in equation A.1. This means that for state $S_1$, executing a component repair and then following the optimal policy has a value at least as large as equation A.1, $V(S_1, u_{jt} = M_C) \geq V(S_2, u_{jt} = M_C)$.

This same argument holds for operating and for resting. If system $j$ in $S_2$ operates or rests, and then continues with the optimal policy, then $S_1$ following the same policy will produce at least the same value.

Therefore, we know that if the decision is anything but combining maintenance for system $j$, and then following the optimal policy, the value in $S_1$ is at least as large as the value in $S_2$. Since the two states have the same value for combined maintenance, and combining maintenance is optimal for system $j$ in state $S_1$, combined maintenance remains optimal for system $j$ in $S_2$. □

**Proof of Proposition 2.4.2**

*Proof.* In $S_1$, we consider system $j$ with hours within the tolerance range for conducting preventive maintenance (tiers 4 and 5 from Section 2.6); system $j$ has a belief of component failure, $b(F) = b^*$; system $j$ is in neither type of maintenance, and the optimal decision is to combine preventive maintenance and component repair, $u_{jt} = M_{PC}$.

We know the basic structure of the optimization problem from Section 2.3. Since combining maintenance is optimal for system $j$ in $S_1$, we know that $V(S_1; u_{jt} = M_{PC}) > V(S_1; u_{jt} \in \{M_P, M_C, Operate, Rest\})$, where $M_{PC}$ is combined maintenance, $M_P$ is preventive maintenance only, and $M_C$ is component repair only.

The reward function, equation 2.1, is linearly decreasing in $b(F)$, and $b_2 = b^* + \Delta_b$, where $\Delta_b > 0$, so:

$$G(S_2) = G(S_1) - \Delta_b(1 - \delta) \tag{A.2}$$

In state $S_2$, we hold the other systems constant, and only adjust $b(F)$ for system $j$. This means the operating constraint is still met, and that maintenance capacity exists for $S_2$, as it did for $S_1$. We wish to show that $V(S_2; u_{jt} = M_{PC}) > V(S_2; u_{jt} \in \{M_P, M_C, Operate, Rest\})$, and we consider each decision in turn.

Combined maintenance resets both the component and the hours, which resets $S_1$ and $S_2$ into identical states, so $V(S_1; u_{jt} = M_{PC}) - G(s_2) = V(S_2; u_{jt} = M_{PC}) - G(s_1)$ gives

$$V(s_2; u_{jt} = M_{PC}) = V(s_1; u_{jt} = M_{PC}) - \Delta_b(1 - \delta) \tag{A.3}$$

We now establish that the value of $S_1$ minus the current reward offset shown in equation A.3, when we execute one of the other actions for system $j$, $u_{jt} \in \{M_P, M_C, Fly, Rest\}$ followed by the optimal policy, is at least as large as the value of $S_2$ when we execute the same action followed by its optimal policy.

Because component maintenance resets $b(F)$, states $S_1$ and $S_2$ are identical afterwards, and therefore have the same subsequent optimal policy, so $V(s_1; u_{jt} = M_C) - \Delta_{b(F)}(1-\delta) = V(s_2; u_{jt} = M_C)$, and therefore $V(s_1; u_{jt} = M_{PC}) > V(s_2; u_{jt} = M_C)$.

Preventive maintenance resets the hours, and does not not change $b(F)$. Consider, for $S_2$, executing preventive maintenance for system $j$ and then following the optimal policy, as in equation A.4, where $S_2' = S'|S = S_2, u_{jt} = M_P$.

$$V(S_2, u_{jt} = M_P) = G(S_2) + \gamma E[G(S_2')] + \gamma^2 E[V(S_2'')] \tag{A.4}$$

If $S_1$ follows an identical policy to the one in equation A.4, then it will have a larger value. State $S_1$ has a larger current reward, and the difference continues after preventive maintenance. The difference in reward will change if $b(F)$ changes, which could happen from two possible actions in the final term of equation A.4. First, during the subsequent stages using the optimal policy of $S_2$, the components are repaired, in which case the states become identical. Second, $b(F)$ could change if the subsequent stages using the optimal policy from $S_2$ include operating. From condition 2, we know that after operating, since $b_2 > b_1$, $E[b_2'] \geq E[b_1']$. This means that state $S_1$ following the policy from equation A.4 will continue to have a value at least equal to the value for state $S_2$. So, for state $S_1$, executing preventive maintenance and then following the optimal policy has a value at least as large as equation A.4 plus the current reward offset, $V(S_1, u_{jt} = M_C) \geq V(S_2, u_{jt} = M_C) + \Delta_b(1-\delta)$.

This same argument holds for operating and for resting. If in $S_2$ system $j$ operates or rests, and then continues with the optimal policy, then $S_1$ following the same policy will produce the same value, offset by the current reward.

So, the value when state $S_1$ executes one of the four non-combined actions and then follows the optimal policy is at least as large as the value for state $S_2$ executing the same action followed by its optimal policy, plus the current reward offset. Therefore $u_{jt} = M_{PC}$ remains optimal for $b > b^*$. □

**Proof of Corollary 2.4.2.1**

*Proof.* This follows directly from Proposition 2.4.2, under the updated belief transitions described in Section 2.3; if the belief of failure, $b(F)$, for any system is directly a function of the exogenous model, then $b(F)$ will take one of three values, $b(F) \in \{0, 1-\zeta, \zeta\}$. So, if $P(comp = F | o = \text{“}Failing\text{”}) = \zeta$, then when $\zeta$ increase, $b(F)$ increases, and the result holds from Proposition 2.4.2. □

## A.3 Additional Use-Case Information

As a use-case for this method, we look at a U.S. Army unit's fleet maintenance for MH-60 helicopters.

### A.3.1 Use-case Background

Because of the risk of a crash, aviation maintenance is focused on regular preventive maintenance inspections, which include detailed checks at various intervals, frequently based on flight hours. These maintenance operations have many facets, but they generally occur on one of two planning horizons: a short-term horizon focused on immediate repairs and periodic inspections, and a long-term horizon focused on intensive multi-week inspections that include tearing down, inspecting, and re-assembling large portions of the aircraft. These maintenance horizons typically have different maintenance teams, so the team that is focused on the regularly reoccurring regular checks is the same team that will execute the unscheduled repairs when a component breaks. This short-term horizon is our focus.

Decisions about maintenance impact which aircraft are available, and decisions on which

aircraft fly impact the required maintenance tasks since the number of flight hours drives the bulk of the inspections. Each week, the unit has to allocate resources and accomplish its assigned tasks, while remaining prepared for unscheduled events. Unscheduled maintenance can disrupt the plan, both reducing aircraft availability, and in the extreme, making the unit unable to meet its mission requirement. Advanced knowledge of these disruptions would allow a unit to account for the required time and resources when building the schedule. However, the incorporation of these predictions into the decision process is non-trivial because the flight and maintenance schedules impact each other, and because of the unit's limited maintenance capacity. The primary metric the army uses to assess maintenance success in a unit is aircraft availability, or the number of aircraft available to fly a mission on that day. This is frequently measured as a percentage of total aircraft, and referred to as the Operational Readiness (OR) rate.

The Department of Defense established the Joint Artificial Intelligence Center in 2018 to leverage AI for national security. One of the two initial initiatives was (and remains) predictive maintenance, with the goal of improving the availability of military airframes by minimizing maintenance down time. One of the primary efforts within the initiative is producing predictive models that help units and agencies address the challenges and disruptions of unscheduled maintenance. An initial proof-of-concept model for predicting component-level faults focused on a specific mode of failure on the engines of the H-60 helicopter, an airframe common across the military services. When engines build up sand, the heat from the engine can turn the sand into glass, and as the glass builds up, the engine runs at hotter temperatures. If the sand continues to "glass", then operating the engine could be unsafe. There is a built-in safety to prevent this unsafe operation, where, on the ground, the engine will not start if the temperature reaches a certain threshold. This prevents a catastrophic failure, but the aircraft is not available for operation. The proof-of-concept model is a binary prediction for this type of "hot-start failure", at a certain time horizon. For example, predicting this type of failure within the next 10 flight hours.

### A.3.2 Use-case Sensitivity Analysis

This section presents the results on the maintenance index from varying four parameters: the reward function, the initial conditions, the flying hour distribution, and the discount factor. The key take-away is that the general maintenance index described in Section 2.6 is not sensitive to these parameters. We show the sensitivity by modifying Figure 2-6 with different rows associated with different parameter values.

Our reward function provides no reward for broken aircraft or aircraft in maintenance, and provides a reward of 1 for aircraft with a healthy component. For aircraft with a failing component, it provides a value of $\delta \in (0, 1]$, where a value of 1 matches current practice, since healthy and failing components are indistinguishable. We can take the base index plot and vary the reward function to see how our actions change as $\delta$ changes.

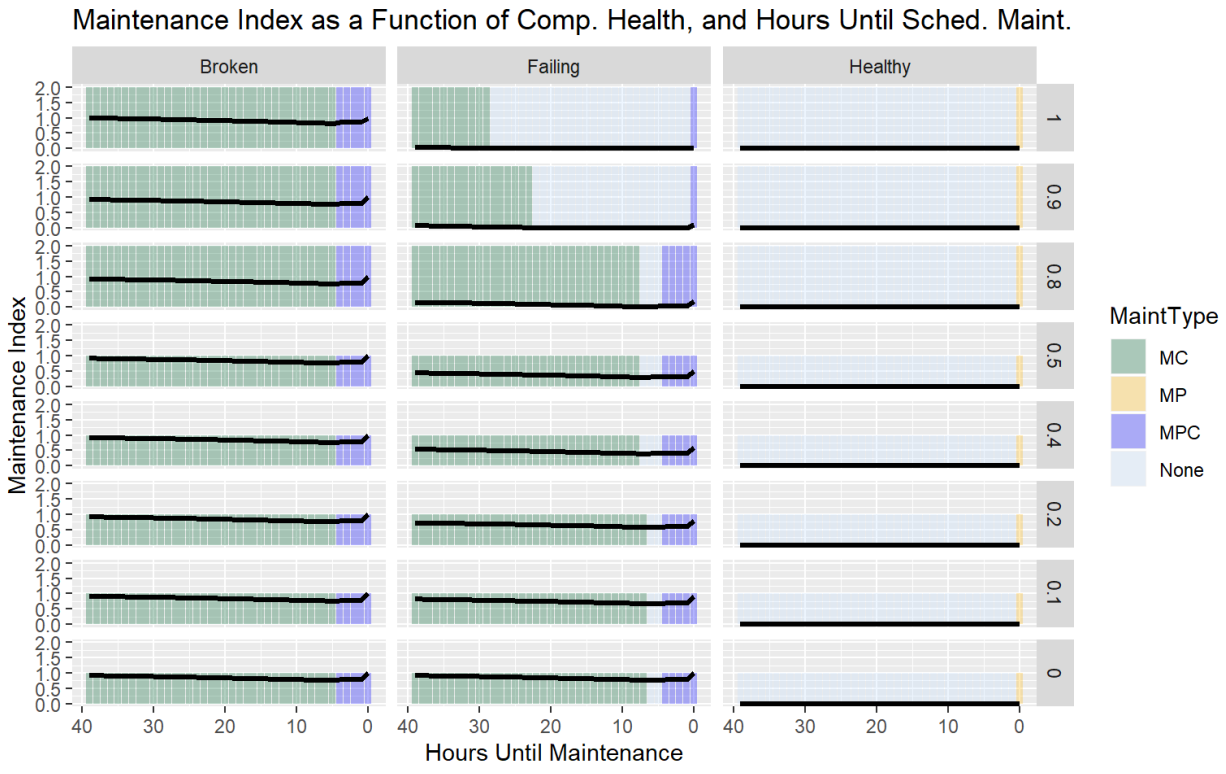


Figure A-1: A plot of the maintenance index and heuristic actions for varying the reward function.

In Figure A-1, we see the base plot for a specific instance, with various levels of $\delta$ as the rows. The top row shows the index and policy when the failing component receives the same reward as the healthy component. We can see in the middle panel for the failing component that at the required preventive maintenance inspection, we combine repairs (tier 4), but that we do not execute the preventive inspection prior to its required point (tier 5). As the reward decreases, we see a change when the reward is 0.8. In this case, the additional blue in the middle panel shows that we execute combined repair if we have a failing component once we reach the beginning of the tolerance window, hour 36. Because it is better to have a healthy component, it makes sense to repair a failing one when we can, in this case with a combined repair.

The initial conditions do not impact our policy. We see in Figure A-2 that for each of the five initial condition cases considered, we end up with the same index and the same policy. These five cases for the initial condition are: uniform across the healthy states, uniform across the failing states, some aircraft initially in maintenance, and two cases of random subsets of healthy states.

The lack of sensitivity to an initial condition means that our heuristic policy can be computed in advance, offline, and does not need to be re-solved for a specific instance.

The flying hour distribution, for similar instances, does not impact our policy. We see in Figure A-3 that for various ranges in a triangular distribution (with a peak at 4 hours and the range varying from 5 to 10 hours), the index and policy are the same.

The discount factor significantly impacts the model solution, but the variation in the solutions highlights the discount factors that make it applicable for our application.

Figure A-4 depicts the base plot for a specific instance, with various levels of $\gamma$, the discount factor, as the rows. In the top row, with a discount factor of 0.5, we see that when the component is broken (left panel), we execute a repair, and if the break happens at hour 40, we execute the repair and conduct the preventive maintenance. If the component is failing,

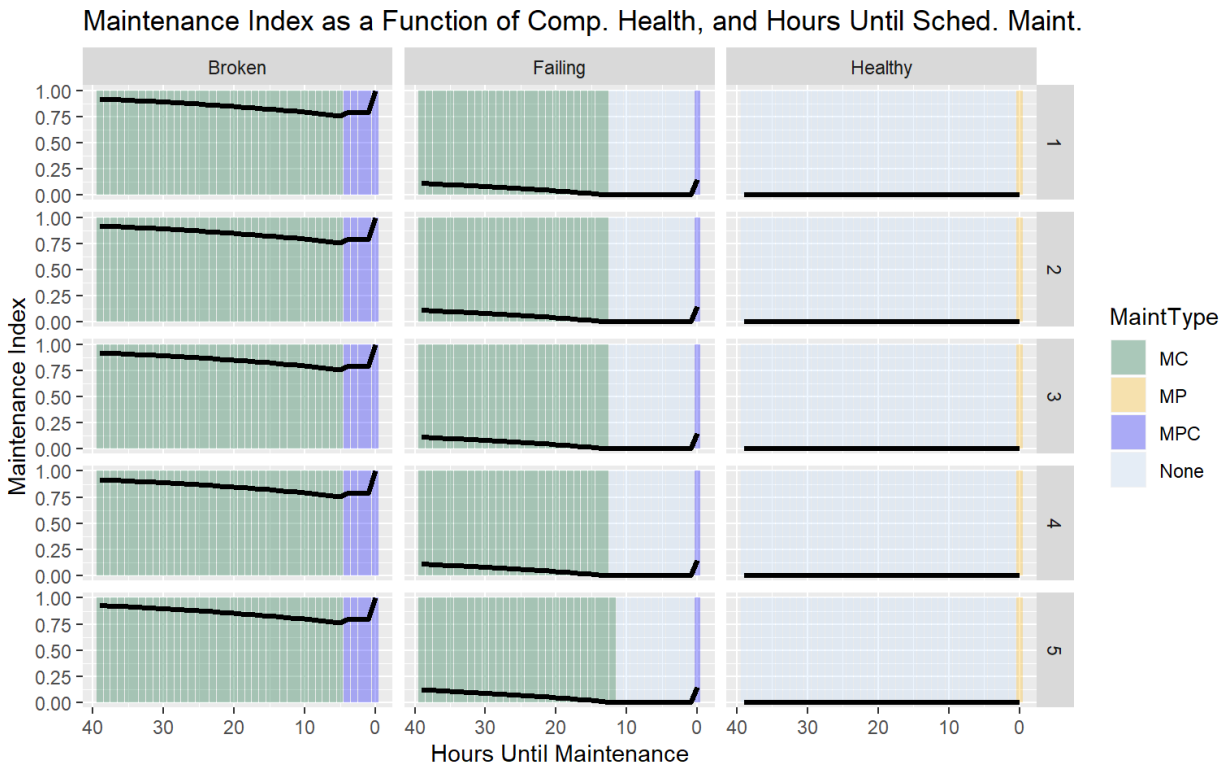


Figure A-2: A plot of the maintenance index and heuristic actions for varying the initial conditions, for five cases.

the only action we take is to conduct regular preventive maintenance at hour 40, with no repair.

In the bottom row, for a discount factor of 0.9, we see that when the component is failing, we execute repairs for low flight hours, and we combine maintenance at hour 40. This shows us that only with a discount factor of 0.9 - "looking" farther into the future - does it make sense to begin paying attention to possible failing components and their repair. Aviation units are of course focused on long-term as well as short-term readiness, and we therefore adopt a discount factor greater than 0.9 for the remainder of the analysis.

We can also assess the necessary health signal quality as the discount factor varies. We see slight variations in the location of the threshold between the region where the prediction model performance is high enough for the policy to match the certain case, and when it is

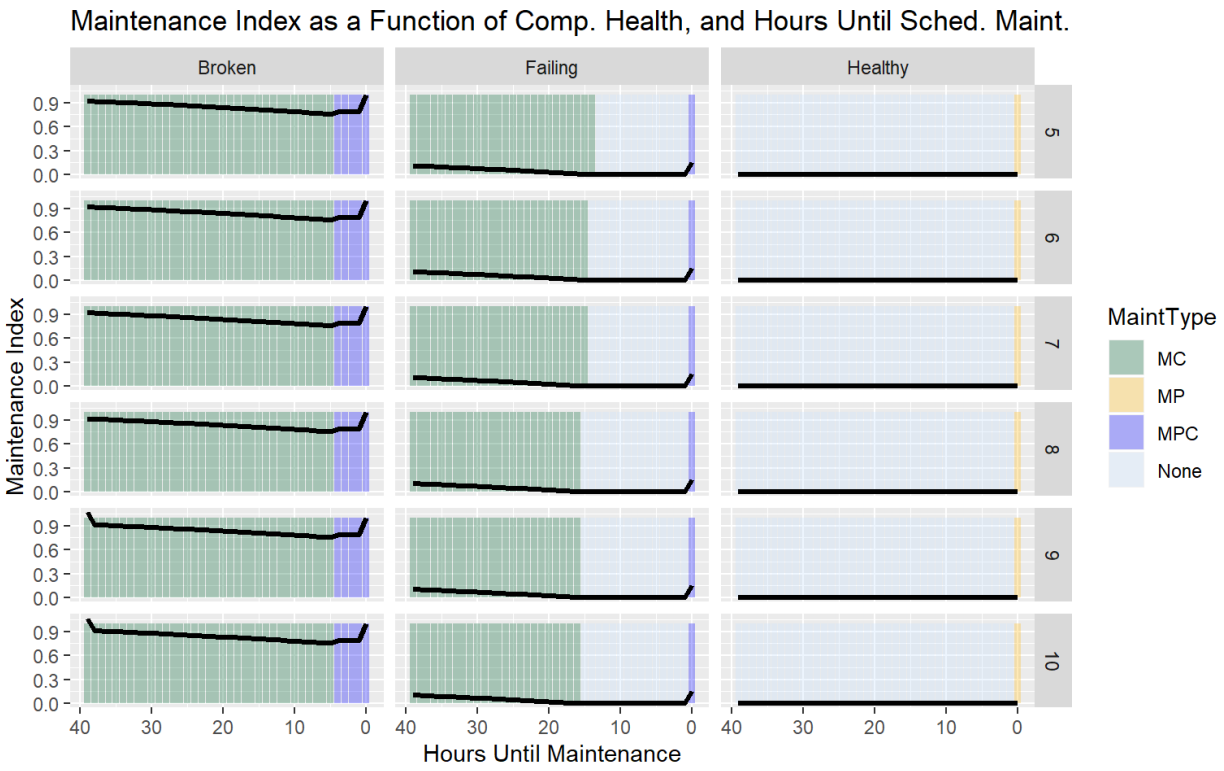


Figure A-3: A plot of the maintenance index and heuristic actions for varying flying hour distributions (triangular; varied peaks).

not.

Figure A-5 shows the threshold between regions for Opportunistic Component Maintenance for varying discount factors.

### A.3.3 Use-case Simulation

We simulate the change in Operational Readiness rate, the unit's primary performance metric for up-time, with the LP-driven heuristic policy to confirm the insight about how good the health signal's source prediction model must be before it provides value.

We simulate the readiness of an aviation unit with various repair policies to determine the effectiveness of our policy and its impact on unit downtime. The execution of our simulation is based on iterating the maintenance practices in a unit over a multi-year period. We

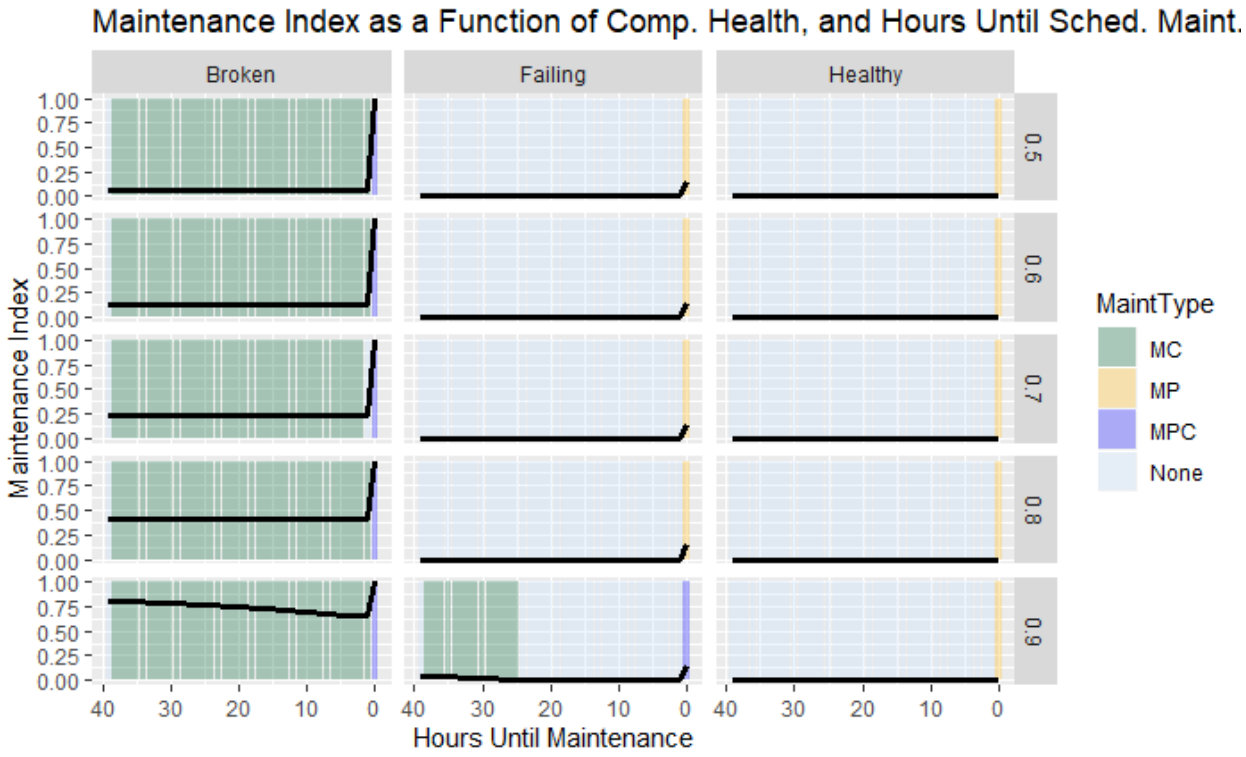


Figure A-4: A plot of the maintenance index and heuristic actions for varying discount factors.

use historical maintenance and flight records from a United States Army aviation unit as a baseline for the input parameters.

For an individual fault, we evaluate the benefit of a policy, for scheduling, with equation A.5. This takes the ratio of the OR rate improvement from using that policy over the baseline to the OR rate improvement that would happen from the complete absence of this fault. This shows us the percent of this fault's downtime contribution that could be removed by using the selected policy.

$$\text{Benefit of Policy} = \frac{ORRate_{Policy} - ORRate_{Baseline}}{ORRate_{NoFault} - ORRate_{Baseline}} \tag{A.5}$$

We compare three policies, and define a baseline.

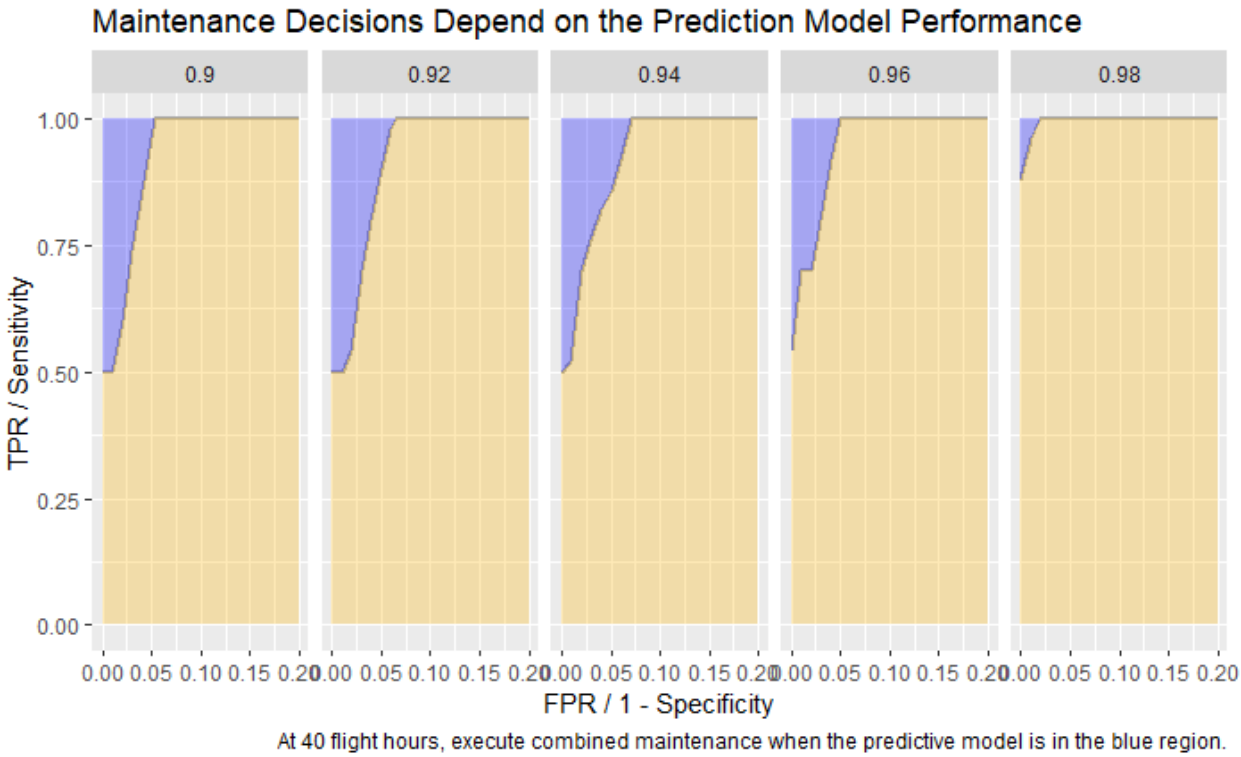


Figure A-5: A depiction of the necessary quality of the health signal's source predictive model as the discount factor varies.

(i) **Heuristic Policy, with Fully Observable Component Degradation Process:** We have full knowledge of the state of the component, and we execute preventive and corrective maintenance using the index policy defined here.

(ii) **Heuristic Policy, with Partially Observable Component Degradation Process:** We have partial knowledge of the state of the component, and we execute preventive and corrective maintenance using the index policy defined here.

(iii) **Myopic Policy, with Partially Observable Component Degradation Process:** We have partial knowledge of the state of the component, and we execute a naive policy where we treat a failing component as in need of immediate maintenance (as if it is broken).

(iv) **Baseline:** We have no knowledge of the state of the component unless it breaks, and

we then we execute repairs.

The first policy, with fully observable system, shows the maximum possible benefit from incorporating signal-driven pre-emptive repairs, assuming a perfect signal. We compare in the realistic setting of a partially observable system the LP-driven heuristic developed previously, to a myopic implementation where repairs are executed on failing components as they are for broken components.

For a representative fault, we see the following results from the simulation, and see similar results for other parameters on part reliability and maintenance length. A single fault has a low prevalence, and so the difference in the overall OR rate is small. However, the eventual operational goal is to have many component health signals. The key result is that there are benefits from having a component health signal, but they don't realize until the health signal's source prediction model is almost perfect. For the partially observable system below, we evaluated a prediction model with sensitivity of 0.97, specificity of 0.99, and a precision of 0.97 - exceptionally high. While this was right above the threshold for these parameters, in simulation it does no better than the baseline.

| Policy | OR Rate | Benefit |
|---|---|---|
| Baseline | 88.58 % | – |
| LP Heuristic, Fully Observable System | 88.72 % | 27% |
| LP Heuristic, Partially Observable System | 88.58 % | 0% |
| Myopic, Partially Observable System | 88.39 % | -37 % |

Table A.1: Simulated OR Rate, and the percent of the component's impact on downtime that could be removed by various policies.

There is clearly a benefit from incorporating advanced knowledge of a part into scheduling. In this example, we mitigate 27 % of the component fault's impact on OR rate through smart maintenance combinations, *if* we have a perfect health signal. But even for an exceptionally good health signal source prediction model, we see no measurable gain over the baseline. If we don't smartly combine maintenance (tiers 4 and 5) when we have a health signal, as in the myopic policy, we actually hurt the OR rate.

# Appendix B

# Supplement for Chapter 3

## B.1 Graph and Path Generation

### B.1.1 Generating the Graph

We generate the graph, $\mathcal{G}$ from the set of assignment categories, $S$, and the time horizon, $T$.

**Algorithm 3:** Generate the Graph, $\mathcal{G} = (V, E)$

```
Result: G = (V, E)
Input  : S, T
Create officer source node b_0
for t ∈ {1, ..., T} do
    for s ∈ S do
        Create node (s, t)
        if t > 1 then
            for k ∈ S do
                Create edge e from (k, t − 1) to (s, t)
            end
        else
            Create edge e from b_0 to (s, 1)
        end
    end
end
```

### B.1.2 Path-based Formulation

The primary purpose for using a path-based formulation is to capture a history for each officer. One possibility for avoiding the computational difficulties of the path-based formulation would be to capture the guidance with edge and node adjustments. For example, if one element of guidance was that someone in a blue job could never move to a red job, then we could induce a sub-graph of $\mathcal{G}$ with no edges connecting blue and red jobs. This might then enable us to use an edge-based, flow-style, formulation. However, many of the restrictions in practice do not have a sub-graph variant. Consider a five period expansion of the sample graph in Figure 3-2. Table B.1 gives examples of some professional development restrictions that cannot be modeled by inducing a sub-graph. In many cases, this is because another path exists that meets the criteria, but uses one of the same edges as the path that violates the constraint. Additionally, the path-based version allows us to consider a concise way for

adding flexibility, with the addition of paths.

| Example Restriction | Path Violation | Problem with Subgraph Version |
|---|---|---|
| 'Must Contain Blue' | GGGGG | Can't remove all G-G Edges |
| 'If Blue, Two Periods' | BBBRR | Can't remove all B-B edges |
| 'Once Blue, No Green' | BBRRG | Can't remove all R-G edges |
| 'If Red, Blue Before' | GGRRB | Can't remove all G-R edges |
| 'If Multiple B, Together' | BGBRR | Can't remove all B-G edges |
| 'If Red, No Blue Before' | BGGRR | Can't remove all B-G edges |

Table B.1: Example assignment guidance that requires path-based formulations and cannot be expressed by inducing a sub-graph of $\mathcal{G}$.

### B.1.3 Guidance-allowed Paths

In many cases we can map the professional development guidance into rules that we can use to determine if a certain path is allowed. This is the motivating logic behind the set of constraints in **PPC**. In our example $\mathcal{G}$ from Figure 3-2, we consider all of the possible paths in $P$, and then apply 9 realistic rules. On the left of Figure B-1, we depict the 27 paths as the three color combination for the three periods. We index rule $i$ as $d_i$, and depict which paths would still be allowed under that rule (green check) and which are not allowed (red x). With these 9 applied rules, only 2 of the 27 paths are allowed.

This relative sparsity of paths holds on larger examples as well. With $|S| = 8$, $T = 8$, and 19 realistic rules, $|P| = 1.7e7$, but after all the rules are applied, only 109 paths are allowed (0.001% of the total).

### B.1.4 Generating Paths from Data

The key formulation input is the set of allowed paths, $P_A$, which we can estimate from assignment histories.

As the granularity of the time period increases, the combinatorial number of possible paths increases, and more people use unique paths. To balance the need for a more granular time

| Path | Period 1 | Period 2 | Period 3 | $d_1$ *Precedence* Red Only After Blue | $d_2$ *Precedence* Green Only After Blue | $d_3$ *MustContain* Must Include Blue | $d_4$ *FreezeAfter* After Blue, Same Color | $d_5$ *IfTimingReq* If Blue, One Period | $d_6$ *NotAllowedAfter* Once Red, No Blue | $d_7$ *OneBlock* If Green, Together | $d_8$ *NotBefore* If Red, No Green Before | $d_9$ *ReqTransition* If Green To X, Must be Red |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | | | ✓ | ✓ | ✓ | ✓ | X | ✓ | ✓ | ✓ | ✓ |
| 2 | | | | ✓ | ✓ | ✓ | ✓ | X | ✓ | ✓ | ✓ | ✓ |
| 3 | | | | ✓ | ✓ | ✓ | ✓ | X | ✓ | ✓ | ✓ | ✓ |
| 4 | | | | ✓ | ✓ | ✓ | X | ✓ | ✓ | ✓ | ✓ | X |
| 5 | | | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 6 | | | | ✓ | ✓ | ✓ | X | ✓ | ✓ | ✓ | X | ✓ |
| 7 | | | | ✓ | ✓ | ✓ | X | X | X | ✓ | ✓ | ✓ |
| 8 | | | | ✓ | ✓ | ✓ | X | ✓ | ✓ | ✓ | ✓ | ✓ |
| 9 | | | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 10 | | | | ✓ | X | ✓ | ✓ | X | ✓ | ✓ | ✓ | X |
| 11 | | | | ✓ | X | ✓ | ✓ | ✓ | ✓ | X | ✓ | X |
| 12 | | | | ✓ | X | ✓ | ✓ | ✓ | ✓ | ✓ | X | X |
| 13 | | | | ✓ | X | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | X |
| 14 | | | | ✓ | X | X | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 15 | | | | X | X | X | ✓ | ✓ | ✓ | ✓ | X | ✓ |
| 16 | | | | X | X | ✓ | ✓ | ✓ | X | ✓ | X | ✓ |
| 17 | | | | X | X | X | ✓ | ✓ | ✓ | X | X | ✓ |
| 18 | | | | X | X | X | ✓ | ✓ | ✓ | ✓ | X | ✓ |
| 19 | | | | X | ✓ | ✓ | ✓ | X | X | ✓ | ✓ | ✓ |
| 20 | | | | X | ✓ | ✓ | ✓ | ✓ | X | ✓ | ✓ | ✓ |
| 21 | | | | X | ✓ | ✓ | ✓ | ✓ | X | ✓ | ✓ | ✓ |
| 22 | | | | X | X | ✓ | ✓ | ✓ | X | ✓ | ✓ | X |
| 23 | | | | X | X | X | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 24 | | | | X | X | X | ✓ | ✓ | ✓ | ✓ | X | ✓ |
| 25 | | | | X | ✓ | ✓ | ✓ | ✓ | X | ✓ | ✓ | ✓ |
| 26 | | | | X | X | X | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 27 | | | | X | ✓ | X | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

Figure B-1: The 27 possible paths in our example $\mathcal{G}$ from Figure 3-2, annotated as allowed or not allowed based on the rule in the associated column.

period without omitting possible paths not included in the assignment history, we use a decomposition approach, depicted in Figure B-2.

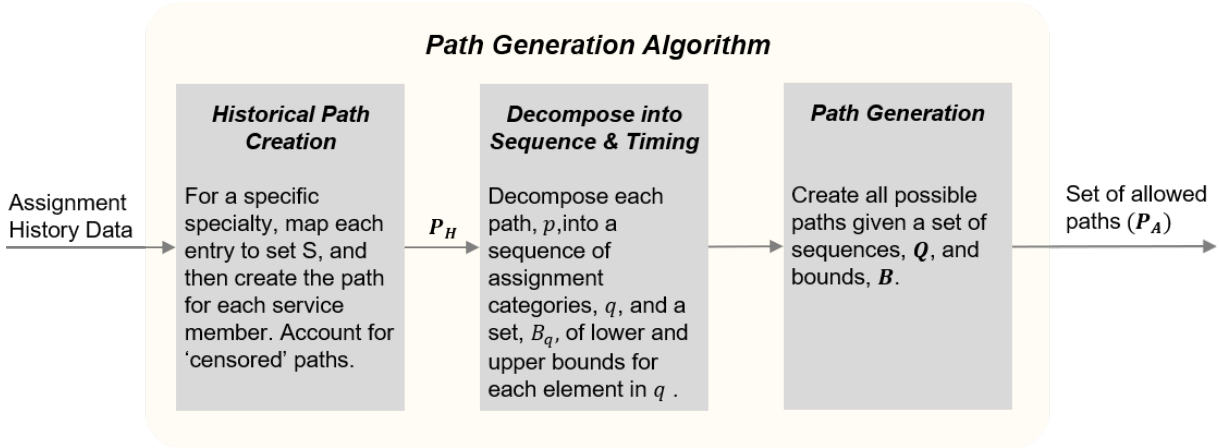


Figure B-2: A description of the path generation algorithm that takes assignment history data, creates historically used paths, decomposes them into sequences and timing, and creates the input set of allowed paths, $P_A$.

We take as an input the assignment history data for personnel in the specialty of interest, and then map the assignments to $S$ while grouping them to account for time periods. This

gives us the historical path set $P_H$. To find very similar paths that are missing from $P_H$ but should be in $P_A$, we then decompose the historical paths into sequences and timing. A sequence is the list of assignment categories for a service member, in order, but with no consideration of timing. The timing estimate is then a lower and upper bound on when certain assignments are held for similar sequences. We then generate all possible paths given the historical sequences and timing. In our numerical results in Section 3.6, we mapped the use-case branch monthly assignments into historical paths, and with a time horizon of $T = 16$, there were more than 600 unique full paths. When we categorize the sequences of the 1220 officers with continued service through year 10, we see that a majority of officers used a very small set of sequences, as shown in Figure B-3 (described in the middle box of Figure B-2). We also consider the truncated sequences from officers who depart the branch, along with the estimated time bounds for each assignment, to generate $P_A$.

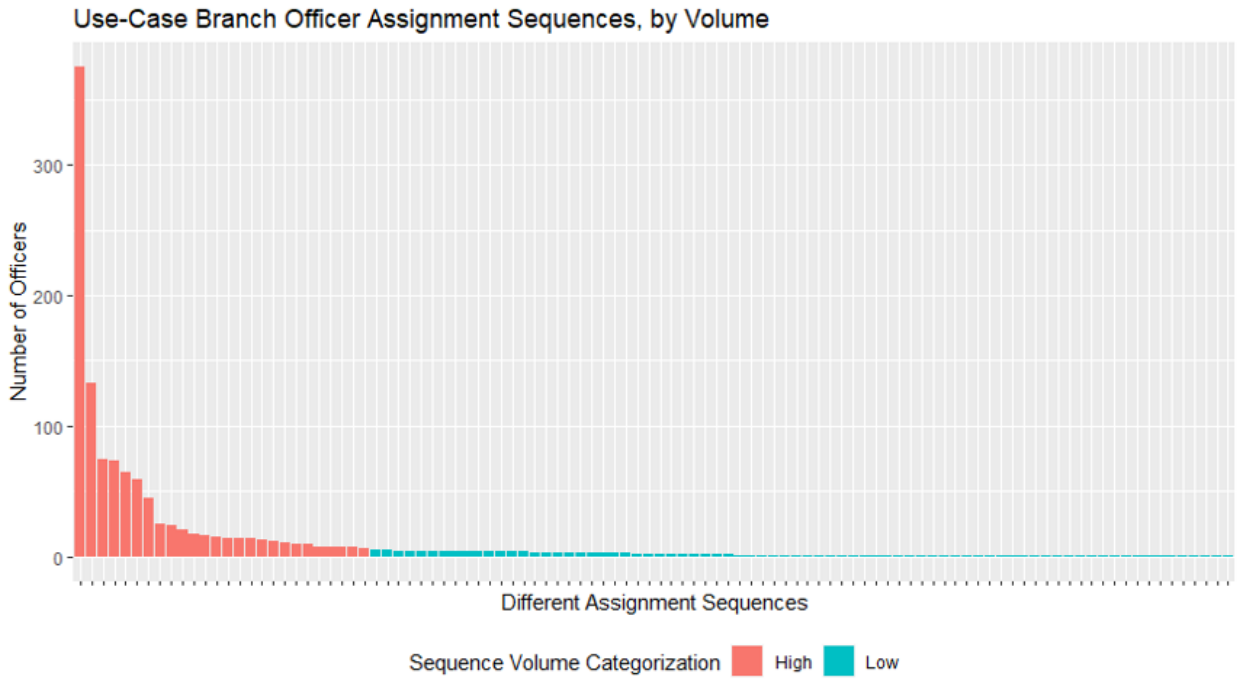


Figure B-3: A view of officer assignment sequences (horizontal axis) and the number of officers who completed each sequence, in our use-case branch. A sequence is an ordered set of assignment categories, with no specification on timing.

## B.2 Notes on the Optimization Formulation as a Set Function

We have the base LP formulation, $\mathbf{MPRP(P_A)}$, which determines the volume of personnel that should use each path in the set $P_A$ to minimize an overall readiness shortfall. The optimal objective value of this formulation for any input set $P_A$ is the set function $Z(P_A) = \mathbf{MPRP(P_A)}$. We consider adding $k$ paths from a ground set $\Omega$ to the set $P_A$, in order to minimize (improve) the value of $Z(P_A)$. For the set of $k$ paths selected from $\Omega \setminus P_A$, defined as $P_k$, we define the function $F(k)$ as the minimum value of $Z(P_A \cup P_k)$, that is: $F(k) = \min_{P_k : P_k \subseteq \Omega \setminus P_A, |P_k| = k} Z(P_A \cup P_k)$ (equivalent to $\mathbf{MPRPF_k(P_A)}$). We wish to establish sufficient conditions for the supermodularity of $Z$ and the convexity of $F$. Unfortunately, the conditions described here do not apply to the use-case.

First, we describe a greedy algorithm for solving $\mathbf{MPRP(P_A)}$, and establish when that greedy algorithm is optimal. We then describe the conditions for when $Z$ is supermodular, leveraging the optimal greedy algorithm. We then describe the conditions for when $F$ is convex, leveraging the optimal greedy algorithm, and provide an example that shows the convexity of $F$ does not imply the supermodularity of $Z$.

Algorithm 4 iteratively determines $f_p$, the volume of personnel on paths in $P_A$. Paths are selected by an index, $\lambda_p$, which is the amount the objective value would change based on the maximum allocation of available volume to that path. The objective is the weighted sum of the readiness shortfall for the different assignment categories, $s \in S$, and a single path can impact the objective portions of different assignment categories. The index is re-calculated when the readiness shortfall in an assignment category drops to 0. Feasibility is maintained as no allocation is allowed to break the assignment category upper bound or the overall resource cap.

Consider the example network with three paths in Figure B-4 with three paths numbered $1, 2, 3$ from the top. If $\lambda_2 > \lambda_1$ and $\lambda_2 > \lambda_3$, then when we solve $\mathbf{MPRP(P_A)}$ with Algorithm

4 we first maximize use of path 2. But if that use reaches the capacity limit of Green (and/or and Red), then $f_1 = 0$ (and / or $f_3 = 0$) because there is no remaining capacity for path 1 (and / or path 3). However, it might be optimal to have $f_2 = 0$ and to maximize the use of $f_1$ and $f_3$, particularly as path 1 and path 3 do not share any common $s \in S$.

In a sense, path 2 has 'blocked' the use of two other paths, and the combined use of those two paths could be better than the use of just path 2. This example motivates Lemma B.2.1 which describes sufficient conditions for Algorithm 4 to solve $\mathbf{MPRP}(\mathbf{P_A})$ to optimality. This condition effectively prevents the 'blocking' described above.

**Lemma B.2.1.** *If there does not exist a subset of three paths $p, q, l \in \Omega$ : $p$ and $q$ share a common $s_1 \in S$ and $p$ and $l$ share a common $s_2 \neq s_1 \in S$, then the optimal solution to $\mathbf{MPRP}(\mathbf{P} \in \mathbf{\Omega})$, $f^*$, can be found with a greedy algorithm.*

## Conditions for the Supermodularity of $Z(P_A)$

The supermodularity of the set function $Z(P_A)$ would allow for additional certainty when leveraging Algorithm 4, and when using $\mathbf{GFA_k}$. This would mean that adding a path to the smaller set, $P$, has a more negative change; equivalently, that adding a path to the larger set makes less of difference (it makes doesn't make it as negative). If $Z$ is supermodular, then $-Z$ is submodular (so readiness is submodular). Unfortunately, sufficient conditions for supermodularity are very restrictive.

**Proposition B.2.2.** *When there does not exist a subset of three paths $p, q, l \in \Omega$ : $p$ and $q$ share a common $s_1 \in S$ and $p$ and $l$ share a common $s_2 \neq s_1 \in S$, then $Z : 2^{\Omega} \to \mathbb{R}$ is supermodular.*

## Conditions for the Convexity of $F(k)$

If $F$ is convex, then we have a better understanding of the behavior of $\mathbf{MPRPF_k}$ as we add paths.

**Proposition B.2.3.** *When there does not exist a subset of three paths $p, q, l \in \Omega$ : $p$ and $q$ share a common $s_1 \in S$ and $p$ and $l$ share a common $s_2 \neq s_1 \in S$, $F(k)$ is convex-extensible; $\exists \tilde{F} : \mathbb{R} \to \mathbb{R}$, that is convex, where : $\tilde{F}(k) = F(k) \quad \forall k \in \mathbb{Z}_+ \cup \{0\}$.*

Even under these conditions, the convexity of $F$ does not imply the supermodularity of $Z$. Consider the following example, with parameters below, depicted in Figure B-4. In this case, $F$ is convex, but $Z$ is not supermodular.

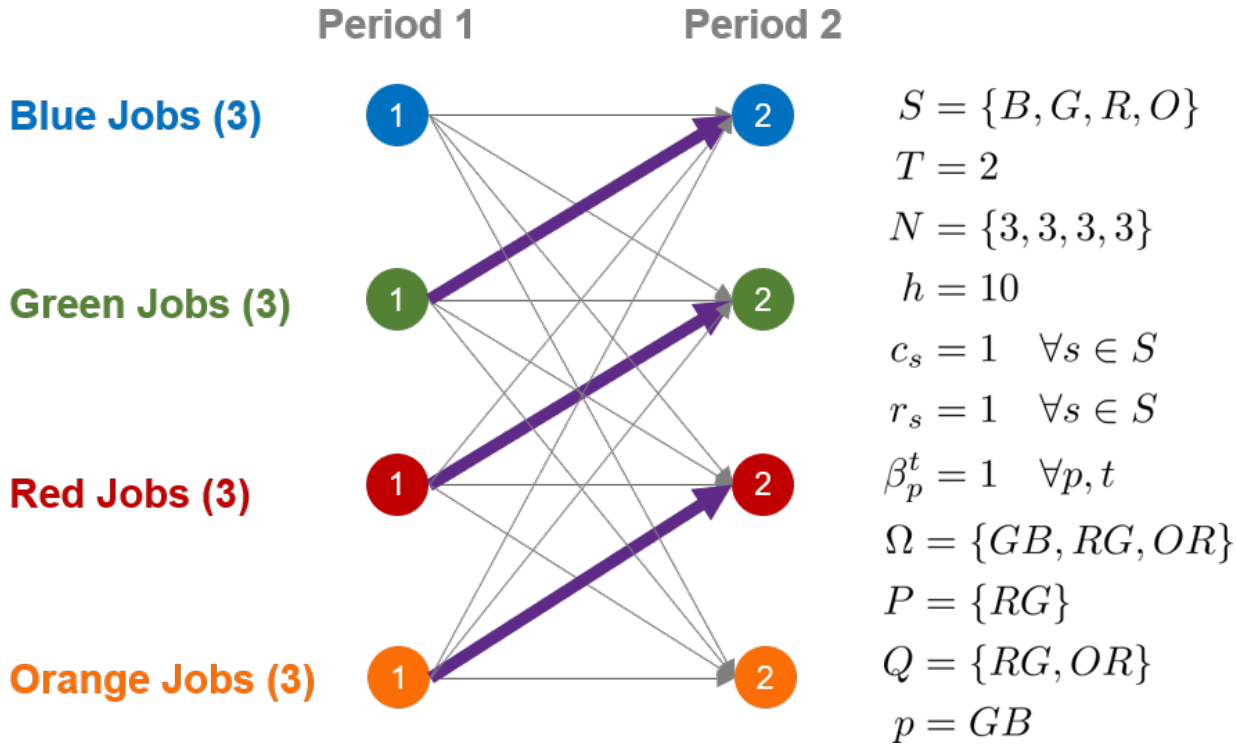


Figure B-4: Simple three path example for when Algorithm 4 is not optimal, and a numerical example showing that the convexity of $F(k)$ does not imply the supermodularity of $Z$.

**$Z$ is not Supermodular:**

$$
\begin{aligned}
P &\subset Q \\
Z(P) &= \mathbf{MPRP}(\{RG\}) = 6 \\
Z(Q) &= \mathbf{MPRP}(\{RG, OR\}) = 6 \\
Z(P \cup \{p\}) &= \mathbf{MPRP}(\{RG\} \cup \{GB\}) = 6 \\
Z(Q \cup \{p\}) &= \mathbf{MPRP}(\{RG, OR\} \cup \{GB\}) = 0 \\
Z(P \cup \{p\}) - Z(P) &= 0 \\
Z(Q \cup \{p\}) - Z(Q) &= -6 \\
Z(P \cup \{p\}) - Z(P) &\nleq Z(Q \cup \{p\}) - Z(Q)
\end{aligned}
$$

**$F$ is Convex:**

$$
\begin{aligned}
P_A &= \{OR\} \\
Z(P_A) &= \mathbf{MPRP}(\{OR\}) = 6 \\
F(0) &= 6 \\
F(1) &= \mathbf{MPRP}(\{OR, GB\}) = 0 \\
F(2) &= 0
\end{aligned}
$$

## B.3 Proofs

**Proof of Proposition 3.4.1**

*Proof.* Let $f^*_{\mathbf{MPRPF_k}}, d^*_{\mathbf{MPRPF_k}}$ be the optimal solution to $\mathbf{MPRPF_k}(\mathbf{P_A}, \mathbf{P'})$. When $k$ increases by 1, one additional value of $f_p, p \in P'$ can be positive. Our existing solution $f^*_{\mathbf{MPRPF_k}}, d^*_{\mathbf{MPRPF_k}}$ is still feasible in the new problem, so we only consider a new value of $f_p > 0$ if it improves the solution. Let $f^*_{\mathbf{GFA_k}}$, $d^*_{\mathbf{GFA_k}}$, be the optimal solution to the final

iteration of $\mathbf{MPRP}(\tilde{\mathbf{P}}_\mathbf{A})$ in $\mathbf{GFA_k}(\mathbf{P_A})$. When $k$ increases by 1, the algorithm can iterate one additional time, but only if there exists a path $p$ that can improve the value of the solution. □

**Proof of Proposition 3.4.2**

*Proof.* The first inequality holds since in $\mathbf{MPRP}(\mathbf{P})$, the objective has non-negative weights on a sum of variables, $d_s \geq 0$. The second inequality holds as $\mathbf{MPRPF_k}(\mathbf{P_A}, \mathbf{P} \setminus \mathbf{P_A})$ considers all of the paths in $P$, but selects only a subset that are not in $P_A$. The third inequality holds because $P' \subseteq P \setminus P_A$, and so the feasible region of $\mathbf{MPRPF_k}(\mathbf{P_A}, \mathbf{P} \setminus \mathbf{P_A})$, is not smaller than the feasible region of $\mathbf{MPRPF_k}(\mathbf{P_A}, \mathbf{P'})$. The fourth inequality holds since $\mathbf{GFA_k}(\mathbf{P_A})$ can select $k$ paths from the set under consideration in $\mathbf{MPRPF_k}(\mathbf{P_A}, \mathbf{P'})$ which selects the optimal $k$ subset. The final inequality holds as $\mathbf{GFA_k}(\mathbf{P_A})$ starts with the solution to $\mathbf{MPRP}(\mathbf{P_A})$. □

**Proof of Lemma 3.5.1**

*Proof.* Define paths $p = \tilde{s}$ as paths that start, remain, and end on $s$, and set $\tilde{S}$ as the collection of those paths. If $\frac{1}{\sum_{t=1}^{T} \beta^t} rN \leq b$ then we can decouple the flow through the system, and $f_{\tilde{s}} = \frac{1}{\sum_{t=1}^{T} \beta^t} rn_s \quad \forall s \in S$ is feasible, and $d_s = 0 \quad \forall s \in S$.

If $\frac{1}{\sum_{t=1}^{T} \beta^t} rN > b$, then we iteratively assign $f_{\tilde{s}} = \frac{1}{\sum_{t=1}^{T} \beta^t} rn_s$ for each $s$ until $\sum_{\tilde{s} \in \tilde{S}} f_{\tilde{s}} = b$. For any $s$ where we allocated personnel volume in full, $d_s = 0$; for any $s$ where we allocated no volume, $d_s = rn_s$; if there is an $s$ with partial volume, it has a $d_s < rn_s$. We sum the lower bound constraints, using only the paths in $\tilde{S}$, and $\sum_{s \in S} d_s \geq rN - \sum_{\tilde{s} \in \tilde{S}} \sum_{t=1}^{T} \beta^t f_{\tilde{s}} = rN - \sum_{t=1}^{T} \beta^t b$. With equal $r$ and $c$ for each assignment category, no flow allocation between paths $\tilde{s}$ or an allocation to a path outside of $\tilde{S}$ can decrease the right hand side. The iterative assignment to $f_{\tilde{s}}$ is optimal. $\mathcal{Z}^*_{\mathbf{MPRP(P)}} = \max\big(0, c(rN - b\sum_{t=1}^{T} \beta^t)\big)$.

Consider the system with $|S| = 1$. There is only one path, $p = 1$, and all jobs are part of the same assignment category, $n_1 = N$. If $\frac{1}{\sum_{t=1}^{T} \beta^t} rN \leq b$ then $f_1 = \frac{1}{\sum_{t=1}^{T} \beta^t} rN$ is feasible, and

$d_1 = 0$. If $\frac{1}{\sum_{t=1}^{T} \beta^t} rN > b$ then $f_1 = b$, which is feasible, and $d_1 = rN - b\sum_{t=1}^{T} \beta^t$. We have used all available personnel volume, and the solution cannot improve. $f_1^* = \min(b, \frac{1}{\sum_{t=1}^{T} \beta^t} rN)$ is optimal and $\mathcal{Z}^*_{\mathbf{MPRP(P_1)}} = \max(0, c(rN - b\sum_{t=1}^{T} \beta^t))$. □

**Proof of Proposition 3.5.2**

*Proof.* In a steady-state system, the number of personnel is based on retention and cohort size, $O_A = b\sum_{t=1}^{T} \beta^t$. Applying the definitions and subtracting each term from $N$, the above proposition is equivalent to: $D \geq \mathcal{Z}^*_{\mathbf{MPRP(P_A)}} \geq N - b\sum_{t=1}^{T} \beta^t$. The first constraint holds because $\mathcal{Z}^*_{\mathbf{MPRP(P_A)}} = \sum_s d_s$ which is the number of unfilled positions, given the optimal allocation of personnel. The second constraint holds from Propositions 3.4.2 and 3.5.1, as $R_A$ measures the readiness shortfall in a system with no professional development constraints, **MPRP**(**P**). □

**Proof of Lemma B.2.1**

*Proof.* We have a solution, $f^*$, found using Algorithm 4 that is feasible by construction.

If $Z(P) = 0$, it cannot improve. If $Z(P) > 0$, there $\exists x \in S : d_x > 0$. In order to decrease $d_x$ and improve the objective, we would need to allocate additional volume to a path, $j : x \in S_j$.

There are two possibilities.

First, consider the case where $\exists j \in P_A : x \in S_j$, and path $j$ has no binding assignment category constraints, $s \in S_j$. Since the greedy algorithm did not allocate additional volume to $f_j$ to decrease $d_x$, we know that the overall volume $h$ is binding, which terminates the algorithm, and path $j$ was the final path to get a volume allocation. Therefore, $\forall p \neq j : f_p > 0$, we know that $\lambda_p \geq \lambda_j$. Shifting volume from path $p : f_p > 0$ to path $j$ increases (worsens) the objective since the index is the objective value change for a single path decision, and only the overall volume constraint was binding.

Second, consider the case where $\nexists g \in P_A : x \in S_g$ and path $g$ has no binding assignment

category constraints, $s \in S_g$. For every path $j : x \in S_j$, some other assignment category $y \neq x \in S$ is binding.

Consider path $q : x, y \in S_q$ and path $p : y \in S_p$. If $\lambda_q \geq \lambda_p$ then we have maximized the use of the capacity at $y \in S$ for decreasing $d_x$. If $\lambda_p > \lambda_q$ then we could re-allocate volume from path $p$ to path $q$ to decrease $d_x$. However, that single adjustment increases (worsens) the objective. We could improve the objective only if in addition to shifting volume from path $p$ to path $q$, $\exists z \in S_p$, and we are able to allocate additional volume to path $l : z \in S_l$. This would mean that using path $p$ before path $q$ or path $l$ prevented a better allocation to paths $q$ and $l$. But the existence of path $l$ violates the condition. $f^*$ is optimal.

□

**Proof of Proposition B.2.2**

*Proof.* We wish to establish that for $P \subseteq Q \subseteq \Omega$, $Z(P \cup \{p\}) - Z(P) \leq Z(Q \cup \{p\}) - Z(Q), \quad \forall p \in \Omega \setminus Q$. So, we are interested in solutions to $Z$ for four input sets: $P, Q, P \cup \{g\}, Q \cup \{g\}$, where $g \in \Omega \setminus Q$. For notation, on the optimal solution $f$ that solves for $Z$, we superscript the input set, as in $f^{P \cup \{g\}}$.

Based on the condition on path subsets in the proposition, if there exists a path $j$ that shares an $s_1$ with path $g$, then no other path shares $s_2 \neq s_1$ with path $g$. That is, if $\exists j : s_1 \in S_j, s_1 \in S_g$ then $\nexists l : s_2 \in S_l, s_2 \in S_g$. We can then split set $Q$ into two disjoint sets based on the paths that share a common $s_1$ with path $g$, set $Q_g$, and paths that do not share, $Q'_g$. $Q_g = \{p \in Q : \exists s \in S_p, s \in S - g\}$. $Q = Q_g \cup Q'_g$. We can then group the assignment categories $s \in S$ based on path $g$ and set $Q_g$.

$$
\begin{aligned}
&S^1 : \text{Set of } s \text{ shared between path } g \text{ and paths } p \in Q_g.\ s \in S_g \cap_{p \in Q_g} S_p \\
&S^2 : \text{Set of } s \text{ only used by path } g.\ s \in S_g, s \notin \cup_{p \in Q} S_p \\
&S^3 : \text{Set of } s \text{ not used by path } g \text{ and used by at least one path } j \in Q_g\ . \\
&\quad s : s \notin S_g, \exists j \in Q_g : s \in S_j \\
&S^4 : \text{Set of } s \text{ not used by } g \text{ or paths in } Q_g.\ s \notin S_g \cup_{j \in Q_g} S_j
\end{aligned} \tag{B.1}
$$

Based on the subset condition, $\nexists p \in Q'_g : s \in S^3, s \in S_p$. In other words, the assignment categories accessible by path $g$ and paths that share a common $s_1$ with path $g$ are only accessible by those paths.

We can then define $D_i = \sum_{s \in S^i} c_s d_s$ as the portion of the objective associated with category $i$, and the objective value is then $Z = D_1 + D_2 + D_3 + D_4$. We are interested in the difference in objective values based on changing the input set, and we superscript based on the common input set, so $\Delta D_1^P = D_1^{P \cup \{g\}} - D_1^P$. $Z(P \cup \{g\}) - Z(P) = \Delta D_1^P + \Delta D_2^P + \Delta D_3^P + \Delta D_4^P$ and $Z(Q \cup \{g\}) - Z(Q) = \Delta D_1^Q + \Delta D_2^Q + \Delta D_3^Q + \Delta D_4^Q$. We are therefore interested in the sign of $Z(P \cup \{g\}) - Z(P) - (Z(Q \cup \{g\}) - Z(Q))$.

We consider adding path $g \in \Omega \setminus Q$ to sets $P$ and $Q$. We calculate the path index, $\lambda_p$, calculated at each iteration of the greedy algorithm, with updated $F$, $d_s$, and $S'_p$.

$\lambda_p = \sum_{s \in S'_p} c_s d_s - \sum_{s \in S'_p} c_s (d_s - \hat{\beta}_p^s (h - F))_+$. We proceed with three cases.

**Case 1:** When $\lambda_g \leq \lambda_p \quad \forall p \in P$ where $f_p^P > 0$ then there is no use for path $g$: $f_g^{P \cup \{g\}} = 0$ and $f_g^{Q \cup \{g\}} = 0$. So $Z(P \cup \{g\}) - Z(P) = Z(Q \cup \{g\}) - Z(Q) = 0$.

**Case 2:** Consider the case when the constraint on $h$ doesn't bind, and when $\exists p \in P$ with $f_p^P > 0 : \lambda_g > \lambda_p$. In this case, when we consider $Z(P \cup \{g\})$, it is optimal to use path $g$.

If the constraint on $h$ is not binding, then the only reason to not use $f_g$ until an assignment category capacity is met is if a path that shares a capacity is more valuable. If that more

valuable path exits in $Q \setminus P$, then we might allocate more volume to path $g$ when adding it to set $P$, and $f^{\{}Q \cup \{g\}\}_g = f^{\{}P \cup \{g\}\}_g$. Consequently, $\Delta D_2^P \leq \Delta D_2^Q$. This also means that $\Delta D_1^P \leq \Delta D_1^Q$ and $\Delta D_3^P \leq \Delta D_3^Q$, as there is possibly less benefit to shifting to path $g$ when starting with set $Q$. The paths in $Q'_g$ that impact $s \in S^4$ are not impacted by any adjustment i nthe capacity limitation from using path $g$ since there isn't an overall binding volume capacity, and $\Delta D_4^P = \Delta D_4^Q = 0$. Consequently, $Z(P \cup \{g\}) - Z(P) \leq Z(Q \cup \{g\}) - Z(Q)$.

**Case 3:** Consider the case when the constraint on $h$ does bind, and when $\exists p \in P$ with $f_p > 0 : \lambda_g > \lambda_p$. In this case, when we consider $Z(P \cup \{g\})$, it is optimal to use path $g$.

If the overall volume capacity, $h$, is binding, then the same arguments for $S^1, S^2, S^3$ hold, but not there could be a different adjustment to $S^4$. Specifically, there could be a shift in volume from paths in $Q'_g$ to path $g$. But since the paths in $Q'_g$ impact $D_4$, the shift when adding path $g$ to set $Q$ would come from paths that are at least as valuable as those when adding path $g$ to set $P$. So $\Delta D_4^P \leq \Delta D_4^Q$. Therefore $Z(P \cup \{g\}) - Z(P) \leq Z(Q \cup \{g\}) - Z(Q)$.

□

**Proof of Proposition B.2.3**

*Proof.* $F(k) = \min_{P_k : P_k \subseteq \Omega \setminus P_A, |P_k| = k} Z(P_A \cup P_k)$. From Lemma B.2.1, we know that the greedy approach from Algorithm 4 is optimal. So for $F(k)$, $P_k$ consists of the $k^{th}$ largest index values from $\Omega \setminus P_A$ computed during the Algorithm. Similarly, for $F(k+1)$, $P_{k+1}$ consists of the $k+1^{st}$ largest index values from $\Omega \setminus P_A$. So $P_{k+1} = P_k \cup \{i\}$, where $i$ is the next selected path, and $P_{k+2} = P_{k+1} \cup \{j\}$, where $j$ is path selected after $i$.

From Proposition B.2.2, $Z$ is supermodular, so $Z(P_k \cup \{i\}) - Z(P_k) \leq Z(P_k \cup \{j\} \cup \{i\}) - Z(P_k \cup \{j\})$. So $F(k+1) - F(k) \leq F(k+2) - Z(P_k \cup \{j\})$. Since $Z(P_k \cup \{i\}) \leq Z(P_k \cup \{j\})$ based on optimally selecting path $i$ before path $j$, $F(k+1) - F(k) \leq F(k+2) - Z(P_k \cup \{j\}) \leq F(k+2) - Z(P_k \cup \{i\})$, and $F(k+1) - F(k) \leq F(k+2) - F(k+1)$.

□

**Algorithm 4:** Greedy Solution Approach to $\mathbf{MPRP}(\mathbf{P_A} \subseteq \mathbf{\Omega})$

**Result:** $f_p^*$ and $\mathcal{Z}^*_{\mathbf{MPRP}(\mathbf{P_A} \subseteq \mathbf{\Omega})}$

**Input :** $P_A, S, T, h, r_s, n_s, c_s, \beta_p^t, \delta_s^t(p)$

Define $\hat{\beta}_p^s = \sum_{t=1}^{T} \beta_p^t \delta_s^t(p) \quad \forall p \in P_A, \forall s \in S$

Define $S_p = \{s \in S : \exists t \text{ with } \delta_s^t(p) = 1\} \quad \forall p \in P_A$

Initialize $S'_p = S_p \quad \forall p \in P_A$

Initialize $F = 0$, $f_p^* = 0 \quad \forall p \in P_A$

Initialize $\lambda_p = \sum_{s \in S'_p} c_s d_s - \sum_{s \in S'_p} c_s (d_s - \hat{\beta}_p^s (h - F))_+ \quad \forall p \in P_A$

Initialize $K = P_A$

**while** $\exists \lambda_p > 0$ *and* $F < h$ *and* $K \neq \emptyset$ **do**

  $p = \underset{p \in K}{\text{argmax}}\ \lambda_p$

  $f_p^* = \min(h - F, \frac{r_i n_i - \sum_{k \in P_A} f_k^* \hat{\beta}_k^i}{\hat{\beta}_p^i} \quad \forall i \in S'_p, \frac{n_j - \sum_{k \in P_A} f_k^* \hat{\beta}_k^j}{\hat{\beta}_p^j} \quad \forall j \in S_p \setminus S'_p)$

  $F = \sum_{p \in P} f_p^*$

  **if** *minimized by* $i \in S'_p$ **then**

    $S'_p = S'_p \setminus \{i\} \quad \forall p : i \in S_p$

    $d_s = r_s n_s - \sum_{p \in P} f_p^* \hat{\beta}_p^s \quad \forall s \in S$

    $\lambda_p = \sum_{s \in S'_p} c_s d_s - \sum_{s \in S'_p} c_s (d_s - \hat{\beta}_p^s (h - F))_+ \quad \forall p \in P_A$

  **else if** *minimized by* $j \in S_p \setminus S'_p$ **then**

    $K = K \setminus \{p : j \in S_p\}$

**end**

# Appendix C

# Supplement for Chapter 4

## C.1 Formulation Variants and Weighted Sum Method Details

### C.1.1 Stand-alone Readiness Formulation

We can capture the readiness portion of **OAT** as a stand-alone formulation by removing the matching variables and constraints.

$\mathbf{CURRENT_{Stage1}}$ :

$$\min_{d,r,y} \quad \sum_{u \in U} d_u$$

$$s.t. \quad r_u = \tilde{r}_u + \frac{1}{N_u} \sum_{j \in J_u} y_j \qquad \forall u \in U \qquad \text{(C.1a)}$$

$$d_u \geq \gamma_u^- (\underline{g}_u - r_u) \qquad \forall u \in U \qquad \text{(C.1b)}$$

$$d_u \geq \gamma_u^+ (-\overline{g}_u + r_u) \qquad \forall u \in U \qquad \text{(C.1c)}$$

$$\sum_{j \in J} y_j = m \qquad \text{(C.1d)}$$

$$d_u \geq 0 \qquad \forall u \in U \qquad \text{(C.1e)}$$

$$y_j \in \{0,1\} \qquad \forall j \in J \qquad \text{(C.1f)}$$

In $\mathbf{CURRENT_{Stage1}}$, the primary decision variable is $y_j$, a binary selector for including job $j$ in the selected subset, $J'$. Constraint C.1a determines the projected readiness based on how many jobs in that unit are filled. Constraints C.1b and C.1c determine the deviation of the projected readiness from the unit's target readiness. Constraint C.1d ensures that enough jobs are selected to balance the market, where $m$ is the number of applicants.

### C.1.2 Min-cost Flow Formulation

To create a tractable approach, $\mathbf{OAT}_F$ re-casts **OAT** as a min-cost network flow with side constraints.

$\mathcal{G}$ : Graph representation with nodes $V$ and arcs $A$.

$b_v$ : The demand at each node $v$.

$u_{ij}$ : The capacity of arc $(i, j)$.

$c_{ij}$ : The cost of using arc $(i, j)$.

$\delta^+(i)$ : The set of nodes that $i$ connects to.

$\delta^-(i)$ : The set of nodes that connect to $i$.

Graph $\mathcal{G} = (V, A)$ is the base for the model.

**Algorithm 5:** Generate the Graph, $\mathcal{G} = (V, A)$

**Result:** $\mathcal{G} = (V, A)$

**Input :** $O$, $J$, $U$, $J_u \quad \forall u \in U$

Create nodes for every applicant, $o \in O$

Create nodes for every job, $j \in J$

Create two nodes, $\{\underline{u}, \overline{u}\}$ for every unit, $u \in U$

Create penalty-under node, $P_U$

Create penalty-over node, $P_O$

**for** $o \in O$ **do**

  **for** $j \in J$ **do**

    Create arc from $o$ to $j$

  **end**

**end**

**for** $j \in J$ **do**

  Create arc from $j$ to $\underline{u} : j \in J_u$

**end**

**for** $u \in U$ **do**

  Create arc from $\underline{u}$ to $\overline{u}$

  Create arc from $P_U$ to $\underline{u}$

  Create arc from $P_U$ to $\overline{u}$

  Create arc from $\overline{u}$ to $P_O$

**end**

### Capacity, Demand, and Cost

The capacity of each arc connecting job $j$ to unit $u$ is $u_{j\underline{u}} = 1$. Other arcs are uncapacitated.

We want to ensure that every applicant matches, so the demand at each applicant node, $o$, is $b_o = -1$. There is no demand at any job or penalty node. The demand at each unit's first node is the number of applicants needed to bring that unit to its lower goal, $b_{\underline{u}} = \lceil N_u(\underline{g}_u - r_u)^+ \rfloor$. The demand at each unit's second node is the number of applicants needed

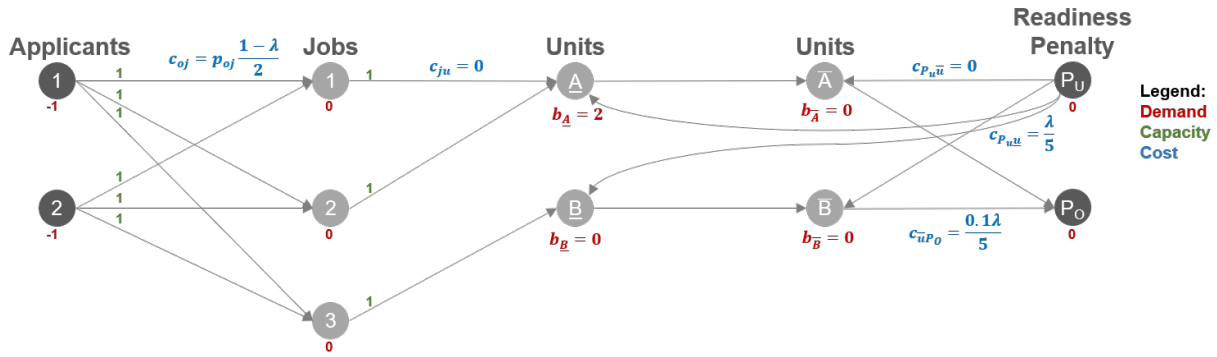


Figure C-1: An example graph for a 2 applicant, 3 job, 2 unit network. Demand is in red, capacities are in green, and cost is in blue.

to bring that unit to its upper goal from its lower goal, $b_{\overline{u}} = \lceil N_u((\overline{g}_u - r_u)^+ - (\underline{g}_u - r_u)^+) \rfloor$.

The cost of an arc connecting an applicant to a job is associated with applicant preference, and $c_{oj} = p_{oj}\frac{1-\lambda}{m}$. The cost of an arc connecting a job to a unit's lower node, $\underline{u}$, is 0. The cost of an arc connecting a unit's lower and upper nodes is 0. The cost on the arcs connected to the penalty nodes is associated with unit readiness. The cost of an arc connecting $P_U$ to a unit's lower node, $\underline{u}$, is $c_{P_U\underline{u}} = \frac{\lambda\gamma_u^-}{N_u}$. The cost of an arc connection $P_U$ to a unit's upper node, $\overline{u}$, is 0. The cost of an arc connecting unit's upper node $\overline{u}$ to $P_O$ is $c_{\overline{u}P_O} = \frac{\lambda\gamma_u^+}{N_u}$.

### Formulation

**OAT**$_F$ formulates the problem as a min-cost capacitated network flow with the addition of a stability constraint.

$\mathbf{OAT}_F$ :

$$\min_{x} \quad \sum_{(i,j)\in A} c_{ij}x_{ij}$$

$$s.t. \quad \sum_{j\in\delta^-(i)} x_{ji} - \sum_{j\in\delta^+(i)} x_{ij} = b_i \qquad \forall i \in V \setminus \{P_u, P_o\} \qquad \text{(C.2a)}$$

$$x_{ij} \leq u_{ij} \qquad \forall (i,j) \in A \qquad \text{(C.2b)}$$

$$x_{oj} + \sum_{k:o\succ_j k} x_{kj} + \sum_{k:j\succ_o k} x_{ok} \leq 1 \qquad \forall (o,j) \in A : o \in O, j \in J \qquad \text{(C.2c)}$$

$$x_{ij} \in \mathcal{Z}^{0+} \qquad \forall (i,j) \in A \qquad \text{(C.2d)}$$

In $\mathbf{OAT}_F$ $x_{ij}$ is the flow on each arc $(i,j) \in A$. Constraint C.2a ensures that inbound and outbound flows balance. Constraint C.2b enforces the capacity of each edge. Constraint C.2c enforces the stability of the matching for the applicants and the selected jobs; more details below.

The cost function is a combination of the average applicant preference received, and the total readiness deviation for units. When the flow from applicants meets a unit's demand exactly, there is no flow to and from a penalty node with that unit, and consequently no readiness cost. If the flow from applicants falls between the lower and upper goals, then it satisfies the lower demand exactly, and there is no cost from the penalty node to satisfy the remainder of the upper demand. When the flow from applicants is short of a unit's demand, then the formulation results in flow from penalty-under to the unit's lower demand, at a readiness cost, and flow to the upper demand at no cost. When the flow from applicants is over a unit's demand, then the formulation results in flow from a unit's upper node to penalty-over, at a readiness cost.

### C.1.3 Algorithm Details for the Weighted-Sum Approach

The algorithm for Section 4.4, weighted-sum portion is detailed below.

**Algorithm 6:** Generate a portion, $\tilde{Z}_{NS}$, of the non-dominated, supported frontier, $Z_{NS}$, where each point on the frontier has two components, $(Q, S)$.

**Result:** $\tilde{Z}_{NS}$

**Input :** $\mathbf{OAT}_{F-PR}, \epsilon$

*Notation: superscript for the scaling parameter, $\lambda$, used to find the point; subscript for the two points used in the computation to find a new point.*

$z^0 = (Q^0, S^0)$ from solution to $\mathbf{OAT}_{\lambda-F-PR}$ with $\lambda = 0$

$z^1 = (Q^1, S^1)$ from solution to $\mathbf{OAT}_{\lambda-F-PR}$ with $\lambda = 1$

$\tilde{Z}_{NS} = \{(Q^0, S^0), (Q^1, S^1)\}$

Set $z_1 = (Q_1, S_1) = (Q^0, S^0)$

Set $z_2 = (Q_2, S_2) = (Q^1, S^1)$

$Z_{out} = \emptyset$

$\Delta = \epsilon + 100$

**while** $\Delta > \epsilon$ **do**

$\tilde{\lambda} = \frac{S_1 - S_2}{Q_1 - Q_2 + S_1 - S_2}$

Solve $\mathbf{OAT}_{\lambda-F-PR}$ with $\tilde{\lambda}$

**if** $(Q^{\tilde{\lambda}}, S^{\tilde{\lambda}}) \in \tilde{Z}_{NS}$ **then**

$Z_{out} = Z_{out} \cup (z_1, z_2)$ *(no additional points between this pair of points)*

**else**

$\tilde{Z}_{NS} = \tilde{Z}_{NS} \cup \{(Q^{\tilde{\lambda}}, S^{\tilde{\lambda}})\}$ *(add point to frontier)*

**end**

$$\underset{z^a, z^b \in \tilde{Z}_{NS}: (z^a, z^b) \notin Z_{out}, a,b \text{ adjacent}}{\operatorname{argmax}} \sqrt{(S^a - S^b)^2 + (Q^a - Q^b)^2}$$

**if** *solution is infeasible* **then**

$\Delta = -1$

**else**

$\Delta = \sqrt{(S^a - S^b)^2 + (Q^a - Q^b)^2}$

$z_1 = z^a$; $z_2 = z^b$

**end**

**end**

## C.2 Proofs

### Proof of Proposition 4.4.1

*Proof.* We can construct a solution in the following manner. Find any feasible matching, $\mu$, between applicants and jobs. This stable matching is guaranteed to exist with strict preference list inputs [Roth], and we can find it with a deferred acceptance algorithm. For every matched applicant and job, set $z_{oj} = 1$, otherwise $z_{oj} = 0$. This satisfies constraints 4.1f, 4.1g, and 4.1j. Additionally it satisfies constraint 4.1e and determines the value of $y_j$ for each matched or unmatched job. This ensures the integrality of $y$, constraint 4.1i, and that $m$ jobs are selected, constraint 4.1d. The known values of $y$ then determine the value of $r$ with constraint 4.1a, and the value of $d$ with constraints 4.1b, 4.1c, and 4.1h.

Consider a solution to **OAT** that is not stable, where, without loss of generality, applicant 1 prefers job 1 to his match, $\mu(o_1) = \hat{j}$, and his match prefers someone else to applicant 1. In this case, for $o_1$ and his match $\hat{j}$, constraint 4.1g would have a first term of $z_{o_1\hat{j}} = 1$. Because $\exists k : k \succ_{\hat{j}} o_1$, the second term is at least 1, and because $j_1 \succ_{o_1} \hat{j}$ the third term is at least 1. But this violates the constraint and is not feasible. The solution cannot be unstable. □

### Proof of Proposition 4.4.2

*Proof.* Consider an optimal solution to $\mathbf{OAT}_\lambda$ of $\{r^*, d^*, y^*, z^*\}$ with objective value of $\mathcal{Z}^*_{\mathbf{OAT}_\lambda} = \lambda \sum_{u \in U} d^*_u + (1-\lambda)\frac{1}{m}\sum_{o \in O}\sum_{j \in J} p_{oj} z^*_{oj}$. We set $x_{oj} = z^*_{oj}\ \forall o \in O, j \in J$. We set $x_{j\underline{u}} = y_j\ \forall j \in J$.

For every $u \in U$: if $r^*_u \leq \underline{g}_u$ then $x_{P_U\underline{u}} = (\underline{g}_u - r^*_u)N_u$, $x_{\underline{u}\overline{u}} = 0$, $x_{P_U\overline{u}} = (\overline{g}_u - \underline{g}_u)N_u$, and $x_{\overline{u}P_O} = 0$; else if $\underline{g}_u < r^*_u \leq \overline{g}_u$ then $x_{P_U\underline{u}} = 0$, $x_{\underline{u}\overline{u}} = (r^*_u - \underline{g}_u)N_u$, $x_{P_U\overline{u}} = (\overline{g}_u - r^*_u)N_u$, and $x_{\overline{u}P_O} = 0$; otherwise, as $r^*_u > \overline{g}_u$ then $x_{P_U\underline{u}} = 0$, $x_{\underline{u}\overline{u}} = (r^*_u - \underline{g}_u)N_u$, $x_{P_U\overline{u}} = 0$, and $x_{\overline{u}P_O} = (r^*_u - \overline{g}_u)N_u$. Then:

$$
\begin{aligned}
\mathcal{Z}_{\mathbf{OAT}_F} &= \sum_{(i,j)\in A} c_{ij}x_{ij} \\
&= \sum_{o\in O}\sum_{j\in J} c_{oj}x_{oj} + \sum_{u\in U} c_{P_U\underline{u}}x_{P_U\underline{u}} + \sum_{u\in U} c_{\overline{u}P_O}x_{\overline{u}P_O} \\
&= \sum_{o\in O}\sum_{j\in J} p_{oj}\frac{1-\lambda}{m}x_{oj} + \sum_{u\in U}\frac{\lambda\gamma_u^-}{N_u}x_{P_U\underline{u}} + \sum_{u\in U}\frac{\lambda\gamma_u^+}{N_u}x_{\overline{u}P_O} \\
&= \sum_{o\in O}\sum_{j\in J} p_{oj}\frac{1-\lambda}{m}x_{oj} + \sum_{u\in U}(\frac{\lambda\gamma_u^-}{N_u}((\underline{g}_u - r_u^*)N_u)^+ + \frac{\lambda\gamma_u^+}{N_u}((r_u^* - \overline{g}_u)N_u)^+) \\
&= (1-\lambda)\frac{1}{m}\sum_{o\in O}\sum_{j\in J} p_{oj}x_{oj} + \lambda\sum_{u\in U}(\gamma_u^-(\underline{g}_u - r_u^*)^+ + \gamma_u^+(r_u^* - \overline{g}_u)^+) \\
&= (1-\lambda)\frac{1}{m}\sum_{o\in O}\sum_{j\in J} p_{oj}z_{oj}^* + \lambda\sum_{u\in U} d_u^* \\
&= \mathcal{Z}^*_{\mathbf{OAT}}
\end{aligned}
$$

Consider an optimal solution to $\mathbf{OAT}_F$ of $x^*$ with objective value of $\mathcal{Z}^*_{\mathbf{OAT_F}} = \sum_{(i,j)\in A} c_{ij}x_{ij}^*$. We set $z_{oj} = x_{oj}^*\ \forall o \in O, j \in J$. We set $y_j = x_{j\underline{u}}^*\ \forall j \in J$.

For every $u \in U$ there are three possible configurations of penalty flows. If $x_{P_U\underline{u}}^* > 0$ then we know $x_{\overline{u}P_O}^* = 0$, otherwise the objective would increase. Similarly, if $x_{\overline{u}P_O}^* > 0$ then $x_{P_U\underline{u}}^* = 0$. So, if $x_{P_U\underline{u}}^* > 0$ we set $r_u = \underline{g}_u - \frac{x_{P_U\underline{u}}^*}{N_u}$ and $d_u = \gamma_u^-(\underline{g}_u - r_u)$. If $x_{\overline{u}P_O}^* > 0$ we set $r_u = \overline{g}_u + \frac{x_{\overline{u}P_O}^*}{N_u}$ and $d_u = \gamma_u^+(-\overline{g}_u + r_u)$. If both $x_{P_U\underline{u}}^* = 0$ and $x_{\overline{u}P_O}^* = 0$, then we set $r_u = \underline{g}_u$ and $d_u = 0$. Then:

$$
\begin{aligned}
\mathcal{Z}_{\mathbf{OAT}} &= (1-\lambda)\frac{1}{m}\sum_{o\in O}\sum_{j\in J} p_{oj} z_{oj} + \lambda \sum_{u\in U} d_u \\
&= \sum_{o\in O}\sum_{j\in J}(1-\lambda)\frac{1}{m} p_{oj} x^*_{oj} + \lambda \sum_{u\in U}(\gamma^-_u(\underline{g}_u - r_u))^+ + \lambda \sum_{u\in U}(\gamma^+_u(-\overline{g}_u + r_u))^+ \\
&= \sum_{o\in O}\sum_{j\in J}(1-\lambda)\frac{1}{m} p_{oj} x^*_{oj} + \lambda \sum_{u\in U}\gamma^-_u \frac{x^*_{P_U\underline{u}}}{N_u} + \lambda \sum_{u\in U}\gamma^+_u \frac{x^*_{\overline{u}P_O}}{N_u} \\
&= \sum_{o\in O}\sum_{j\in J}(1-\lambda)\frac{1}{m} p_{oj} x^*_{oj} + \sum_{u\in U}\frac{\lambda\gamma^-_u}{N_u} x^*_{P_U\underline{u}} + \sum_{u\in U}\frac{\lambda\gamma^+_u}{N_u} x^*_{\overline{u}P_O} \\
&= \sum_{o\in O}\sum_{j\in J} c_{oj} x^*_{oj} + \sum_{u\in U} c_{P_U\underline{u}} x^*_{P_U\underline{u}} + \sum_{u\in U} c_{\overline{u}P_O} x^*_{\overline{u}P_O} \\
&= \mathcal{Z}^*_{\mathbf{OAT}_F}
\end{aligned}
$$

□

**Proof of Proposition 4.4.3**

*Proof.* Consider $\mathbf{OAT}_{F-PR}$. Select any subset of jobs, $J' \subseteq J : |J| = |O|$ and set $x_{j\underline{u}} = 1\ \forall j \in J'$. With a fixed $J'$, the problem decomposes into two parts. To the left of the job nodes, we now set $x_{oj} = 0\ \forall o \in O, j \notin J'$. We now have the classic stable matching problem with equal-sized sets $O$ and $J'$. An applicant-optimal solution always exists and is integral ([63]). To the right of the job nodes, we now have a min cost network flow with a demand of $b = -1$ at every node $j \in J'$. We can now solve for the min-cost flow with integral inputs to determine the flow on the penalty arcs, and get an integral optimal solution (see for example, Ahuja et al., 1993). This is true for any subset of jobs $J' \subseteq J : |J| = |O|$.

Since the demand at each applicant node, $o \in O$, is -1, we know that total flow out of set $O$ and into all of the nodes in set $J$ is $m$. Since the demand for all nodes in $J$ is 0, with flow balance, we know that any feasible solution must always have a total flow out of set $J$ of $m$ and $\sum_{j\in J} x_{j\underline{u}} = m$. Every feasible solution will include a subset of jobs $J' \subseteq J : |J| = |O|$,

and we can relax the integrality constraint in for all arcs $(i, j) \in \mathcal{A} : j \notin J$, and the optimal solution to $\mathbf{OAT}_{\lambda-F-PR}$ is integral. □

**Proof of Proposition 4.4.4**

*Proof.* Each iteration of Algorithm 6 takes two known objective-space points as inputs, and it finds one of those points when solving the ensuing **OAT** then those points are not considered in subsequent iterations (storing these pairs of points in $Z_{OUT}$). So, as long as there are a finite number of possible points in the objective space, the algorithm will terminate. There are a finite number subsets $J' \subset J$ and there are a finite number of matchings for each of those subsets. Each selected subset maps to a single readiness value, and each matching maps to a single satisfaction value. There are therefore a finite number of possible objective space points.

[Known result for the weighted-sum method; see [25]] When $\epsilon = 0$ Algorithm 6 will continue to search between pairs of adjacent points until there are no more points to find. Consider a pair of adjacent points, $(Q_1, S_1)$ and $(Q_2, S_2)$. Consider a third point, $(Q_3, S_3)$ such that $Q_1 < Q_3 < Q_2$. When $S_3$ is below the line connecting points 1 and 2, the minimization of the convex combination will find one of the two end points first. When $S_3$ is above the line connecting points 1 and 2, then the minimization will find it, and it is part of the extreme supported set of non-dominated points. The algorithm only terminates once it has found all of such points. □

**Proof of Proposition 4.4.5**

*Proof.* This is true because of the criteria in Definition 4.4.1. Consider a starting solution with selected job subset $J'$ and stable matching $\mu$. We only execute a one-swap if the entering job, $j_b \in J \setminus J'$ is more preferred by an applicant than his current match, $\mu(\tilde{o}) = j_a$. The remainder of the matching remains the same, and thus satisfaction is improved. We can only execute the one-swap if the entering job, $j_b$ prefers $\tilde{o}$ to any other applicant who would

prefer $j_b$ of their current match. This ensures the stability of the solution since we started with a stable solution and only add $j_b$ in a way that it will not prefer any other applicants to its match. □

**Proof of Proposition 4.4.6**

*Proof.* Consider a one-swap chain initiated by $j_e \in J \setminus J'_0$, with the applicant-optimal matching $\mu_0$, to job subset $J'_0$, and subsequent matchings and subsets indexed by $i$.

To describe our logic, we consider subsets of the matching where instead of all $m$ applicants, we look at the match just for $k$ of the applicants, $1 < k \leq m$. Consider just two applicants, $o_1$ and $o_2$, and their respective matches. In the original matching, $\mu_0(o_1) = j_a$ and $\mu_0(o_2) = j_b$. If in any subsequent matching, $i$, during the one-swap chain we look at their respective matches again, we could see a different pair of jobs if $o_1$ and/or $o_2$ have executed one-swaps. If we see the same job subset for these two applicants, $\{j_a, j_b\}$ then it must produce the same matching, $\mu_i(o_1) = j_a$ and $\mu_i(o_2) = j_b$. One swaps must be strictly applicant satisfaction improving, and since we started with the applicant-optimal matching, we cannot return to the same subset of jobs and applicants, but have a different matching.

This same logic is why $j_e$ will always remain in any subsequent subsets in the chain, $J'_i$, since if it departs, we would have a subset equal to the original subset. But since the original matching was applicant-optimal and each subset change had a matching $\mu_i$ that improved satisfaction for applicants, this would not be possible, and $j_e$ must remain future subsets, $J'_i$.

We can apply this logic to larger subsets of applicants. If we consider all of the applicants except the one now matched to $j_e$, there are $m$ possible job subsets of size $m-1$ that could be matched to these $m-1$ applicants. But any repeat of the selected subset, with a different matching, would not be strictly improving if we started with the applicant-optimal subset originally. Hence, there can be at most $m$ one-swaps.

□

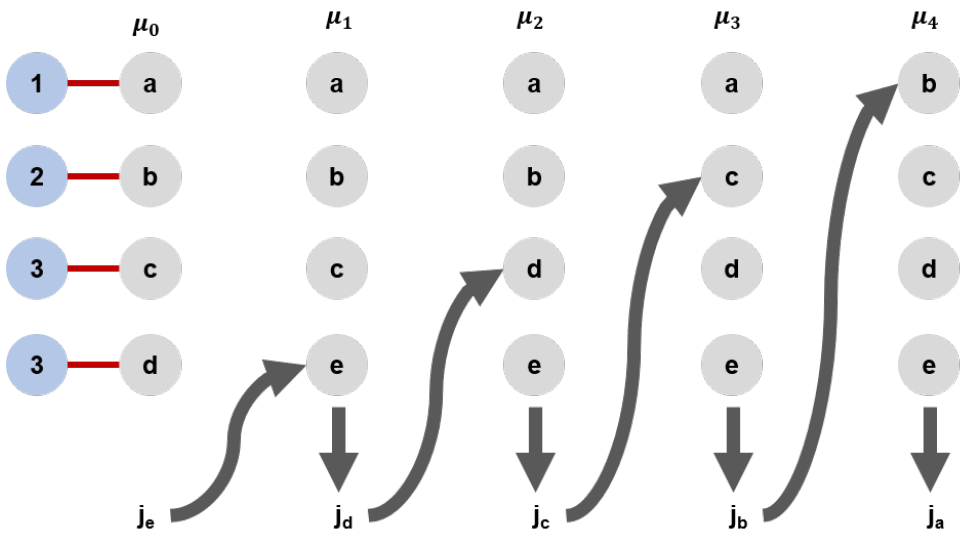


Figure C-2: A 4 applicant, 5 job example where $j_e$ is the initiating job in a one-swap chain. After $m$ (4) one-swaps, job $j_a$ exits the selected subset. For job $j_a$ to re-enter, we would get a subset of jobs equal to one of the 4 already encountered after the first one-swap. But since we started with an applicant-optimal one-swap, and each one-swap is applicant improving, this is not possible. Consider $j_a$ attempting to re-enter in a fifth one-swap for applicant 2, replacing $j_c$. If that was the case, we would be in our two job example showing this is not possible when we focus on a subset of the matching.

### Proof of Proposition 4.4.7

*Proof.* The one-swap chain algorithm (1) finds an initial solution, (2) executes a number of one-swap chains that depends on the selected depth, $\tau$, and the sizes of the applicant and job input sets, where (3) each one-swap chain consists of a number of one-swaps, and (4) each one-swap requires a known series of operations. We consider each of these portions in turn.

(1) We can initiate the one-swap chain algorithm by solving the model of the current process by solving **OAT** with $\lambda = 1$. We use the job subset portion of the solution, and then apply the deferred acceptance algorithm to that job subset and the applicants. The network flow variant of **OAT** that is just the readiness portion is formulation C.1 and has constraints that are linear in the number of jobs, and a number of variables that is quadratic in the number of jobs. The complexity using a polynomial-time linear programming algorithm is at least

as good as $O(n^3)$ ([62]). The deferred acceptance algorithm, given a subset of the jobs equal in size to the number of applicants, has complexity $O(m^2)$ ([29]).

(2) At each step in the algorithm from $1...\tau$, the algorithm computes at most $n-m$ one-swap chains: one for each job not in the selected subset. So, the algorithm then computes at most $\tau(n-m)$ one-swap chains.

(3) From Proposition 4.4.6, we know a one-swap chain has at most $m$ one-swaps when starting from an optimal solution. There is no guarantee that each one-swap chain results in an applicant-optimal matching given the new subset, and we could at the beginning of each step from $2...\tau$ apply the deferred acceptance algorithm in $O(m^2)$ to ensure it is applicant optimal.

(4) Each one-swap in the chain requires one pass through all of the applicant preferences, and then a single computation based on the result of that pass. So a one-swap takes $O(m)$ time.

This means that given an initial solution, the algorithm computes all of the required one-swap chains in $\tau(n-m)m^2$ time. There are at most $\eta = \tau(n-m)m$ different solutions found during the one-swap chains. From the set of objective-space points from these solutions, we compute the Pareto frontier, which can be done in $O(\eta log(\eta))$ time ([43]).

□

# Bibliography


[1] 10 U.S.C. Readiness Reporting System. § 117, 2018.

[2] Abderrahmane Abbou and Viliam Makis. Group Maintenance: A Restless Bandits Approach. *INFORMS Journal on Computing*, 31(4):719–731, 2019.

[3] Oussama Mazari Abdessameud, Filip Van Utterbeeck, Johan Van Kerckhoven, and Marie Anne Guerry. Military manpower planning towards simultaneous optimization of statutory and competence logics using population based approaches. *ICORES 2018 - Proceedings of the 7th International Conference on Operations Research and Enterprise Systems*, 2018-Janua:178–185, 2018.

[4] Atila Abdulkadiro and Tayfun Sönmez. Matching Markets: Theory and Practice. *Advances in Economics and Econometrics*, 1:3–47, 2013.

[5] Kolos Csaba Ágoston, Péter Biró, Endre Kováts, and Zsuzsanna Jank. College admissions with ties and common quotas: Integer programming approach. *European Journal of Operational Research*, 2021.

[6] Kolos Csaba Ágoston, Péter Biró, and Iain McBride. Integer programming methods for special college admissions problems. *Journal of Combinatorial Optimization*, 32(4):1371–1399, 2016.

[7] Kolos Csaba Ágoston, Péter Biró, and Richárd Szántó. Stable project allocation under distributional constraints. *Operations Research Perspectives*, 5:59–68, 2018.

[8] Haris Aziz, Unsw Sydney, and Data Csiro. Matching Market Design with Constraints. 2021.

[9] Jackson Barnett. DoD JAIC Funding, 2021.

[10] Peter B Baumgarten. *Optimization of United States Marine Corps Officer Career Path Selection*. PhD thesis, Naval Postgraduate School, 2000.

[11] Dimitris Bertsimas and Jose Nino-Mora. Restless Bandits, Linear Programming Relaxations, and a Primal-Dual Index Heuristic. *Operations Research*, 48(1):80–90, 2000.

[12] Jeff Bezanson, Stefan Karpinski, Viral B. Shah, and Alan Edelman. Julia: A Fast Dynamic Language for Technical Computing. pages 1–27, 2012.

[13] Lin Boliang, Wu Jianping, Lin Ruixi, Wang Jiaxi, Wang Hui, and Zhang Xuhui. Optimization of high-level preventive maintenance scheduling for high- speed trains. *Reliability Engineering and System Safety*, 183(April 2018):261–275, 2019.

[14] K. Bouvard and V Cocquempot. Condition-based dynamic maintenance operations planning and grouping . Application to commercial heavy vehicles. *Reliability Engineering and System Safety*, 96:601–610, 2011.

[15] Fatih Camci. System Maintenance Scheduling With Prognostics Information Using Genetic Algorithm. *IEEE Transactions on Reliability*, 58(3):539–552, 2009.

[16] Philip Cho, Vivek Farias, John Kessler, Retsef Levi, Thomas Magnanti, and Eric Zarybnisky. Maintenance and Flight Scheduling of Low Observable Aircraft. 2014.

[17] Philip Y Cho. *Optimal scheduling of fighter aircraft maintenance*. PhD thesis, 2011.

[18] Matthew F. Dabkowski, Samuel H. Huddleston, Paul Kucik, and David Lyle. Shaping senior leader officer talent: How personnel management decisions and attrition impact the flow of Army officer talent throughout the officer career model. *Proceedings - Winter Simulation Conference*, pages 1407–1418, 2010.

[19] Bram de Jonge and Philip A. Scarf. A review on maintenance optimization. *European Journal of Operational Research*, 285(3):805–824, 2020.

[20] Department Defense. Mission Initiatives, 2022.

[21] Rommert Dekker and Frank van der Duyn Schouten. A Review of Multi-Component Maintenance Models with Economic Dependence. *Mathematical Methods of Operations Research*, pages 411–435, 1997.

[22] Iain Dunning, Joey Huchette, and Miles Lubin. JuMP: A modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017.

[23] Federico Echenique and Vijay V Vazirani. Two-Sided Markets: Stable Matching (DRAFT). *Forthcoming in Online and Matching-Based Market Design*, 2021.

[24] Lars Ehlers, Isa E. Hafalir, M. Bumin Yenmez, and Muhammed A. Yildirim. School choice with controlled choice constraints: Hard bounds versus soft bounds. *Journal of Economic Theory*, 153:648–683, 2014.

[25] Matthias Ehrgott and Xavier Gandibleux. A survey and annotated bibliography of multiobjective combinatorial optimization. *OR Spektrum*, 22(4):425–460, 2000.

[26] Aytek Erdil and Haluk Ergin. What's the matter with tie-breaking? Improving efficiency in school choice. *American Economic Review*, 98(3):669–689, 2008.

[27] Aytek Erdil and Haluk Ergin. Two-sided matching with indifferences. *Journal of Economic Theory*, 171:268–292, 2017.

[28] Daniel Fragiadakis, Atsushi Iwasaki, Peter Troyan, Suguru Ueda, and Makoto Yokoo. Strategyproof matching with minimum quotas. *ACM Transactions on Economics and Computation*, 4(1), 2015.

[29] D. Gale and L. S. Shapley. College admissions and the stability of marriage. *American Mathematical Monthly*, 69(1), 1962.

[30] Saul I. Gass. Military manpower planning models. *Computers and Operations Research*, 18(1):65–73, 1991.

[31] John Gittins, Kevin Glazebrook, and Richard Weber. *Multi-Armed Bandit Allocation Indices: 2nd Edition*. 2011.

[32] Paul L. Goethals and Natalie M. Scala. Eliminating the weakest link approach to army unit readiness. *Decision Analysis*, 15(2):110–130, 2018.

[33] Matthew S Goldberg, Suzanne M Holroyd, and Gregory G Hildebrandt. Measuring Military Readiness and Sustainability. *RAND Corporation*, 1991.

[34] Kyle Greenberg and Parag A Pathak. Mechanism Design meets Priority Design : Redesigning the US Army's Branching Process. *NBER Working Papers*, (June), 2021.

[35] Andrew O. Hall and Michael C. Fu. Optimal Army officer force profiles. *Optimization Letters*, 9(8):1769–1785, 2015.

[36] Adriaan Van Horenbeek and Liliane Pintelon. A dynamic predictive maintenance policy for complex multi-component systems. *Reliability Engineering and System Safety*, 120:39–50, 2013.

[37] Mark E.T. Horn, Tarek Elgindy, and Antonio Gomez-Iglesias. Strategic workforce planning for the Australian Defence Force. *Journal of the Operational Research Society*, 67(4):664–675, 2016.

[38] Yuichiro Kamada and Fuhito Kojima. Efficient matching under distributional constraints: Theory and applications. *American Economic Review*, 105(1):67–99, 2015.

[39] J. Kays, W. Carlton, M Lee, and W. Ratliff. Analysis of Operational Readiness Rates. *United States Military Academy, Operations Research Center*, (July), 1998.

[40] Michael Jong Kim, Rui Jiang, Viliam Makis, and Chi Guhn Lee. Optimal Bayesian fault prediction scheme for a partially observable system subject to random failure. *European Journal of Operational Research*, 214(2):331–339, 2011.

[41] Michael Jong Kim and Viliam Makis. Joint optimization of sampling and control of partially observable failing systems. *Operations Research*, 61(3):777–790, 2013.

[42] George Kozanidis. A multiobjective model for maximizing fleet availability under the presence of flight and maintenance requirements. *Journal of Advanced Transportation*, 43(2):155–182, 2009.

[43] H. T. Kung, F. Luccio, and F. P. Preparata. On Finding the Maxima of a Set of Vectors. *Journal of the ACM (JACM)*, 22(4):469–476, 1975.

[44] Retsef Levi, Thomas Magnanti, Jack Muckstadt, Danny Segev, and Eric Zarybnisky. Maintenance Scheduling for Modular Systems : Modeling and Algorithms. *Naval Research Logistics*, 2014.

[45] William S. Lovejoy. Some Monotonicity Results for Partially Observed Markov Decision Processes. *Operations Research*, 35(5):736–743, 1987.

[46] Marco E. Lübbecke and Jacques Desrosiers. Selected topics in column generation. *Operations Research*, 53(6):1007–1023, 2005.

[47] L. M. Maillart, T. G. Yeung, and Z. Gozde Icten. Selecting test sensitivity and specificity parameters to optimally maintain a degrading system. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 225(2):131–139, 2011.

[48] David Manlove. *Algorithmics of Matching Under Preferences*. World Scientific Publishing Inc., Singapore, 2013.

[49] David O Marlow and Robert F Dell. Optimal short-term military aircraft fleet planning. *Journal of Applied Operational Research*, 9(1):38–53, 2017.

[50] Mark A Milley. 39th Chief of Staff of the Army Initial Message to the Army. page 1, 2015.

[51] Anthony Przybylski, Xavier Gandibleux, and Matthias Ehrgott. Two phase algorithms for the bi-objective assignment problem. *European Journal of Operational Research*, 185(2):509–533, 2008.

[52] John F Raffensperger and Linus E Schrage. Paradigm Measuring for Military Readiness. *Military Operations Research*, 3(5):21–34, 1997.

[53] Yongyi Ran, Xin Zhou, Pengfeng Lin, Yonggang Wen, and Ruilong Deng. A Survey of Predictive Maintenance: Systems, Purposes and Approaches. *IEEE Communications Surveys & Tutorials*, pages 1–36, 2019.

[54] Alvin E. Roth and Elliott Peranson. The redesign of the matching market for american physicians: Some engineering aspects of economic design. *American Economic Review*, 89(4):748–780, 1999.

[55] Alvin E Roth, Uriel G Rothblum, and John H VandeVate. Stable Matchings, Optimal Assignments, and Linear Programming. *INFORMS*, 18(4):803–828, 1993.

[56] Natalie Scala and James Howard. *Handbook of Military and Defense Operations Research*. CRC Press.

[57] United States Government. National Defense Strategy. page 23, 2018.

[58] U.S. Army. Army Unit Status Reporting. *AR: 220-1*, (April), 2010.

[59] U.S. Army. Army Strategic and Operational Readiness. *AR: 525-30*, (October), 2011.

[60] U.S. Army. *DA-PAM 600-3: Officer Professional Development and Career Management*. Number April. 2019.

[61] U.S. Army G-3/5/7. Army Readiness Guidance. *Stand To*, pages 1–3, 2016.

[62] Pravin M. Vaidya. Algorithm for Linear Programming Which Requires O(((m plus n)n^2 plus (m plus n)^1.5n)L) Arithmetic Operations. *Conference Proceedings of the Annual ACM Symposium on Theory of Computing*, 47:29–38, 1987.

[63] John H. Vande Vate. Linear programming brings marital bliss. *Operations Research Letters*, 8(3):147–153, 1989.

[64] Vedat Verter and Bahar Y. Kara. A path-based approach for hazmat transport network design. *Management Science*, 54(1):29–40, 2008.

[65] P. Whittle. Restless bandits: activity allocation in a changing world. *Journal of Applied Probability*, 25(A):287–298, 1988.

[66] Fuhui Wu, Qingbo Wu, and Yusong Tan. Workflow scheduling in cloud: a survey. *Journal of Supercomputing*, 71(9):3373–3418, 2015.

[67] Mark Zais and Dan Zhang. A Markov chain model of military personnel dynamics. *International Journal of Production Research*, 54(6):1863–1885, 2016.