Coordinated Planning of Air and Space Assets: An Optimization and Learning Based Approach

by

Eric John Robinson

B.S. Mathematics, United States Air Force Academy (2011)

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

Master of Science in Operations Research

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2013

Author .....
Sloan School of Management
May 17, 2013

Approved by .....
Mark Abramson
Member of the Technical Staff
The Charles Stark Draper Laboratory
Technical Supervisor

Certified by .....
Hamsa Balakrishnan
Associate Professor of Aeronautics and Astronautics
Thesis Supervisor

Accepted by .....
Professor Dimitris J. Bertsimas
Boeing Professor of Operations Research
Co-director, Operations Research Center

This thesis was prepared at The Charles Stark Draper Laboratory, Inc. under Grant Number NNX12H81G sponsored by the National Aeronautics and Space Administration Earth Science Technology Office Advanced Information Systems Technology-11 program.

Coordinated Planning of Air and Space Assets: An Optimization and Learning Based Approach

by

Eric John Robinson

Submitted to the Sloan School of Management
on May 17, 2013, in partial fulfillment of the
requirements for the degree of
Master of Science in Operations Research

Abstract

There are many organizations that use unmanned assets, such as satellites or drones, to collect information. This may include taking pictures of the ground, gathering infrared photos, taking atmospheric pressure measurements, or any conceivable form of data collection. Often these separate organizations have overlapping collection interests or flight plans that are sending sensors into similar regions. However, they tend to be controlled by separate planning systems which operate on asynchronous scheduling cycles. We present a method for coordinating various collection tasks between the planning systems in order to vastly increase the utility that can be gained from these assets. This method focuses on allocation of collection requests to scheduling systems rather than complete centralized planning over the entire system so that the current planning infrastructure can be maintained without changing any aspects of the schedulers. We expand on previous work in this area by inclusion of a learning method to capture information about the uncertainty pertaining to the completion of collection tasks, and subsequently utilize this information in a mathematical programming method for resource allocation. An analysis of results and improvements as compared to current operations is presented at the end.

Technical Supervisor: Mark Abramson
Title: Member of the Technical Staff
The Charles Stark Draper Laboratory

Thesis Supervisor: Hamsa Balakrishnan
Title: Associate Professor of Aeronautics and Astronautics

Acknowledgments

All of the work completed here would not have been possible without the love and support that I received during my time at MIT. I want to thank Dr. Steve Kolitz and Mark Abramson at Draper Laboratory for giving me the opportunity to come work with them on such interesting research. The extra time, effort, and care that they put into giving me feedback on my work was a blessing throughout my time at MIT. The professional development that I gained by working with them will stay with me throughout my life.

My thesis advisor at MIT, Professor Hamsa Balakrishnan, has been equally as important in helping me to comprehend and solve an abstract research problem. Her insights during both years led me down many paths that I would not have even thought to explore, ultimately pushing me into a final working model. I am incredibly grateful that she took me in as an extra student when she already had a full schedule.

To all of my friends at the MIT Operations Research Center, thank you for a wonderful two years. John and Will, going rock climbing with both of you was a great time. Allison and Fernanda, thanks so much for taking care of my puppy when Kimber and I were out of town. I am going to miss all of my friends at the ORC, but am very grateful to have spent time with them.

To my friends from the Academy, thank you for all of the fun times over the past two years. Dylan, Kyle, Beth, Steph, and Evan, thank you for the good times at our Bible study—I learned a lot and had a great time. Chris and Monica—Kimber and I thoroughly enjoyed spending time with you both; I hope that you enjoy a wonderful life together! John and Dan, thank you for being great roommates in our first year. Ryan, it was great having you in my Draper office, especially on those days that creative ideas just were not flowing.

To Paul, Katie, and Amy—I had a great time coaching and spending time with all of you. Thanks for inviting me to go snowboarding, rock climbing, and all sorts of other activities over the past two years—you are all great friends.

To my puppy Hazel, I have enjoyed running around with you outside and watching you be entertained by the most arbitrary objects.

To my family at home, thank you for the love and support that you have shown to me. Throughout high school, the Academy, and MIT, you have always been there, putting up with my schedule and pushing me to do well. I am truly blessed to have a family that wants me to excel and take advantage of great opportunities.

To my beautiful wife Kimber, you are the love of my life and I want you to know that none of this would have been possible without you. This last year that we spent together has been the best year of my life, even when we both were at home doing work late on a Friday night. Thank you for reading through all of my work and giving me feedback, especially when I know that you were just as busy. When it comes to combining love with brilliance, nobody does it better than you.

Finally, I want to thank the Lord for His extreme generosity toward me. I pray that I will be able to use these gifts and opportunities in a manner pleasing to Him. "And whatever you do, whether in word or deed, do it all in the name of the Lord Jesus, giving thanks to God the Father through him."

Eric J. Robinson, 2nd Lt, USAF

May 17, 2013

THIS PAGE INTENTIONALLY LEFT BLANK

Contents

THIS PAGE INTENTIONALLY LEFT BLANK

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 1

Introduction

1.1 Background

Currently, there are many organizations in the United States that use unmanned assets, such as satellites or drones, to collect information. This may include taking pictures of the ground, gathering infrared photos, taking atmospheric pressure measurements, or any conceivable form of data collection. Often these separate organizations have overlapping collection interests or flight plans that are sending sensors into similar regions. Exploiting such common interests between the organizations could potentially free up asset sensing time to be used elsewhere. Unfortunately, different organizations often have command and control stations for their assets that are spread across the nation. Even within an organization, separate missions might control their assets from different locations. This separation can make exploitation of common collection interests a nontrivial task because it might be difficult to gather all of the representatives for a meeting or even a telephone conference. Each of these organizations also has different objectives—a problem that further isolates each organization/mission and complicates mutual correspondence. An automated, coordinated approach to assigning collection tasks among these segregated missions/organizations, or "stovepipes," has the potential to significantly reduce the problem of isolation in order to

increase the realized utility of the collections made by all assets between the organizations.

Organizations collect data for various purposes. The National Aeronautics and Space Administration (NASA) collects data for scientific research; The Department of Defense (DOD) collects data for intelligence, surveillance and reconnaissance. Often these organizations utilize their own air and/or space assets in order to best manage the collection parameters—time, location of targets, desired level of resolution, and type of data being obtained (infrared, visible light, atmospheric pressure, etc.) for their targets of interest. Even within organizations (e.g., branches of DOD or NASA) there may be separate, generally unshared, collection assets. The stovepiped nature of these planners prevents cross-communication among separate sensing assets, creating inefficiencies that could be avoided through coordinated planning. Such isolation between operations of separate planners means that requests may not be allocated to the most appropriate sensors based on their specific requirements. Additionally, some missions require multiple simultaneous sensor data collections at a single location (called dual collections) that can be very difficult to accomplish with the limited assets available in a stovepiped system. Using a coordinated approach, the likelihood of fulfilling such dual collection requests can increase because the number and type of available assets is much larger. Conceptually, the coordinated approach is instantiated in the *coordination planner* (CP), which utilizes an algorithmic method for analyzing the parameters of various collection tasks within a set of planners (i.e., missions or organizations) in order to determine the best allocation of collection requests to planners. By performing this analysis and allocation, the CP effectively increases the ability of the assets on each of the stovepiped planners to complete all collection requests.

1.2 Contributions

We present a method for coordinating various collection tasks between the planning systems in order to vastly increase the utility that can be gained from air and space assets.

This method focuses on allocation of collection requests to scheduling systems rather than complete centralized planning over the entire system so that the current planning infrastructure can be maintained without changing any aspects of the schedulers. We expand on the previous work done in [1] by inclusion of a learning method to capture information about the uncertainty involved in data collection from such assets. We subsequently utilize this information in a mathematical programming method for resource allocation. With the information that we gain using on-line learning, this mathematical programming formulation has the ability to model many tradeoffs pertaining to the assignment of collection tasks to planning systems.

We begin the design of the CP by addressing how to predict whether a certain coordination request will be accepted or rejected, and subsequently completed or failed—a research topic that has not been previously considered. To do this, we combine prior intuition and knowledge about the individual planning systems with real-time data about which requests were accepted, rejected, completed, or failed using Bayesian analysis, which is then translated into probability estimates of being accepted and then completed. We also add a forward-looking component to estimate the probability that a request will be sent in the future to aid in the decision making process. Estimates are updated off-line between iterations in order to take all available observations into account.

We incorporate these probability estimates into an integer programming (IP) model which pairs the requests to planners. This IP is designed to model a variety of tradeoffs, such as duplicating requests to maximize completion probabilities vs. sending a wide variety of requests to maximize overall utility, or delaying requests until a later time if another request is more urgent. We apply this solution technique to a few potential scenarios, the results of which indicate that these models produce vastly improved utility in the sensing collection results obtained over the current stovepiped approach, whether "utility" is number of requests completed, average priority of the requests (in a system where certain requests are more important than others), or any other objective.

The results suggest that software implementations of the proposed CP could reap considerable benefits over current planning methods. This has practical significance as we intend on implementing a variation of this research in an ongoing collaboration with NASA to help utilize air and space sensing assets more effectively.

1.3 Thesis Outline

This thesis develops and analyzes a proposed method for coordinated planning through seven chapters. An overview of the thesis organization follows.

Chapter 2 develops the coordinated planning construct. This chapter begins by introducing the operational concept through a review of previous work. This review is followed by a summary of other previous literature relevant to the coordinated planning problem. Various scenarios are presented in which a CP could be of use, including climate studies and intelligence/reconnaissance settings. A short discussion about uncertainty and tradeoffs inherent to the CP is presented. This is followed by a description of the versatility of the CP, including the possibility of implementing the CP within a web service, as part of sensor webs with different levels of data sharing between assets, or even within a monetized setting. The chapter concludes with a short overview of how requests can be obtained from customers, as well as how the CP can handle simultaneous or related requests.

Chapter 3 defines the context of the coordination planning problem considered in this thesis. All of the terminology relevant to the problem is defined at the beginning, including the coordination system interface and an explicit definition of what it means to send a request for data collection. The concepts of planners and assets are fully developed along with the idea of planning cycles used for modeling. The coordinated planning problem is then explicitly defined, which includes definitions of all the specifications, constraints, and other considerations for creating a CP. The chapter closes by showing the flow of information in a coordinated system as compared to other system types (such as stovepiped systems), which

includes a discussion of how to incorporate knowledge about uncertainty into the problem.

Chapter 4 formalizes the problem into mathematical notation and produces an integer programming formulation to solve the problem of allocating requests to planners in a given iteration of coordinated planning. The data, metrics, and decisions are defined at the beginning, followed by a discussion of the modeling approach that is taken for this problem. A rigorous definition of the mathematical notation is presented, immediately followed by a development of how to value request-planner pairings as well as how to incorporate knowledge about uncertainty to create an expected value. The chapter closes by discussing practically sized problems and limitations in the modeling approach presented.

Chapter 5 shows how to estimate the probabilities introduced in Chapter 4 to incorporate any information about the uncertainty inherent to the problem into the mathematical programming formulation. Two Bayesian methods are proposed—a simple "coin flip" model and a more complicated Bayesian logistic regression model. A novel linear programming based method is presented for constructing a prior parameter distribution in the Bayesian logistic regression model, which is designed to be able to incorporate prior beliefs (as well as confidence in those beliefs) into the construction of the prior. This linear programming based method is designed to work for any Bayesian logistic regression model and therefore has applicability beyond the scope of this thesis. The chapter concludes with a brief discussion of the benefits and drawbacks to each Bayesian model.

Chapter 6 conducts an empirical analysis of the methods presented in this thesis. The first experiment performed is a direct analysis of the tractability of the formulation. This is important since integer programs are known to become inefficient for large problem sizes. The next experiment involves investigation of how well the methods of this thesis compare to a stovepiped system in a few different scenarios, including a discussion of the value gained by using information to estimate probabilities. The final experiment involves testing the sensitivity of the mathematical programming formulation to errors that might arise in probability estimation.

Chapter 7 provides a short description of differences between the research presented in this paper and that presented in previous research, as well as recommendations for further research relevant to the coordinated planning problem. This includes assumptions that could be relaxed, interesting modeling approaches that could be pursued, or other research that might be interesting.

Chapter 2

Coordinated Planning Background

Coordinated planning can be implemented within any situation that requires the use of advanced sensor systems that exist in satellites, unmanned aerial vehicles (UAVs), underwater vehicles, or ground vehicles. Studies requiring these assets have become increasingly more prevalent in the past few years. One of the major recent uses of sensor systems involves employing satellites or UAVs to study Earth's climate. The Earth Observing System, designed by NASA, is one such example of a set of sensors being used for climatology [3]. Within this system, NASA has launched many satellites. The Suomi National Polar-orbiting Partnership (NPP) satellite is one of these, launched in October 2011. This satellite orbits Earth about 14 times per day, observing nearly the entire surface in this time period. The sensors on board NPP perform many different climate-related operations, such as creating global models of temperature and moisture profiles for use by meteorologists, monitoring the ozone levels near the poles, measuring atmospheric and oceanic properties, and examining both emitted and reflected radiation from Earth's surface [3]. The Aqua satellite, also in the NASA Earth Observing System, collects information about "Earth's water cycle, including evaporation from the oceans, water vapor in the atmosphere, clouds, precipitation, soil moisture, sea ice, land ice, and snow cover on the land and ice. Additional variables also being measured by Aqua include radiative energy fluxes, aerosols, vegetation cover on the land,

phytoplankton and dissolved organic matter in the oceans, and air, land, and water temperatures" [4]. These are just two examples of the many satellites currently in orbit collecting climate-related information, all of which have some sort of overlapping interests and may even contain some of the same sensor models. As such, implementing a coordinated planning scheme within satellite planners of the Earth Observing System, or any climate related satellites, could prevent redundant gathering of the same data, while spreading collection demands more evenly across the satellites for better sensor utilization.

Recent interest in examining natural disasters has also increased, furthering the need to efficiently coordinate between sensor planners. The Hurricane and Severe Storm Sentinel (HS3) , a NASA investigation designed to enhance "understanding of the processes that underlie hurricane intensity change in the Atlantic Ocean basin," is one such example of a mission trying to learn more about natural disasters [5]. The United States Forest Service (USFS) has also recently been employing UAVs and satellites to help image active wildfires, reducing the risks of "smoke, excessive thermal wind drafts, and unfamiliar terrain" on the pilots that usually do the imaging in airplanes or helicopters [6]. The USFS also uses UAVs to collect data for invasive species professionals [7]. With all of these potential applications, coordinated planning could provide a massive benefit through the sharing of information between these organizations.

Science and forestry are not the only areas that could benefit from coordinated planning. The intelligence and reconnaissance communities utilize a tremendous amount of autonomous vehicles and sensing assets to complete missions. However, the United States intelligence and reconnaissance communities are divided into many separate organizations, including the National Security Agency, Central Intelligence Agency, National Reconnaissance Office, and individual intelligence agencies of the armed services. This division suggests that there is a large amount of overlapping desires for data collection which could be pooled in a manner that allows each individual organization to gain more utility from data collections. Indeed, the concept of coordination is already recognized as being important; according to the Joint

Doctrine for Targeting, which defines how targets for remote sensing should be created and collected, a "primary consideration" for developing targeting plans "is the joint force's ability to **coordinate**, deconflict, prioritize, synchronize, integrate, and assess joint targeting operations" [8]. Clearly a coordinated planning framework is in line with this objective, and certainly could improve the overall utility of sensing data collected for the intelligence and reconnaissance communities.

2.1 Previous Literature

The initial groundwork for developing a CP was performed by Thomas Herold in [1]. Herold goes into depth describing the operational concept of the CP in a real-world context, which he uses to motivate a description of the coordinated planning problem. Herold provides a linear programming formulation to allocate requests across to the mission planners for the available sensors in a manner by optimizing a construct that he calls the *value function*. Herold's value function uses a weighted combination of quantities relating to request priority, observation quality on a given sensor, distance for a UAV to fly from its home base, satellite viewing angle, length of the time window for request completion, and whether or not the request should be collected simultaneously with other requests. Herold then provides a study on how changing the weights for these attributes affects the performance of the CP using a controlled test scenario. Using these results, Herold proposes a method for choosing the weights by solving small optimization problems using data obtained by simulating the system. The results from Herold show that using a coordinated approach can provide a significant benefit in a sensor system.

Related analysis pertaining to centralized planning of multiple viewing assets (satellites, UAV, etc.) has been widely studied. In [9], Sakamoto considers the problem of efficiently planning missions for a collection of UAVs in a centralized manner. He proposes a robust mixed-integer programming formulation in order to create UAV mission plans that have a

high likelihood of being feasible in a stochastic environment. In [10], Blair Negron solves a very similar problem, planning missions for multiple UAVs given a set of tasks in three dimensions. Negron solves a very general problem, including time windows, observation duration, and location information for each task, as well as maximum altitude, minimum altitude, endurance, and travel time between locations as inputs for the UAV (this notion of a data collection task presented by Negron is the basis for the concept of a request presented in Chapter 3). By including such a large amount of generality in her model, the resulting mathematical programming formulation that Negron develops becomes inefficient for large applications. To fix this issue, Negron develops a meta-heuristic that creates mission plans very efficiently without sacrificing much value from optimality, thereby allowing quick solutions even for very large problems.

In [11], the authors approach control of unmanned assets for a wide array of tasks (search, target classification, attack, damage assessment, etc.). The solution approach utilizes a hierarchical division of the problem into multiple layers of control. The authors construct and simulate an auction-based formulation to determine how to best assign tasks to various groups of vehicles. A main insight is that allowing multiple assets to cooperate on a single task provides better global results. However, the hierarchical method employed for the control of UAV task assignment and completion in [11] still addresses a version of collection planning in which all of the agents work together toward the same overall objective—not for the objectives of stovepiped planners. Thus, this article still solves a more centralized problem than the the one considered in this thesis.

The authors of [12] solve a problem of completing a large set of tasks with a small number of UAVs by utilizing a mathematical program for centralized assignment of tasks to assets, and then creating a separate scheduling algorithm to decide the paths taken by the individual assets. The main two differences between this type of problem and the coordinated planning problem are (1) the coordinated planning problem assigns tasks to planners, not to individual assets, and (2) the coordinated planning problem allows planners to have their

own scheduling/control algorithms for their assets. While these articles provide insights about other potential solution approaches for the coordinated planning problem, they all focus on highly centralized planning that does not take into account the issue of planning for asynchronous, distributed systems (i.e., stovepipes).

Work has also been performed relating to efficient tasking of satellites. The authors of [13] develop a tool for the centralized planning of a vast collection of points of interest for which a large number of satellites are available. The authors split the decisions hierarchically—first assigning tasks to satellites, then separately planning the task start and end times for each satellite to maximize the total value of the targets obtained. Both sets of decisions are performed using integer programming approaches that optimize periodically over time. The resulting real-time models are capable of handling around 100 satellites and many different points of interest. The first stage of decisions presented in [13] that assigns tasks to satellites is very similar to the decisions faced in Chapter 4. The system analyzed in [13] has full, centralized control of the satellites, knowing that no other outside tasks will be assigned to those assets. In contrast, the problem developed in Chapters 3 and 4 do not actually task assets, but send requests for data collection to the mission planners for those assets, which has much more uncertainty do to the possibility of rejection and lack of knowledge about the currently planned schedules for each asset.

The authors of [14] address the uncertainty inherent in the planning of photographs taken by a single satellite, using a mathematical programming formulation that is motivated by a Markov Decision Process. Their model considers the probability that a photograph would actually be completed if it is incorporated into the schedule for the current day, as well as the probability that the photograph will be selected and subsequently completed for a future day under a given policy, to design a schedule that maximizes the total expected value of realized photographs subject to any feasibility constraints. The latter probability is calculated by looking at the number of remaining feasible opportunities each satellite would have to potentially schedule the photograph. The authors suggest that these probabilities

can be determined either by simulation or can be adaptively learned in an on-line manner through a machine learning based approach (although the actual method for learning these probabilities was left for future research). They recommend the learning approach as it allows the formulation to adapt to changes in the system over time. This approach is used as motivation for the inclusion of forward-looking probabilities in Chapter 4 involving the chance that a request will be sent by the CP and ultimately completed by some individual planner at a future time.

A more general problem of dynamically assigning an abstract resource to an abstract task over time is considered in [15]. The problem, coined by the authors as "The Dynamic Assignment Problem," is solved using a combination of network optimization and approximate dynamic programming (ADP) Techniques. The basic framework assumes that each resource can serve only one task at any given moment, but allows the tasks to appear dynamically over time. An exact dynamic programming formulation is developed that models this problem. However, the state and decision spaces are far too large to admit a solution of the exact problem, which motivates the use of ADP. The paper develops a method utilizing ADP that iteratively solves many network assignment problems to obtain a good, although suboptimal, solution. The results show that the algorithm can significantly outperform myopic optimization methods, or those that do not include forward looking knowledge about the future when making decisions. One of the major difficulties in using the methods presented by [15] is that the solution method is heavily dependent on the efficiency obtained by solving a network optimization problem to allocate resources to tasks. Unfortunately, if other outside constraints are included, if the resources can accept more than one task in any given assignment iteration, or if there are nonlinear elements in the objective, the mathematical programming formulation might lose the efficiency inherent in the structure of a network optimization problem. As a result, the aggregation-based ADP methods that the authors use become too inefficient to implement in practice. Since all of these issues exist in the coordinated planning problem, this thesis does not implement the methods of [15].

2.2 Uncertainty and Tradeoffs in Coordinated Planning

The CP developed by Herold addresses a deterministic problem. One of the main differences in the current work is to relax deterministic assumptions so that the CP can take uncertainty into account. Characterizing this uncertainty can be difficult, as the factors that contribute significantly to uncertainty may not be obvious. Successful completion of any given request can be dependent upon natural factors such as cloud cover, appropriate weather, physical condition of the sensing assets, and even systemic factors, such as the number of requests currently being serviced by a specific planner. This paper will discuss one method for the incorporation of knowledge about uncertainty, under the assumption that certain probabilities can be accurately estimated.

A very important piece of incorporating uncertainty into the coordinated planning approach involves the ability to accurately value various tradeoffs that must be considered when performing collection management. This is no trivial task as the number of potential tradeoffs that could be modeled is quite large. Examples of considerations for a CP include:

1. Sending a low priority request now that has few or no future opportunities for collection vs. sending a high priority request now that has many future opportunities for collection

2. Sending a request now to a planner with low probability of completion vs. waiting for a time when the probability of completion is higher (such as a request that may be in view of a satellite very briefly over the next hour but might be in view for a very long time three hours from now)

3. Sending a request to the planner with the most desirable sensing asset even if it has a low probability of completion vs. sending a request to the planner for a less desirable asset that may have a much higher probability of completion

4. Sending a low priority request if there is an asset with a high probability of being able

to complete the request vs. not sending the low priority request

5. Sending a high value request multiple times to maximize the chance that it will be completed vs. sending the high value request once to be able to send other less valuable requests

6. Evaluating how to send requests in a system of highly saturated, highly constrained assets

7. Evaluating how to send requests if there are budget constraints

The coordinated planning methods discussed in this thesis aim to create a method of allocation that addresses many of these concerns.

2.3 Versatility

With an appropriate design, a coordinated planning approach can be made to apply to a large variety of potential situations. The construct developed in this thesis is designed to be flexible, thereby allowing it to be implemented in many different scenarios.

2.3.1 Web Service Implementation

A *web service*, or "a software system designed to support interoperable machine-to-machine interaction over a network" [16], is a natural setting for implementing a coordinated planning approach. Users who wish to collect climate, intelligence, reconnaissance, or other data from sensing assets could easily upload requests for data to the web service using either an on-line form or a predefined XML schema. The CP would then consider all of the inputs, determine sensing assets that could feasibly complete these requests, and decide an appropriate allocation of requests to the mission planners for these assets so as to maximize overall value subject to various limitations, including budgetary constraints and capacity limits for the rate of requests being sent to the mission planners [17, 13]. The web service would then

be designed to interface with on-line request input services for the planners of various sensing assets, or directly with the internal networks of organizations owning the systems if an agreement could be made with those organizations. This setup immediately overcomes the problem of stovepipes mentioned in Chapter 1 by using the internet to overcome geographical separation, interfacing directly with the various planners, and intelligently deciding the best allocations of requests to the planners for sensing assets.

2.3.2 Sensor Web Architecture

The implementation of the CP as part of a web service could help to fulfill the concept of a *sensor web*, which "consists of intra-communicating, spatially-distributed sensor pods that are deployed to monitor and explore environments" so that "information gathered by one pod is shared and used by other pods" [18]. The sensor web architecture is a conceptual design in which various sensing assets have complete knowledge of the locations and data from all other sensors, so that planning for the collection of data can be coordinated across all the platforms.

In many practical cases it may be desirable (or required) to only have a partial sensor web architecture, in which some data is shared across the various sensing assets but not all of the data. A web-service based CP would be very useful in these situations as well because, by design, the CP leaves control of the assets to the missions and optimizes based on the request that it receives. This type of design could be useful in a situation where different organizations, such as the U.S. military, NASA, the National Oceanic and Atmospheric Administration (NOAA), and the United States Geological Survey (USGS), take advantage of the collective capabilities of all their assets while still maintaining full operational control over their respective sensors. This is an operation that makes sense in much of the current sensing systems, although research has not addressed a good way to plan within this infrastructure. As such, the work that we do in this paper will offer important insight into how to plan for partial sensor webs.

It is even possible to utilize the CP in an environment without any sensor web, but rather as an aid in planning for a single mission. For example, the mission planners for HS3 at NASA do a large amount of research observing the behavior of hurricanes using UAVs [5]. A version of the CP could be used prior to designing the mission plans for any given day to determine if satellites such as Earth-Observing 1 could potentially collect any specific pieces of data, using likelihoods and priorities of various proposed pieces of data to figure out the best way to assign requests, while considering budgetary or capacity constraints [19, 5]. In this case, the importance of controlling for uncertainty would be very high since a single user or group of users would be accepting all of the risk inherent in trusting completion of data requests to external assets.

2.3.3 Monetized Setting

A web-service or web-interface based implementation of the CP could also be used in a for-profit setting. In this situation, the coordination planning algorithm would be run by a single owner with some level of access to the resources of various assets. Requests for data would be input to the CP web interface along with the amount of money that the user would be willing to pay to have the request completed. The CP, knowing its own budget and any capacity constraints of the planners, as well as any costs associated with completing a request, would take these requests and assign them to planners in order to maximize the total expected profit over some planning horizon.

2.3.4 Obtaining Requests from Users

We have been assuming in our discussion that users know exactly what type of data they want to have collected. However, often users do not know the exact location, sensor type, or time that they want to collect data. In this case, a CP with probability estimates for the completion of requests could be of use as it would be able to present different options to the user along with probabilities of completion for each option. These options would be

generated by knowledge of where the sensors in the coordination system would be located over time. The user would then select one or more of the options to input as requests. However, design of this suggestion system is beyond the scope of this thesis.

2.3.5 Related Requests

Sometimes users may have requests which must be completed by different planners. For example, a user may desire *simultaneous collections* of information pertaining to a single target, such as a satellite image of a hurricane at the same time that an aerial vehicle obtains pressure or temperature readings. More generally, a user may have a set of requests for similar sets of data, referred to as *related requests*, which must be completed by different assets. These types of situations can be handled much more easily through the use of a CP than by using stovepiped planners as there are many more options available for servicing requests.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 3

Modeling Assumptions and Problem Definition

As shown in the previous chapter, there are many potential uses and implementations for a coordinated planning system. Thus, it is important to clearly define the modeling assumptions that will be used in order to identify the capabilities and limitations of a CP.

3.1 Terminology

We begin by defining the main terminology used throughout this paper. Some of these terms have already been introduced (e.g., *request* and *planner*) and will now be formally defined; others will be completely new. These terms have been constructed in a manner that should agree with common conventions that exist in sensor planning literature, although some terms are unique to the idea of coordinated planning. We also note that while many of the terms in this paper are consistent with those of the previous work done in [1], some have been slightly altered. Thus, it is important to follow the definitions listed here while reading this paper. Each important term will be introduced in boldface italics, with all subsequent references in plain text.

3.1.1 Coordination System Interface and Requests

At the highest level is the ***coordination system interface***. This is the actual computer program, person, or other device which collects ***requests*** from people or machines who wish to use the CP Technology, known as ***users.*** These requests are the unique collection specifications for obtaining a piece of data as defined by a user. These specifications could potentially be defined in different ways. For example, requests must be associated with some location, known as a ***target***, since we are attempting to collect data either on Earth's surface or in its atmosphere. However, the manner in which a request is defined could vary. A user may wish to view an entire region of Earth and therefore define a ***regional target***, or the user may desire an image or piece of data at a single ***point target*** defined by latitude, longitude, and altitude (see Figure 3-1 for comparison).

Figure 3-1: Regional Targets (left) vs. Point Targets (right)—Figure courtesy of [1]

A summary of all specifications required for each point target request is given in Table 3.1. We assume that any regional target can be discretized into point targets. This discretization may be done by users or it may be done by the coordination system interface. We note that this is a realistic assumption because many planners require that requests external to their own desires should be given with point targets in order to reduce the burden of scheduling.

A request will also define the specific ***task*** that must be completed, which is the actual piece of data to be collected. This could be a picture, infrared image, altitude measurement, barometric pressure reading, temperature reading, etc. A list of some example task types is given in Table 3.2.

Type of Data Input	Description
ID	A unique alphanumeric identifier associated with this request
Task Type	Type of collection task (see Table 3.2 for examples)
Priority	A value in the interval $[0, 1]$, where a higher value represents a more important request
Longitude	Longitude of the location to be serviced by the request
Latitude	Latitude of the location to be serviced by the request
Altitude	Altitude of the location to be serviced by the request
Duration	The minimum amount of time required to service this request
Related Requests	A list of all requests related to this one, if none then this value is left empty
Start Time Window *startTW*	The earliest possible start time for observing the target associated with this request
End Time Window *endTW*	The latest possible end time for observing the target associated with this request

Table 3.1: List of Request Specifications

Task Type	Description
Ground Image	Image of a specific location on the ground
Cloud Image	Image of the cloud structure in a specific location, usually to analyze a specific weather event, such as a hurricane
Infrared Ground Image	Ground Image using an infrared sensor
Infrared Cloud Image	Cloud image using infrared sensor
Topological Survey	Topological or altitude reading at a specific location
Pressure Reading	Measurement of the atmospheric pressure at a certain location

Table 3.2: Example Task Types

Sometimes users may have requests which must be completed by different planners. For example, a user may desire ***simultaneous collections*** of information pertaining to a single target, such as a satellite image of a hurricane at the same time that an aerial vehicle obtains pressure or temperature readings. In this example, the simultaneous collection would need to be completed by different assets, although this may not always be the case. More generally, any set of requests which must be completed by different sensors we refer to as ***related requests***. We assume for simplicity that all requests within a set of related requests have the same set of feasible planners, modeling the realistic desire to have multiple simultaneous collections from some set of planners.

3.1.2 Planners and Assets

A ***planner*** is the entity that schedules requests for completion on various ***assets***, which could be UAVs, airplanes, satellites, ground vehicles, or underwater vehicles, although the software and simulations developed in this paper will limit assets to being UAVs or satellites. Formally, the planner is a function that takes requests as inputs, and produces an ordered set of times in which an asset or set of assets will complete some or all of the requests that

are feasible (i.e., a mission plan) as outputs. For example, a planner could be a computer program that produces a UAV flight path to image a specific set of locations, or a human writing a flight path on paper.

3.1.2.1 Planning Cycles

Each planner has its own ***planning cycle*** which consists of ***planning***, ***upload***, and ***execution phases***, where the length of the execution phase is equal to the combined lengths of the planning and upload phases. Requests can only be completed during an execution phase. We assume that for a given execution phase, each planner will only consider feasible requests submitted during the planning phase immediately prior to that execution phase. For example, examining Figure 3-2, we see that in order for a request to be completed by the UAV planner during execution phase 2, it must be submitted during planning phase 2 for that planner. We do not assume that planners have synchronized cycles, meaning that they can all have customized phase start and end times as shown in Figure 3-2 for an example with one UAV planner and three satellite planners. We note that while the entire time line is covered by planning, upload, and execution phases in this figure, this coverage may not always exist due to breaks in planning for maintenance purposes. However, handling extraordinary breaks in the cycle is beyond the scope of this thesis.

3.1.2.2 Assets and Sensors

Each planner schedules incoming requests using ***sensors*** on one or more assets. The sensor(s) carried on each asset will perform the actual tasks associated with the requests, such as capturing an image or recording a temperature. Assets are further differentiated as being ***taskable*** or ***non-taskable***. A taskable asset is one that is willing to accept requests as input for developing a mission plan, while a non-taskable asset generates its mission plan without regard to external requests [1]. A UAV is non-taskable if its planner chooses a flight path based on its own internal collection needs before taking requests; a satellite is non-taskable

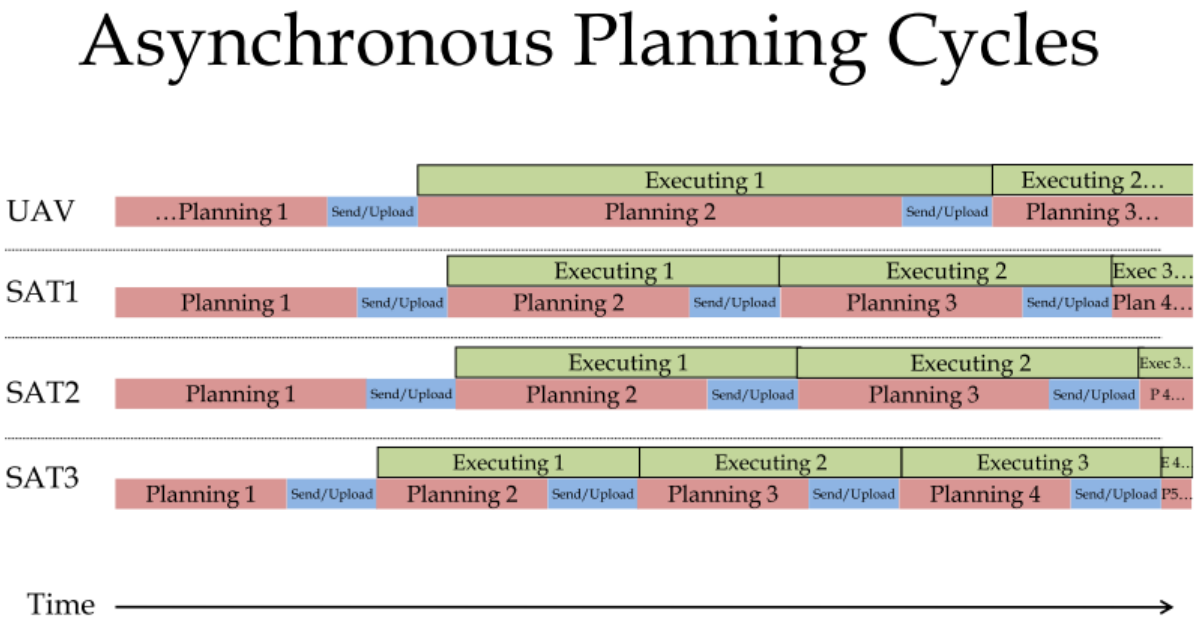

Figure 3-2: Asynchronous Planning Cycles—Figure courtesy of [1]

if the pointing angles of its sensors are known before taking requests. Generally, if a satellite is non-taskable, it is because the satellite is always "on," i.e., it always points at the same angle toward Earth and continuously collects data. In contrast, taskable assets are those in which the associated planner will simultaneously analyze its own data collection needs as well as other external requests when deciding a flight path (UAV) or set of pointing angles (satellite).

3.1.2.3 Asset Specifications

Each asset has a set of specifications for a given execution phase. Those specifications relevant to a UAV are listed in Appendix A, while those relevant to a satellite are given in Appendix B. These specifications define the physical limits of the asset, as well as the general region to be viewed for that execution phase. We assume that all of the specifications for each asset on any given planner are known.

3.1.3 The Coordinated Planning Problem

The ***coordinated planning problem*** which we analyze in this paper involves coordinating collection requests between stovepiped missions to increase the overall utility of the system, without forcing the missions to significantly alter their planning systems. The ***coordination planner*** (***CP***) first takes user-defined requests and prior information about collection interests from the participating planners as inputs in order to determine efficient ***pairings*** of requests with planners. The CP then must send its own ***coordination requests*** to the planners that it considers in its system, separately asking each one to complete some subset of the user-defined requests as determined by the pairings. It is imperative that these requests be sent during the appropriate planning phases for each planner, so we assume that the CP is aware of the planning cycles for each planner.

The CP builds up a ***queue*** of requests over time which is the set of all user-defined requests which have not yet been completed. The CP reviews this queue periodically to create pairings and coordination requests, which we refer to as the ***CP iteration***. The period of this review is referred to as the ***CP iteration length***. In order to ensure that the CP always has at least one opportunity to send requests during each planning period of a given planner, the CP iteration length is assumed to be shorter than the lengths of the planning periods for all of the individual planners. This assumption could be relaxed in reality if needed, but the cost would be that the CP may not have the opportunity to send requests for some execution periods on individual planners.

3.1.3.1 Opportunity Finder

The first step in reviewing the CP queue involves employing an ***opportunity finder***, which is a filter to determine feasible pairings of requests to planners. For example, the opportunity finder would filter out the possibility of pairing a request in Nevada with a planner for a non-taskable UAV that only flies over the Atlantic Ocean, or a satellite that does not pass over Nevada. We assume that this opportunity finder finds all pairings that have a positive

probability of being completed, and removes all other pairings. The opportunity finder utilizes the specifications of the assets to make these determinations. For the analysis that is presented in this thesis, the opportunity finder takes into account the time window of a request, whether the endurance of a UAV is long enough to travel the great-circle distance to the request location and back again [20, 21], and whether the predicted orbit of a satellite using secular J2 perturbation theory has a line-of-sight to the request location [2].

3.1.3.2 Coordination Requests

The next step involves choosing the best pairings and then sending the coordination requests to the planners. Each of these coordination requests can then be ***accepted***, ***rejected***, ***completed***, or ***failed*** by the individual planners—we call this the ***status*** of the coordination requests (see Table 3.3 for explanation of these terms). We require all planners to inform the CP when coordination requests are completed or failed; however, we allow planners the option of informing the CP when requests are accepted or rejected. In order to differentiate these two options, we refer to planners as being ***informative*** if they inform the CP about accepted and rejected requests; otherwise the planners are ***non-informative***. We assume that all planners land in exactly one of these two categories, i.e., we do not have any "partially informative" planners which give feedback for some requests but not for others. We also assume that a user request is completed if and only if at least one coordination request associated with that user request is completed; otherwise the user request is failed.

3.1.3.3 Individual Planner Specifications

We assume that each planner also has a set of specifications which defines the level of interaction that it has with the CP. The first of these, i.e. whether a planner is informative or non-informative, has already been explained. The remainder of the specifications are described here.

1. **Identification (ID):** Each planner has a unique alphanumeric ID tag for reference.

Type of Feedback	Definition
Accepted	Indicates that a planner has selected the given coordination request to be included in the planned schedule for its asset(s).
Rejected	Indicates that a planner has decided not to include the given coordination request in the planned schedule for its asset(s).
Completed	Indicates that the given coordination request has been fulfilled by a planner and the data is ready for delivery to the user. Note that this implies the request must first have been accepted by this planner.
Failed	Indicates that the given coordination request could not be completed due to unforeseen circumstances even though it was accepted by the planner.

Table 3.3: Potential Types of Status Feedback from Planners

2. **Management Authority:** Each planner will have either an ***internal*** or ***external*** management authority. Conceptually, an internally managed planner is one for which the CP has full knowledge of an asset's planning process, location, and schedule, while an externally managed asset is one for which this information is not known. We do not explicitly differentiate between these two types of planners in our analysis as long as all UAV planners provide the information given in Appendix A and all satellite planners provide the information given in Appendix B. We leave the determination of planner-specific important information to future research, which could be used to influence information that could be obtained from internally managed planners.

3. **Assets:** Each planner has assets associated with it which are known to the CP in advance, along with the specifications of those assets.

4. **Capacity:** We assume that each planner has a maximum number of coordination requests that it can receive per unit of time for a given execution period, which is known as the planner's capacity. For example, if the capacity of a planner is 10 coordination requests per hour, then an execution period of length two hours would allow a maximum of 20 coordination requests to be considered. Some of the time this capacity is explicitly defined by the operators of the planners, as is the case for sensor planning services following the specification in [17]. Other times, this constraint exists as a contractual guarantee by the CP to prevent over saturation of planners with coordination requests. In a situation where neither of these hold true, it may still be intelligent to impose the capacity constraints anyway based on the number of requests completed by various planners, although research into how to best choose such values is beyond the scope of this thesis. By imposing these capacity constraints, we can reasonably assume that, given the general location in which an asset will be collecting sensing data, the probability that a planner will complete a given request is negligibly influenced by the other coordination requests being sent to that planner. This allows

for the reasonable independence assumption that if requests $r_1, \ldots, r_k$ are all sent to planner l, then the set of probabilistic events of the form "request r_i is completed by planner l" for all $i \in \{1, \ldots, k\}$ form an independent set of events, as do the events "request r_i is accepted by planner l" for all $i \in \{1, \ldots, k\}$.

5. **Reservation Fee:** A given planner l might impose a "reservation fee" in dollars of $c_l > 0$ per coordination request that it considers. This is more common in planners for commercial assets which must either recoup expenses or make a profit. These funds are assumed to be collected upon submission of the request, although they might be refundable at a later time if the planner does not complete the request. We assume that there is a set amount of funds available to the CP at each iteration for the intent of satisfying necessary reservation fees.

3.1.3.4 Weather and Nature

A final piece of information that must be dealt with in the coordinated planning problem is the issue of weather. There are many potential disturbances that could affect the performance of various assets, including cloud cover, wind speed, precipitation, and other natural factors. For the scope of this thesis, we assume that the manager of the CP has the option to input which natural factors he/she feels have the biggest impact on the probability that a request will be completed. We choose to leave the option open for including these features because different situations in which the CP is employed may require different factors to be considered. We leave the analysis of determining the best features for certain situations for future research. However, we note that it is possible to implement the methods described in this paper without rigorous feature selection research.

3.1.3.5 Other Modeling Assumptions

1. **Budget:** The manager of the CP will likely not have unlimited funds to offset the reservation fees of coordination requests, even if the money is refundable. We model

this idea by assuming that the coordinator has a maximum amount of money per unit of time that is available for reservation fees—or a budget. For example, if the CP iteration length is 2 hours and the budget gives $500 per hour for reservation fees, then we assume that at each iteration the total amount of reservation fees cannot exceed $1,000.

2. **Request Sending Limits:** We allow the CP to send multiple coordination requests to distinct individual planners in an effort to complete a single user request. However, we do not want to send too many coordination requests for any single user request, regardless of the value assigned to that request or its probability of completion, because one of the purposes of the CP is to use resources efficiently. We model this desire by implementing a strict limit on the number of coordination requests that can be generated for each user request per unit of time.

3.2 Flow of Information

General purpose collection management requires a large flow of information between users and planners. This flow of information can be streamlined by the use of coordinated planning as we will now show. For comparative purposes, we also explain the information flow used currently ("stovepiped systems"), as well as the ideal, yet usually impractical, situation ("synchronized planning systems").

3.2.1 Coordinated System

The coordinated system which we utilize maintains all of the same communication channels as the stovepiped system, but adds in two extra additional options for users. The first additional option allows the user to input requests to the coordination system interface rather than directly to the planners. The second option allows users to still input requests directly to the planners, after first querying the coordination system interface via a web service to

obtain information pertaining to the likelihood that their request(s) would be completed by various planners, including which planners would actually have the opportunity to complete that request. All of this information flow is depicted in Figure 3-3.

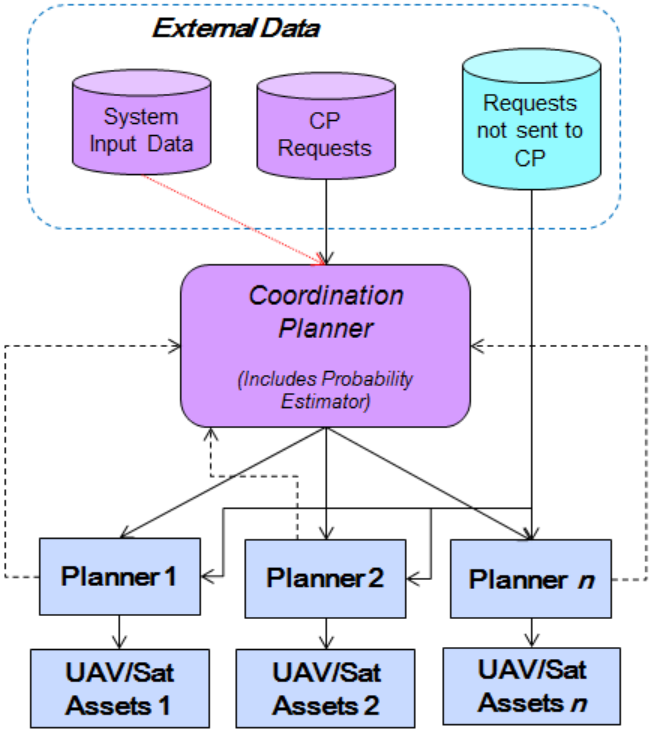

Figure 3-3: Information Flow in a Coordinated Planning System

We see that the CP does not actually perform any of the scheduling for the individual planners. It is important to note that when employing a coordinated system, we do not alter any of the current infrastructure—rather we add to what already exists. Thus, we allow for planners to exhibit asynchronous planning cycles as mentioned previously, which means that deadlines for submission of requests could be different for each planner.

3.2.2 Coordinated Planning Iteration

The coordinated planning iteration consists of two phases: the information gathering phase and the coordinated planning phase. The information gathering phase begins at periodically spaced ***epochs*** in time, where the period between epochs is the aforementioned CP iteration

length. This phase continues until the next epoch, at which point a new coordinated planning phase and information gathering phase are initiated. This coordinated planning phase ends when all coordination requests for the current iteration have been sent to planners, and is in general much shorter than the information gathering phase. This process is illustrated in Figure 3-4. Note that information gathering and coordinated planning phases overlap such that the i^{th} coordinated planning phase starts simultaneously with information gathering phase $i + 1$, although the coordinated planning phase is much shorter and therefore ends earlier.

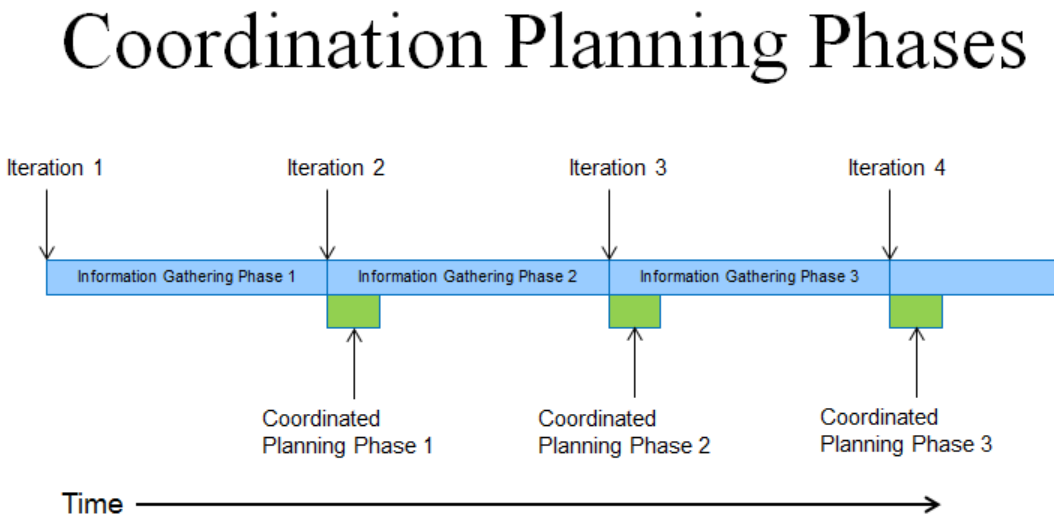

Figure 3-4: Phases Within a Coordinated Planning Iteration

3.2.2.1 Information Gathering Phase

During the information gathering phase, users input their requests to the CP via the coordination system interface, which adds the requests to the queue. Also during this phase the CP receives feedback from the individual planners pertaining to the status of previously assigned coordination requests. Any requests that are completed during this phase are removed from the queue, and the completed collection data is made available to the appropriate users. All other notifications, i.e., accepted, rejected, and failed, are parsed into data which is stored for later use. We assume that all informative planners return information

about accepted/rejected coordination requests at some time after the coordination request has been sent but before the execution period begins for the appropriate request-planner pairing. We do not make any explicit assumptions within our mathematical model as to when planners inform the CP about completed/failed requests other than it has to be after the execution period has ended for the appropriate pairing. However, for analysis purposes, we will only consider scenarios in which the CP is informed of completions and failures immediately following the end of the appropriate execution periods. This models the idea that planners should desire to make this information available as quickly as possible.

3.2.2.2 Coordinated Planning Phase

During the coordinated planning phase, all of the data collected from the information gathering phase is reviewed in order to convert user requests into coordination requests which can be assigned to the planners. This process begins by passing all requests in the queue through the opportunity finder to determine feasible pairings of requests to planners. Then, a ***probability estimator*** reviews all of the stored data pertaining to the results of prior coordination requests (i.e., the defining attributes of the coordination requests, as well as whether they were accepted/rejected, completed/failed and by which planner) to create probability estimates for acceptance and completion of all feasible pairings of requests to planners. This information is sent to an optimization algorithm which determines efficient pairings relative to some predefined utility (e.g., number of requests completed), and then these pairings are sent to the appropriate planners as coordination requests. The coordinated planning iteration is depicted pictorially in Figure 3-5. In this flowchart, the steps associated with the coordination planning phase are located inside of the dotted line, and the information gathering steps are outside this line (more details on the mathematics behind these steps are given in the following chapters).

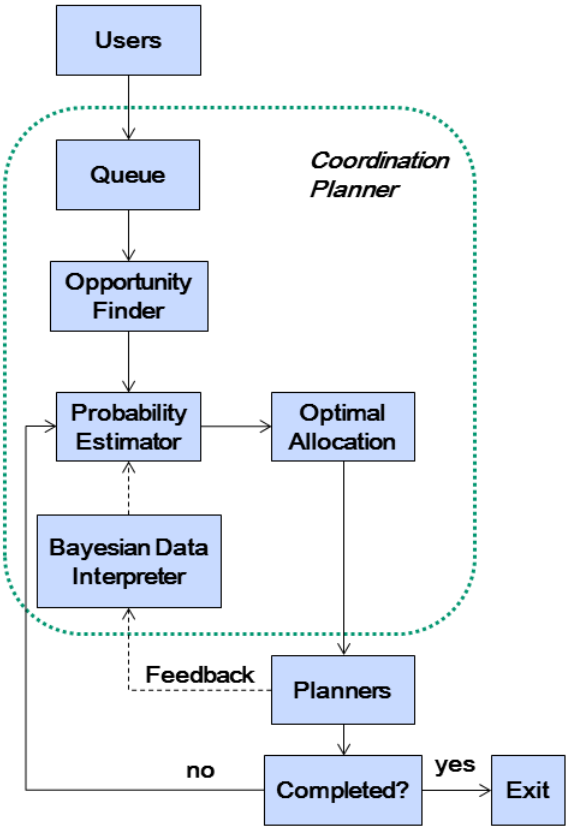

Figure 3-5: Single CP Iteration

3.2.3 Other System Designs

3.2.3.1 Stovepiped Systems

Presently, planners tend to act in the stovepiped manner described in Section 1.1 where the operators of various assets do not communicate with operators of other assets. In this system, users input requests directly to the planners following the solid arrows in Figure 3-6, and planners return completed data requests to the users following the dotted arrows in the figure.

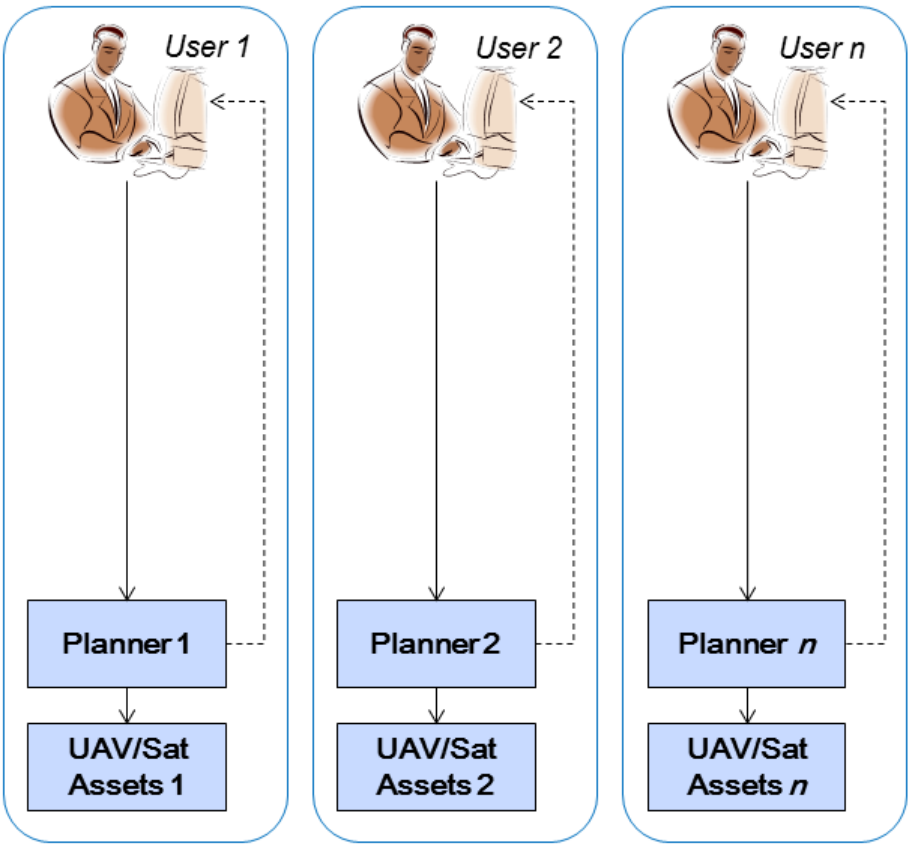

Figure 3-6: Information Flow in a System of Stovepiped Planners

3.2.3.2 Synchronized Planning Systems

The ***synchronized planning system*** architecture is a theoretical concept that has been researched extensively [1, 13]. In this construct, some subset(s) of the planners are forced to

have the exact same planning cycle as shown in Figure 3-7 (hence the name "synchronized planning systems"), rather than exhibiting the asynchronous planning cycles shown in Figure 3-2 which are allowed in a coordinated planning system.

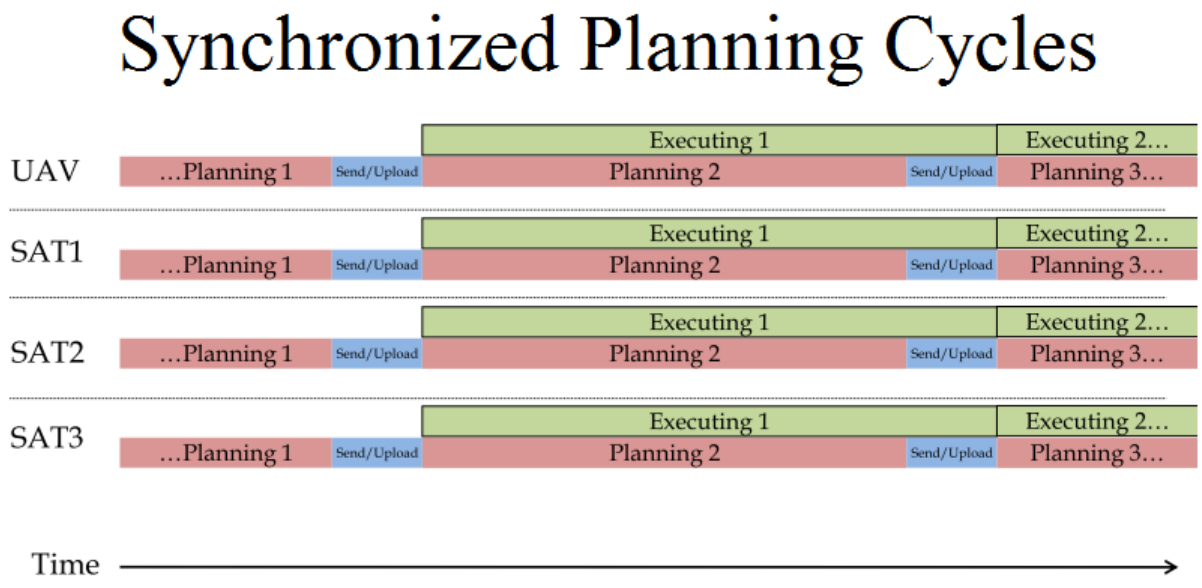

Figure 3-7: Synchronized Planning Cycles

3.2.3.3 Brief Comparison with Coordinated System

As has already been mentioned, stovepiped systems present a vast array of problems and inefficiencies in collection management. The lack of communication in these systems forces users to bear the burden of finding the best possible sensor(s) to use for their specific requests, which can lead to users trying to locally optimize their schedules without regard for other users that may want to employ some of the same assets. Users may also be unaware of the benefits of certain assets, or simply not have the personnel contacts to use other assets that would be well-suited for their tasks. Some other major disadvantages of stovepipes include their inability to efficiently find ***piggybacking*** opportunities (i.e. chances to add their requests onto the previously scheduled plans of a different asset which may be operating in a desirable location), or from pooling requests between various users to find a more efficient allocation of requests to sensors.

All of these difficulties suggest that a highly centralized synchronized planning system could be quite useful. Theoretically, a synchronized system could have the ability to control large subsets of the planners simultaneously from a centralized platform, and thus streamline collection operations. Although there are some domains in which it makes sense to implement this type of system, often synchronized planning is highly impractical. We will rarely be able to implement a synchronized planning system as this would require a complete overhaul of the existing planning infrastructure due to the stovepiped nature of current planners. Computer interfaces between synchronized assets/planners would need to be redesigned and headquarters of such mission planners would likely need to be relocated so that the operators of synchronized planners/assets could work together. The individual planners would have to give up their respective planning cycles. For these reasons we do not focus on centralized or synchronized systems, even if theoretically more efficient.

A coordinated planning scheme has the potential to maintain many of the benefits of both stovepiped and synchronized planning systems while eliminating their problems. With coordination, we eliminate the lack of communication inherent to the current system of stovepipes by providing a single, automated platform which can interface with each planner to send requests and receive information. This is done without forcing planners into a single synchronized planning cycle but rather allows them to maintain their own respective cycles. In addition, planners are given the liberty to choose their own planning algorithms and control their assets, and are even allowed to set their schedules before considering any coordination requests (these would be "non-taskable" planners). Thus, by coordinating, we can eliminate the communication gap between stovepipes without implementing the impractical restrictions on the planners that come attached to a synchronized system, which can ultimately yield large increases in the collective utility of all users in the system. All that is asked in return is for the planners to give legitimate consideration to the coordination requests that they receive, as well as give timely, appropriate feedback to the CP pertaining to accepted or rejected request status updates (for informative planners), and completed/failed

status updates (for all planners). While not required, it is highly encouraged for planners of non-taskable assets to inform the CP of orbital dynamics for satellites (usually this is publicly available), and scheduled flight paths or takeoff and flight range information for UAVs. In addition, planners are encouraged to put themselves in the "informative" category in order to provide extra information to the CP, although this is not required either. Since the "users" quite often are the owners and operators of individual planners and assets, the burden involved for a planner to join a coordinated system is small compared to the gains that can be realized by their respective operating agencies.

Chapter 4

Mathematical Model

Section 3.2 presented the algorithmic flow within a coordinated system during a given coordinated planning iteration. One part of this involved generating coordination requests by deciding intelligent pairings of user requests to planners. From a user's perspective, this decision process is simply a black box that determines which planner will have the opportunity to service the requests. It is the main purpose of the CP to design an effective algorithm to act as this black box. The next two chapters are devoted to developing the mathematics behind the algorithm by first formulating the problem and then examining the uncertainty aspect.

4.1 Background

4.1.1 Data, Metrics, Decisions

We begin by exploring the data, metrics, and decisions available for a decision tool. At any given planning iteration, we have available to us a very large amount of data: a list of user requests and their specifications, a list of planners in the coordinated system and their specifications during the current iteration, sets of related requests, a budget of how much money per unit time can be allocated to reservation fees, a planner capacity of coordination

requests per hour, and a maximum number of coordination requests that can be allocated to user requests per unit time. Given this data, we must be able to choose the best allocation of coordination requests to planners in any given iteration. The metric for making these decisions is a concept of utility referred to as the ***value*** of a user request on a planner. This value represents a numerical measure of the benefit to the CP if a coordination request associated with a user request were to be completed by a given planner.

4.1.2 Mathematical Modeling Approach

The concept of having uncertain information that is revealed over time fits naturally into the dynamic programming (DP) paradigm. However, there are many difficulties associated with using DP for this problem. In DP, a state space must be defined such that at any point in time where decisions take place, the state space incorporates all relevant information for those decisions. Decisions are then created for every possible instantiation of the state space over time. While it would be possible to create a well-defined state space for the coordinated planning problem, it would contain many abstract quantities that would make enumeration of the entire state space impossible. Even with simplifying assumptions and discretization of continuous components within the state space, it would be very difficult to obtain a tractable problem. Often such issues can be circumvented using various approximate dynamic programming methods. Unfortunately, our problem makes the application of such methods difficult due to the massive number of decisions that must be considered at each iteration, which makes searching a list of decisions very slow. This creates a problem for both DP, which requires a probability distribution to be defined over all forms of uncertain information, and approximate DP, which usually requires the availability of a high-fidelity simulation for the evolution of the system. Therefore, approximate DP requires the existence of well-defined probability distributions for uncertain information—a difficult issue to resolve for the coordinated planning problem (for more background into DP and approximate DP methods, we refer the reader to [22, 23]).

In contrast, the binary nature of decisions involving the assignment of coordination requests to planners, in combination with the quantitative measure of utility, suggests that iteratively solving a sequence of binary integer programs (IP) also naturally fits into the structure of the problem (for a background on IP methods, see [24]). In contrast to DP, no state space is required for IP, and the massive amounts of decisions that need to be made can be put into a very specific mathematical programming structure which allows us to search the solution space much more efficiently than would be possible in DP. However, by abandoning DP, we lose the built-in consideration of uncertainty. In order to counter this, we will utilize a probability estimation technique, described in Chapter 5, to estimate approximate likelihoods that a coordination request will be completed if sent to a given planner. This allows us to model uncertainty using iterative learning techniques to translate features of requests, planners, and nature into simple probabilities that can adapt with the system rather than trying to create probability distributions over abstract quantities. As such, this is the modeling approach that we will take—iterative solution of an integer program over time.

4.2 Mathematical Programming Formulation

4.2.1 Notation and Definitions

Recall from Table 3.1 that each *user* request has a unique alphanumeric ID, and from Section 3.1.3.3 that each planner has a unique alphanumeric ID. For the remainder of the paper, we will always refer to requests and planners by their ID tags. An arbitrary request will therefore be referred to as "request r" where r represents the request ID; an arbitrary planner will similarly be referred to as "planner l." Further, as shown by these examples, any instance of the word "request" will always refer to a user request unless we explicitly use the term "coordination request" defined in Section 3.1.3. We will also occasionally refer to an "execution period t," which refers to the execution period with ID t (see Figure 3-2).

Although we know the planning cycles, there may be some information that is not available for future execution periods. This is particularly the case in UAV planners which have assets flying highly variable routes between different execution periods (this will almost never be the case for a satellite planner). We refer to those execution periods for which we have all of the specifications defined as ***known execution periods*** as they represent the potential opportunities for pairing requests with planners.

Prior to determining the appropriate assignments of user requests to planners, we will need to determine which pairings should be considered during the current planning iteration. This involves first using the opportunity finder to determine the set of all feasible request-planner pairings, and then removing the set of all pairings (r, l) for which a coordination request associated with the next execution period t_{next} of planner l has already been sent. For example, suppose that request 1 is feasible during t_{next} of planner 2. If during one of the previous coordinated planning iterations we had sent a coordination request to planner 2 asking it to complete request 1 during t_{next} of planner 2, then we would remove this from our list of potential pairing considerations. Thus, at each coordinated planning iteration, we look at this subset of the feasible pairings. Table 4.1 defines the applicable set notation that will be used for a given coordinated planning iteration.

In addition to these sets, we also define a few other values. We recall from Section 3.1.3 that we must consider the budget, planner capacity constraints, and request sending limits when determining the best assignments of requests to planners. We also need to explicitly define terms to represent CP iteration length and probability estimates. To model these concepts, we introduce the quantities defined in Table 4.2.

4.2.2 Example

In order to better understand this structure, consider the following example situation of a given coordinated planning iteration. Suppose that there is a queue of three requests $r1$, $r2$, and $r3$, and the system contains two individual planners, A and B. Based on this system,

Set	Definition	Arbitrary Element
R	Set of all requests r in the CP queue	Some request r
L	Set of all individual planners l in the coordinated system	Some planner l
F	Set of all potential request-planner pairs (r, l) for the current execution period	An ordered request-planner pair (r, l)
R_l	Set of all requests r such that (r, l) is a potential pairing	Some request r
L_r	Set of all planners l such that (r, l) is a potential pairing	Some planner l
D	Indexed set containing all sets of related requests (i.e. requests which must be completed by different planners).	A set of related requests $\{r_1, \ldots, r_N\}$ where $N \geq 2$.
$D(i)$	The i^{th} element of the set D, representing a set of related requests	Some request r
$L_{D(i)}$	Set of all planners on which the related requests $r \in D(i)$ are feasible	Some planner l
RFE_{rl}	The set of all remaining feasible known execution periods t for request r on planner l	Some execution period t
S_{rl}	The set of all execution periods t in which a coordination request associated with r has already been sent to planner l	Some execution period t

Table 4.1: Set Definitions

Quantity	Definition
$CPiterationLength$	Time length in hours of a coordinated planning iteration
$budget$	Number of dollars per hour allocated to reservation fees
$capacity_l$	Number of coordination requests per hour allowed by planner l
$maxRequest$	Maximum number of coordination requests per hour allowed for each user request
b	$b = budget * iterationLength$—i.e., the budget available per iteration
c_l	reservation fee charged by planner l per coordination request
n_l	$n_l = \lfloor capacity(l) * iterationLength \rfloor$—i.e., the number of coordination requests per iteration allowed by planner l
N^{max}	$N^{max} = \lfloor maxRequest * iterationLength \rfloor$—i.e., the maximum number of coordination requests per iteration allowed for each user request

Table 4.2: Important Quantities

the opportunity finder determines that the feasible planners for each request during the next execution periods of planners A and B are those shown in Table 4.3. In addition, the requests $r1$ and $r2$ are related, while $r3$ is a single request. Based on this knowledge, the quantities shown in Tables 4.4 and 4.5 can be defined. Note that a few more sets—RFE_{rl} for $r \in \{r1, r2, r3\}$ and $l \in \{A, B\}$—would still need to be obtained from the opportunity finder and past data before the integer programming formulations in the next section could be implemented to determine where each coordination request would be sent.

ID	Priority	Feasible Planners for e_{plan}	Related Request Set
$r1$	1	A and B	$\{r1, r2\}$
$r2$	0.7	A and B	$\{r1, r2\}$
$r3$	0.5	B	none

Table 4.3: Requests in Example Queue

R	L	F	R_A
$\{r1, r2, r3\}$	$\{A, B\}$	$\{(r1, A), (r1, B), (r2, A), (r2, B), (r3, B)\}$	$\{r1, r2\}$

Table 4.4: Example Sets (Part I)

R_B	L_{r1}	L_{r2}	L_{r3}	D	$D(1)$	$L_{D(1)}$
$\{r1, r2, r3\}$	$\{A, B\}$	$\{A, B\}$	$\{B\}$	$\{\{r1, r2\}\}$	$\{r1, r2\}$	$\{A, B\}$

Table 4.5: Example Sets (Part II)

4.2.3 Integer Programming Formulation

Recall from Section 3.2 that planning is done in an iterative fashion. In particular, the CP builds a queue of user requests over time that it reviews periodically to determine assignments of user requests to planners, which it then submits to planners as coordination requests. Using this as motivation, we now introduce the integer programming (IP) formulation to optimize the assignments of requests to planners in a single planning iteration. For the remainder of this section, we assume that we are in a single coordinated planning iteration at the time epoch denoting the beginning of the coordinated planning phase. We also assume that the sets in Table 4.1 are known.

4.2.3.1 Request Values

We begin by defining a notion of value for each feasible request-planner pairing at any given iteration. This value, denoted v_{rl}, is designed to represent the utility to the CP if the coordination request associated with user request r were to be completed by a planner l. However, we can only claim the value for a single completed coordination request per user request because each user request only needs a single collection of data. Thus, we define the ***utility*** of a past request r to be the maximum of the set of all values v_{rl}, where planner $l \in L_r$, such that a coordination request associated with r was completed by planner l.

A potential construction of the values v_{rl} was described by Herold in [1]. He introduces a concept called a *value function* that contains a weighted sum of quantitative request features to produce a single "value" for each potential request-planner pairing. We suggest using a similar approach where each value v_{rl} is a linear combination of quantitative features that are considered to be valuable, normalized to the interval $[0, 1]$ with weights that sum to unity across all of the attributes, and constructed to have linear increase in value (i.e., if

we increase an arbitrary one of these three features by some amount ε, the value associated with this feature increases in an amount directly proportional to ε). We suggest removing features from the value function that are only related to uncertainty and not to the value of the request, such as the number of remaining feasible known execution periods for a request r on a planner l. By doing this, the uncertainty component can be explicitly addressed using methods in the next chapter without requiring the use of simulation to determine approximate weights for such features.

Supposing we construct the value function with $m \geq 0$ features, a request-planner pairing (r, l) having instances $f_1(r, l), \ldots, f_m(r, l)$ of those features would have a total value of $v(r, l) = M^{scale} \sum_{i=1}^{m} w_i f_i(r, l)$, where each $w_i \geq 0$ is the percent importance the i^{th} feature to the CP, so that $\sum_{i=1}^{m} w_i = 1$, and M^{scale} is a scaling factor used to protect against low-precision error tolerances in computer implementations of optimization routines. For this project, we set $M^{scale} = 10$ and use the features described in Table 4.6, all of which were part of the "utility related" features given in [1].

Symbol	Description
$priority_r$	Priority of request r in the interval $[0, 1]$
$qualObservation_{rl}$	Observation quality produced by planner l for coordination requests having the same task type as r, continuous on the interval $[0, 1]$
$related_r$	Binary feature taking value 1 if r is part of a related set of requests and taking value 0 otherwise (has a positive weight if collecting a set of related requests yields extra value to the CP)

Table 4.6: Features Used to Determine Values of Request-Planner Pairings

4.2.3.2 Non-linearity in the Objective Function

The qualitative objective for this problem is to maximize the ***expected total utility*** at each planning iteration. Recall from the previous section that the utility of a request r is the value that is actually realized by the CP, which is the maximum of the set of all values v_{rl}

such that a coordination request associated with r was completed by planner l. Thus, the ***total utility*** is the sum of the individual request utilities for each request r in the queue. However, when deciding where to send coordination requests, we do not know which ones will be completed. To model this, we introduce random variables Z_{rl} which take a value of v_{rl} if a coordination request associated with r is completed by planner l *during any execution period* and 0 otherwise. We also introduce integer decision variables x_{rl} to take a value of 1 if we choose to send a coordination request associated with r to planner l *for the next execution period*, and 0 otherwise. Under this model, the utility V_r of request r is $V_r = \max_{l \in L_r} \{Z_{rl}\}$, so the expected total utility of all requests is $E\left\{\sum_{r \in R} V_r\right\}$. Based on our definitions, the probability mass function for Z_{rl} is

$$P\left(Z_{rl} = z\right) = \begin{cases} 1 - q\left(r, l, x_{rl}\right) & \text{if } z = v_{rl} \\ q\left(r, l, x_{rl}\right) & \text{if } z = 0 \\ 0 & \text{otherwise,} \end{cases} \tag{4.1}$$

where $q\left(r, l, x_{rl}\right)$ is the probability that no coordination requests associated with request r will be completed by planner l during any execution period as a function of the decision x_{rl}. Using the assumption mentioned in Section 3.1.3.3 that the events of requests being sent, accepted, or completed are independent between different execution periods, this probability can be expanded as

$$\begin{aligned} q\left(r, l, x_{rl}\right) = & \left(1 - x_{rl} \times P_a(r, l, t_{next}) \times P_c(r, l, t_{next})\right) \\ & \times \left[\prod_{t \in RFE_{rl}} \left(1 - P_s(r, l, t) \times P_a(r, l, t) \times P_c(r, l, t)\right)\right] \\ & \times \left[\prod_{t \in S_{rl}} \left(1 - P_a(r, l, t) \times P_c(r, l, t)\right)\right]. \end{aligned} \tag{4.2}$$

where

- $P_s(r, l, t)$ is the probability that a coordination request associated with r will be ***sent*** to planner l for completion during execution period t,
- $P_a(r, l, t)$ is the probability that a coordination request associated with r will be ***accepted*** by planner l for completion during execution period t, given that it has already been sent,
- $P_c(r, l, t)$ is the probability that a coordination request associated with r will be ***completed*** by planner l during execution period t, given that it has already been sent and accepted,
- RFE_{rl} and S_{rl} are as defined in Table 4.1,
- t_{next} is the next execution period for planner l.

The functions $P_s(r, l, t)$, $P_a(r, l, t)$, and $P_c(r, l, t)$ are assumed to be known and comprise what we refer to as the ***probability estimators***. We will discuss how to learn these functions over time using prior beliefs and data in Chapter 5. We note that $P_a \equiv 1$ for non-informative planners since we receive no information about rejection, implying that acceptance probabilities for non-informative planners are bundled with completion probabilities. In addition, once a coordination request for the pairing (r, l) during execution period t has been sent or accepted, we update $P_s(r, l, t) = 1$ or $P_a(r, l, t) = 1$, respectively, for that pairing. Under this model, the expected utility of request r is $E\{V_r\} = E\left\{\max_{l \in L_r}\{Z_{rl}\}\right\}$, which can be calculated using Algorithm 4.1. Hence, the objective to be maximized, or the expected total utility of all requests, is simply

$$E\left\{\sum_{r \in R} V_r\right\} = \sum_{r \in R} E\{V_r\}. \tag{4.3}$$

4.2.3.3 Linearized Formulation

Directly applying (4.1), (4.3), and Algorithm 4.1 would yield an objective function that is nonlinear in the decision variables x_{rl}. However, the structure of the value function is such

Algorithm 4.1 Expected Utility of a Request

Inputs: request r, decision variables x_{rl} for all $l \in L_r$

1. Sort the set L_r into a new indexed set $\tilde{L}_r$ ordered according to increasing values v_{rl} for all $l \in L_r$. In other words, if $\tilde{L}_r(i)$ denotes the i^{th} planner of this set, and $v(i)$ the value of the associated request-planner pairing $\left(r, \tilde{L}_r(i)\right)$, then $v(i) \leq v(j)$ for all $i \leq j$.
2. Initialize $v_{total} = 0$ and $i = 1$.
3. While $i \leq |\tilde{L}_r|$, loop through the following:
   (a) Define $l = \tilde{L}_r(i)$, the i^{th} element of $\tilde{L}_r$.
   (b) Define $p = q(r, l, x_{rl})$, the probability that no coordination requests associated with request r will be completed by planner l during any execution period.
   (c) Update $(p)(v_{total}) + (1-p)v(i) \rightarrow v_{total}$.
   (d) Update $i + 1 \rightarrow i$.
4. Output $E\{V_r\} = v_{total}$.

that the only nonlinear pieces are multiplicative interactions between decision variables associated with the same request. There are never any nonlinear interactions between decision variables associated with different requests. We can use this fact to circumvent the problem of the nonlinear objective function by introducing extra binary decision variables y_{rG} into the formulation which represent ***composite decisions***. Specifically, each decision variable y_{rG} takes a value of 1 if we send the request r to each planner $l \in G$, for some set $G \subseteq L_r$, but not to any other planners; otherwise we set $y_{rG} = 0$. We introduce one such variable for each potential (r, G) pair such that $r \in R$ and $G \in T_r$, where T_r is the set of all subsets of L_r with at most N^{max} elements (including the empty set), and $N^{max} > 0$ is the maximum number of coordination requests per iteration allowed for each user request (as defined in Table 4.2). For each of these composite variables, we introduce a ***composite utility*** k_{rG} which is defined to be the expected utility of request r if a coordination request associated with r is sent to each of the planners in G, or

$$k_{rG} = E\{V_r\} \tag{4.4}$$

where $E\{V_r\}$ is calculated using Algorithm 4.1 for request r being sent to all planners $l \in G$, and the decision variables x_{rl} for all $l \in L_r$ take the values

$$x_{rl} = \begin{cases} 1 & \text{if } l \in G \\ 0 & \text{otherwise.} \end{cases} \tag{4.5}$$

We must have $y_{rG} = 1$ for exactly one set $G \subseteq T_r$ so that we do not count the value of more than one set G for any given request r. Therefore we must include the constraints

$$\sum_{G \in T_r} y_{rG} = 1 \quad \forall r \in R \tag{4.6}$$

in our integer programming formulation. We also know that if some $y_{rG} = 1$, then it must be true that $x_{rl} = 1$ for all $l \in G$, so we must include the constraints

$$y_{rG} \leq x_{rl} \quad \forall r \in R, G \in T_r, l \in G. \tag{4.7}$$

Since we are performing a maximization, the constraints (4.6) and (4.7) are also sufficient ensure that if all estimated probabilities are in the open interval $(0, 1)$ with $x_{rl} = 1$ for all $l \in G$ and $x_{rl} = 0$ for all $l \notin G$, then $y_{rG} = 1$. This is because $k_{rG} = E\{V_r\}$ strictly increases if an extra planner l is added to G, so we have the relationship that if $G_1 \subset G_2$, then $k_{rG_1} < k_{rG_2}$. Thus, in order to maximize the objective and satisfy constraint (4.6) we must set $y_{rG'} = 1$ where $G' = \{l | x_{rl} = 1\}$, which is exactly the correct composite variable. (Even if some of the estimated probabilities are not in the open interval $(0, 1)$, we still obtain an optimal solution as the only possible variables y_{rG} that would be set to 1 in a maximization are those where (r, G) satisfies $k_{rG} = k_{rG'}$.)

In addition to adding the constraints to satisfy the definitions of y_{rG} in conjunction with the request sending limits, we need to translate our budget, capacity, and related request requirements into mathematical constraints, using the notation in Table 4.2. Since each

planner charges a reservation fee of c_l per submitted request for which we have a budget of b dollars per iteration, the budget constraint is

$$\sum_{r \in R} \sum_{l \in L_r} c_l x_{rl} \leq b. \tag{4.8}$$

The capacity constraints are simple as well. Since we can send at most n_l coordination requests to each planner, capacity constraints are

$$\sum_{r \in R_l} x_{rl} \leq n_l \qquad \forall l \in L. \tag{4.9}$$

The related request constraints are slightly more difficult. An arbitrary set of related requests $D(i)$ for some index has the requirement that none of the requests $r \in D(i)$ can be sent to the same planner. Thus, for any given index i and planner $l \in L_{D(i)}$, at most one of the variables x_{rl} is allowed to be nonzero. This concept is modeled with the mathematical constraints

$$\sum_{r \in D(i)} x_{rl} \leq 1 \qquad \forall i \in \{1, \ldots, |D|\}, l \in L_{D(i)}. \tag{4.10}$$

Combining these constraints, the goal is to choose values for the variables x_{rl} and y_{rG} that solve

$$
\begin{aligned}
\max \quad & \sum_{r \in R} \sum_{G \in T_r} k_{rG} y_{rG} && && (4.11)\\
\text{st} \quad & \sum_{r \in R} \sum_{l \in L_r} c_l x_{rl} \leq b && \\
& \sum_{r \in R_l} x_{rl} \leq n_l && \forall l \in L \\
& \sum_{r \in D(i)} x_{rl} \leq 1 && \forall i \in \{1, \ldots, |D|\}, l \in L_{D(i)} \\
& \sum_{G \in T_r} y_{rG} = 1 && \forall r \in R \\
& y_{rG} \leq x_{rl} && \forall r \in R, G \in T_r, l \in G \\
& x_{rl} \in \{0,1\} && \forall r \in R, l \in L_r \\
& y_{rG} \in \{0,1\} && \forall r \in R, G \in T_r.
\end{aligned}
$$

Once this integer program is solved, we simply send a coordination request for the user request r to each planner l such that $x_{rl} = 1$.

4.2.3.4 Size of Solution Space

The number of decision variables in this composite variable formulation has the potential to grow very quickly at a rate of $O\left(|R|\ |L|^{N^{max}}\right)$for a fixed value of N^{max}, assuming every request is feasible on every planner. Fortunately, practically sized problems will generally have less than 20 planners and 1000 requests at any iteration, for which the exact solution to this integer program can be found efficiently using the built-in branch-and-bound techniques of the solver CPLEX [25]. In addition, requests generally are only feasible on a small subset of the planners considered in the coordinated system, which further decreases the number of variables required to solve a realistic problem. Even for those requests which are feasible on most of the individual planners, realistically the marginal return of increasing N^{max} by

1 decreases very quickly for larger values of N^{max} since the marginal increase in expected value decreases very quickly with the number of times that a request is sent (see analysis for comparison of various values of N^{max}).

4.2.3.5 Forward-Looking Component

The incorporation of the probability estimates P_s gives the formulation the ability to include knowledge about potential eventualities in the future when making decisions. Although other methods could have been used to address the forward-looking piece, they would not have nearly the same benefits. By estimating the probability that a request will be sent to a given planner during a single future estimation period, a very specific event has been constructed that lends itself to collecting observations. Thus, learning techniques can be used to fit this probability on-line, giving the formulation the ability to adapt to the actual state of the system over time. As a result, using these probabilities helps to retain much of the flexibility of DP at a much lower computational complexity. In fact, the structure of the formulation even resembles a form of on-line approximate optimistic policy iteration where the probability estimates are used to parametrize the future cost-to-go, being updated every few observations for improvement. Other methods for addressing this future cost-to-go, such as adding a linear penalty to requests that have more remaining feasible execution periods or iteratively performing an open-loop optimization for all future decisions, tend to lose the flexibility of adapting to the system, and might even become intractable very quickly (as would be the case with the open-loop optimization).

4.2.3.6 Heuristic Integer Programming Formulation

Even though the full formulation that has just been developed is tractable for many real-world scenarios, there might be some situations in which the problem size is much larger. For example, intelligence and reconnaissance missions might need to coordinate many assets when choosing a target [8], or future space operations might involve hundreds of small cube

satellites that collect data [13]. In order to handle these cases, a suboptimal formulation is presented that eliminates the exponential growth of the number of decision variables. To motivate this heuristic, consider a situation where it is assumed *apriori* that a coordination request associated with r would be accepted and completed if sent to planner l. Then the deterministic coordinated planning formulation of [1] can be slightly modified to include the budget constraint and request sending limits, resulting in the following formulation:

$$\begin{aligned}
\max \quad & \sum_{r\in R}\sum_{l\in L_r} v_{rl}x_{rl} && && (4.12)\\
\text{st} \quad & \sum_{r\in R}\sum_{l\in L_r} c_l x_{rl} \le b && \\
& \sum_{r\in R_l} x_{rl} \le n_l && \forall l \in L \\
& \sum_{l\in L} x_{rl} \le N^{max} && \forall r \in R \\
& \sum_{r\in D(i)} x_{rl} \le 1 && \forall i \in \{1,\ldots,|D|\}, l \in L_{D(i)} \\
& x_{rl} \in \{0,1\} && \forall r \in R, l \in L_r.
\end{aligned}$$

Rather than using the deterministic value v_{rl} for the coefficients, we could use the *marginal expected value* $\tilde{k}_{rl}$ *on planner* l that would be gained by sending a coordination request associated with r to planner l during this period, but not to any other planners. Conceptually, the marginal expected value is simply v_{rl} scaled by the net increase in completion probability of request r on a planner l gained by sending r to l for the next execution period t_{next} of l. Mathematically, this is calculated as

$$\tilde{k}_{rl} = v_{rl}\left[q\left(r,l,0\right) - q\left(r,l,1\right)\right] \tag{4.13}$$

where $q\left(r,l,x\right)$ is defined as in equation (4.1). Replacing each instance of v_{rl} with $\tilde{k}_{rl}$ in the

formulation 4.12 gives the *heuristic formulation*

$$
\begin{aligned}
\max \quad & \sum_{r \in R} \sum_{l \in L_r} \tilde{k}_{rl} x_{rl} && && (4.14)\\
\text{st} \quad & \sum_{r \in R} \sum_{l \in L_r} c_l x_{rl} \leq b \\
& \sum_{r \in R_l} x_{rl} \leq n_l && \forall l \in L \\
& \sum_{l \in L_r} x_{rl} \leq N^{max} && \forall r \in R \\
& \sum_{r \in D(i)} x_{rl} \leq 1 && \forall i \in \{1, \ldots, |D|\}, l \in L_{D(i)} \\
& x_{rl} \in \{0, 1\} && \forall r \in R, l \in L_r.
\end{aligned}
$$

Note that the formulation 4.12 is equivalent to 4.14 with the myopic estimators $P_s \equiv 0$, $P_a \equiv 1$, and $P_c \equiv 1$. As will be seen in Chapter 6, the heuristic formulation can be used to obtain results that are slightly lower than the full formulation, yet can be found much more quickly. In addition, the overhead of constructing composite variables, discussed in Section 6.1, is completely eliminated from this problem, allowing solvers such as CPLEX to stop searching after a set time limit and return the best current solution. This time limit gives a guarantee on efficiency, and evidence in Section 6.1 suggests that solutions produced are still optimal or close to optimal (just requiring more time for the solver to actually prove optimality).

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 5

Estimating Uncertainty

The integer programs of the previous chapter rely on the probability estimators P_s, P_a, and P_c defined in Section 4.2.3.2. Determining these functions can be difficult due to non-stationary levels of request saturation and the absence of prior data to analyze. These difficulties motivate the creation of an algorithm that can estimate probabilities by exploiting information about requests with similar specifications, as well as any prior knowledge that may be available about how the planners operate. This is the problem that we aim to solve in this chapter—how to efficiently and effectively utilize on-line observations of request completions/failures on each planner in combination with prior knowledge to approximate the functions P_s, P_a, and P_c.

We will be using a Bayesian approach to handle this problem. In Bayesian statistics, we use probability models concerning observed and unobserved quantities in order to make inferences from data. This is done by first creating a conditional likelihood distribution, which is the probability distribution for the response variables conditioned on the observation of a single data point. The conditional likelihood distribution incorporates the use of Bayesian parameters that are fitted by the data over time. The biggest difference between a frequentist approach and a Bayesian approach is that a Bayesian approach maintains a probability distribution that allows the process of parameter fitting to be done via Bayes' Rule on the

model parameters rather than minimizing a loss function to obtain a point estimate. In addition, a Bayesian approach allows for prior knowledge to be input into the model via a prior distribution on the parameters, which can be very helpful in small datasets [26]. This structure fits nicely into an on-line, nonstationary system where distributions change over time, because the presence of a Bayesian prior distribution and use of Bayes' rule allows regularization, prevents overfitting, and incorporates beliefs about the system (see [26] for more information).

5.1 Notation

Before explicitly defining any models, we would like to make a few comments regarding notation that will be used. Boldface is used to differentiate vectors and matrices from scalars, whether they are random or deterministic. Thus, x_i denotes the i^{th} component of a vector $\boldsymbol{x}$, whereas $\boldsymbol{x_i}$ denotes an entire vector with subscript i. This is done to follow the convention of using the notation $\boldsymbol{x_i}$ to denote distinct statistical observations.

To simplify notation and for ease of reading, the Bayesian likelihood expressions of random entities will be written as $p(\cdot)$ where the inside will be filled with lowercase symbols associated with a particular instance of a random variable or vector. Thus, $p(\boldsymbol{z})$ represents the probability density function (PDF) or probability mass function (PMF), whichever is appropriate, of a random vector $\boldsymbol{Z}$ evaluated at $\boldsymbol{z}$. Similarly, $p(\boldsymbol{z})p(w|\boldsymbol{z})$ would be used to express the product of the PDF (or PMF) of $\boldsymbol{Z}$ evaluated at $\boldsymbol{z}$ with the conditional PDF (or PMF) of W evaluated at w, given that $\boldsymbol{Z} = \boldsymbol{z}$. As is typical in Bayesian statistics, the same lowercase letters also are used to indicate associated random entities—shorthand meant to reduce the amount of notation that must be used. For example, in the statement $P(y = 1) = 0$, y clearly represents a random variable. However, in $p(y_i|\boldsymbol{x_i})$, the vector $(\boldsymbol{x_i}, y_i)$ is a specific observation. The context surrounding these symbols will define whether they represent random entities or specific observations.

5.2 Bayesian Coin Flip Model

5.2.1 Motivation

The Bayesian coin flip model (BCF) is used to make inferences about a dataset from a population where the response variable is binary, such as a coin flip that returns "heads" or "tails." In this model, we have a dataset $D = \{y_1, \ldots, y_m\}$ of m observations $y_i \in \{0, 1\}$, all of which are assumed to be conditionally independent and identically distributed according to a common probability mass function, so that an arbitrary observation y has a distribution

$$p(y|\theta) = \begin{cases} \theta & \text{if } y = 1 \\ 1 - \theta & \text{if } y = 0 \\ 0 & \text{otherwise} \end{cases}$$

with given parameter θ. It is assumed that the parameter θ is a random variable, which has some known prior probability distribution $p(\theta)$ before the data y_i are observed. As the data are obtained, the distribution for θ is updated using Bayes' Rule to obtain a posterior distribution for θ,

$$p(\theta|y_1, ..., y_m) = \frac{p(y_1, ..., y_m|\theta)p(\theta)}{\int_\theta p(y_1, ..., y_m|\theta)p(\theta)d\theta}.$$

where m is the number of observed data points [26].

5.2.2 Selection of the Prior Distribution

For the Bernoulli likelihood $p(y|\theta)$, if the prior $p(\theta)$ is a beta distribution with parameters α and β, then the posterior $p(\theta|y)$ is also a beta distribution with new parameters $\alpha + y$ and $\beta + 1 - y$. Thus, the beta distribution is referred to as the *conjugate prior* for the Bernoulli likelihood. Using conditional independence, this implies that the posterior after m observations $p(\theta|y_1, ..., y_m)$ is a beta distribution with parameters $\alpha + \sum_{i=1}^m y_i$ and $\beta + m - \sum_{i=1}^m y_i$, respectively. This information can easily be used to obtain a maximum a posteriori

(MAP) point estimate for θ, or the mode of the posterior distribution, for values of $\alpha, \beta \geq 1$ as

$$\hat{\theta}_{MAP} = \begin{cases} \frac{\alpha + \sum_{i=1}^{m} y_i - 1}{\alpha + \beta + m - 2} & \text{if } \alpha + \beta + m > 2 \\ 0.5 & \text{if } \alpha = \beta = 1,\ m = 0 \end{cases}. \tag{5.1}$$

For more information on these calculations, the reader is referred to [26].

The prior distribution has two major purposes. The first of these is to express beliefs about the values of the parameters in question using probability distributions. The second is to provide regularization against overfitting the posterior distribution to the observed data, which has the added benefit of maintaining positive probabilities for rare events that will not be able to influence the posterior distribution as often in a finite dataset. However, if the beliefs are minimal, it can be useful to create a prior distribution that is unimodal yet has a large variance. This allows the data to overcome the prior more quickly while still giving some regularization when small amounts of data are present, ensuring that the estimates for the parameters do not fluctuate too wildly. In the context of the BCF model, this could be done by selecting a mean $\mu \in (0, 1)$ for the beta prior distribution to be a reasonable value for θ, and then selecting the parameters α and β that would result in this mean while simultaneously maximizing the variance and maintaining a unimodal distribution. Mathematically, using μ, this maximization would be done by choosing prior parameters α and β that solve

$$\max \quad \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)} \tag{5.2}$$

$$\text{st} \quad \frac{\alpha}{\alpha + \beta} = \mu \tag{5.3}$$

$$\alpha \geq 1 \tag{5.4}$$

$$\beta \geq 1. \tag{5.5}$$

The objective 5.2 gives the variance of a beta distribution, constraint 5.3 ensures that the

mean of the beta distribution is the chosen prior belief μ, and constraints 5.4 and 5.5 ensure that the beta distribution is unimodal (modes where $p(\theta) = \infty$ are allowed as well). The constraints of this optimization were designed to give a sensible distribution—for this reason, the value of the objective function in this optimization will be referred to as the *maximum sensible variance* of a beta distribution for the remainder of this thesis. The solution to this optimization problem, proven in Appendix C, is given in Algorithm 5.1.

Algorithm 5.1 α and β Yielding the Maximum Sensible Variance of a Beta Distribution

1. Choose $\mu \in (0, 1)$ to be a reasonable approximation to the Bayesian parameter θ.
2. Set $\lambda = \frac{1-\mu}{\mu}$.
3. Set $\alpha = \max\left(1, \frac{1}{\lambda}\right)$.
4. Set $\beta = \lambda\alpha = \max(\lambda, 1)$.
5. Output α, β as the prior parameters for the beta distribution.

5.2.3 Application to Coordinated Planning

Every event for which probabilities are being predicted in the coordinated planning problem can be modeled with a binary set of outcomes—a request r is either sent or not sent by the CP, accepted or not accepted by a planner l (given that it was sent), and completed or not completed by l (given that it was sent and accepted). To estimate these probabilities using the BCF model, observations need to be separated according to whether they refer to a request being sent, accepted or completed, and also which planner was associated with the observation. Thus, a total of $3|L|$ groups of observations will be collected into the sets $SentData(l)$, $AcceptData(l)$, and $CompleteData(l)$, recording these datasets for all planners $l \in L$ (recall the definition of L in Table 4.1). Observations for this were constructed as follows:

- For a given planner l, an observation $y^{sent}_{(r,l,t)} \in SentData(l)$ where $y^{sent}_{(r,l,t)} = 1$ indicates that a coordination request associated with the request-planner-execution period triple

(r, l, t) was sent, otherwise $y^{sent}_{(r,l,t)} = 0$.

- An observation $y^{accept}_{(r,l,t)} \in AcceptData(l)$ takes the value $y^{accept}_{(r,l,t)} = 1$ if the coordination request was accepted for the triple (r, l, t), otherwise $y^{accept}_{(r,l,t)} = 0$ if the coordination request was sent but not accepted.

- An observation $y^{complete}_{(r,l,t)} \in CompleteData(l)$ takes the value $y^{complete}_{(r,l,t)} = 1$ if a coordination request was completed for the triple (r, l, t), otherwise $y^{complete}_{(r,l,t)} = 0$ if the coordination request was sent and accepted but not completed.

It is important to note that for the data in each of the $AcceptData(l)$ sets, only coordination requests that have already been sent are considered, and for $CompleteData(l)$, only coordination requests that have already been sent and accepted are considered.

These observations are recorded in the appropriate datasets as they are received over time. By design, the observations are separated into groups that do not depend upon the request or execution period; thus, the probability estimators P_s,P_a, and P_c only vary by planner, so that $P_s(\cdot, l, \cdot)$, $P_a(\cdot, l, \cdot)$, and $P_c(\cdot, l, \cdot)$ are all constant for a given l . Using this fact, each of the observations $y \in SentData(l)$ is assumed to come from a Bernoulli distribution with parameter $\theta = P_s(\cdot, l, \cdot)$, all $y \in AcceptData(l)$ come from a Bernoulli distribution with parameter $\theta = P_a(\cdot, l, \cdot)$, and all $y \in CompleteData(l)$ come from a Bernoulli distribution with parameter $\theta = P_c(\cdot, l, \cdot)$. The point estimates for the three probability estimators are then obtained using the MAP estimate 5.1 so that

$$P_s(\cdot, l, \cdot) = \frac{\alpha^{sent}_l + \sum_{y \in SentData(l)} y - 1}{\alpha^{sent}_l + \beta^{sent}_l + |SentData(l)| - 2}$$

$$P_a(\cdot, l, \cdot) = \frac{\alpha^{accept}_l + \sum_{y \in AcceptData(l)} y - 1}{\alpha^{accept}_l + \beta^{accept}_l + |AcceptData(l)| - 2}$$

$$P_c(\cdot, l, \cdot) = \frac{\alpha^{complete}_l + \sum_{y \in CompleteData(l)} y - 1}{\alpha^{complete}_l + \beta^{complete}_l + |CompleteData(l)| - 2}$$

where the prior parameter pairs $(\alpha_l^{sent}, \beta_l^{sent})$, $\left(\alpha_l^{accept}, \beta_l^{accept}\right)$, $\left(\alpha_l^{complete}, \beta_l^{complete}\right)$ are selected by creating appropriate prior beliefs for the associated P_s, P_a, or P_c and then using Algorithm 5.1 to find the desired parameters. These point estimates are calculated at the beginning of each coordinated planning iteration to obtain the appropriate probability estimators.

5.3 Bayesian Logistic Regression Model

5.3.1 Classical Logistic Regression

A common model in classical statistics used to extract a probability estimate is called *logistic regression*. This model estimates the probability $P(y = 1|\boldsymbol{x})$, where $y \in \{0, 1\}$ is a binary response variable which has a dependent relationship with its associated attribute vector $\boldsymbol{x}$. This is done by first observing m data points of the form $(\boldsymbol{x_i}, y_i)$ for all $i = 1, \ldots, m$, which are assumed to follow the same joint distribution as $(\boldsymbol{x}, y)$. The method then makes the simplifying assumption that, given $\boldsymbol{x_i}$, each y_i is conditionally independent of all other y_j, and that the log odds ratio for the common distribution y can be expressed as a linear combination of the observed attributes in $\boldsymbol{x}$ so that

$$\log\left(\frac{P(y = 1|\boldsymbol{x}, \boldsymbol{\lambda})}{P(y = 0|\boldsymbol{x}, \boldsymbol{\lambda})}\right) = \boldsymbol{\lambda}^T\boldsymbol{x}$$

for some parameter vector $\boldsymbol{\lambda}$. Rearranging this expression yields

$$P(y = 1|\boldsymbol{x}, \boldsymbol{\lambda}) = \frac{e^{\boldsymbol{\lambda}^T\boldsymbol{x}}}{1 + e^{\boldsymbol{\lambda}^T\boldsymbol{x}}} \tag{5.6}$$

which is readily recognizable as the logistic function of $\boldsymbol{\lambda}^T\boldsymbol{x}$. The regression coefficients $\boldsymbol{\lambda}$ are then estimated from the data via maximum likelihood estimation (see [27] for more details on ML estimation), where the log-likelihood is found by applying assumption (5.6) to all of the observations y_i which are conditionally independent given their respective attribute vectors

$\boldsymbol{x_i}$. Examining (5.6), we see that the probability $P(y = 1|\boldsymbol{x}, \boldsymbol{\lambda})$ is monotonically increasing in the argument of the associated logistic function $\boldsymbol{\lambda}^T\boldsymbol{x}$. This offers a very convenient modeling framework because the probability that needs to be estimated, $P(y = 1|\boldsymbol{x}, \boldsymbol{\lambda})$, generally increases or decreases monotonically in any given predictive attribute.

5.3.2 Bayesian Logistic Regression Background

For the coordinated planning problem, the classical version of logistic regression described in the previous section would be able to translate observed data into probability estimates, but it still does not provide a method for incorporating prior knowledge about the distribution $p(y|\boldsymbol{x})$. This is important in the coordinated planning problem as decisions will be made after having observed little data, suggesting that a Bayesian form might be appropriate.

Bayesian logistic regression (BLR) is not a new concept. The authors of [28] utilize BLR to analyze datasets for text categorization where the number of predictor variables is very high and even exceeds the number of observations. To do this, they implement Laplace prior distributions with zero mean to favor sparsity in the final estimates of the parameters. The work in [29] also employs a zero-mean Laplace prior for sparsity in analyzing neuronal spiking data from multiple electrodes. In [30], the authors expand the concept of BLR for binary response variables to include categorical response variables as well, giving the added capability of predicting probabilities or performing classification in any situation with finite outcomes. Instead of focusing on using the Bayesian model to induce sparsity, a major contribution of this thesis involves developing a method that can translate many simultaneous and even contradictory beliefs into a Gaussian prior distribution that encodes and quantifies the level of confidence in these beliefs.

For this model to work, we need to make a few assumptions. In order to control for differences between planners, we assume that we run separate BLRs for each planner. We assume that we can translate the *specifications* associated with the request-planner pairing into appropriate real-valued *attributes* such that request-planner pairings with similar attributes

result in similar probabilities obtained from P_s, P_a, or P_c, and request-planner pairings with identical attributes will have identical completion probabilities. It is important to note that, since we are performing logistic regression on separate planners, the chosen set of attributes may be planner-dependent. For example, a request having specifications including latitude, longitude, and altitude may be assigned to two separate planners, which we label 1 and 2. Suppose now that planner 1 has UAV assets, and planner 2 has satellite assets. The request-planner pairing of this request with planner 1 may translate these specifications into the attribute "distance from home base," whereas planner 2 may translate these specifications into the attribute "distance from the space track of the satellite asset to the latitude and longitude of the request." These transformations of specifications into attributes will be discussed in more detail at the end of the chapter.

5.3.3 Methodology

5.3.3.1 Conditional Likelihood and Probability Model

Assume that we have a dataset of m observations $D = \{(\boldsymbol{x_1}, y_1), \ldots, (\boldsymbol{x_m}, y_m)\}$. Within D, each $y_i \in \{0, 1\}$ is the response variable for the i^{th} observation, and each $\boldsymbol{x_i}$ is a vector of observed attributed related to the i^{th} observation. As we did in classical logistic regression, we begin by making the assumption that the log odds ratio for an arbitrary unknown observation y can be expressed as a linear combination of the attributes in $\boldsymbol{x}$ so that

$$\log\left(\frac{P(y=1|\boldsymbol{x},\boldsymbol{\lambda})}{P(y=0|\boldsymbol{x},\boldsymbol{\lambda})}\right) = \boldsymbol{\lambda}^T\boldsymbol{x}$$

for some vector of Bayesian parameters $\boldsymbol{\lambda}$. Rearranging this expression results in a generalized linear model with the logistic link function given in [28], implying

$$p(y|\boldsymbol{x},\boldsymbol{\lambda}) = \begin{cases} \text{logit}^{-1}(\boldsymbol{\lambda}^T\boldsymbol{x}) = \frac{e^{\boldsymbol{\lambda}^T\boldsymbol{x}}}{1+e^{\boldsymbol{\lambda}^T\boldsymbol{x}}} & \text{if } y = 1 \\ 1 - \text{logit}^{-1}(\boldsymbol{\lambda}^T\boldsymbol{x}) = \frac{1}{1+e^{\boldsymbol{\lambda}^T\boldsymbol{x}}} & \text{if } y = 0 \end{cases} \tag{5.7}$$

where $\text{logit}(p) = \log\left(\frac{p}{1-p}\right)$ is the logit link function (see [26] for details). We assume that all observations y_i are independent conditioned on $\boldsymbol{\lambda}, \boldsymbol{x_i}$. To make this model Bayesian, the parameters $\boldsymbol{\lambda}$ are assumed to be random, and as such are given a prior probability density $p(\boldsymbol{\lambda})$. In addition, it is assumed that $\boldsymbol{\lambda}$ is independent of all observations $\boldsymbol{x_i}$, i.e., $p(\boldsymbol{\lambda}|\boldsymbol{x_i}) = p(\boldsymbol{\lambda})$. In order to include an intercept term in linear portion of the model, it is assumed that the first attribute of every observation $\boldsymbol{x_i}$ is the constant 1 so that $x_{i1} = 1$. Assuming that we have a proper prior distribution $p(\boldsymbol{\lambda})$ for the parameters, i.e., that the integral $\int_{\boldsymbol{\lambda}} p(\boldsymbol{\lambda})\,\mathrm{d}\boldsymbol{\lambda}$ converges to 1, Bayes' rule applied to a single observation yields

$$\begin{aligned} p(\boldsymbol{\lambda}|\,\{(\boldsymbol{x}, y)\}) &= \frac{p(\boldsymbol{\lambda}, \boldsymbol{x}, y)}{p(\boldsymbol{x}, y)} \\ &= \frac{p(y|\boldsymbol{\lambda}, \boldsymbol{x})p(\boldsymbol{\lambda}, \boldsymbol{x})}{p(y|\boldsymbol{x})p(\boldsymbol{x})} \\ &= \frac{p(y|\boldsymbol{\lambda}, \boldsymbol{x})p(\boldsymbol{\lambda})p(\boldsymbol{x})}{p(y|\boldsymbol{x})p(\boldsymbol{x})} \\ &\propto p(y|\boldsymbol{\lambda}, \boldsymbol{x})p(\boldsymbol{\lambda}). \end{aligned} \tag{5.8}$$

This can be extended to all m observations in D, yielding a posterior distribution

$$p(\boldsymbol{\lambda}|D) \propto \left[\prod_{i=1}^{m} p(y|\boldsymbol{\lambda}, \boldsymbol{x_i})\right] p(\boldsymbol{\lambda})$$

which can be approximated via the *Metropolis-Hastings algorithm*, a form of *Markov Chain Monte-Carlo Simulation*. This algorithm produces a sample of approximately independent and identically distributed data points $\boldsymbol{\lambda_j}$ from the posterior distribution $p(\boldsymbol{\lambda}|D)$. For more information, the reader is referred to [26].

Once $p(\boldsymbol{\lambda}|D)$ has been obtained, the posterior predictive distribution $p(y|\boldsymbol{x})$ can be determined using either a point estimate of $\boldsymbol{\lambda}$ and the equation 5.7, or Monte-Carlo simulation of the integral

$$
\begin{aligned}
P(y=1|\boldsymbol{x},D) &= \int_{\boldsymbol{\lambda}} P(y=1|\boldsymbol{\lambda},\boldsymbol{x},D)\, p(\boldsymbol{\lambda}|D)\, \mathrm{d}\boldsymbol{\lambda} \\
&= \int_{\boldsymbol{\lambda}} P(y=1|\boldsymbol{\lambda},\boldsymbol{x})\, p(\boldsymbol{\lambda}|D)\, \mathrm{d}\boldsymbol{\lambda} \\
&= \int_{\boldsymbol{\lambda}} \frac{e^{\boldsymbol{\lambda}^T\boldsymbol{x}}}{1+e^{\boldsymbol{\lambda}^T\boldsymbol{x}}}\, p(\boldsymbol{\lambda}|D)\, \mathrm{d}\boldsymbol{\lambda}.
\end{aligned} \tag{5.9}
$$

5.3.3.2 Prior parameter Distribution

The Bayesian prior distribution was constructed to regularize observed data through beliefs about the probabilities being estimated so that obsolete data could be discarded over time. These beliefs were constructed using quantitative *belief statements* of the form:

"We estimate that the probability $P(y=1|\boldsymbol{x}^{\boldsymbol{k}})$ given some attributes $\boldsymbol{x}^{\boldsymbol{k}}$ is f_k, but we have c_k confidence that it is within the interval $[a_k, b_k]$."

In these statements, $0 < a_k < f_k < b_k < 1$ represent estimates about the conditional probability $P(y=1|\boldsymbol{x}^{\boldsymbol{k}})$, and $c_k \in (0,1)$ is some fractional level of confidence in the statement. For example, suppose the attributes for a request answer the following questions:

1. Is the target location or target type a high-value area for the planner to which it will be sent (in this case "high-value" would need to be explicitly defined)? *yes*

2. What is the cloud cover forecast over the target area as a fraction between 0 and 1 inclusive? *0.95*

Then one particular example of a confidence statement could take the following form, where the first attribute is the intercept term 1, the second attribute is "1" to represent the "yes" from the first question, and the final attribute is the answer to the second question:

"We estimate that the completion probability $P_c(r,l,t) = P\left(y_{(r,l,t)}^{complete} = 1|\boldsymbol{x}^{\boldsymbol{k}}\right)$ for a request-planner-execution period triple (r,l,t) having attributes $\boldsymbol{x}^{\boldsymbol{k}} = (1,1,0.95)$ is 0.8, but we have 90% confidence that it is within the interval $[.65, .9]$."

Suppose that we have n such confidence statements, so $k = 1, \ldots, n$. We will use the information from these statements to produce a prior distribution on $\boldsymbol{\lambda}$. We assume a multivariate normal distribution with independent component random variables for this prior on $\boldsymbol{\lambda}$ due to its unimodal structure, convenient parametrization in terms of a mean vector and covariance matrix, and property that linear combinations of its component random variables are still normal. While this may not induce sparsity in the results as explained in [29], the nice structure of the Gaussian will allow for injection of outside beliefs that may not be afforded by using a sparsity prior. In order to build this prior distribution, we will first assume that the j^{th} component of $\boldsymbol{\lambda}$ has a univariate normal distribution with mean μ_j and variance v_j for all $j = 1, \ldots, d$, where d is the length of $\boldsymbol{\lambda}$. Thus, $\boldsymbol{\lambda}$ must have a multivariate normal distribution with mean vector $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_d)$ and covariance matrix $\text{diag}(\boldsymbol{v})$, where $\boldsymbol{v} = (v_1, \ldots, v_d)$. Once we have constructed $\boldsymbol{\mu}$ and $\boldsymbol{v}$, we will have a completely well-defined model for Bayesian inference.

In order to figure out good values for $\boldsymbol{\mu}$ and $\boldsymbol{v}$, let us examine the form of our beliefs. We can interpret the values f_k to be estimates for the percent of observations with attribute vector $\boldsymbol{x^k}$ that would be expected to have the outcome $y = 1$ rather than $y = 0$. Thus, it makes sense to construct the mode $\boldsymbol{\mu}$ of our prior distribution on $\boldsymbol{\lambda}$ to maximize the pseudo-likelihood function $L(\boldsymbol{\lambda})$ where

$$L(\boldsymbol{\lambda}) = \prod_{k=1}^{n} \left(\frac{e^{\boldsymbol{\lambda}^T \boldsymbol{x^k}}}{1 + e^{\boldsymbol{\lambda}^T \boldsymbol{x^k}}} \right)^{f_k} \left(\frac{1}{1 + e^{\boldsymbol{\lambda}^T \boldsymbol{x^k}}} \right)^{(1-f_k)}. \tag{5.10}$$

We therefore set the values of $\boldsymbol{\mu}$ to be

$$\begin{aligned} \boldsymbol{\mu} = \boldsymbol{\lambda^*} &= \underset{\boldsymbol{\lambda}}{\text{argmax}} \quad L(\boldsymbol{\lambda}) \\ &= \underset{\boldsymbol{\lambda}}{\text{argmin}} - \sum_{k=1}^{n} \left[f_k \text{log} \left(\frac{e^{\boldsymbol{\lambda}^T \boldsymbol{x^k}}}{1 + e^{\boldsymbol{\lambda}^T \boldsymbol{x^k}}} \right) + (1 - f_k) \text{log} \left(\frac{1}{1 + e^{\boldsymbol{\lambda}^T \boldsymbol{x^k}}} \right) \right] \\ &= \underset{\boldsymbol{\lambda}}{\text{argmin}} \sum_{k=1}^{n} \left[f_k \text{log} \left(1 + e^{-\boldsymbol{\lambda}^T \boldsymbol{x^k}} \right) + (1 - f_k) \text{log} \left(1 + e^{\boldsymbol{\lambda}^T \boldsymbol{x^k}} \right) \right]. \end{aligned} \tag{5.11}$$

We now turn our attention to finding the values of the variance vector $\boldsymbol{v}$ using the confidence interval $[a_k, b_k]$ and the scalar c_k. Define θ^k in terms of the random vector $\boldsymbol{\lambda}$ such that

$$\theta^k = \text{logit}^{-1}(\boldsymbol{\lambda}^T \boldsymbol{x^k}) = \frac{e^{\boldsymbol{\lambda}^T \boldsymbol{x^k}}}{1 + e^{\boldsymbol{\lambda}^T \boldsymbol{x^k}}}.$$

Assume that each confidence interval is "centered" in the interval $[0, 1]$ in the sense that, for any given attribute vector $\boldsymbol{x^k}$, we have

$$P(\theta^k \leq a_k | \boldsymbol{x^k}) = P(\theta^k \geq b_k | \boldsymbol{x^k}). \tag{5.12}$$

where

$$P(\theta^k \in [a_k, b_k] \mid \boldsymbol{x^k}) = c_k.$$

Combining this assumption with the confidence statement, we see that

$$P(\theta^k \leq a_k | \boldsymbol{x^k}) = P(\theta^k \geq b_k | \boldsymbol{x^k}) = \tfrac{1-c_k}{2}.$$

Manipulating this expression, we have

$$\begin{aligned} \frac{1 - c_k}{2} &= P(\theta^k \leq a_k | \boldsymbol{x^k}) \\ &= P(\text{logit}(\theta^k) \leq \text{logit}(a_k) | \boldsymbol{x^k}) \\ &= P(\boldsymbol{\lambda}^T \boldsymbol{x^k} \leq \text{logit}(a_k) | \boldsymbol{x^k}) \\ &= \phi \left(\frac{\text{logit}(a_k) - \boldsymbol{\mu}^T \boldsymbol{x^k}}{\sqrt{\boldsymbol{x^k}^T \text{diag}(\boldsymbol{v}) \boldsymbol{x^k}}} \right) \end{aligned} \tag{5.13}$$

where $\phi(\cdot)$ denotes the standard normal cumulative distribution function. In this expression, the second equality holds by monotonicity of the logit$(\cdot)$ function, and the third equality holds by definition of θ^k. The final equality holds because each of the λ_j are normal random variables with mean μ_j and variance v_j, independent of other λ_i or $\boldsymbol{x^k}$, implying that $\boldsymbol{\lambda}^T \boldsymbol{x^k}$

is normal with mean $\boldsymbol{\mu}^T\boldsymbol{x^k}$ and standard deviation $\sqrt{\boldsymbol{x^k}^T\text{diag}(\boldsymbol{v})\boldsymbol{x^k}}$. By similar reasoning, we have

$$1 - \frac{1-c_k}{2} = \frac{1+c_k}{2} = P(\theta^k \leq b_k|\boldsymbol{x^k}) = \phi\left(\frac{\text{logit}(b_k) - \boldsymbol{\mu}^T\boldsymbol{x^k}}{\sqrt{\boldsymbol{x^k}^T\text{diag}(\boldsymbol{v})\boldsymbol{x^k}}}\right). \tag{5.14}$$

Unfortunately, it is very possible that some of our belief statements are unintentionally in direct conflict with each other or with some of the modeling assumptions used to derive the equations (5.13) and (5.14), implying that existence of a solution $\boldsymbol{v}$ to the system of equations (5.13) and (5.14) for all $k = 1, \ldots, n$ is not guaranteed. Thus, our objective in selecting $\boldsymbol{v}$ will be to choose parameters that either solve the system of equations (5.13) and (5.14) for all $k = 1, \ldots, n$ if our belief statements are appropriate, or are "close" in some sense to solving the system if no solution exists. To accomplish this goal, we note that having already selected the mode $\boldsymbol{\mu}$ of our prior distribution, conflicting belief statements make us less confident that the mode should actually be located at $\boldsymbol{\mu}$. This in turn implies that the confidence levels c_k given in the belief statements should actually be interpreted as upper bounds on our confidence about the location of the mode $\boldsymbol{\mu}$. In terms of probability statements, we interpret this bound to mean

$$P(\theta^k \in [a_k, b_k] \mid \boldsymbol{x^k}) \leq c_k. \tag{5.15}$$

We choose to enforce this bound by constraining feasible values of $\boldsymbol{v}$ to satisfy

$$\begin{aligned} P(\theta^k \leq a_k|\boldsymbol{x^k}) &= \phi\left(\frac{\text{logit}(a_k) - \boldsymbol{\mu}^T\boldsymbol{x^k}}{\sqrt{\boldsymbol{x^k}^T\text{diag}(\boldsymbol{v})\boldsymbol{x^k}}}\right) \geq \frac{1-c_k}{2} \\ P(\theta^k \geq b_k|\boldsymbol{x^k}) &= 1 - \phi\left(\frac{\text{logit}(b_k) - \boldsymbol{\mu}^T\boldsymbol{x^k}}{\sqrt{\boldsymbol{x^k}^T\text{diag}(\boldsymbol{v})\boldsymbol{x^k}}}\right) \geq \frac{1-c_k}{2}. \end{aligned} \tag{5.16}$$

We see that the first line of these constraints involving a_k will be automatically satisfied for any $a_k, \boldsymbol{\mu}$ such that

$$\text{logit}(a_k) - \boldsymbol{\mu}^T \boldsymbol{x^k} \geq 0.$$

This property is a direct result of the symmetry of the normal distribution about its mean $\boldsymbol{\mu}^T \boldsymbol{x^k}$ combined with the fact that $c_k \geq 0$. Similarly, the second line of constraints will be automatically satisfied for any $b_k, \boldsymbol{\mu}$ such that

$$\text{logit}(b_k) - \boldsymbol{\mu}^T \boldsymbol{x^k} \leq 0.$$

Since constraints that are automatically satisfied for all $\boldsymbol{v} > \mathbf{0}$ can be ignored, we can rewrite (5.16) in the equivalent linear form

$$\begin{aligned} \boldsymbol{x^k}^T \text{diag}(\boldsymbol{v}) \boldsymbol{x^k} &\geq \left(\frac{\text{logit}(a_k) - \boldsymbol{\mu}^T \boldsymbol{x^k}}{\phi^{-1}\left(\frac{1-c_k}{2}\right)} \right)^2 \quad \forall k : \text{logit}(a_k) - \boldsymbol{\mu}^T \boldsymbol{x^k} < 0 \\ \boldsymbol{x^k}^T \text{diag}(\boldsymbol{v}) \boldsymbol{x^k} &\geq \left(\frac{\text{logit}(b_k) - \boldsymbol{\mu}^T \boldsymbol{x^k}}{\phi^{-1}\left(\frac{1+c_k}{2}\right)} \right)^2 \quad \forall k : \text{logit}(b_k) - \boldsymbol{\mu}^T \boldsymbol{x^k} > 0. \end{aligned} \tag{5.17}$$

Thus, a vector $\boldsymbol{v}$ satisfies (5.17) if and only if it also satisfies (5.16). Also, any vector $\boldsymbol{v}$ satisfying (5.16) has the property that

$$\begin{aligned} P(\theta^k \in [a_k, b_k] \mid \boldsymbol{x^k}) &= 1 - P(\theta^k \notin [a_k, b_k] \mid \boldsymbol{x^k}) \\ &= 1 - P(\theta^k \leq a_k | \boldsymbol{x^k}) - P(\theta^k \geq b_k | \boldsymbol{x^k}) \\ &\leq 1 - \frac{1 - c_k}{2} - \frac{1 - c_k}{2} \\ &= c_k \end{aligned}$$

meaning it also satisfies (5.15). By limiting our solution space with the constraints (5.16), we retain a notion of confidence interval centrality similar to that given by assumption (5.12). We also want to reduce our set of potential selections for $\boldsymbol{v}$ to be those in which individual components v_i each exhibit a similar amount of uncertainty relative to the respective Bayesian

parameters λ_i that they describe. We do this to ensure that we are never overly confident in one parameter, which helps to prevent over-fitting of our prior distribution to possibly incorrect beliefs. We model this restriction on $\boldsymbol{v}$ by imposing the constraints

$$\frac{v_i}{\gamma_i^2} \leq r^2 \frac{v_j}{\gamma_j^2}, \quad \forall i, j \in \{1, \ldots, d\} : i \neq j \tag{5.18}$$

where γ_i represents the mean value of the i^{th} component of the vectors $\boldsymbol{x^k}$, and $r \geq 1$ is a constant allowing control over how much spread we are allowing between the elements of $\boldsymbol{v}$ (a value for r that is closer to 1 represents less spread). We divide each v_i by γ_i^2 in order to correct between different scales on the units being used for each attribute of $\boldsymbol{x^k}$. As a final method for protecting against poorly written confidence statements, we impose the vector constraint

$$0 \leq v_{min} \leq v_i \leq v_{max} \quad \forall i \in \{1, \ldots, d\} \tag{5.19}$$

on the set of feasible values for $\boldsymbol{v}$, where v_{min} and v_{max} are constants giving a region of satisfactory values for $\boldsymbol{v}$. Once we have identified all feasible $\boldsymbol{v}$ satisfying our bounds in (5.16) through (5.19), our goal is to search this feasible set to find the one giving the "most" information pertaining to our beliefs. To complete this objective, we note that

$$\text{var}\left(\text{logit}(\theta^k) \mid \boldsymbol{x^k}\right) = \boldsymbol{x^k}^T \text{diag}(\boldsymbol{v}) \boldsymbol{x^k}. \tag{5.20}$$

Qualitatively, smaller magnitudes of $\text{var}\left(\text{logit}(\theta^k) \mid \boldsymbol{x^k}\right)$ for a particular k indicate that we have more information pertaining to requests with parameters $\boldsymbol{x^k}$. Different magnitudes of the vector $\boldsymbol{x^k}$ can inflate or deflate the above variance, though, so we divide (5.20) by $\|\boldsymbol{x^k}\|_2^2$ as a form of normalization, allowing us to compare on an absolute scale a measure of the amount of variability that $\boldsymbol{v}$ creates in the k^{th} belief statement. Since lower belief variability indicates more information, we aim to choose the vector $\boldsymbol{v}$ to minimize the sum over all k of

the normalized variances given by the expression

$$\frac{1}{\|\boldsymbol{x^k}\|_2^2}\boldsymbol{x^k}^T \text{diag}(\boldsymbol{v})\boldsymbol{x^k}. \tag{5.21}$$

Combining (5.16) through (5.21), we select $\boldsymbol{v}$ via the linear program (LP)

$$\min_{\boldsymbol{v}} \quad \sum_{k=1}^{n} \frac{1}{\|\boldsymbol{x^k}\|_2^2}\boldsymbol{x^k}^T \text{diag}(\boldsymbol{v})\boldsymbol{x^k} \tag{5.22}$$

$$\text{st} \quad \boldsymbol{x^k}^T \text{diag}(\boldsymbol{v})\boldsymbol{x^k} \geq \alpha_k^2 \qquad \forall k : \text{logit}(a_k) - \boldsymbol{\mu}^T\boldsymbol{x^k} < 0 \tag{5.23}$$

$$\boldsymbol{x^k}^T \text{diag}(\boldsymbol{v})\boldsymbol{x^k} \geq \beta_k^2 \qquad \forall k : \text{logit}(b_k) - \boldsymbol{\mu}^T\boldsymbol{x^k} > 0 \tag{5.24}$$

$$\frac{v_i}{\gamma_i^2} \leq r^2\frac{v_j}{\gamma_j^2} \qquad \forall i, j \in \{1, \ldots, d\} : i \neq j \tag{5.25}$$

$$0 \leq v_{min} \leq v_i \leq v_{max} \qquad \forall i \in \{1, \ldots, d\} \tag{5.26}$$

where the constants α_k, β_k are given by

$$\alpha_k = \frac{\text{logit}(a_k) - \boldsymbol{\mu}^T\boldsymbol{x^k}}{\phi^{-1}\left(\frac{1-c_k}{2}\right)}$$
$$\beta_k = \frac{\text{logit}(b_k) - \boldsymbol{\mu}^T\boldsymbol{x^k}}{\phi^{-1}\left(\frac{1+c_k}{2}\right)}.$$

We note that the box constraints (5.26) form a closed, bounded set. This implies that if the feasible set of this LP is nonempty, then it must form a bounded polytope, guaranteeing the existence of an optimal solution. By construction of this LP, as long as v_{max} is sufficiently large, then this LP has a nonempty feasible set, and thus an optimal solution v^* (see proof in Appendix D). By combining the maximum likelihood approach for constructing $\boldsymbol{\mu}$ with this linear program for constructing $\boldsymbol{v}$, we can translate any positive number of qualitative belief statements into an informative prior distribution on the Bayesian parameters.

5.3.4 Application to Coordinated Planning

As in the BCF model, every event for which probabilities are being predicted in the coordinated planning problem can be modeled with a binary set of outcomes—a request r is either sent or not sent, accepted or not accepted by a planner l (given that it was sent), and completed or not completed by l (given that it was sent and accepted). Thus, the same datasets $SentData(l)$, $AcceptData(l)$, and $CompleteData(l)$, are collected for the Bernoulli response variables on all planners l. In addition to these, the attribute sets $SentAttributes(l)$, $AcceptAttributes(l)$, and $CompleteAttributes(l)$ are also recorded, where any attribute vector $\boldsymbol{x_{(r,l,e)}}$ in one of these sets is associated with the response $y_{(r,l,e)}$ in the corresponding dataset, i.e., $SentAttributes(l)$ corresponds with $SentData(l)$, $AcceptAttributes(l)$ with $AcceptData(l)$, and $CompleteAttributes(l)$ with $CompleteData(l)$.

The observations are recorded in the appropriate sets as they are received over time. Unlike the BCF model, the probability estimators P_s, P_a, and P_c can be dependent on requests, planners, and execution periods, not just planners. Three separate BLRs are run for each planner, one for each probability type. The point estimates of $\boldsymbol{\lambda}$ for the three probability estimators are then obtained by finding the mean of the samples obtained via posterior simulation (using the package MCMCpack [31] in R), so that

$$
\begin{aligned}
P_s(r,l,t) &= \frac{e^z}{1+e^z} \text{ where } z = \left[\boldsymbol{\lambda}^{\boldsymbol{sent}}(l)\right]^T \boldsymbol{x}^{\boldsymbol{sent}}_{\boldsymbol{(r,l,t)}} \\
P_a(r,l,t) &= \frac{e^z}{1+e^z} \text{ where } z = \left[\boldsymbol{\lambda}^{\boldsymbol{accept}}(l)\right]^T \boldsymbol{x}^{\boldsymbol{accept}}_{\boldsymbol{(r,l,t)}} \\
P_c(r,l,t) &= \frac{e^z}{1+e^z} \text{ where } z = \left[\boldsymbol{\lambda}^{\boldsymbol{complete}}(l)\right]^T \boldsymbol{x}^{\boldsymbol{complete}}_{\boldsymbol{(r,l,t)}}
\end{aligned}
$$

with $\boldsymbol{\lambda}^{\boldsymbol{sent}}(l), \boldsymbol{\lambda}^{\boldsymbol{accept}}(l), \boldsymbol{\lambda}^{\boldsymbol{complete}}(l)$ being the point estimates obtained from the appropriate BLRs, and $\boldsymbol{x}^{\boldsymbol{sent}}_{\boldsymbol{(r,l,t)}}, \boldsymbol{x}^{\boldsymbol{accept}}_{\boldsymbol{(r,l,t)}}, \boldsymbol{x}^{\boldsymbol{complete}}_{\boldsymbol{(r,l,t)}}$ being the appropriate attribute vectors of the triple (r,l,t) for the given probability. These point estimates are calculated at the beginning of each coordinated planning iteration to obtain the appropriate probability estimators.

5.3.5 Attributes for Bayesian Logistic Regression

Having developed the probability estimation methods, a few suggested attributes are defined for use in the BLRs. These attributes are designed to be on a similar scale for ease of constructing prior beliefs and to minimize the chances of unnecessary computational errors. In fact, most are in the interval $[0, M^{scale}]$, where $M^{scale} = 10$ for this project. Also, recall throughout this discussion that acceptance is guaranteed for non-informative planners, i.e., $P_a \equiv 1$. However, completion probabilities are much lower for non-informative planners since rejection information is never given to the CP. We suggest that the reader review the notation defined in Chapter 4 before reading this section.

5.3.5.1 Probability of Request Being Sent by the Coordination Planner

We run a single BLR incorporating all past observations of sent and unsent pairings to estimate the probability that request r will be sent to planner l during a future feasible execution period t (see Chapter 5 for an explanation of how these observations are collected over time). As such, it is important to identify attributes that represent quantities affecting the decisions of the CP, not the decisions of the individual planners. To do this, we selected the following attributes for a pairing (r, l) during execution period t.

1. **Number of Related Requests:** This attribute is simply the number of requests related to r. If r is not in a set of related requests, this attribute is zero. If r is in a set of related requests $D(i)$, this attribute is $|D(i)| - 1$. Since we are only considering unrelated requests and dual collects in our analysis, this attribute is always zero (if r is unrelated) or one (if r is in a dual collect). Although no related requests were included in the simulation, this attribute is recorded as a potential suggestion for real-world implementations.

2. **Expected Pairing Value:** This attribute is simply $v_{rl} \times P_a(r, l, t) \times P_c(r, l, t)$, i.e., the value of the request-planner pairing (r, l) multiplied by the probability $P_a(r, l, t)$

that r would be accepted by l if sent, and the probability $P_c(r, l, t)$ that r would be completed by l if sent and accepted. This attribute represents how valuable (r, l) would be if period t was the only chance that we had to sent the request to any planner. By including this attribute we are able to give the regression a sense of how valuable the pairing is in the current execution period as compared to other execution periods.

3. **Current Request Value:** This is just the expected value of all accepted assignments of request r to any planner in the system, including l, during past execution periods ($P_a(r, l, t) \equiv 1$ for accepted requests and non-informative planners). This represents the expected value that would be obtained if we never sent any more coordination requests associated with r to any planner. We can calculate this attribute using Algorithm 4.1 with input request r, all $x_{rl} = 0$, and replacing P_s to be $P_s \equiv 0$ before calculating $q(r, l, x_{rl} = 0)$.

4. **Current Planner Value:** This attribute is calculated as

$$v(r, l) \times \left[1 - \prod_{t \in S_{rl}} (1 - P_a(r, l, t) \times P_c(r, l, t))\right].$$

This is the expected value of all accepted assignments of request r to planner l during past execution periods for which we do not yet have completion information ($P_a(r, l, t) \equiv 1$ for accepted requests and non-informative planners). For our analysis, this value will always be either zero if r has never been sent to l, or the expected value for (r, l) during the period immediately prior to t. This is because we assume that all planners give completion information immediately following the end of their execution periods, which, when combined with the fact that a user request r is removed from the queue when any single coordination request associated with r is completed, implies that we can have at most one single assignment of (r, l) that is open without completion information.

5. **Nonlinear Extensions:** We also include the square-root and squared values of each of these attributes to give the regression more modeling flexibility (for a total of 12 attributes). We do not need to worry about the correlations introduced by adding these attributes as we are only looking to improve predictive capability of the regression output, and the proper prior parameter distribution ensures that highly correlated attributes do not make the problem unstable.

Using these attributes, the linear programming based prior distributions were constructed as follows. We began by generating 200 random attribute vectors, each of which would be utilized in a separate belief statement as required in Section 5.3.3.2. However, since we do not have any strong apriori beliefs specific to individual attributes about the probability that a request would be sent, each of the belief statements is given the same values $f = 0.2$, $a = 0.1$, $b = 0.8$, $c = 0.9$. These values represent our apriori belief that significantly less than half of the request-planner pairings will be sent, but at the same time the value of $f = 0.2$ can adapt to values closer to 0 or 0.5 easily. The values of our interval $[a, b]$ and our confidence c are intentionally set so that our interval of possibilities is wide since we are not sure exactly where this probability lies but are confident that it is not too extreme.

5.3.5.2 Probability of Request Acceptance by Planner

Different planners may have very different planning processes. As a result, separate planners most likely do not value attributes in the same way. For example, two satellites on separate planners may be designed quite differently. The first satellite may have a very limited capacity for servicing requests due to limited on-board memory, whereas the second satellite may not have this problem. The planner for the first satellite may place more weight on the type of request in this case in order to complete those requests that most align with its objectives, whereas the planner for the second satellite may place more weight on having a small viewing angle away from nadir in order to service as many requests as possible. In order to control for effects such as these, we run a separate BLR for each planner to estimate

the probability that a coordination request would be accepted given that it was sent to the planner. Some of the attributes are specific to the type of assets on the planners and as such are labeled either "UAV Planners Only" or "Satellite Planners Only." Otherwise, the attributes apply to all planners, and are designed to quantify real-world trade-offs that must be considered for a pairing (r, l) during execution period t. We note that these estimates only apply to informative planners as non-informative planners do not tell the CP when they accept or reject a request.

1. **Normalized Great-Circle Distance (UAV Planners Only):** This attribute represents how far away a request is from the UAV home base as a percent of the maximum range of all UAVs operating on the planner. To calculate this attribute, we first calculate the great-circle distance $Dist_{GC}$ between the UAV home base and the request location using the algorithm described in [21, 20, 2]. We then divide this quantity by the maximum range $maxRange$ of all UAVs on the planner, which is calculated as the product of the maximum UAV speed and the UAV endurance, converting units as necessary so that the final attribute quantity is dimensionless. Thus, the value of this attribute is

$$normGCD = \frac{Dist_{GC}}{maxRange}.$$

   For our analysis, each planner has the same specifications for its UAVs. We include this attribute because it should be more difficult to find UAV routes for locations far from the home base.

2. **Normalized Viewing Angle (Satellite Planners Only):** This attribute represents the mean viewing angle, in radians, required for the satellite on planner l to service request r during execution period t. It is calculated by using J2 secular theory to propagate the satellite orbit over the length of the execution period in time steps of 60 seconds [2]. At each time step, if the request is feasible on the satellite, we calculate the absolute value of the viewing angle away from nadir that would be required to

service the request using the Earth Centered-Earth Fixed (ECEF) coordinates of the satellite and the request. We then find the mean of all such feasible angles, which yields the value of this attribute $normViewAngle$. We include this as an attribute because algorithms for taskable satellites must design their mission plans to include the time required to rotate a satellite sensor.

3. **Request Observation Duration as a Fraction of Feasible Time:** This attribute is the observation duration $Duration_r$ required for request r divided by the total time T_{feas} that the request would be feasible during the execution period t of planner l, giving a value of

$$durFrac = \frac{Duration_r}{T_{feas}}.$$

The amount of time feasible is defined to be the total length of time that the opportunity finder determines r would be feasible on l during execution period t. We include this as an attribute because if the request observation duration is a high percentage of the time that the request is feasible, there is little flexibility for the planner to fit the request into its schedule, thus decreasing the probability that it will be accepted.

4. **Infeasible Time as a Fraction of Execution Period Length:** This attribute is the amount of time that the request is *not* feasible T_{infeas} during execution period t, divided by the length T_{length} of execution period t on planner l. Thus, the value of this attribute is

$$infeasFrac = \frac{T_{infeas}}{T_{length}} = \frac{T_{length} - T_{feas}}{T_{length}} = 1 - \frac{T_{feas}}{T_{length}}$$

where T_{feas} is the length of time that the request is feasible during execution period t, which is calculated in the same manner as for the previous attribute. The execution period length T_{length} is assumed to be known for each planner. This attribute is selected as it provides a measure of the flexibility that the planner has for scheduling this request—a higher value means less flexibility and hence a higher chance of rejection. We choose to use time infeasible rather than time feasible for this attribute so that all

three attributes have similar directional effects on the acceptance probability. In other words, we have designed the model in such a way that we expect an increase in any of the aforementioned attributes to decrease the acceptance probability, and a decrease in any attribute to increase the acceptance probability.

5. **Nonlinear Extensions:** As we did when estimating the probability that a request would be sent, we include the squares and square-roots of the above four attributes to give the regression more flexibility.

6. **Target Types:** Since various assets might favor certain target types over others, or may fail more often when trying to complete requests with certain target types, it may be useful to include binary indicator variables for any target types considered. However, for our analysis, this attribute was assumed to be irrelevant in the simulation and therefore was not used.

Using these attributes, the linear programming based prior distributions were constructed as follows. We began by generating 200 random attribute vectors, each of which would be utilized in a separate belief statement as required in Section 5.3.3.2. For each of these attribute vectors, we created our belief statement quantities to reflect the idea that larger values of $normGCD$, $normViewAngle$, $durFrac$, and $infeasFrac$ should decrease the probability of acceptance. However, since it is not known how much of an effect these attributes will have, the values for f need to remain relatively similar, and likely at or below 0.5 since most requests will not be accepted. To reflect these beliefs, first the quantity $score = normGCD + durFrac + infeasFrac$ or $score = (2/\pi)\, normViewAngle + durFrac + infeasFrac$ was calculated for UAV planners or satellite planners, respectively. The value f was set so that if $score > 0.75$, then $f = 0.3$; if $0.25 < score \leq 0.75$, then $f = 0.4$, otherwise $f = 0.5$. The interval $[a, b]$ was set to be relatively wide with a low confidence $c = 0.5$ in order to reflect the fact that our estimate is very imprecise, setting $a = \frac{f}{2}$ and $b = \frac{1+f}{2}$, which are the midpoints between 0 to f , and f to 1 for a or b, respectively. We assume that we do

not have prior knowledge of which target types are preferred by each planner, so we do not differentiate our prior beliefs by target type.

5.3.5.3 Probability of Request Completion by Planner

We also run a separate BLR for each planner l to estimate the probability that a coordination request would be completed, given that it was sent by the CP and accepted by l. Separate regressions are run for each planner for similar reasons identified in the previous section (different planners may have different probabilities of being down for maintenance, cloud cover may be different in their respective geographical regions, etc.). We utilize the same set of attributes for these regressions that we used for estimating the probability of acceptance as they still provide a decent representation of the characteristics of pairings.

The linear programming based prior distributions, though, were constructed using different beliefs than those used for the the estimation of acceptance probability as being accepted implies a high chance of completion. Specifically, we kept the same process to generate the prior distributions, except that we changed the values of f that we used. We still wanted our belief statement quantities to reflect the idea that probabilities should have similar quantities, with larger values of $normGCD$, $normViewAngle$, $durFrac$, and $infeasFrac$ decreasing the probability of acceptance. The value f was set so that if $score > 0.75$, then $f = 0.8$; if $0.25 < score \leq 0.75$, then $f = 0.9$, otherwise $f = 0.95$ (where $score$ was calculated in the same way as for the probability of acceptance). This scheme reflects the idea that once a request has been accepted, it has a very high chance of being completed. If the planner was not informative, then the value f was set so that if $score > 0.75$, then $f = 0.8 \times 0.3$; if $0.25 < score \leq 0.75$, then $f = 0.9 \times 0.4$, otherwise $f = 0.95 \times 0.5$—simply a merger of the acceptance and completion probabilities for informative planners via the multiplication rule for probabilities. The values of $a,\ b,\ c$ were calculated using the same process as used for constructing acceptance priors.

5.4 Discussion

5.4.1 Empirical Performance

Prior to implementing these estimation methods in practice, it is important to gain some insight into their behaviors. The BLR method takes advantage of problem-specific information in the form of prior beliefs and observation attributes, whereas the BCF method inputs very little problem-specific information. As such, it is important to understand the relative performances of these two methods when the prior beliefs are accurate/inaccurate, as well as when the observation attributes are relevant/irrelevant. Figure 5-1 summarizes the performance of the methods for a scenario in which the prior belief statements were accurate, and Figure 5-2 summarizes the performance in a different scenario where the prior belief statements were inaccurate. Both scenarios were simulated using four different observation attributes in the interval $[0, 1]$ in addition to the constant attribute, with Bernoulli outcomes being drawn randomly from a conditional likelihood distribution $p(y|\boldsymbol{x})$ defined by these attributes. After 10, 25, 50, 100, and 200 observations were obtained, both methods were used to estimate $P(y = 1|\boldsymbol{x})$ for 10,000 initializations $\boldsymbol{x}$ distributed uniformly in the space of possible attributes. The root mean square error (RMSE) of these estimates as compared to the actual probabilities from the likelihood model $p(y|\boldsymbol{x})$ is plotted on the vertical axis of the figures. The figures show results compared to the RMSE obtained by estimating $P(y = 1|\boldsymbol{x})$ to be the "benchmark" unconditional probability $P(y = 1)$, obtained by simulation using the conditional likelihood distribution $p(y|\boldsymbol{x})$, for all $\boldsymbol{x}$.

The left side of each figure shows the error in a situation where $p(y|\boldsymbol{x})$ has a high dependence on the attributes $\boldsymbol{x}$, while the right shows a situation where the likelihood $p(y|\boldsymbol{x})$ has a very small dependence on $\boldsymbol{x}$. These figures suggest that the BLR has the best long-run performance, regardless of the scenario. However, because the prior distributions were purposefully constructed to have high variability, they conform to the data very quickly. Thus, when the number of observations is small, the extra complexity of the BLR can cause

error due to overfitting. While this issue could be addressed by creating stronger confidence statements in the construction of the prior, this approach is not recommended if the beliefs are weak. The extra error causes the simple BCF model to outperform BLR when attributes are not as relevant and the number of observations is small. The prior beliefs also affect the initial errors, but are quickly overcome by the data as the confidence statements used in constructing these priors were weak.

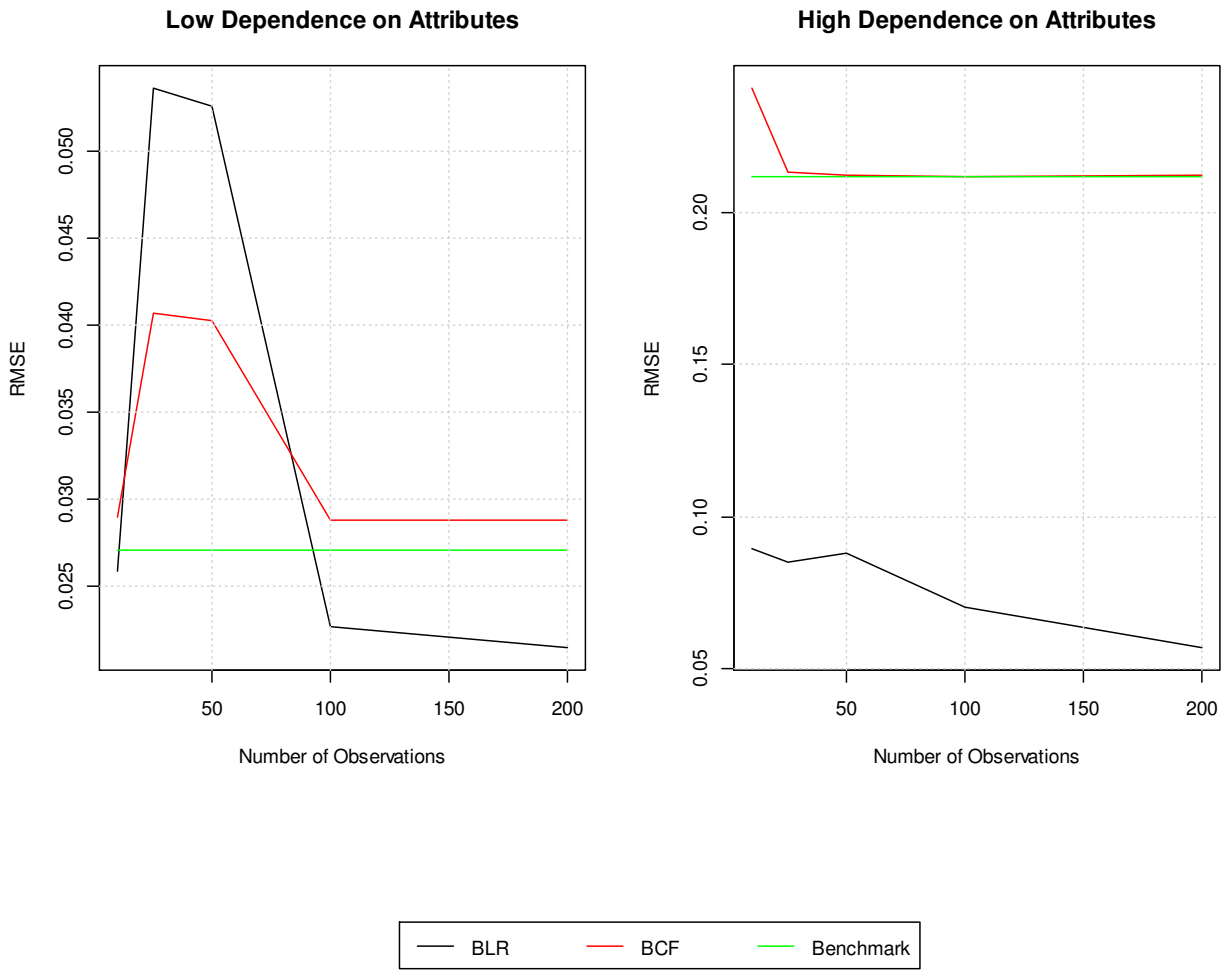

Figure 5-1: Performance Using Accurate Belief Statements

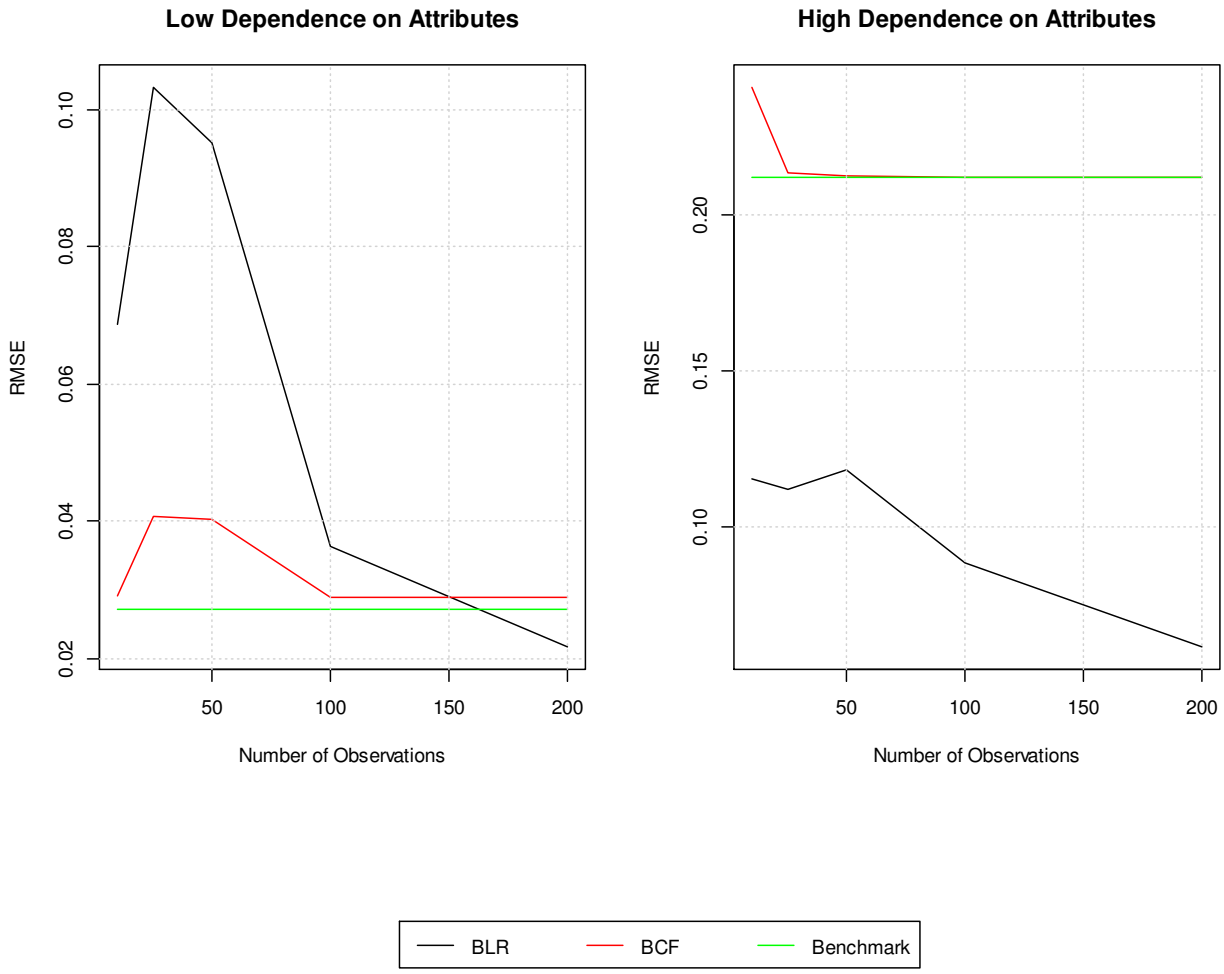

Figure 5-2: Performance Using Inaccurate Belief Statements

5.4.2 General Comparisons

Both of the methods described in this chapter have their relative merits for estimating probabilities. The main benefits of using the BCF model involve its simplicity. In this model, very little data is needed before the posterior converges, as shown in Figures 5-1 and 5-2. The posterior MAP estimates are also very easy to obtain as they exist in closed form. Since the prior is a beta distribution, it is very easy to mold it to reflect beliefs about the probabilities. One of the biggest advantages of this method is in coordinated planning scenarios where certain types of request specifications are encountered more frequently than others. In this case, attribute-based methods can have added error due to the lack of rare observations, which is not an issue for the BCF model as no distinctions are made between requests or execution periods. Attribute-based methods can also be more prone to overfitting and therefore tend to have higher errors in small datasets, which is especially a problem when the attributes that are recorded do not have much relevance to the actual probabilities. However, the simplicity of the BCF model also has the potential to create poor performance. By neglecting to differentiate between request types, the BCF model will not be able to determine if certain request specifications are more favorable to planners than others, which may be quite valuable.

In contrast, the BLR model adds complexity that has the ability to recognize such patterns in the specifications of requests that are favored by planners. This is especially true if highly relevant, specialized attributes are available with a large dataset. While the prior distribution provides some protection against overfitting to small datasets and lack of observations with rare attributes, the method is still susceptible to such errors as seen by the error spikes in the small datasets in Figures 5-1 and 5-2. This problem is especially amplified when predicting the acceptance and completion probability estimators P_a and P_c, respectively, as there will necessarily be much fewer of these observations due to the fact that requests must first be sent before being accepted or completed. Further, requests that are rarely sent may not be represented very well in the acceptance and completion datasets, which could be an

issue. Despite these difficulties, this method can still work very well in some situations, and as will be seen in Chapter 6, adds tremendous value to predicting P_s as opposed to the BCF model.

One final consideration when choosing which method to use involves non-stationarity in the system. It is possible, and even likely, that some periods of time will be more saturated by requests with open time windows and other periods of time. In these situations, the requests that are open during peak demand times will necessarily have a lower probability of being sent to a given planner. If the fluctuations in the demand are high enough or the time scale over which the demand fluctuates is long enough to collect sufficient data at each demand level, the probability estimation methods should be able to adapt to this system. Even when these situations do not exist, it can be useful to have estimation methods that adapt over time so that some of the initial data points that are collected before the system reaches a steady-state do not play as large of a factor in the estimation. To address this issue, two additional parameters were introduced into each dataset: *minObservations* and *numObservations*. The first of these, *minObservations*, gives the minimum number of observations that a dataset must have before being used for probability estimation. Any fewer observations and the prior distribution will be used to make the estimates. The second parameter, *numObservations*, specifies the number of data points to maintain over time. For example, suppose that $minObservations = 10$ and $numObservations = 300$ for the dataset $SentData(l)$ when used in a BCF model. Then before 10 observations of potential pairings being sent/not sent are observed, $\hat{\theta}_{MAP}$ is set completely from the prior distribution. Once at least 10 observations exist, $\hat{\theta}_{MAP}$ is estimated using Bayes' rule to obtain the posterior. Once the 300^{th} observation is obtained, every time a new observation enters the system, the oldest one exits.

This method has two major benefits: it adapts to non-stationary systems, and it always insures that prior beliefs provide some regularization in the estimates. The difficulty with using this method, though, is that there is a tension between having enough data points to

overcome random error and maintaining few enough data points that the system can adapt to non-stationarity. For the BCF model this is not a difficult task as very little data is needed to be confident that the estimates are good. This can be more challenging for the BLR where a few hundred observations need to be kept to have confidence that random error is not having a significant effect. In fact, if more than a few attributes are used, the value of *numObservations* would need to be increased unless the prior beliefs were very accurate.

5.4.3 Implementation in the Coordinated Planning Problem

The comparisons between the BCF model and the BLR model present a difficult choice for implementation. Taking into account their relative strengths and weaknesses, three methods were considered for how to implement them in practice.

1. **Estimates via BLR Model** where all of the probabilities were estimated using the BLR technique.

2. **Estimates via BCF Model** where all of the probabilities were estimated using the BCF model.

3. **Mixed estimates** where the probability estimator P_s was determined using BLR, and P_a and P_c were determined using the BCF model. This partition was chosen for the mixed estimates since much more sent data exists than accepted or completed data, and the attributes of requests entering the queue for sending are not affected by the decisions of the CP. In contrast, the attributes of the requests entering the individual planner queues are affected by such decisions.

4. **Mixed estimates followed by solely using BLR** where the mixed method is used until sufficient data is collected for the estimators P_a and P_c to be determined using logistic regression, and the time scale of non-stationarity is long enough to maintain sufficient data. In this case, "sufficient data" must include observations of rare events as well, which may take a long time and would require research into how much data must

initially be obtained. For this reason, analysis of this method is left to future research as it is beyond the scope of this thesis, but it should be noted that this method has the potential to combine the benefits of both estimation techniques while minimizing the costs.

Chapter 6

Analysis and Results

Having developed a collection of mathematical tools to address the Coordinated Planning Problem, we turn our attention to analyzing the performance of our methods. To this end, we address four major issues. The first issue involves assessing the tractability of the mathematical programming formulation presented in Chapter 4. We address this item in Section 6.1, examining the computational limits of the formulation and how that relates to realistically sized problems. The remaining three issues all involve investigating how our newly developed mathematical tools should behave in reality. In their current form, the methods that have been developed can be applied quite broadly. As such, we develop a few scenarios in Section 6.2 for experimentation. The remaining sections address the following three hypotheses:

1. The coordinated planning methods described in this paper, including the math program, probability estimation, and attribute selections, can provide a significant increase in the total value of serviced requests over stovepiped operations.

2. Estimating probabilities using the methods of Chapter 5 can provide a significant increase in the total value of all serviced requests over myopic planning methods.

3. Our mathematical programming formulation can provide a significant increase in the

total value of serviced requests over stovepiped operations even if errors exist in probability estimation.

6.1 Tractability Analysis

As mentioned in Section 4.2.3.4, the mathematical programming formulation (4.11) has the issue that the number of decision variables can potentially grow very quickly at a rate of $O\left(|R|\ |L|^{N^{max}}\right)$ for a fixed value of N^{max} if every request is feasible on every planner. We also mentioned that practically sized problems will generally have less than 20 planners and 1000 requests at any iteration, for which the exact solution to this integer program can be found efficiently using the built-in branch-and-bound techniques of the solver CPLEX [25]. We now explicitly examine the computational time required to solve various problem sizes.

6.1.1 Design

At any given planning iteration, the size of the mathematical programming formulation grows most significantly in three quantities: the maximum number of times a request is allowed to be sent N^{max}, the number of requests in queue at a given iteration $|R|$, and the number of planners in the coordinated planning system $|L|$. We note that a fourth potential quantity influencing the size of the problem is the mean number of planners upon which each request is feasible. However, this is highly dependent upon the requests in queue at a given iteration, so for our tractability analysis we examine the "worst-case" scenario where every request in queue is feasible on every planner. We construct problems such that 10% of the requests are related requests. However, to simplify analysis, all related request sets contain exactly two requests, representing the common simultaneous collections mentioned in Section 3.1.1, while also introducing a necessary element of computational complexity inherent to related requests. Each planner was assumed to have a capacity of $ceiling\left(1.25 \times \frac{|R|}{|L|}\right)$ and reservation fee uniformly selected from the interval $[0, 10]$ cost units, with total budget of

$5 \times |R|$ cost units. We choose these values as they allow the possibility for every request to be sent while also adequately constraining the system.

For a given value of N^{max}, $|R|$, and $|L|$, we created test problems where each request-planner pair (r, l) was given a value v_{rl}, probabilities $q(r, l, 0)$ and $q(r, l, 1)$ defined in Section 4.2.3.2. Each v_{rl} was selected uniformly on the interval $[0, 10]$, each $q(r, l, 0)$ was selected uniformly on $[0, 1]$, and each $q(r, l, 1)$ was selected uniformly on the interval $[0, q(r, l, 0)]$ since $q(r, l, 1) \leq q(r, l, 0)$ must hold.

Under this design, 10 test problems were created for various values of N^{max}, $|R|$, and $|L|$. These problems were solved using both the full and heuristic formulations with the CPLEX libraries for Java and an Intel Core i5 processor [25]. Since coordinated planning will be performed in real time, the amount of computation time was limited to 120 seconds for each formulation to find a solution. If CPLEX could not prove optimality of a solution in this amount of time, the problem was terminated and all remaining problems of that size were skipped. After this, all problems were solved again with a timeout limit of 20 seconds, and an overall timeout limit of 120 seconds for the combined construction and solution of the optimization problem. This 120 second limit was still necessary because the time required to set up the constraint matrix for the composite formulation is not trivial. For example, with $N^{max} = 3$ and $|L| = 100$, there are 166,751 possible composite variables per request. Since construction of each composite variable requires long strings of multiplication, addition of stored probabilities, and construction of many constraints through various loops, this process can be computationally intensive. If CPLEX could completely construct the constraint matrix within 120 seconds but could not prove optimality in that time, the upper bound for the optimality gap produced was recorded at the end of the run. Appendix E lists the mean runtimes for a cross-section of the results obtained, mean runtimes for provably optimal solutions across all 10 test problems, as well as their optimality gap bounds. The results are described in the next section.

6.1.2 Results

Figure 6-1 shows representative plots for the behavior of the formulations in situations where CPLEX proved optimality. The top row of plots holds the number of planners constant at 30 and shows runtime results for $N^{max} = 1$ and $N^{max} = 2$ against the number of requests used. The bottom row of plots holds the number of requests constant at 250 and shows runtime results for $N^{max} = 1$ and $N^{max} = 2$ against the number of planners used. All test problems were solved to optimality for these plots within the 120 second time limit. For the situations analyzed, it is clear that the full formulation is slower than the heuristic formulation. For $N^{max} > 1$, the full formulation becomes intractable much more quickly than the heuristic formulation, breaking the two minute barrier at smaller numbers of planners and requests. This can be seen in Figure 6-1 by the exponential shape of the black lines on the plots for $N^{max} = 2$. In contrast, the time required to solve the heuristic formulation increases in a much more linear fashion for the test samples. This is likely due to the Heuristic formulation having much stronger linear programming relaxations, which would increase efficiency in the branch-and-bound algorithm used by CPLEX.

Interestingly, there were a few cases where CPLEX could not prove optimality of the heuristic solutions. However, the solutions found generally had upper bounds for their optimality gaps with order of magnitude around 10^{-4} found within 20 seconds (see Appendix E). These small gaps suggest that the solutions obtained are likely optimal, but CPLEX could not prove optimality efficiently. Even if they were not optimal, the gap is so small that it would be negligible in a real-world application. Unfortunately, due to the inefficiency of the composite construction phase, the full formulation is still intractable for some of the situations considered, even with the timeout limit. From these results, it can be seen that the full formulation is tractable for most real-world situations, but even if it is not, limiting the solution time or solving the heuristic formulation should give good results.

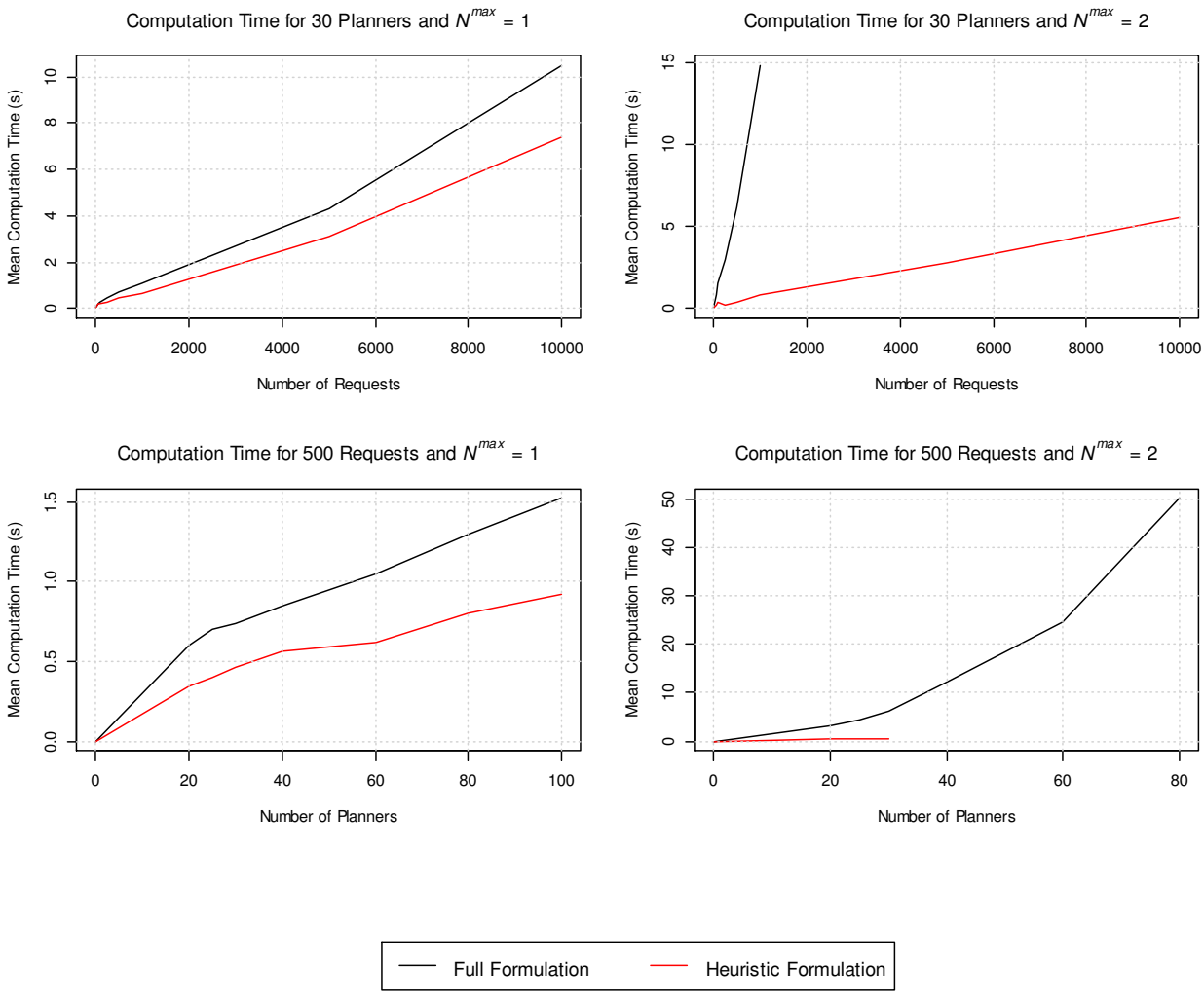

Figure 6-1: Tractability of Full and Heuristic Formulations

6.2 Experimental Setup

Before implementing the mathematical models that we have developed within an actual coordination system, we must understand the behavior and performance of our tools. In particular, we need to know if our methods actually provide any improvement over the current methods, and if so, how much improvement is provided. In addition, we would like to evaluate the sensitivity of the model to errors in the inputs. The next few sections summarize experiments and results for a few coordinated planning scenarios in an effort to address these issues.

6.2.1 Coordination System Framework

The experimental scenarios developed in this chapter are motivated by potential instantiations of the framework discussed in Section 2.3.2 where we are trying to coordinate the mission scheduling of a few UAV and satellite planners. Each scenario contains two UAV planners and six satellite planners. In addition, we assume that each satellite planner has exactly one satellite asset. All planners have capacity constraints, some more than others, and reservation fees vary. The overall design simulates the realistic actuality that a low percentage of requests sent to any given planner are completed.

For most of the tests, the planners are assumed to be non-informative. A few runs are performed with informative planners to give a glimpse of the value of acceptance information, although an in-depth experiment showing the relative merits of informative versus non-informative planners is beyond the scope of this thesis and as such is left to future research. To simulate the fact that real-world responses are not obtained immediately, it is assumed throughout the analysis that if a planner is informative, it waits until the end of a given planning period to inform the CP of which requests were accepted or rejected for that planning period. Further, all planners send a report to the CP at the end of their respective execution periods detailing which coordination requests were just completed or failed. The

CP iteration length is set to be 0.5 hour, and all individual planners have planning periods that are longer than 0.5 hour. This is realistic because planning is often done on the order of many hours or a few days, and the half-hour periodic interval would facilitate easier implementation in a real-world situation.

Each scenario performs planning for 1000 requests with $N^{max} = 3$, 10 target types, and 10 sensor types, where each target-to-sensor pair has a unique observation quality chosen uniformly on the interval $[0, 1]$. The number of sensors on assets for any given planner was chosen uniformly on the set $\{1, 2, 3\}$, and each sensor was chosen uniformly across the set of all 10 sensors. Once the planners were instantiated, they were not changed for any of the trials. This setup simulates the concept that a real-world system can have many target types and sensor types, with planners that have multiple sensors.

Each request was given a single target type selected uniformly from all possibilities, as well as a priority drawn uniformly on the interval $[0, 1]$. The latitude of the request was chosen uniformly on the interval $[35°, 45°]$ and the longitude on the interval $[-120°, -105°]$ (this represents a region in the Western United States). The altitude was chosen to have a 50% chance of being at ground level, and a 50% chance of being above ground. Given that the altitude was above ground, it was selected uniformly on the interval $[0, 4]$ kilometers above ground level. For simplicity, when performing satellite feasibility calculations, we assume that "ground level" meant zero feet above the reference ellipsoid for Earth, which makes little difference as we will only have an error on the order of a few kilometers. The minimum observation duration length for each request was selected on the interval $[0.017, 0.17]$ hours (approximately 1 to 10 minutes) as this represents a realistic set of possible duration lengths. In order to simplify analysis, no related requests were included in these scenarios. This is implemented as a control to ensure that testing of the formulation is not influenced by the existence of special scenarios. Empirical studies of related requests are left to further research if necessary. The value v_{rl} of all feasible request-planner pairs was determined using

the equation

$$v_{rl} = 10\,(0.5 \times priority + 0.3 \times qualObservation_{rl} + 0.2 \times related_r)$$

where $priority_r$, $qualObservation_{rl}$, and $related_r$ are defined in Table 4.6, except that $qualObservation_{rl}$ was calculated using the mean of all observation qualities for sensors on planner l pairing with the target type of r. Note that the value for $related_r$ will always be zero in this scenario—this quantity is left in the value function with a positive weight to facilitate accurate comparison with potential future empirical studies about related requests. Any requests not completed by the end of the scenario or by the end of their respective time windows were considered failed and therefore did not contribute to the total value obtained.

6.2.2 Comparison of Methods

Using the experimental setup of Section 6.2, four different scenarios were designed. The horizon length of all scenarios was 36 hours to simulate 1.5 days of planning. The next few paragraphs explain these scenarios, followed by an experimental design used to test the performance of the methods against certain benchmarks, and a discussion of the results.

6.2.2.1 Scenario 1

This scenario was designed to represent a situation in which the full formulation (4.11) and probability estimator developed in this thesis should perform very well as compared to myopic optimization benchmarks. The time windows of the requests had a wide variability—being drawn uniformly from 0 to 32 hours. The purpose of this was to create a situation in which a myopic formulation should not be optimal. To accommodate this variability, request $startTW$ values were drawn uniformly on the interval 0 to 4 hours after the start of the scenario. The probabilities of acceptance for the planners were all designed to be highly related to the attribute $infeasFrac$ (which was the only attribute used in the BLR models),

and all planners were informative.

6.2.2.2 Scenario 2

The second scenario was designed to be exactly the same as the first, except that all of the planners were non-informative. By comparing the results with the first scenario, a sense of the value gained by receiving acceptance information can be obtained. However, it is important to caveat the results with the fact that this is a very isolated test which incorporates many assumptions, as described in Section 6.2.1. Therefore, examination of the results of this comparison should not be taken to represent the exact quantity of improvement that would be achieved in a real situation.

6.2.2.3 Scenario 3

The third scenario was designed to represent a practical, real-world situation. In this scenario, the length of the request time windows varied from short to long, with values for *startTW* spread throughout the scenario. Specifically, the lengths of the time windows were drawn uniformly on the interval $[0, 15]$ hours, with *startTW* being drawn uniformly from 0 to 24 hours after the start of the scenario. None of the planners were informative in this situation. In addition, acceptance/completion of requests was decided using realistic planner simulations that incorporated the total capacity of the planners during their execution periods, the number of requests actually sent to the planners for each execution period, the saturation of the planners from outside requests, various levels of difficulty implied by the attributes of the request, and random effects. The attributes used for the BLR models were constructed as described in Section 5.3.5.

6.2.2.4 Scenario 4

The final scenario was designed to represent a situation where the marginal benefit of using the stochastic methods over myopic methods should be small. By design, myopic methods

are not forward looking, so they tend to work best in situations where the future has little effect on present decisions. To simulate this idea, all of the requests in this scenario were given shorter time windows drawn uniformly on the interval $[0, 2]$ hours, implying that mutually exclusive intervals of time have very few of the same feasible requests; in fact any two time intervals separated by more than two hours would be guaranteed to not have any of the same feasible requests. The start of the time windows $startTW$ was drawn uniformly between 0 and 34 hours after the start of the scenario. The realistic simulation for acceptance/completion used in scenario 3 was again included for this scenario, with the alteration that the attributes of Section 5.3.5 had absolutely no effect on the acceptance/completion of requests.

6.2.2.5 Design

A total of 30 sets of 1000 requests were generated according to the specifications outlined in each of the four scenarios, for a total of 120 request sets. The possibility that some planners may be saturated with requests from sources external to the CP was simulated by decreasing the percent of coordination requests completed by some of the planners. Within a given scenario, the simulations of the planners remained the same for all 30 request sets.

Each request set was planned over the full horizon using nine different planning methods. The three methods implemented the full formulation (4.11) with each of the first three probability estimation methods explained in Section 5.4.3. The next three methods repeated these tests for the heuristic formulation (4.14). The seventh method implemented coordinated planning via the full formulation without probability estimation by setting $P_s \equiv 0$, $P_a \equiv 1$, and $P_c \equiv 1$, representing the results that would be obtained by optimizing with a myopic policy that ignores stochasticity. The eighth method implemented coordinated planning using the myopic optimization formulation (4.12) that was used to motivate the heuristic formulation (4.14), which ignores both stochasticity and nonlinear effects. The final method did not use any coordinated planning but simply simulated the results that would

be obtained by using the current stovepiped operations. These last three methods served as benchmarks for comparison with the first six methods.

The general probability estimation techniques in Chapter 5 were tailored for specific application to this experiment. The parameter *minObservations* satisfied $minObservations = 10$ for any of the BCF models, $minObservations = 100$ for the BLR models pertaining to P_a and P_c, and $minObservations = 200$ for the BLRs pertaining to P_s. These values were chosen as they allowed for the methods to overcome the data errors evident in small datasets. The BCF model had the smallest value of *minObservations* due to its quick convergence. The BLR model for P_s was given a larger threshold than for P_a and P_c because the beliefs used for P_s were given a lower level of confidence and therefore higher susceptibility to data error. Similarly, $numObservations = 300$ for all of the estimation methods because this was large enough to overcome excessive data error while also being small enough to adapt to any non-stationarity in the time windows of the requests, which existed toward the end of each scenario when requests began to expire without being replaced by new requests.

6.2.2.6 Results

The purpose of this experiment was to test the following two hypotheses presented at the beginning of the chapter:

1. The coordinated planning methods described in this paper, including the math program, probability estimation, and attribute selections, can provide a significant increase in the total value of serviced requests over stovepiped operations.

2. Estimating probabilities using the methods of Chapter 5 can provide a significant increase in the total value of all serviced requests over myopic planning methods.

The first of these hypotheses is addressed in Figure 6-2, which shows the mean values obtained within each of the four scenarios when running the planning horizon separately for each of the 30 request test sets. The three methods shown in this bar chart include the full formulation using BLR for all estimations, the Heuristic formulation using BLR

for all estimations, and the current stovepiped operations. In every single scenario, the stovepiped operations produced values that were much lower than those produced by any of the forms of coordinated planning. In fact, some of the values shown in Figure 6-2 for coordinated planning are more than double the associated values for the stovepiped methods. In addition, the full formulation has the ability to exploit probabilistic information much more easily than the heuristic as seen by the large differences in the results for scenarios 1 and 2 (full yields 21.1% and 17.7% more value than heuristic). The difference is smaller for scenarios 3 and 4, with the full formulation yielding 11.4% and 6.2% improvements over the heuristic formulation, respectively, although still quite significant in a sample of size 30. Thus, the coordinated planning methods described in this paper can certainly provide a large increase in value over the current stovepiped operations, especially in a situation where the full formulation is tractable.

It is interesting that the total values obtained between scenarios 1 and 2 are quite similar for both formulations—in fact, scenario 2 produced better average results than scenario 1 for the heuristic formulation—which suggests that, at least in the most optimistic situations, an informative planner that communicates with the CP as late as possible is not much better than a non-informative planner. However, it is important to caveat this observation with the fact that the difference between informative and non-informative planners may be more pronounced if non-informative planners fail to return completion/failure data in a timely manner.

The second hypothesis is addressed in Figure 6-3 in which the three methods of probability estimation described in Section 5.4.3 are compared against the myopic implementations created by setting $P_s \equiv 0$, $P_a \equiv 1$, and $P_c \equiv 1$. The comparisons are controlled by formulation to see if the results are different between the full and heuristic integer programs. The vertical axes of the plots in Figure 6-3 show the percent increase over the myopic benchmark in each of the scenarios that was obtained by using the indicated method for probability estimation (as such the myopic benchmark is simply the horizontal axis because it has zero

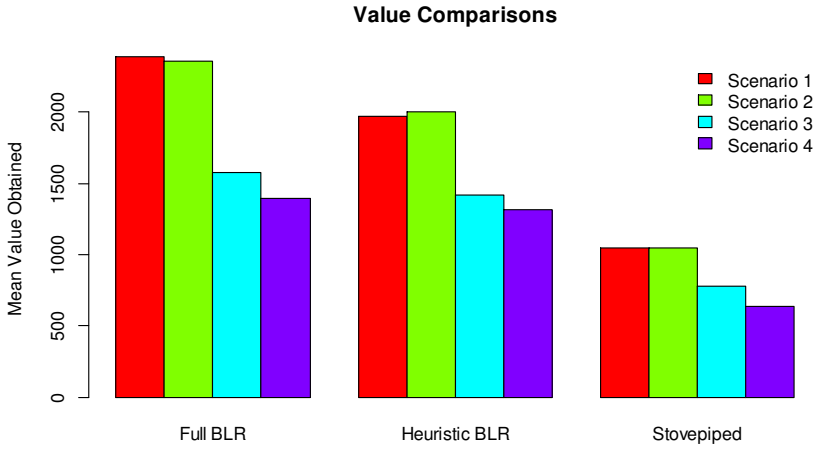

Figure 6-2: Value of Formulations

benefit over itself). It is clear from this figure that the probability estimation provided a large amount of benefit regardless of formulation, suggesting that the probability estimation procedures have the ability to reduce uncertainty in the optimization. However, the pessimistic scenario 4 produced some difficulties for the estimation procedure. The full formulation was still able to do as well or better than the myopic benchmark for this scenario, but the heuristic did slightly worse. This is an important piece of information as it suggests that using probability estimation with the full formulation tends to outperform myopic methods even in the worst situations; similarly, the heuristic formulation should outperform its myopic benchmark in the majority of potential scenarios.

The BLR estimation procedure worked very well in both cases. However, it seems that the mixed BLR and BCF techniques of Section 5.4.3 are the most robust to scenario changes for the full formulation, outperforming the estimations that solely use BLR techniques in all but scenario 1 (which was purposefully tailored to the strengths of BLR). This makes sense as the acceptance and completion data are highly dependent on the decisions of the CP, so a simple estimation technique for these probabilities can reduce error. It is significant, though, that solely using the simple BCF estimation methods tends to do worse than either of the other two estimation procedures, although it is still better than the myopic benchmarks.

This suggests that BLR models should be used to determine the probability estimator P_s in any situation.

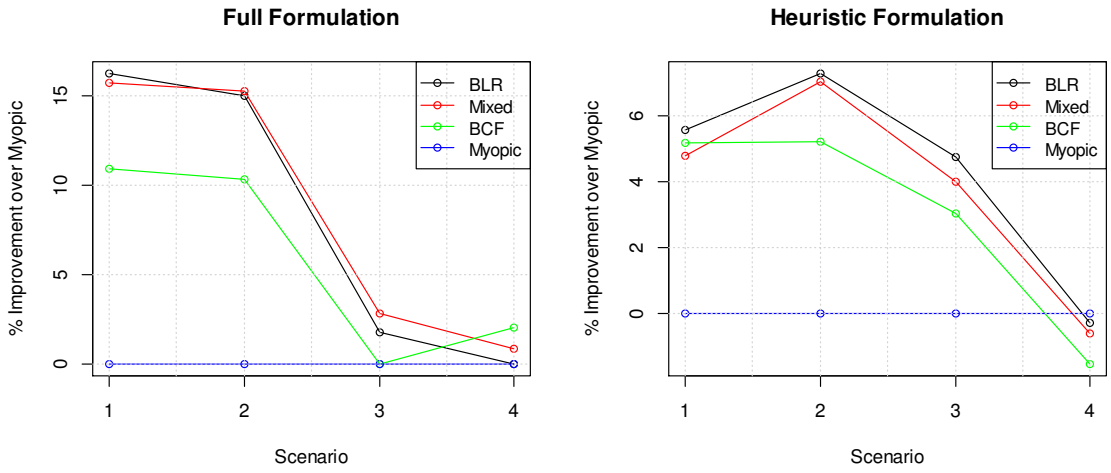

Figure 6-3: Value of Probability Estimation

The final comparison performed was between the myopic implementations of the full and heuristic formulations to see the differences in the absence of probability estimation. Figure 6-4 displays these results, which show that the myopic benchmark for the full formulation is clearly higher than the myopic benchmark for the heuristic formulation in all scenarios, implying that there is value to be obtained if the full formulation is in a tractable situation. This result also suggests that intelligently reducing the space of composite variables to expand the tractability of the full formulation could be valuable in the future.

In general, the results of the experiments imply that both the estimation procedure and the design of the formulations provided essential contributions to the increased performance of the stochastic methods over the myopic and stovepiped methods. However, probability estimation should never be implemented without careful consideration of the model being used. As with any machine learning based method, improper application may yield poor results. Thus, the models should aim to include as few attributes as possible to describe the system while still giving accurate predictions, even when data is scarce.

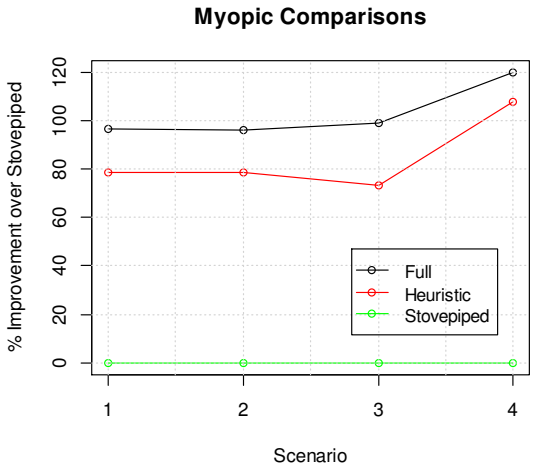

Figure 6-4: Myopic Benchmark Comparisons

6.2.3 Sensitivity

The purpose of this section is to test the hypothesis that the mathematical programming formulations presented in Chapter 4 can provide a significant improvement over stovepiped operations even if errors exist in probability estimation. This is useful if the algorithm for probability estimation is changed in the future, or if we are worried that the probabilities of our system may be hard to estimate.

6.2.3.1 Design

Perfect quantification of the errors inherent to a coordinated planning scenario is impossible; endless numbers of potential use-case scenarios could be conceived, each having unknown "true" probabilities. In light of this fact, the goal for the sensitivity analysis was to examine the behavior of the coordination planning formulations in a single theoretical situation where all of the assumptions of the model are met, and "true" acceptance and completion probabilities exist. To this end, 30 sets of 1000 requests were generated according to the specifications of scenario 3 in Section 6.2.2.3 for a 36-hour horizon with the CP iteration length set at 0.5 hour for each test case, although all test runs assumed informative planners (which return information immediately prior to the appropriate execution periods). However, instead of

simulating the output of the individual planning algorithms as we did in the previous experiment, we assigned each request-planner pairing a probability of being accepted or completed for each feasible execution period. The acceptance probability was drawn from a beta distribution with mean $\bar{p} = 0.4$ using the associated maximum sensible variance (as described in Section 5.2.2). The completion probability was drawn from another beta distribution with mean 0.9 using the associated maximum sensible variance, modeling the fact that completion probability should be high since the request has already been accepted. The distribution of acceptance probabilities and the constant completion probability were both held constant across all test runs for each planner. These are the "true" probabilities of acceptance and completion, from which the events of acceptance and completion of all coordination requests were simulated. It is important to note that the results presented in the next section have different numerical values than those seen in the previous section. This is a direct result of the fact that probabilities were drawn from distributions which might not represent the behaviors of the individual planning algorithms used for the previous analysis.

We note that the probability of being sent, which we denote p_s for this section, is directly dependent upon the formulation. As a result, we cannot simulate errors in the probability of a request being sent in a future execution period of any given planner. Thus, we cannot create a "true" probability that a coordination request associated with a request-planner pairing will be sent. However, it is possible to run the analysis by viewing the sent probability as a parameter that places value on future possibilities rather than a quantity that needs to be estimated. As such, we analyze the sensitivity of the CP with the sent probability set at the thresholds shown in Table 6.1. For each of these thresholds, we take potential request-planner pairings and test the sensitivity of the mathematical programming formulation to various levels of error in the estimates of the "true" probabilities that are used for simulation.

We examine a few different types of error in this analysis. The first type of error that we examine is random noise about the true probabilities. Essentially, this tests how well our formulation performs if the probabilities that are input to the math program are perturbed

from the true estimates. To test random noise errors, let p represent an arbitrary probability that must be estimated (this could be an acceptance or completion probability). To simulate noise in our estimates, the estimate for each such probability p is drawn from the interval $[\max(0, p-\varepsilon),\ \min(1, p+\varepsilon)]$ for some error level $\varepsilon \in [0, 1]$.

The second type of error that we examine is error due to overly conservative or extreme probability estimates. This type of error tends to produce probabilities that are either abnormally clustered around 0.5 (which we refer to as conservative estimates), or are abnormally pushed to the extremes of 0 or 1 (which we refer to as extreme estimates). Conservative and extreme errors appear when estimating probabilities using various machine-learning methods [32], often requiring empirical data and some form of calibration, such as the sigmoid training method presented in [33]. Thus, it is important to determine the performance of our formulation when introduced to error of this type. To simulate this behavior, we construct the estimates as follows. We first perturb each true probability p using the random noise simulation to obtain a new estimate $\tilde{p}$ drawn uniformly on $[\max(0, p-\varepsilon),\ \min(1, p+\varepsilon)]$ for some error level $\varepsilon \in [0, 1]$. We then take the new estimates $\tilde{p}$ and create a final estimate $\hat{p}$ that solves $\frac{\hat{p}}{1-\hat{p}} = \left(\frac{\tilde{p}}{1-\tilde{p}}\right)^{\delta}$ for some $\delta > 0$, so that the final estimate is

$$\hat{p} = \frac{\tilde{p}^{\delta}}{(1-\tilde{p})^{\delta} + \tilde{p}^{\delta}}. \tag{6.1}$$

Using this transformation, any value of $0 < \delta < 1$ pushes the estimates $\hat{p}$ closer to 0.5, and any value of $\delta > 1$ pushes the estimates $\hat{p}$ closer to the extremes 0 and 1 ($\delta = 1$ does not change the estimate). The final values $\hat{p}$ are then given to the math program for use in the trial runs. Conceptually, the parameter δ represents the extent to which an arbitrary probability estimator is conservative ($0 < \delta < 1$) or extreme ($\delta > 1$). Figure 6-5 shows the effect the mapping 6.1 from $\tilde{p}$ to $\hat{p}$ for a few values of δ.

Under this design, 30 test scenarios were generated. Each scenario was then run two times for each of the 80 $(p_s, \varepsilon, \delta)$ triples obtained by selecting each of p_s, ε, δ from the values

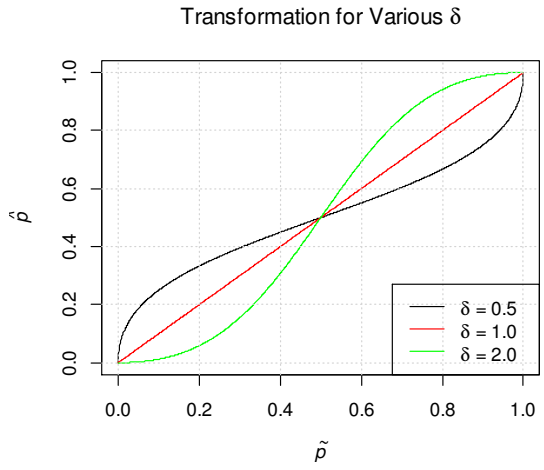

Figure 6-5: Effects of Various δ on $[0, 1]$

shown in Table 6.1. In the first of these two runs, the full formulation (4.11) was used to implement coordinated planning, and in the second run the heuristic formulation (4.14) was used for coordinated planning. Each of the test scenarios was then run two final times using the following benchmarks for comparison. The first of these implemented coordinated planning via the optimization formulation (4.12), referred to as the "myopic benchmark" in the results. The second was done without using a coordinated planning method, but rather by using the stovepiped system representing current operations, where each request was blindly submitted to a single planner. This method was referred to as the "stovepiped benchmark" in the results. Neither of these final two benchmark planning methods take probability estimates into account, so there was no need to run the test scenarios for different values of the $(p_s, \varepsilon, \delta)$ triples. The total value of all completed requests was recorded for each of the test runs. The results are described in the next section.

Parameter	p_s	ε	δ
Values	0, 0.1, 0.2, 0.3	0, 0.1, 0.2, 0.3	0.5, 0.75, 1.0, 1.33, 2.0

Table 6.1: Sensitivity Parameter Values

6.2.3.2 Results and Analysis

The goal of the sensitivity analysis was to examine how well the full formation (4.11) worked under different levels of error in the probability estimates. The results obtained from the sensitivity experiment are presented in Figure 6-6, Figure 6-7, and Table 6.2. The figures are designed to separately visualize how varying each of the parameters p_s, ε, δ affects the results.

The first part of testing the effects of error in the probability estimates examined differences in the value obtained by varying ε. Since ε models random noise, the results of this analysis should indicate whether or not random noise in the probability estimates provides reservations about using the full formulation for coordinated planning. The results for this analysis are summarized in Figure 6-6, which shows the mean value obtained across all 30 test scenarios as ε increases. There are four plots, each one giving results for a constant value of p_s. In any single plot, a separate curve is given for each value of δ using one of the four planning methods described in the previous section.

Only two of the four planning methods are shown in Figure 6-6. This is because the mean values obtained for the stovepiped and myopic benchmarks were both much less than the value obtained for the full and heuristic formulations under any of the situations considered (see Table 6.2). Thus, it is clear that even with a large amount of error, incorporating knowledge about probabilities into the coordinated planning process provides far superior results than the current stovepiped operations or myopic CP implementations. This does not necessarily mean that including knowledge about probabilities will always have this effect; however the difference is large enough to conclude that coordinated planning with knowledge about probabilities can drastically increase the total value obtained from all requests.

It is clear from Figure 6-6 that both the full formulation and the heuristic formulation are sensitive to ε. This result seems to hold true regardless of the values for p_s or δ as seen by the fact that as ε increases, both the black lines and the red lines decrease. The plots also show that the full formulation provided a clear benefit over the heuristic formulation.

In fact, in many of the situations considered, the full formulation provided more than a 10% increase in the mean value obtained across the scenarios than was obtained by the heuristic formulation—a gain that is especially significant considering that the standard deviations for each sample with constant $(p_s, \varepsilon, \delta)$ were between 1% and 3% of the mean value obtained for all of the planning methods (see Table 6.2).

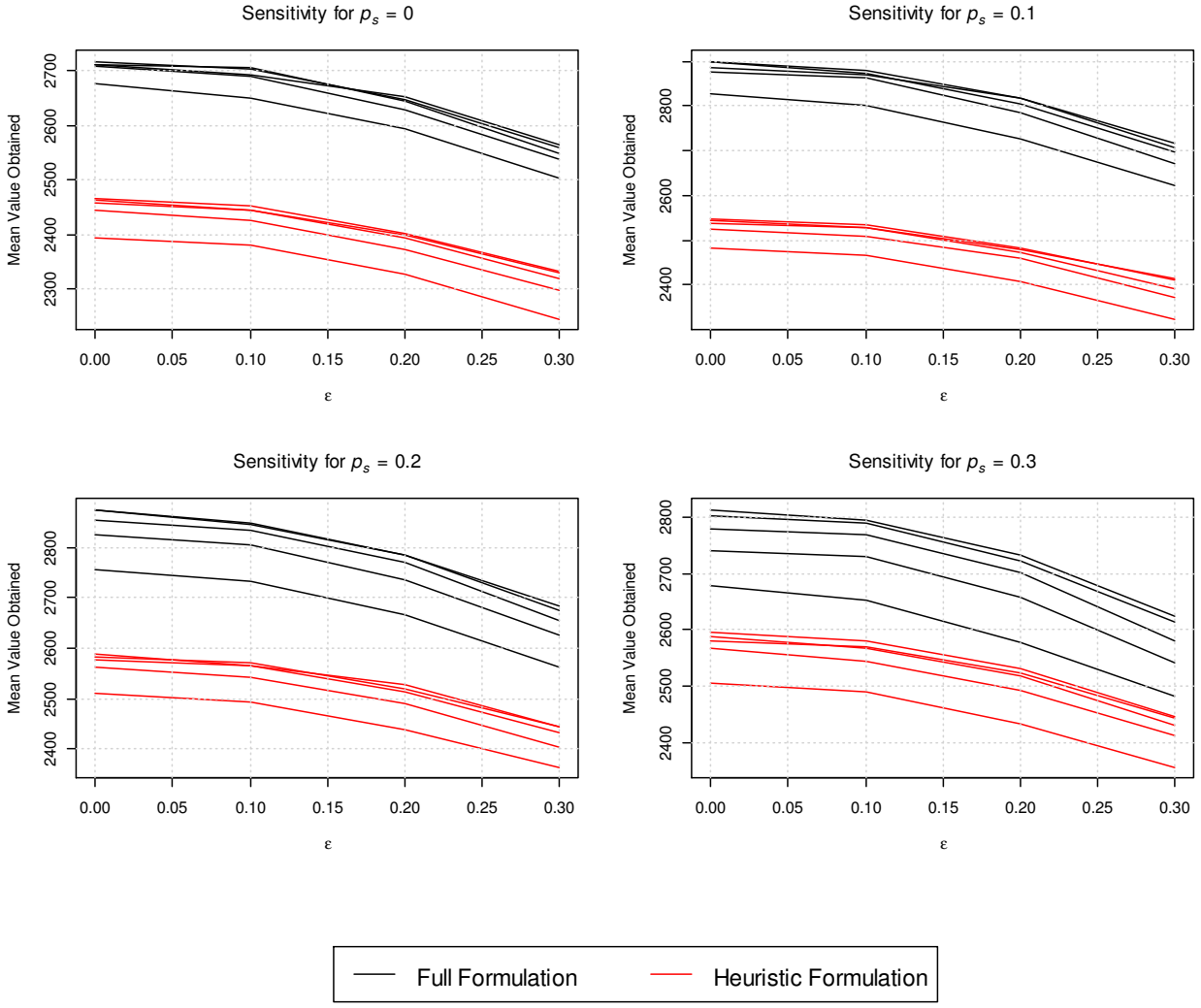

Figure 6-6: Sensitivity of Formulations to ε

The second part of testing examined differences in the value obtained by varying δ—indicating whether errors due to overly conservative ($\delta < 1.0$) or overly extreme ($\delta > 1.0$)

estimates should induce reservations about using the full formulation for coordinated planning. The results for this analysis are summarized in Figure 6-7, which shows the mean value obtained across all 30 test scenarios as δ increases. There are four plots, each one giving results for a constant value of p_s. In any single plot, a separate curve is given for each value of ε using one of the four planning methods described in the previous section. Again, the mean values for the stovepiped and myopic benchmarks are not shown in these plots as they are both much less than the value obtained for the full and heuristic formulations under any of the situations considered. A few of the arcs in Figure 6-7, such as the full formulation for some of the low values of ε and p_s (not labeled on the plots), yield the least mean values for low values of δ, increase monotonically to a maximum at $\delta = 1.0$, and then decrease slightly on the path to $\delta = 2.0$. The maximum at $\delta = 1.0$ makes intuitive sense—this is the point where δ does not contribute any extra error into the estimates. It also makes sense that low values of δ produce worse results than high values of δ, even for symmetric values of $\log(\delta)$ about $\log(1.0) = 0$. This is because values of $\delta < 1.0$, or $\log(\delta) < 0$, bring all estimates closer to 0.5, thereby removing making it more difficult to distinguish between high and low probabilities. In contrast, values of $\delta > 1.0$, or $\log(\delta) > 0$, move all probabilities that are greater than 0.5 closer to 1, and all probabilities less than 0.5 closer to 0. Thus, even though this introduces error, it successfully separates higher probabilities from lower probabilities.

This separation works so well that many of the curves actually produce their best results at $\delta = 1.33$ or $\delta = 2.0$, which is counter-intuitive considering the extra error introduced at these values of δ. Thus, it seems more desirable to have extreme probability estimates than to have conservative estimates. However, it is not recommended that extra error be injected into a real-world situation to make estimates more extreme because it is not clear how much scaling would be acceptable in practice before the estimates would become so extreme that they would overcome the actual pairing values and reduce performance.

Figures 6-6 and 6-7 also reveal how the full and heuristic formulations perform as the parameter p_s changes, indicating whether or not the planning methods are robust to errors in

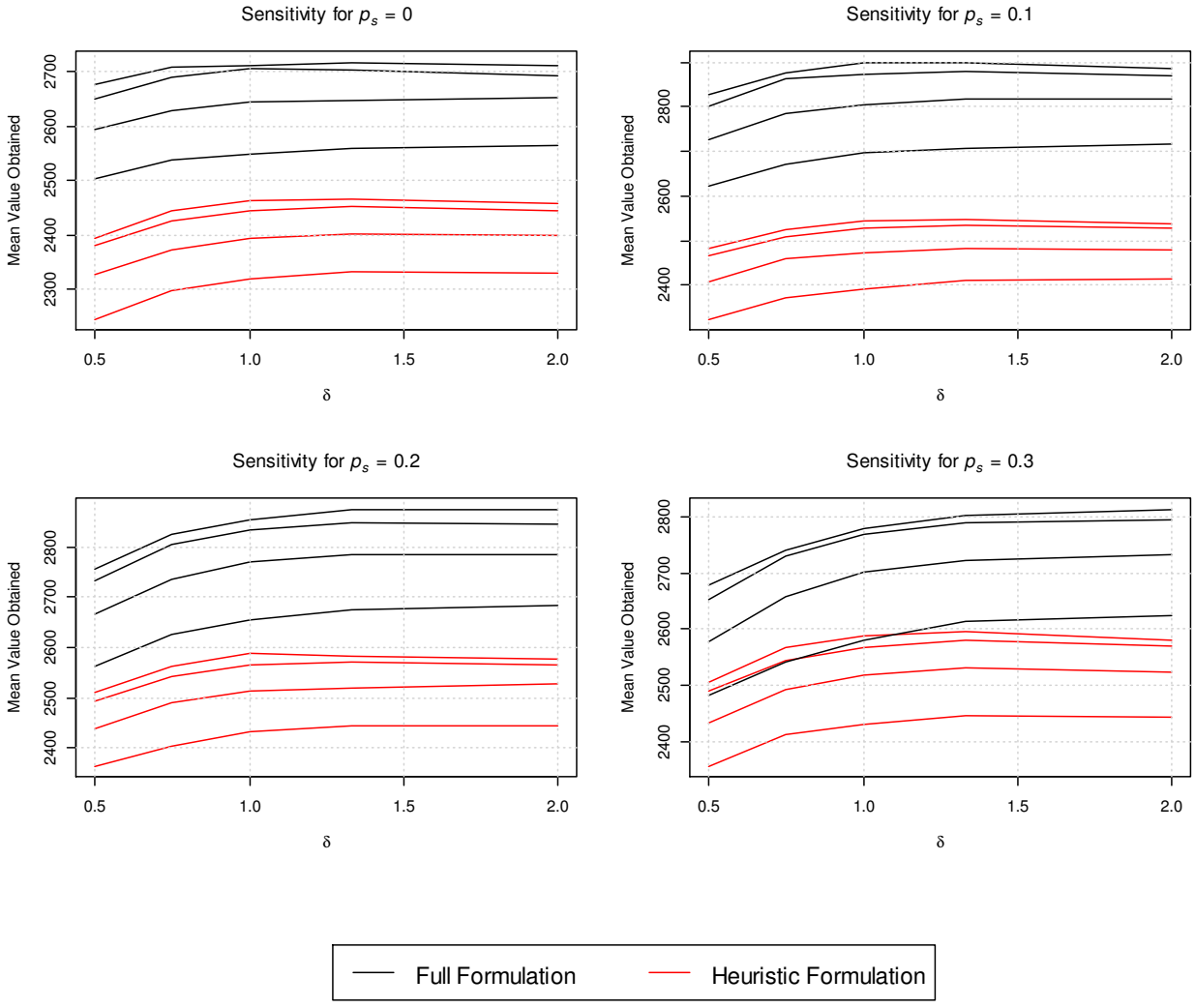

Figure 6-7: Sensitivity of Formulations to δ

the probability estimator $P_s(r, l, t)$. These figures show that the full formulation is slightly more sensitive to changes in p_s than the heuristic formulation, evidenced by the larger variability of the black lines than the red lines as p_s changes. In addition, certain values of p_s seem to amplify the effects of ε and δ for both methods. In fact, the results suggest that p_s is the most significant of the three parameters as neither ε nor δ alone have nearly the same impact of amplifying other errors. However, it is important to note that because there are many more potential pairings than the budget and total capacities of the planners could allow, the values returned by $P_s(r, l, t)$ must be very low on average. Thus, considering the range 0 to 0.3 for p_s likely spans the entire range of outputs for $P_s(r, l, t)$. This fact suggests that errors in $P_s(r, l, t)$ should not be of concern because all of the results for both formulations far surpassed the values obtained by the myopic and stovepiped benchmarks.

The overall benefits of the coordinated planning methods analyzed can be seen by examining Table 6.2. Clearly both the full and heuristic formulations produced much better results than the myopic and stovepiped benchmarks. Another positive result shown in Figure 6.2 is that at any given amount of error, both formulations are very consistent in the amount of value that they produce, yielding sample standard deviations that are less than 3% of the associated sample means. These low standard deviations imply that in any single real-world scenario, a large amount of confidence can be placed that neither the full nor the heuristic formulation will perform poorly.

Method	Min Sample Mean	Max Sample Mean	Min Sample Standard Deviation	Max Sample Standard Deviation
Full	2482.68	2900.48	50.08	70.73
Heuristic	2245.44	2597.14	46.83	66.65
Full Myopic	1955.88		57.13	
Heuristic Myopic	1653.88		49.05	
Stovepiped	1027.06		53.12	

Table 6.2: Mean and Standard Deviation Comparisons

6.3 Conclusions

The collective results of the experiments performed support all of the hypothesis presented in the introduction to this chapter. The tractability analysis showed that the full formulation is efficient in many practically sized problems, with the heuristic formulation being efficient through a much larger theoretical size that may be encountered in the future. The scenarios considered in Section 6.2.2 provided strong evidence that the coordinated planning methods described in this paper, including the mathematical programming formulations, probability estimation, and attribute selections, can provide a significant improvement over stovepiped operations. The sensitivity analysis showed that even with errors in probability estimates, both the full and heuristic formulations should outperform current operations. In fact, the current "stovepiped" operations performed considerably worse than coordinated planning via either the full or heuristic formulation in all of the results.

Chapter 7

Further Research and Conclusions

7.1 Suggestions for Future Research

The models created in this thesis provide many contributions toward finding the best solution for coordinated planning. However, there are still many potential directions for future work. Some of the directions that could offer immediate improvement are described in this section.

7.1.1 Expansion of Problem Scope

There are many conceivable real-world situations that could benefit from coordinated planning. However, the problem currently restricts requests and planners to satisfy the assumptions set forth in Chapter 3. One of the consequences of this restriction is that requests occur only at single point targets. Part of the reason for this restriction is that realistic planners, such as the ones presented in [10] and [13], take point targets as inputs. However, it could be helpful to research ways to include spatial targets into the problem as well. Two potential ways for doing this involve either developing a good user interface for the CP that can accept spatial requests and appropriately discretize them into point targets, or developing a new opportunity finder to determine if spatial targets would be feasible on various planners, subsequently performing the estimation of acceptance and completion probabilities

using appropriate attributes. Another restriction of the assumptions is that only satellite and UAV assets were considered. It is quite possible that planners within a coordinated planning system control other types of assets as well, such as ground vehicles or unmanned underwater vehicles. Research into the appropriate types of attributes for use in BLR models of such situations could prove to be valuable.

Another potential extension of the problem would be to consider the possibility that planners have unknown capacity rates. This is very possible and models the idea that some planners might not explicitly limit the number of requests that they can receive, but in practice cannot complete more than a certain number. As a result, there is a limit on the number of requests that should be sent by the CP, because sending too many might result in planners completing low-value requests instead of high-value requests. To combat this problem, two methods are suggested for further research. One possibility could be to analyze data requirements and capacities of other known planners, creating an estimate of the unknown capacity rates. Another possibility could be to assume a Bayesian likelihood model for the unknown capacity, updating the MAP estimate of the capacity parameter(s) as observations are received pertaining to the number of requests completed per execution period of a given planner. This might be too computationally intensive or undesirable to implement, in which case a simpler model could be developed, or a guess based on simulations/prior data could be used.

7.1.2 Mathematical Methods

Even without expanding the problem scope there are still many ways to provide new insights into the work that has already been completed. A great deal of work has been done in this thesis to estimate various probabilities using a BLR model. While the results suggested that this estimation method worked quite well, it may be possible to do better. Future research into various types of prior distributions, or even completely different models such as neural networks or beta regression, may be of value. However, one of the major benefits of the BLR

model in this paper was the ability to construct a Gaussian prior distribution that reflected any beliefs. The ability to do this depended directly on the special structure and assumptions involved in logistic regression models. Implementing an analogous Bayesian approach in a neural network may be much more difficult (or even impossible) due to increased model complexity. Beta regression may exhibit a similar problem as well. However, if prior beliefs are insignificant, these may be viable options to pursue.

Since uncertainty is inherent in this problem, it may be useful to examine robust discrete optimization approaches to finding a solution [34]. Such work might involve trying to replace probability estimation with a robust optimization formulation. Unfortunately, taking this approach would likely involve creating an uncertainty set for the objective function that is dependent on the vector of decision variables for allocating requests to planners. As such, the usual strong duality results would not apply to this situation; a specialized iterative algorithm would likely need to be developed to solve the problem. Even if a successful algorithm could be developed, it is not clear that the solution would do any better because acting conservatively might not be a good approach for this problem. A second, more practical, approach would be to use robust optimization to control the error in the probability estimates, using an error bound derived from the posterior parameter distributions of the BLR models. This approach certainly has the potential to provide protection against poor estimates. However, the approach also exhibits many modeling difficulties that might be difficult to overcome, such as highly nonlinear relationships between estimates that would make modeling of correlations between cost function coefficients in a robust formulation very difficult. It may be possible to construct a robust model that ignores these effects, but this could potentially loose many of the benefits of the probability estimation. Further research into this area is necessary to make definite conclusions.

A significant benefit could also be obtained by development of an intelligent method for reducing the space of composite variables, which could allow the full formulation (4.11) to be tractable in the large problems that may be experienced in the future. One potential

method for doing this could involve designing a generalized assignment problem that selects a subset of planners for consideration by each request, using v_{rl} values in the objective with no other side constraints. Such a problem could be solved efficiently, using the output to generate the composites for the full formulation. This is just one possible suggestion, and all results should be tested against the heuristic formulation to ensure that the model provides extra improvement.

7.1.3 Suggestions for New Request Attributes

One of the most important items for future research involves determining which request attributes result in the best probability estimates. Specifically, research should be done to determine the best attributes for acceptance and completion probabilities in various types of coordinated planning scenarios. Further research could also be done to find better attributes for use in determining P_s; however, these attributes are much less dependent on the actual scenario and much more dependent on the coordination planning algorithm employed. Assuming that the algorithm does not change significantly, the attributes used for the analysis in Chapter 6 should suffice. The attributes used to estimate probabilities of acceptance and completion, though, depend heavily on the scenario. As such, it may be useful to analyze which attributes work best in different scenarios. In order to facilitate this research, a few realistic scenarios are listed here along with suggested attributes to explore.

7.1.3.1 Cloud Cover and Weather Data

Situations may exist in which cloud cover or other weather data can be easily extracted. In this case, some of the attributes that may be useful to examine include:

- Actual cloud cover at the request target location. This would be used when recording observations; the predicted cloud cover would be used for probability estimation.
- Actual wind speed at the request target location (for UAVs). Again, actual wind speed

would be used when recording observations; predicted wind speed would be used for probability estimation.

- Chance of precipitation at the request target location (for UAVs). Actual observations for this would simply record either a 1 or 0 if there was or was not precipitation at the target location; predicted chance of precipitation would be used for probability estimation.

- Any other relevant weather-related issues.

7.1.3.2 Planner-Specific Information

It may be possible to gain highly detailed information about the operations or mission plans of particular planners or their respective assets, outside of the generic information used for the analysis in this thesis. If any such information can be obtained, it should be exploited to aid the probability estimation process. For example:

- If the flight plans for a UAV asset are known, the distance from a request target location to the closest leg of the UAV flight path could be used as an attribute, allowing the planner to efficiently identify piggybacking opportunities (see Section 3.2.3.3 for a description of piggybacking).

- If a planner l has multiple assets with different specifications, or controls more than one type of asset, great care would need to be taken to ensure that attributes are well-defined for any request r. For example, planner l might control satellite s and UAV u. It does not make sense to include an attribute for mean viewing angle on satellite s or distance from the home base of UAV u, because if a request r is feasible on s but not on u, the "distance from home base" attribute is ill-defined. (This problem was implicitly controlled in the analysis of Chapter 6 by assuming that satellite planners had only one asset, and that UAV planners had assets with the same specifications.) One potential method for circumventing this issue would replace asset-specific attributes such as

"mean viewing angle" with artificial "difficulty scores" which are calculated using the asset-specific attributes, where the maximum difficulty score represents an infeasible request. If this method were to be employed, care would need to be taken to ensure that the transition from a maximum difficulty score to a slightly smaller difficulty score naturally represents the difference between an infeasible request and an almost infeasible request. Research into how to construct these difficulty scores could greatly expand the applicability of the models developed in this thesis.

- If a satellite planner has more than one satellite, the number of satellites on which a request is feasible could be added as a new attribute. In addition, the mean viewing angle would need to be adapted, potentially to be the mean viewing angle across all feasible satellites. Similar modifications to the time-based attributes would need to be done as well.

7.2 Major Contributions

The work presented in this thesis is an extension of that done in [1]. As such, a description of the main assumptions from previous work that have been relaxed, along with new contributions, are listed here.

1. The assumption from [1] that a request is guaranteed to be completed once it has been accepted is relaxed in this thesis.

2. The assumption from [1] that all planners must be informative (i.e., send information to the CP when requests have been accepted or rejected) is relaxed in this thesis.

3. For those planners that are informative, the assumption from [1] that acceptance and rejection information is relayed to the CP before the next iteration of coordinated planning is relaxed in this thesis.

4. The concept of a dual or simultaneous collection is generalized in this thesis using the concept of "related requests" that can include more than two requests.

5. The model in this thesis is designed to maximize an expected value based on probabilities that can be initialized using prior beliefs and updated via on-line learning.

6. The nonlinear increase in expected value gained by sending a request multiple times is addressed in this thesis.

7.3 Conclusions

A few important implications follow directly from the results of this thesis. Coordinated planning needs to be performed intelligently using knowledge about the future and controlling for uncertainty in order to provide the most benefit. Choosing probability estimation via BLR as the preferred method for controlling uncertainty created the added benefit that the prior beliefs about the behaviors of various planners, derived from simulation, data, or expert knowledge, could be incorporated into the system, which could then be combined in real-time with observed data to produce probability estimates. These estimates could be used either for coordinated planning, or could be released to the users to give them information about which requests should be considered for submission to a coordination planner. In short, the evidence shows that coordinated planning can provide a significant benefit to air and space data collection missions at a small marginal cost to the owners of the assets, without even requiring much change in the current infrastructure. Incorporating coordinated planning into the operations of air and space assets will be a necessity in the coming years to ensure efficient use of all assets.

THIS PAGE INTENTIONALLY LEFT BLANK

Appendix A

UAV Specifications

Specification	Description
Base Latitude	The latitude of the home base location for the UAV
Base Longitude	The longitude of the home base location for the UAV
Endurance	The maximum length of continuous time (in hours) that the UAV can be working a mission
Max Speed	The maximum speed (in knots) of the UAV
Sensor Type	The actual sensors on board the UAV

Table A.1: UAV Specifications

THIS PAGE INTENTIONALLY LEFT BLANK

Appendix B

Satellite Specifications

Specification	Description
a	Semi-major axis of the satellite orbit
e	Eccentricity of the satellite orbit
i	Inclination of the satellite orbit
Ω	Right Ascension of the Ascending Node for the satellite orbit
ω	Argument of Perigee for the satellite orbit
M_0	Mean anomaly of the satellite at the start of the scenario
θ_0	Greenwich sidereal time at the start of the scenario

Table B.1: Satellite Specifications (see [2] for details)

THIS PAGE INTENTIONALLY LEFT BLANK

Appendix C

Parameters for Maximum Sensible Variance in a Beta Distribution

Consider the optimization problem for determining the parameters α, β of a beta distribution that result in the *maximum sensible variance* (as described in Section 5.2.2) of a beta distribution with given mean $\mu \in (0, 1)$

$$\max \quad \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} \tag{C.1}$$

$$\text{st} \quad \frac{\alpha}{\alpha+\beta} = \mu \tag{C.2}$$

$$\alpha \geq 1 \tag{C.3}$$

$$\beta \geq 1. \tag{C.4}$$

Clearly any feasible solution has $\alpha + \beta > 0$ since $\alpha, \beta \geq 1$. Thus, solving constraint C.2 for β yields $\beta = \left(\frac{1-\mu}{\mu}\right)\alpha = \lambda\alpha$, where $\lambda \triangleq \frac{1-\mu}{\mu}$. Substituting for β yields the equivalent optimization problem

$$\begin{aligned} \max \quad & \frac{\lambda}{(1+\lambda)^2(\alpha+\lambda\alpha+1)} \\ \text{st} \quad & \alpha \geq 1 \\ & \alpha \geq \frac{1}{\lambda}. \end{aligned}$$

Now, $\frac{\lambda}{(1+\lambda)^2}$ is constant and $\lambda > 0$, so the objective is monotonically decreasing in α. Thus, the optimal solution will be to set α^* to be as small as possible, so that $\alpha^* = \max\left(1, \frac{1}{\lambda}\right)$. This implies that $\beta^* = \lambda\alpha^* = \max(\lambda, 1)$. ■

Appendix D

Proof of LP Feasibility for Prior Calculations

Recall the linear program for creating a prior parameter distribution in a BLR model, along with the notation defined in Chapter 5:

$$\underset{\boldsymbol{v}}{\text{minimize}} \quad \sum_{k=1}^{n} \frac{1}{\|\boldsymbol{x^k}\|_2^2} \boldsymbol{x^k}^T \text{diag}(\boldsymbol{v}) \boldsymbol{x^k} \tag{D.1}$$

$$\text{subject to} \quad \boldsymbol{x^k}^T \text{diag}(\boldsymbol{v}) \boldsymbol{x^k} \geq \alpha_k^2 \qquad \forall k : \text{logit}(a_k) - \boldsymbol{\mu}^T \boldsymbol{x^k} < 0 \tag{D.2}$$

$$\boldsymbol{x^k}^T \text{diag}(\boldsymbol{v}) \boldsymbol{x^k} \geq \beta_k^2 \qquad \forall k : \text{logit}(b_k) - \boldsymbol{\mu}^T \boldsymbol{x^k} > 0 \tag{D.3}$$

$$\frac{v_i}{\gamma_i^2} \leq r^2 \frac{v_j}{\gamma_j^2} \qquad \forall i, j \in \{1, \ldots, d\} : i \neq j \tag{D.4}$$

$$0 \leq v_{min} \leq v_i \leq v_{max} \qquad \forall i \in \{1, \ldots, d\} \tag{D.5}$$

The following four conditions are sufficient to guarantee feasibility of this LP:

1. The first attribute of $\boldsymbol{x^k}$ is the constant, i.e., $x_1^k = 1$ for all k.

2. $\gamma_i \neq 0$ for all i.

3. $v_{min} \leq \gamma_i^2 \left[\max_{\left\{k \middle| \text{logit}(a_k) - \boldsymbol{\mu}^T \boldsymbol{x^k} < 0\right\}} \left\{\alpha_k^2\right\} \right]$ or $v_{min} \leq \gamma_i^2 \left[\max_{\left\{k \middle| \text{logit}(b_k) - \boldsymbol{\mu}^T \boldsymbol{x^k} > 0\right\}} \left\{\beta_k^2\right\} \right]$, $\forall i \in \{1, \ldots, d\}$.

4. $v_{max} \geq \gamma_i^2 \left[\max_{\left\{k \middle| \text{logit}(a_k) - \boldsymbol{\mu}^T \boldsymbol{x^k} < 0\right\}} \left\{\alpha_k^2\right\} \right]$ and $v_{max} \geq \gamma_i^2 \left[\max_{\left\{k \middle| \text{logit}(b_k) - \boldsymbol{\mu}^T \boldsymbol{x^k} > 0\right\}} \left\{\beta_k^2\right\} \right]$, $\forall i \in \{1, \ldots, d\}$.

Proof: Assume these conditions hold. Define

$$k_\alpha = \max_{\left\{k \middle| \text{logit}(a_k) - \boldsymbol{\mu}^T \boldsymbol{x^k} < 0\right\}} \left\{\alpha_k^2\right\}$$

$$k_\beta = \max_{\left\{k \middle| \text{logit}(b_k) - \boldsymbol{\mu}^T \boldsymbol{x^k} > 0\right\}} \left\{\beta_k^2\right\}.$$

Consider the solution $\tilde{v}_i = \gamma_i^2 \max\{k_\alpha, k_\beta\}$ for all i. This solution satisfies constraints D.2 and D.3 because $\gamma_1 = 1$, so $\tilde{v}_1 = \max\{k_\alpha, k_\beta\}$. As a result,

$$\boldsymbol{x^k}^T \text{diag}(\tilde{\boldsymbol{v}}) \boldsymbol{x^k} \geq \tilde{v}_1 = \max\{k_\alpha, k_\beta\} \geq \alpha_k^2$$

$$\boldsymbol{x^k}^T \text{diag}(\tilde{\boldsymbol{v}}) \boldsymbol{x^k} \geq \tilde{v}_1 = \max\{k_\alpha, k_\beta\} \geq \beta_k^2$$

hold true by condition 1 since $\left(x_1^k\right)^2 = 1^2 = 1$, and $\left(x_i^k\right)^2 \tilde{v}_i \geq 0$ since $\tilde{v}_i = \gamma_i^2 \max\{k_\alpha, k_\beta\} \geq 0$. Now, since $\gamma_i \neq 0$ by condition 2,

$$\frac{\tilde{v}_i}{\gamma_i^2} = \max\{k_\alpha, k_\beta\}$$

for all i, so

$$\max\{k_\alpha, k_\beta\} = \frac{\tilde{v}_i}{\gamma_i^2} = \frac{\tilde{v}_j}{\gamma_j^2} \leq r^2 \frac{\tilde{v}_j}{\gamma_j^2}$$

since $r \geq 1$ (by definition in Chapter 5), thereby satisfying constraints D.4. By construction, constraints D.5 are clearly satisfied since conditions 3 and 4 imply that

$$0 \leq v_{min} \leq \gamma_i^2 \max\{k_\alpha, k_\beta\} = \tilde{v}_i \leq v_{max}, \ \forall i \in \{1, \ldots, d\}.$$

Thus, $\tilde{v}$ is a feasible solution for v, so the feasible set of the LP is nonempty. ■

THIS PAGE INTENTIONALLY LEFT BLANK

Appendix E

Tractability Results

These tables show the mean time required for CPLEX to prove optimality for the 10 test problems generated, along with corresponding values of N^{max}, $|L|$, and $|R|$. These run-times should be considered approximate upper-bounds for problems of this size as they were designed so that each request was feasible on all planners.

A star in the "Time to Optimality" column indicates that the time required to construct the problem and prove optimality was longer than 120 seconds. A star in the "Optimality Gap Bound at 20 Seconds" column indicates that the best upper bound for the optimality gap was greater than 1 at 20 seconds, so the formulation should be run longer to obtain a better bound on the optimality gap. Note that in many cases it is possible for the formulation to obtain a very good result in not much extra time—e.g., the full formulation with 100 planners, 10000 requests, and $N^{max} = 1$ finds an optimal solution in 33.8071 seconds on average for the problems considered, yet at 20 seconds does not even have a reasonable optimality gap bound.

N^{max}	$\|L\|$	$\|R\|$	Time to Optimality (sec)	Optimality Gap Bound at 20 Seconds
1	20	250	0.3665	0
1	20	1000	0.9344	0
1	20	10000	5.9768	0
1	60	250	0.7410	0
1	60	1000	1.7361	0
1	60	10000	20.1350	0
1	100	250	0.9844	0
1	100	1000	2.6803	0
1	100	10000	33.8071	*
2	20	250	1.6795	0
2	20	1000	6.2765	0
2	20	10000	*	*
2	60	250	10.6787	0
2	60	1000	56.5532	*
2	60	10000	*	*
2	100	250	75.4695	*
2	100	1000	*	*
2	100	10000	*	*
3	20	250	12.8111	0
3	20	1000	58.8462	*
3	20	10000	*	*
3	60	250	*	*
3	60	1000	*	*
3	60	10000	*	*
3	100	250	*	*
3	100	1000	*	*
3	100	10000	*	*

Table E.1: Tractability Results for Full Formulation

N^{max}	$\|L\|$	$\|R\|$	Time to Optimality	Optimality Gap Bound at 20 Seconds
1	20	250	0.2372	0
1	20	1000	0.6399	0
1	20	10000	4.3523	0
1	60	250	0.4492	0
1	60	1000	1.2573	0
1	60	10000	15.2104	0
1	100	250	0.5337	0
1	100	1000	2.1014	0
1	100	10000	24.8572	*
2	20	250	0.2309	0
2	20	1000	0.6045	0
2	20	10000	4.0498	0
2	60	250	1.0500	0
2	60	1000	*	5.2047×10^{-5}
2	60	10000	*	3.2889×10^{-5}
2	100	250	1.0436	0
2	100	1000	*	5.7627×10^{-5}
2	100	10000	*	5.7304×10^{-5}
3	20	250	0.2217	0
3	20	1000	0.5164	0
3	20	10000	4.2948	0
3	60	250	0.6350	0
3	60	1000	*	3.9649×10^{-5}
3	60	10000	*	3.5024×10^{-5}
3	100	250	0.8283	0
3	100	1000	*	5.8939×10^{-5}
3	100	10000	*	3.6685×10^{-5}

Table E.2: Tractability Results for Heuristic Formulation

THIS PAGE INTENTIONALLY LEFT BLANK

Acronyms

ADP	approximate dynamic programming
BCF	Bayesian coin flip
BLR	Bayesian logistic regression
CP	coordination planner
DOD	Department of Defense
DP	dynamic programming
ECEF	Earth Centered-Earth Fixed
HS3	Hurricane and Severe Storm Sentinel
IP	integer programming
MAP	maximum a posteriori
NASA	National Aeronautics and Space Administration
NOAA	National Oceanic and Atmospheric Administration
NPP	National Polar-orbiting Partnership
PDF	probability density function
PMF	probability mass function
RMSE	root mean square error
UAV	unmanned aerial vehicle
USFS	United States Forest Service
USGS	United States Geological Survey

Index of Definitions and Important Terminology Uses

THIS PAGE INTENTIONALLY LEFT BLANK

Bibliography

[1] T. M. Herold, "Asynchronous, distributed optimization for the coordinated planning of air and space assets," Master's thesis, Massachusetts Institute of Technology, 2008.

[2] D. A. Vallado, *Fundamentals of Astrodynamics and Applications*, 2nd ed. El Segundo: Microcosm Press, 2001.

[3] J. Brill. (2013, April) Suomi NPP. National Aeronautics and Space Administration. [Online]. Available: http://npp.gsfc.nasa.gov/mission_details.html

[4] S. Graham. (2013, April) Aqua project science. National Aeronautics and Space Administration. [Online]. Available: http://aqua.nasa.gov/

[5] L. Jenner. (2013, April) NASA–HS3. National Aeronautics and Space Administration. [Online]. Available: http://www.nasa.gov/mission_pages/hurricanes/missions/hs3/index.html

[6] R. Marlaire. (2006, May) NASA and US Forest Service to tame wildfires with UAV capabilities. NASA Ames Research Center. [Online]. Available: http://www.nasa.gov/centers/ames/news/releases/2006/06_35AR.html

[7] Technology and development at the USDA Forest Service. US Forest Service. [Online]. Available: http://www.fs.fed.us/t-d/programs/im/aerial/uav.shtml

[8] "Joint doctrine for targeting," Joint Chiefs of Staff, Tech. Rep. Joint Publication 3-60, January 2002.

[9] P. Sakamoto, "Uav mission planning under uncertainty," Master's thesis, Massachusetts Institute of Technology, 2006.

[10] B. E. L. Negron, "Operational planning for multiple heterogeneous unmanned aerial vehicles in three dimensions," Master's thesis, Massachusetts Institute of Technology, 2009.

[11] P. Chandler and M. Pachter, "Hierarchical control for autonomous teams," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2001, pp. 632–642.

[12] D. Li and J. Cruz, "A robust hierarchical approach to multi-stage task allocation under uncertainty," in *44th IEEE Conference on Decision and Control, and 2005 European Control Conference*. IEEE, 2005, pp. 3375–3380.

[13] M. Abramson, D. Carter, S. Kolitz, M. Ricard, and P. Scheidler, "Real-time optimized earth observation autonomous planning," in *NASA Earth Science Technology Conference, Pasadena, CA*, 2002.

[14] E. Bensana, G. Verfaillie, C. Michelon-Edery, and N. Bataille, "Dealing with uncertainty when managing an earth observation satellite," in *Artificial Intelligence, Robotics and Automation in Space*, vol. 440, 1999, p. 205.

[15] M. Z. Spivey and W. B. Powell, "The dynamic assignment problem," *Transportation Science*, vol. 38, no. 4, pp. 399–419, 2004.

[16] H. Haas and A. Brown. (2004, February) Web services glossary. World Wide Web Consortium. [Online]. Available: http://www.w3.org/TR/ws-gloss/

[17] I. Simonis and J. Echterhoff, *OGC Sensor Planning Service Implementation Standard*, 2nd ed., Open Geospatial Consortium, March 2011.

[18] K. A. Delin and S. P. Jackson, "Sensor web: A new instrument concept," in *Symposium on Integrated Optics*. International Society for Optics and Photonics, 2001, pp. 1–9.

[19] (2011, December) Earth Observing 1 (EO-1). US Geological Survey. [Online]. Available: http://eo1.usgs.gov/

[20] C. Veness. GIS FAQ Q5.1: Great circle distance between 2 points. Movable Type Scripts. [Online]. Available: http://www.movable-type.co.uk/scripts/gis-faq-5.1.html

[21] R. Sinnett, "Virtues of the haversine," *Sky & Telescope*, vol. 68, no. 2, p. 159, August 1984.

[22] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 3rd ed. Belmont: Athena Scientific, 2005, vol. 1.

[23] ——, *Dynamic Programming and Optimal Control: Approximate Dynamic Programming*, 4th ed. Belmont: Athena Scientific, 2012, vol. 2.

[24] D. Bertsimas and J. N. Tsitsiklis, *Introduction to Linear Optimization*. Belmont: Athena Scientific and Dynamic Ideas, 1997.

[25] *IBM ILOG CPLEX Optimization Studio CPLEX User's Manual*, 12th ed., IBM Corporation, 2011.

[26] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis*. Boca Raton: Chapman & Hall/CRC, 2004.

[27] D. P. Bertsekas and J. N. Tsitsiklis, *Introduction to Probability*, 2nd ed. Belmont: Athena Scientific, 2008.

[28] A. Genkin, D. D. Lewis, and D. Madigan, "Large-scale bayesian logistic regression for text categorization," *Technometrics*, vol. 49, no. 3, pp. 291–304, 2007.

[29] M. Seeger, S. Gerwinn, and M. Bethge, "Bayesian inference for sparse generalized linear models," in *Machine Learning: ECML 2007*. Springer, 2007, pp. 298–309.

[30] S. M. O'Brien and D. B. Dunson, "Bayesian multivariate logistic regression," *Biometrics*, vol. 60, no. 3, pp. 739–746, 2004.

[31] A. D. Martin, K. M. Quinn, and J. H. Park, *Package 'MCMCpack'*, 1st ed., The Comprehensive R Archive Network, April 2013.

[32] A. Niculescu-Mizil and R. Caruana, "Obtaining calibrated probabilities from boosting," in *Proceedings of the 21st International Conference on Uncertainty in Artificial Intelligence*, 2005, pp. 413–420.

[33] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," *Advances in Large Margin Classifiers*, vol. 10, no. 3, pp. 61–74, 1999.

[34] D. Bertsimas and M. Sim, "Robust discrete optimization and network flows," *Mathematical Programming*, vol. 98, no. 1-3, pp. 49–71, 2003.