Asynchronous, Distributed Optimization for the Coordinated Planning of Air and Space Assets

By

Thomas Michael Herold
B.S. Operations Research
United States Air Force Academy, 2008

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 2010

Copyright © 2010 Thomas M. Herold. All rights reserved.

The author hereby grants to MIT permission to reproduce and to distribute publicly paper and electronic copies of this thesis document in whole or in part.

Signature of Author:

Sloan School of Management
Interdepartmental Program in Operations Research
14 May 2010
Approved by:

Mark A Abramson
Charles Stark Draper Laboratory, Inc.
Technical Supervisor
Certified by:

Hamsa Balakrishnan
Assistant Professor, Aeronautics and Astronautics
Thesis Advisor
Accepted by:

Dimitris J. Bertsimas
Boeing Professor of Operations Research
Co-Director, Operations Research Center
Asynchronous, Distributed Optimization for the Coordinated Planning of Air and Space Assets

by

Thomas Michael Herold

Submitted to the Sloan School of Management on May 14, 2010 in partial fulfillment of the requirements for the Degree Master of Science in Operations Research

ABSTRACT

Recent decades have seen the development of more advanced sensor and communication systems, with the future certainly holding more innovation in these areas. However, current operations involve “stovepipe” systems in which inefficiencies are inherent. In this thesis, we examine how to increase the value of Earth observations made by coordinating across multiple collection systems. We consider both air and space assets in an asynchronous and distributed environment. We consider requests with time windows and priority levels, some of which require simultaneous observations by different sensors. We consider how these improvements could impact Earth observing sensors in two use areas; climate studies and intelligence collection operations. The primary contributions of this thesis include our approach to the asynchronous and distributed nature of the problem and the development of a value function to facilitate the coordination of the observations with multiple surveillance assets.

We embed a carefully constructed value function in a simple optimization problem that we prove can be solved as a Linear Programming (LP) problem. We solve the optimization problem repeatedly over time to intelligently allocate requests to single-mission planners, or “sub-planners.” We then show that the value function performs as we intend through empirical and statistical analysis.

To test our methodologies, we integrate the coordination planner with two types of sub-planners, an Unmanned Aerial Vehicle (UAV) sub-planner, and a satellite sub-planner. We use the coordinator to generate observation plans for two notional operational Earth Science scenarios. Specifically, we show that coordination offers improvements in the priority of the requests serviced, the quality of those observations, and the ability to take dual collections. We conclude that a coordinated planning framework provides clear benefits.

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

Thesis Advisor: Hamsa Balakrishnan
Title: Assistant Professor
Aeronautics and Astronautics
ACKNOWLEDGEMENTS

I would like to express my profound appreciation to all those who have helped me in my endeavors for the past two years. First, I want to thank Mark Abramson and Dr. Stephan Kolitz of Draper Laboratory. Their commitment to not only my work but to my learning and development over the past two years has been something for which I am very thankful. I will undoubtedly be a more useful asset to the Air Force as a result of their guidance.

I would also like to thank Professor Hamsa Balakrishnan for taking me as her advisee despite her already enormous workload. I am grateful to have had a person of such patience and intellect to work with, and who genuinely cared about my work and my growth as a student.

Next, I want to thank Alex Kahn and Dr. David Carter of Draper Laboratory for their help on this project. I thank Alex for all of his help on the myriad topics he helped me with over the past year. I will miss our baseball discussions as well. I thank David for his patience in teaching me the intricacies of astrodynamics and other new concepts.

I would like to thank the Air Force Academy, MIT, and The Charles Stark Draper Laboratory for giving me this incredible opportunity to further my knowledge.

I must thank the great people and animals that have surrounded me these past few years and shared my experiences. Jake and Chris, I sincerely thank you for your friendship and for helping me get through these times. You guys are the best there is, and will make some of the finest UAV pilots the Air Force has seen, although I hope that isn’t the case. I also feel compelled to give a shout out to the P-Man for his undying and certainly unconditional love. Finally, Zeus, it has been an honor to have shared a home with you for two years.

I want to thank my family for their constant love and support. Mom, Dad, and Jim, I am so very thankful for everything you’ve done for me throughout my life, particularly this past year. I am the person I am today because of you.

Most importantly, I need to thank God for the gifts and opportunities He has given me so far in my life. Sic transit gloria mundi.

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

This thesis was prepared at The Charles Stark Draper Laboratory, Inc. under Internal Company Sponsored Research Projects 22917-001 and 23935-001, Collection Management Optimization and Collection Management Optimization 2.

Publication of this thesis does not constitute approval by Draper or the sponsoring agency of the findings or conclusions contained therein. It is published for the exchange and stimulation of ideas.

Thomas M. Herold, 2nd LT., USAF      May 14, 2010
# Table of Contents

LIST OF FIGURES.................................................................................................... 11

LIST OF TABLES...................................................................................................... 13

1 INTRODUCTION.................................................................................................. 15

1.1 Thesis Overview.................................................................................................. 16

1.2 Contributions...................................................................................................... 17

1.3 Thesis Motivation............................................................................................... 18

2 EARTH-OBSERVING SYSTEM COORDINATION PLANNER
OPERATIONAL CONCEPT ..................................................................................... 21

2.1 Earth-Observing Sensors .................................................................................. 22

2.1.1 Earth-Observing Sensors and Climatology .................................................. 22

2.1.2 Earth-Observing Sensors and Intelligence, Surveillance, and Reconnaissance in a Battlespace ................................................................. 27

2.1.3 Current Operations Framework .................................................................. 29

2.1.4 Future Operations Framework .................................................................... 36

2.2 The Coordination Planner within Larger Frameworks ..................................... 37

2.2.1 Collection Management Process ................................................................ 37

2.2.2 The Coordination Planner within a Web Service ........................................ 37

2.2.3 The Coordination Planner within a Sensor Web ........................................ 38

2.3 Real-World Scenarios ..................................................................................... 40

2.3.1 CLARREO Scenario .................................................................................. 40

2.3.2 WRAP Scenario ....................................................................................... 41

2.4 Earth Phenomena Observation System (EPOS) .............................................. 42

2.4.1 EPOS Functional Overview ...................................................................... 42
5.6.3 Dual Collect Inter-Observation Times ................................................................. 157

5.7 Dual Collection Scenario .......................................................................................... 159

5.8 WRAP Scenario Analysis ...................................................................................... 160

6 CONCLUSIONS AND FUTURE WORK ................................................................. 163

6.1 Summary of Contributions ....................................................................................... 163

6.2 Future Work ........................................................................................................... 165
   6.2.1 Opportunities for Human Interaction ............................................................... 165
   6.2.2 Adding Fidelity to the Coordination Planner .................................................. 166
   6.2.3 Statistical Studies of Real-World Planners ......................................................... 169
   6.2.4 Time-Delay Analysis ....................................................................................... 170
   6.2.5 Auction Algorithms ......................................................................................... 170
   6.2.6 Call-Center Routing ....................................................................................... 171

6.3 Conclusions ............................................................................................................ 171

APPENDIX A – LIST OF ACRONYMS ..................................................................... 173

APPENDIX B – TARGET-TO-SENSOR VALUE LOOKUP TABLE ............................. 177

APPENDIX C – PLANNING CYCLES CALCULATIONS ............................................. 179

APPENDIX D – TOTAL UNIMODULARITY PROOF ................................................. 181

APPENDIX E – REGRESSION OUTPUT ..................................................................... 185

APPENDIX F – PAIRED T-TESTS ............................................................................ 187

REFERENCES ........................................................................................................... 189
LIST OF FIGURES

FIGURE 2-1: FOREST FIRE SIZES OVER TIME [6].................................................................25
FIGURE 2-2: AIRS IMAGE OF VOLCANIC ASH, [14].........................................................27
FIGURE 2-3: JOINT TARGETING CYCLE [15]......................................................................28
FIGURE 2-4: SATELLITES OF THE "A-TRAIN" [19]...............................................................31
FIGURE 2-5: SIMULTANEOUS NADIR OVERPASSES (SNOS) AND SIMULTANEOUS CONICAL
OVERPASSES (SCOS)............................................................................................................33
FIGURE 2-6: ASYNCHRONOUS PLANNING CYCLES.............................................................35
FIGURE 2-7: CURRENT USER INTERACTION WITH "STOVEPIPE" SYSTEMS. ADAPTED FROM [24].
..................................................................................................................................................36
FIGURE 2-8: FUTURE FRAMEWORK OF CENTRALIZED SYSTEM........................................37
FIGURE 2-9: SENSOR WEB DIAGRAM [26]......................................................................39
FIGURE 2-10: PROPOSED CLARREO ORBITS [27]...............................................................41
FIGURE 2-11: EPOS STRUCTURE [30].................................................................................43
FIGURE 3-1: USING POINT TARGETS TO DEFINE AREAS OF INTEREST.........................50
FIGURE 3-2: PLANNING SYSTEMS.....................................................................................53
FIGURE 3-3: RELATIONSHIP BETWEEN PLANNING PHASES AND EXECUTION PHASES ....55
FIGURE 3-4: DEFINING PLANNING PERIODS.......................................................................57
FIGURE 3-5: DEPICTION OF ROUNDING HEURISTIC FOR COMPUTING SATELLITE-TO-TARGET
ACCESSSES ............................................................................................................................65
FIGURE 4-1: COORDINATION IN A HIERARCHICAL, DECENTRALIZED SCENARIO. ADAPTED
FROM [24]..................................................................................................................................89
FIGURE 4-2: FUNCTIONAL ARCHITECTURE ........................................................................91
FIGURE 4-3: PLANNING PERIOD EXAMPLE .......................................................................95
FIGURE 4-4: ASSIGNMENTS AT A SINGLE ITERATION....................................................112
FIGURE 4-5: ASSIGNMENT EXAMPLE. ADAPTED FROM [63].............................................113
FIGURE 4-6: SOFTWARE ARCHITECTURE .........................................................................118
FIGURE 4-7: ALGORITHMIC FLOW ....................................................................................119
FIGURE 4-8: SAMPLE REQUEST DATA...............................................................................120
FIGURE 4-9: SUB-PLANNER DATA....................................................................................121
FIGURE 4-10: SAMPLE OUTPUT..........................................................................................122
FIGURE 5-1: VARYING THE PRIORITY INPUT WEIGHT....................................................126
FIGURE 5-2: VARYING THE DUAL COLLECTION INPUT WEIGHT.....................................127

11
LIST OF TABLES

TABLE 2-1: INTELLIGENCE COLLECTION DISCIPLINES [16]................................................................. 29
TABLE 2-2: SATELLITE MISSION DEFINITIONS [13]......................................................................... 30
TABLE 3-1: REQUEST ATTRIBUTES .................................................................................................. 49
TABLE 3-2: KEPLER ELEMENTS [33]................................................................................................... 61
TABLE 4-1: SETS................................................................................................................................. 93
TABLE 4-2: REQUEST DATA.................................................................................................................. 93
TABLE 4-3: PLANNER DATA ................................................................................................................ 94
TABLE 4-4: SYSTEM DATA.................................................................................................................... 95
TABLE 4-5: SENSOR DATA ................................................................................................................... 96
TABLE 4-6: UAV SUB-PLANNER DATA ................................................................................................. 96
TABLE 4-7: SATELLITE SUB-PLANNER DATA ..................................................................................... 97
TABLE 4-8: VALUE FUNCTION SUMMARY ......................................................................................... 109
TABLE 4-9: DYNAMIC VALUE UPDATE DECISIONS......................................................................... 115
TABLE 4-10: PAIRING INPUT WEIGHTS WITH MOPS......................................................................... 116
TABLE 4-11: ALGORITHMIC FUNCTIONS BY SECTION .................................................................... 119
TABLE 5-1: TRAINING SET REGRESSION RESULTS.......................................................................... 135
TABLE 5-2: TEST SET REGRESSION RESULTS..................................................................................... 136
TABLE 5-3: RELATIONSHIPS FOUND BETWEEN VARIABLES THROUGH REGRESSION .......... 137
TABLE 5-4: SUMMARY OF REGRESSION MODEL PERFORMANCE.................................................. 140
TABLE 5-5: RESULTS FROM EXAMPLES OF WEIGHT VECTOR OPTIMIZATION ............................ 143
TABLE 5-6: ACTUAL VERSUS PREDICTED MOPS............................................................................... 144
TABLE 5-7: EFFECTS OF RANDOM VALUE INCREASES...................................................................... 150
TABLE 5-8: CLARREO SCENARIO ORBIT ANALYSIS CASES............................................................. 155
TABLE 5-9: CASE-BY-CASE RESULTS............................................................................................... 156
1 Introduction

The development of advanced sensor and communication systems has allowed for vastly improved Earth observation systems. Air- and space-based sensors can observe the Earth from a vantage unmatched by multiple ground-based sensors, and as the capabilities of new sensor systems increase, so does our ability to monitor Earth-based phenomena. However, current operations are hindered by the "stovepiped" nature of individual mission systems that limit the amount of coordination that exists amongst various remote-sensing platforms. We refer to this absence as the Coordination Planner Problem (CPP).

The purpose of this thesis is to examine this problem and develop an algorithm and the associated software to coordinate the planning of air- and space-based observation missions in a realistic environment. The algorithm developed addresses four primary challenges presented in the CPP. The first challenge is creating plans in an asynchronous environment, in which individual mission planners operate on different planning cycles. The second challenge results from the distributed nature of the problem in which individual mission planners operate independently of each other and are not entirely subservient to the coordination planner. The third challenge is to create an algorithm that can coordinate the observation of a single location with multiple assets, for example, an air- and a space-based sensor, called a dual collection. The fourth and final challenge deals with creating plans for both air- and space-based sensor assets.
This chapter introduces the various sections of this thesis. The first section gives a brief description of each of the chapters of this thesis. The second section highlights the main contributions of this thesis to the coordinated planning literature. The third section motivates this work and states the goals of this thesis.

1.1 Thesis Overview

This thesis discusses the CPP in six chapters. An overview of each of the remaining five chapters is given in the following paragraphs:

Chapter 2 describes the operational concept for the coordination planner in the context of real-world scenarios. This chapter explains the coordination planner's functional purpose in the context of Draper Laboratory's Earth Phenomena Observation System (EPOS). Most importantly, this chapter describes current operations and the inefficiencies they present. We use the chapter to explain how the coordination planner must operate in an asynchronous environment, as well as how it can be used in the larger context of a collection management process. Then, we outline real-world scenarios to which we can apply the results of this thesis. We introduce two scenarios whose objectives include monitoring Earth phenomena across multiple platforms. Specifically, we describe the Climate Absolute Radiance and Refractivity Observatory (CLARREO) mission and possible applications of coordinated planning within the context of the Wildfire Research and Applications Partnership (WRAP) program.

Chapter 3 explains the development of our model for the CPP. We first define terminology that helps the reader to understand the multiple facets of the CPP and to understand the scope of our model. This chapter also states the key assumptions made for our model, and provides an overview of the inputs, decisions, objectives, and constraints that we model for the CPP. Then, we give a description of the requirements of existing functional planning algorithms for the CPP's sub-planners.

Chapter 4 details the planning algorithm we implement to address the problem. The chapter first reviews literature relevant to the CPP and explains how past work influences the choice of our mathematical model. The chapter goes on to detail the value function that we use for the optimization problems we solve over time. The third section describes these optimization problems that decide which tasks to query on each of the sub-planners at any time. The fourth section explains how our models are incorporated into software.
Chapter 5 presents results of coordinated observation planning using our approaches on various test scenarios. We first describe analysis conducted on the value function to ensure it performs as we intend. Specifically, we analyze the effects of increasing the priority, quality of observations, and dual collect input weights empirically, and analyze the performance of the value function as a whole using statistical models. We then demonstrate how we can embed the statistical models we create within non-linear optimization problems to enable a user of the coordination planner to more accurately choose input weights to meet his objectives. This chapter also conducts analysis on the impact of additional communication with sub-planners. In addition, we discuss the benefit of querying targets on sub-planners repeatedly despite past rejections. We quantify the benefit of coordination compared against a baseline scenario of our creation. The final sections of this chapter analyze the effects of additional UAV and satellite assets in the notional CLARREO and wildfire scenarios.

Chapter 6 summarizes the contributions of this thesis. It also provides recommendations for future work on this topic, including adding fidelity to our coordination planner and addressing the problem from other perspectives. The chapter then summarizes the conclusions found from our work.

1.2 Contributions

In developing an algorithm to address the coordination planner problem in an asynchronous, distributed context, this thesis makes the following contributions:

- A complex value function that is used as an intermediate construct to build observation plans that align with user end objectives. We use a forward-looking value function that accounts for the physical and temporal constraints of the problem and can be tuned to emphasize any of four user Measures of Performance (MOPs).

- An optimization problem that solves the problem of intelligently querying sub-planners at any instance in time and can be solved as a Linear Programming (LP) problem. Our algorithm solves a series of optimization problems over time to intelligently interact with two types of sub-planners and gain the most valuable feedback at the current time. Although the optimization problems are designed to be
Integer Programming (IP) problems, the structure of the constraint matrix is such that they can be solved as LP problems.

- **Integration of the planning algorithm with two types of sub-planners, an Unmanned Aerial Vehicle (UAV) sub-planner and a satellite sub-planner.** The algorithm is embedded in software controlled in MATLAB that interacts with a UAV sub-planner and a satellite sub-planner, each running their own planning algorithms.

- **Empirical and statistical testing and analysis of the value function.** We perform empirical analysis of the value function on large-scale problem instances to demonstrate how a user can emphasize his objectives using the input weight vector. We also provide statistical analysis, in the form of Ordinary Least Squares (OLS), stepwise, and ridge regression to identify and quantify relationships between each value function component and the four MOPs.

- **Development and testing of operational scenarios to demonstrate the effectiveness of the algorithm we develop and to demonstrate the benefit of coordinated planning, in general.** We test the effectiveness of our approach by creating observation plans for relevant operational scenarios. Specifically, we perform demonstrative orbit analysis of the CLARREO mission and provide insight into the marginal benefit of additional UAVs in a notional western United States wildfire scenario.

- **Recommendations for future work on the coordinated planning problem.** Because this thesis is only a first-look at coordinated planning in an asynchronous and distributed framework, we identify areas for improvement for our approaches as well as areas of future work on this class of problems.

### 1.3 Thesis Motivation

The purpose of this thesis is to develop an algorithm to coordinate the observations of target locations with multiple air and space assets in an asynchronous and distributed environment. The plans must satisfy the constraints of satellite and UAV observation planning problems and be able to emphasize user-specified MOPs. The goal of our work is to develop
algorithms that generate observation plans that allow for the collection of valuable data for the Earth Science and intelligence communities.

We apply the algorithm to notional operational scenarios involving multiple, heterogeneous UAVs and satellite-based sensors. The scenarios of interest include the proposed CLARREO mission and a notional WRAP scenario. These scenarios emphasize climate and natural disaster monitoring, respectively. However, our algorithm is applicable to any scenario in which the coordination of the observations of multiple assets is important. In addition to Earth Science observation and intelligence collection missions, we estimate that coordination in a distributed and asynchronous planning framework could provide benefits for climate emissions treaty monitoring and even missile defense missions.
THIS PAGE INTENTIONALLY LEFT BLANK
2 Earth-Observing System Coordination Planner Operational Concept

In this chapter, we present the operational concept for the coordination planner in the context of a number of real-world scenarios. Specifically, we present the coordination planner in the context of Draper Laboratory’s Earth Phenomena Observation System (EPOS). The first section describes the importance of Earth-observing sensors in monitoring climate change and gathering battlespace intelligence. We describe current operations and the inefficiencies they present. The section describes two modes in which the coordination planner could operate: an asynchronous mode, in which the planner optimizes the coordination of asynchronous, distributed data collection assets, and a synchronous mode, which involves optimized coordination of the tasking of heterogeneous sensors on air, space, and ground platforms with synchronized planning cycles. The former case is the focus of this thesis, while the latter is addressed in other studies. The second section describes how the coordination planner can be used in the larger context of a collection management process. We include descriptions of a web-based service for remote sensor users and a futuristic Sensor Web. The third section outlines real-world scenarios to which we can directly apply the results of this thesis. We introduce two scenarios whose objectives include monitoring Earth phenomena across multiple platforms. The scenarios serve as situations for which multi-mission coordination would be
extremely beneficial. The fourth section details the capabilities of EPOS and its functional purpose. This section also describes the EPOS concept in the context of a futuristic Sensor Web. We then describe the role of the coordination planner within EPOS. An outline of the functional purpose of the coordination planner provides a clear meaning of the potential impact the coordinator could have on Earth Science and military intelligence data collection. The fifth section presents our goals in addressing the Coordination Planner Problem (CPP) in this thesis.

2.1 Earth-Observing Sensors

Recent decades have seen the development of more advanced sensor and communication systems, with the future certainly holding more innovation in these areas. Air- and space-borne sensors can observe the Earth from a vantage unmatched by multiple ground-based sensors, and as the capabilities of new sensor systems increase, so does our ability to monitor Earth-based phenomena. In this thesis, we examine how to increase benefits gained by coordinating observations across multiple platforms. In particular, we consider how these improvements could impact the use of Earth-observing sensors in two areas: climate studies and military planning operations. Both areas involve dynamic phenomena for which highly coordinated observations are valuable.

2.1.1 Earth-Observing Sensors and Climatology

In the last 100 years, a number of climate-related issues have become significant on the international stage. Meanwhile, remote sensing missions are now capable of collecting and storing accurate, persistent measurements of climate data on a large scale. For example, archived satellite data enables us to observe the “steady clearing of the world’s rainforests, an apparent annual rise in sea level approaching 2 mm a year and the depletion of the ozone layer by atmospheric pollution” [1]. Moreover, the National Aeronautics and Space Administration (NASA) began the Earth-Observing System (EOS) program in 1991 as a result of a United States Presidential initiative “to provide in-depth scientific understanding about the functioning of Earth as a system”[2]. The EOS Science Plan outlines a number of Earth Science issues of national interest and how current satellite systems gain valuable information on these issues. The following sections describe several applications for which Earth-observing sensors contribute to situational awareness and an understanding of Earth’s climate.
2.1.1.1 Climate Monitoring

This subsection describes three climatology issues for which Earth observing sensors could provide important information. These issues motivate our efforts to improve the efficiency of Earth observation systems.

**Issue 1: Accurate prediction of seasonal precipitation and temperature changes**

Earth-observing sensors contribute to the understanding of both natural and human factors that affect the seasonal climate and related weather phenomena. Orbiting satellites provide measurements of a range of spectral bands, and this data is useful for analyzing changes in atmospheric composition and types of terrain. The Moderate Resolution Imaging Spectroradiometer, or MODIS, instruments are on-board the Aqua and Terra satellites and provide whole-Earth coverage of 36 spectral bands every 1 to 2 days [3]. The fusion of humidity (from the Atmospheric Infrared Sounder (AIRS)), ocean color (MODIS), and cloud-radiation budget (Clouds and the Earth's Radiant Energy System (CERES)) measurements enable scientists to improve models of air and sea interaction that can be used to predict anomalies such as El Niño.

**Issue 2: Ecosystem change and biodiversity**

This issue relates to the problem of species loss as a result of human population growth. Remote sensors can track global changes in terrain patterns, “including expansion and contraction of farmland, urban growth, deforestation, and forest regrowth” [2]. This helps scientists monitor habitat damage for the goal of preserving species.

**Issue 3: Long-term climate change and global warming**

Earth-observing sensors can contribute valuable data to the study of climate change and global warming. The most sophisticated prediction models suggest a global warming of between 1.8° and 4.0° C by the time the amount of carbon dioxide (CO₂) in the atmosphere doubles over pre-Industrial levels [4]. This is projected to occur sometime in the late 21st century. These lower and upper bounds, if realized, would have vastly different effects on the environment. Remote sensors take measurements of the amount of CO₂ and other gases, in the atmosphere and are thus vital to global warming prediction models. Improving the frequency and quality of these data collections can reduce the uncertainty inherent in the predictions.
In addition to monitoring long-term Earth climate and ecosystem changes, air- and space-based sensors can aid in the more immediate task of tracking developing natural disasters and those that are already in-progress.

2.1.1.2 Disaster Monitoring

Earth-observing sensors are instrumental in monitoring natural disasters, as air- and space-based sensors provide the ability to observe such large-scale phenomena. Moreover, remote sensing entails little or no risk to humans. Natural disasters are dynamic systems for which movement prediction is a difficult problem. Coordinating observations of these phenomena over time can improve predictive models. The accuracy of predictive models influences the effectiveness of any safety measures implemented to minimize loss of money, infrastructure, and life. As the human population grows, so do the costs these natural weather phenomena impose on society. One potential application of the results of this thesis is to coordinate the tasking of independent mission planners in order to improve the quality of disaster analysis and prediction models, and hence reduce the potential costs these phenomena could inflict on human populations.

2.1.1.2.1 Wildfires

One specific natural phenomenon for which coordinated planning could be of particular benefit is wildfire. Wildfires have been growing in size and strength since the United States Forest Service (USFS) more actively executed preventative measures in the 1980's. This is largely due to the unforeseen consequences of strict forest fire prevention. After a number of large fires ravaged the northwest U.S. in 1910, the USFS became determined to protect the nation’s forests from wildfire and by 1960, forest fire prevention methods virtually eliminated fires in the U.S. This caused shrubs and trees such as the Douglas fir, a tree that is very susceptible to drought and fire, to flourish. Moreover, brush accumulated on the forest floor up to six feet high, meaning fires could reach from the forest floor to trees that had survived many previous fires [5]. Thus, the preventative measures unintentionally created a situation where even small fires could develop into enormous and uncontrollable ones. One can see in Figure 2-1 that in the 1970’s the average fire encompasses only 20 acres, but by 2002 this statistic had risen to 96 acres per fire, on average.
From 2002-2008, an average of over 7.5 million acres burned each year in the U.S. Costs, both from damage inflicted by wildfires and measures taken to combat the fires, can total several millions of dollars per day. The USFS has since teamed with NASA to demonstrate the use of unmanned aerial vehicles (UAVs) in monitoring wildfires since June 2006. Air- and space-based sensors offer wide-angle views of the fires and electromagnetic radiation measurements that can aid firefighters in predicting the movements of fires and safely combating them.

2.1.1.2.2 Hurricanes

Hurricanes inflict major damage on U.S. and international coasts. It is estimated that over the past century, hurricanes inflicted an annual cost of nearly $10 billion to the United States [7]. These phenomena are yet another example of a natural disaster for which extensive data collection can provide benefit.

Hurricane development involves a combination of complex atmospheric conditions and ocean current patterns. This means coordination amongst air-, space- and ground/sea-based observation platforms is necessary to provide the useful data collection plans. Currently, the National Oceanic and Atmospheric Administration (NOAA) uses two types of aircraft, satellite data, surface ship data, and buoy systems to create their forecasts. The NOAA Aircraft Operations Center, located at MacDill Air Force Base (AFB), has used the P-3 Orion and the Gulfstream IV for air-based reconnaissance since the early 1990's. NOAA combines the data acquired by these assets with measurements gained by U.S. Air Force Lockheed-Martin WC-130J aircraft that conduct the bulk of “hurricane hunting”[8]. As of early 2009, however, under the Global Hawk Pacific 2009 (GLOPAC) mission, various scientific instruments have been placed on an unmanned Global Hawk aircraft that can fly longer and at higher altitudes than
any of the manned aircraft currently in use. Thus, oceanic and atmospheric data that are critical in hurricane path prediction are gathered from many sources but it is rare that these data collection missions are actively coordinated with each other.

2.1.1.2.3 Volcanoes

From 1980-2008, costs from volcanic activity totaled nearly $2.9 billion in economic damage, and an average of 869 human lives per year [9]. While volcanoes caused only 90 American deaths over this period, the U.S. suffered the second most economic damages of all countries.

Magma, a mixture of liquid rock, crystals, and dissolved gas, is expelled onto the Earth's surface under certain conditions and erodes the land on which it flows, forming a volcano. This volcano then becomes a vent through which more magma and its gases will eventually discharge [10]. In addition to the obvious effects of hot magma on the Earth’s surface, volcanic eruptions cause longer-term climate changes. According to Alan Robock, Professor of Meteorology at Rutgers University, a large explosive volcano can cause significant ozone depletion and, in turn, enhanced ultraviolet (UV) radiation for up to 2 years. It can lead to cooler summer temperatures in the northern hemisphere tropics or even significant global cooling for 1-3 years[11].

To mitigate the costs of volcanic eruptions, preparation and early warnings are vital. In addition to using images of volcanoes to identify signs of eruptions, scientists have developed sensors to “detect heat, sulfur dioxide and small changes in the shape of earth's surface” [12]. One space-based instrument useful in identifying and tracking the movement of volcanic ash and changes in heat levels is AIRS, on board NASA’s Aqua satellite. Also, one mission of the satellite NOAA 18, as outlined by the European Space Agency (ESA), involves “volcanic eruption monitoring and detection”[13]. The AIRS instrument measures temperature and water vapor as a function of height. This is valuable in monitoring the effects of volcanic activity. Below is an image taken by AIRS of Soufriere Hills Volcano on Montserrat Island in the Caribbean shortly after it erupted in 2007:
Figure 2-2 is the combination of images taken by AIRS over several passes. It shows the movement of volcanic ash and gases after the eruption. Images such as this one enabled air traffic controllers to redirect commercial airliners around these areas. This type of imagery was of interest shortly after the April 2010 eruption of the Eyjafjallajökull volcano in Iceland, which disrupted air travel in Europe substantially.

Large-scale phenomena such as wildfires, hurricanes, or volcanoes require observations to track how the phenomenon progresses. Thus, coordinating the observations of many different sensors to efficiently track these phenomena is quite useful.

2.1.2 Earth-Observing Sensors and Intelligence, Surveillance, and Reconnaissance in a Battlespace

The concept of coordinated planning is equally applicable to military planning cycles in which intelligence collection and Battle Damage Assessment (BDA) are vital. During a conflict, the Area of Responsibility (AOR) is the region for which a Combatant Commander (CCDR) is accountable. The Joint Force Commander (JFC), subordinate to the CCDR, is tasked with completing the CCDR’s objectives through integrating the joint functions of different services. We focus on the coordinator’s use in improving the joint targeting cycle’s efficiency. The joint targeting process consists of establishing objectives, gathering intelligence, planning for targeting, and assessing the performance of combat operations; this is a dynamic process that
requires timely target selection and highly accurate feedback. As stated in Joint Publication 3-60 (on Joint Targeting), “All potential targets and all targets nominated for attack continually change in importance due to the dynamic nature of the evolving environment in the battlespace” [15]. Figure 2-3 below shows the Joint Targeting Cycle.

![Joint Targeting Cycle](image)

Figure 2-3: Joint Targeting Cycle [15]

Requests for observation begin with an *information need*. An information need answers an abstract question that is of importance to the commander in his AOR. These information needs are sometimes referred to as Essential Elements of Intelligence (EEI). An intelligence collection plan begins by considering all possible sources and methods to satisfy the information need. Most units have access to some or all of the following intelligence gathering sources: Signals Intelligence (SIGINT), Imagery Intelligence (IMINT), Measurement and Signature Intelligence (MASINT), Human-Source Intelligence (HUMINT), Open-Source Intelligence (OSINT), and Geospatial Intelligence (GEOINT). Air-, space-, and ground- based collection platforms collect data in the SIGINT, IMINT, MASINT, and GEOINT realms. A different intelligence agency is responsible for each of these intelligence collection disciplines; we present the intelligence types and the organizations responsible for their collection in Table 2-1:
<table>
<thead>
<tr>
<th>Intelligence Type</th>
<th>Primary Collector</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIGINT</td>
<td>National Security Agency (NSA)</td>
</tr>
<tr>
<td>IMINT</td>
<td>National Geospatial-Intelligence Agency (NGA)</td>
</tr>
<tr>
<td>MASINT</td>
<td>Defense Intelligence Agency (DIA)</td>
</tr>
<tr>
<td>GEOINT</td>
<td>NSA, NGA, DIA, and others</td>
</tr>
</tbody>
</table>

Table 2-1: Intelligence Collection Disciplines [16]

The fact that different agencies collect different types of intelligence highlights the need for coordination amongst the agencies and the assets they control. This is especially true given the chaotic and rapidly changing nature of the battlespace environment.

In that line, many in-theatre targets are deemed “Time-Sensitive-Targets” (TSTs) by the CCDR during target development and prioritization. TSTs are targets that need to be identified, tracked, and possibly attacked as soon as possible. The Joint Doctrine states that TSTs can require “both dynamic prosecution and cross-component coordination and assistance in a time-compressed fashion” [15]. Thus, it is evident that coordinating amongst assets in-theatre and under JFC control (UAVs), assets under the control of other organizations (NSA, NGA, DIA, etc), and even commercial satellites, could be of tremendous benefit to a JFC in this context.

2.1.3 Current Operations Framework

In both the Earth Science and Intelligence Collection (IC) worlds, the structure of current operations makes inter-mission coordination difficult. The following sections describe in some detail this difficulty and the resulting inefficiencies.

2.1.3.1 “Stovepipe” Systems

As discussed above, the observation plans made by individual satellites, UAVs, and, in the future, even Unmanned Surface Vessels (USVs), are usually created by different agencies. Current operations in the Earth Science and IC worlds involve “stovepipe” systems, in which individual missions are managed independently. For example, Earth Observing-1 (EO-1) was launched on November 21, 2000 by NASA on a technology validation/demonstration mission of various instruments and a spacecraft bus. EO-1 has since moved on to a number of Extended Missions, including testing new technologies and adding a high-degree of autonomy to its mission planning and data collection processing. While EO-1’s schedule planning has been adjusted according to the information obtained by other sensors (MODIS, on NASA’s Aqua and Terra satellites) during a series of Sensor Web Experiments [17], it remains under managerial
control of NASA and has not directly coordinated its planning beyond this sensor web. We are unaware of other satellites that participate in coordinated planning with other independent missions. Table 2-2 identifies the missions of five satellites whose ephemerides are notionally used in this thesis:

<table>
<thead>
<tr>
<th>Satellite</th>
<th>Launch Date</th>
<th>Mission</th>
<th>Mission Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aqua</td>
<td>04 May 2002</td>
<td>Atmospheric dynamics/water and energy cycles, cloud formation, precipitation and radiative properties, air/sea fluxes of energy and moisture, sea ice extent and heat exchange with the atmosphere.</td>
<td>NASA</td>
</tr>
<tr>
<td>Terra</td>
<td>18 December 1999</td>
<td>Atmospheric dynamics/water and energy cycles, Atmospheric chemistry, Physical and radiative properties of clouds, air/land exchanges of energy, carbon and water, vertical profiles of CO and methane vulcanology.</td>
<td>NASA</td>
</tr>
<tr>
<td>Tropical Rainfall Measuring Mission (TRMM)</td>
<td>27 November 1997</td>
<td>Monitor and study tropical rainfall.</td>
<td>NASA, Japan Aerospace Exploration Agency (JAXA)</td>
</tr>
<tr>
<td>Satellite Pour l'Observation de la Terre (SPOT) - 5</td>
<td>04 May 2002</td>
<td>Cartography, land surface, agriculture and forestry, civil planning and mapping, digital terrain models, environmental monitoring.</td>
<td>Centre National d'Études Spatiales (CNES, French government space agency)</td>
</tr>
<tr>
<td>NPOESS-1</td>
<td>2013*</td>
<td>Meteorological, climatic, terrestrial, oceanographic, and solar-geophysical applications; global and regional environmental monitoring, search and rescue, data collection.</td>
<td>NOAA</td>
</tr>
</tbody>
</table>

Table 2-2: Satellite Mission Definitions [13]

*Proposed Launch Date, Cancelled as of February 2010

The main reason these "stovepipe" systems provide few coordinated observations is that coordinating with other assets was never their primary mission. Many missions were designed for global surveying, i.e. continuous observation of the Earth. This means that data are eventually available for most locations on the Earth but perhaps not at the times and with the characteristics (resolution, spectral bands observed, etc.) a user might desire. Sensor technology has developed higher resolution instruments at the cost of smaller areas imaged. These new developments add to the need for more careful planning and scheduling, particularly when planning observation schedules for eventual data fusion or calibration purposes.
2.1.3.2 Sensor Cross-Calibration and Cross-Targeting

Simultaneously observing an area with different sensors has a number of benefits. According to the Goddard Space Flight Center, these simultaneous viewings allow for "synergistic measurements where data from several different satellites can be used together to obtain comprehensive information about various key atmospheric components or processes" [18]. In the past, NASA has tried to do so through choosing orbits such that near-simultaneous viewing opportunities were inherently present. However, "formation flying" of satellites is not only difficult, but expensive. It requires close monitoring of satellite trajectories and frequent inputs to maintain these paths. One example is the creation of the "A-Train," a collection of satellites carefully placed in orbits to allow for near-simultaneous viewings. The "A-Train," also known as the EOS PM Constellation (due to the fact that Aura passes over the equator, going north, daily at 1:30 PM), consists of six satellites, some with multiple sensors on-board. The satellites include: Aura, Polarization & Anisotropy of Reflectances for Atmospheric Sciences coupled with Observations from a Lidar (PARASOL), Cloud-Aerosol Lidar and Infrared Pathfinder Satellite Observation (CALIPSO), CloudSat, Aqua, and Orbiting Carbon Observatory (OCO, failed to reach its intended orbit after launch in February 2009). NASA also employs an AM-constellation (the satellites cross the equator, going north, daily at 10:30 AM) with a similar structure and objective. Below is a diagram of the "A-Train" formation:

![Figure 2-4: Satellites of the "A-Train" [19]](image)

The AM and PM constellations operate in this manner because most Earth Science collection instruments are always acquiring data. That is, scientists do not often request that the sensor points at some location. Instead, the continual observations are downloaded to an
archive from which scientists can access the data. There are some sensors, however, (such as Terra’s Advanced Spaceborne Thermal Emission and Reflection Radiometer (ASTER)) that accept requests and are “taskable.” Moreover, as more specialized and finer resolution sensors are produced, tasking of sensors might become increasingly more relevant.

The Goddard Space Flight Center outlines a number of Earth Science-related questions for which simultaneous viewings generate data that could help to provide answers. These questions include:

**Question 1:** What are the aerosol types and how do observations match global emission and transport models?

The measurements of aerosol heights by Cloud-Aerosol Lidar with Orthogonal Polarization (CALIOP, the main instrument on CALIPSO) and aerosol sizes by MODIS (Aqua and Terra) are combined to provide a global distribution of aerosols to test against current models.

**Question 2:** How does cloud layering affect the Earth’s Radiation Budget (ERB)?

Here, NASA uses sensors on CloudSat, as well as MODIS, to provide the first global survey of vertical cloud structure. Data from MODIS enhances the capabilities of CloudSat assets to detect clouds, and vice versa.

**Question 3:** What is the vertical distribution of cloud water/ice in cloud systems?

Mixed phase clouds (clouds composed of both water and ice) are not well understood, leading to incorrect classifications in many weather and cloud forecasts. To create models of mixed phase clouds, simultaneous observations from CALIOP and Polarization and Directionality of the Earth’s Reflectances (POLDER, a sensor on PARASOL) are useful. This is yet another example of the scientific benefit of simultaneous observations.

Many of the observations and measurements discussed previously lose their value if they are slightly inaccurate. Yearly changes in climate are so small that slight errors in measuring these changes can cause grossly inaccurate estimates. According to the Intergovernmental Panel on Climate Change (IPCC), the average near-surface temperature on Earth has increased somewhere between 1° and 1.6° since around 1950 and is projected to continue to increase throughout this century. The accuracy of measurements such as these strongly influences the usefulness of weather prediction models and is important to understanding the global warming issue.
Much literature is available on examples of the direct benefit of sensor calibrations and simultaneous viewings. In [20], the authors address using space- (Measurements of Pollution in the Atmosphere (MOPITT)) and ground-based measurements of ozone (O₃) and carbon monoxide (CO) to diagnose the evolution of the presence of these chemicals in the atmosphere over East Asia. Other research is devoted to improving scientists' ability to calibrate instruments once the samples are obtained, such as that in [21]. This is a difficult problem in itself, which is the reason Simultaneous Nadir Overpasses (SNOs) have become a common way to choose simultaneous viewings post facto [22].

SNOs are moments in time where two (or more) satellites at different altitudes pass directly over the same location on the Earth almost simultaneously (usually within sixty seconds). This means that data collected at these times by each asset involved are from the same angle, and this makes post-processing and calibration considerably easier. Simultaneous Conical Overpasses (SCOs) [23] can also be used. SCOs occur when the sensors involved are not necessarily pointing towards nadir when the overpass occurs. This means that the observations are taken at different angles and through different portions of the atmosphere. There are more opportunities for these viewings, but the task of calibration becomes more difficult. The figure below depicts examples of a SNO and a SCO:

![Diagram of SNO and SCO](image_url)

Figure 2-5: Simultaneous Nadir Overpasses (SNOs) and Simultaneous Conical Overpasses (SCOs)
The concepts discussed above apply to the IC world as well. The value of highly accurate weapons is lessened if intelligence cannot determine target locations accurately and in a timely manner. IC collection assets are more likely to have fine resolution sensors given that their objectives include surveying specific structures and tracking individuals, resulting in a smaller viewable region and fewer opportunities for observations. Thus, the nature of these missions emphasizes the need for coordination among available assets. A dynamic and coordinated planning system could enable near-real-time observation of TSTs by using the sensors of many single-mission planning systems.

Currently, however, most coordinated observations are taken *post facto* from archived data, with little or no real-time coordination generally occurring across sensors. The issue is summarized most succinctly in the following excerpt from the EOS Science Plan document:

> "Simultaneous observations with a group of sensors on the same platform (satellite)... taken together, improve either the accuracy or the scientific content of observations in comparison with measurements from a single instrument. This strategy depends on a close coordination in space and time, and is generally easier, if not absolutely required, on a single satellite." [2]

As a result of the difficulty of coordination, sensor systems are inefficiently utilized compared to what could be achieved if they were coordinated.

2.1.3.3 *Asynchronous Systems*

The way in which these independent missions are managed presents yet another complication; it also means that these missions have their own planning cycles and methods. Planning algorithms, the times at which newly-created plans are uploaded to the assets, and even maintenance periods, differ in different missions. These systems have their own planning objectives and carry out those objectives on their own schedules. The figure below depicts this feature of current operations:
Asynchronous Planning Cycles

Figure 2-6: Asynchronous Planning Cycles

This figure illustrates a notional scenario where there are four independent planners; one plans a UAV mission, while the other three plan satellite missions. Each planner operates on its own schedule in that each creates, uploads, and executes plans at different times. Times at which satellites are unavailable due to scheduled maintenance could also be included in this diagram, which would add to the complexity of the problem.

2.1.3.4 User Communities

Another real-world issue to consider in understanding this problem is the existence of disparate user communities. Currently, each community, whether it consists of scientists, news offices, military personnel, etc., searches for a sensor platform that would satisfy its collection need independently. Not only do they do so independently, but users begin to develop preferences for certain platforms. This could occur because the platform usually provides satisfactory data when it is requested, or simply because the user does not know that other platforms can provide the same or better service. Figure 2-7 illustrates the way in which users most often interact with independent mission planners.
In fact, it is most often the case that users rarely approach more than one planner when trying to obtain the data they desire. This behavior enforces the "stovepiped" nature of current operations.

2.1.4 Future Operations Framework

It would be beneficial for users if the coordinator could schedule tasks on all assets, air, space, and ground, in a synchronous manner where all assets are subject to the same planning cycle, the coordinator's planning cycle. All users seeking some data that could be obtained either from past or future observations would use the coordinator to find what they need. However, as discussed above, current operations do not provide this context. A transformation to this framework would require an overhaul of organizational structure and this is unrealistic, at least in the near future. It is difficult to imagine an environment where, for example, NSA, NGA, and Central Intelligence Agency (CIA) surveillance assets plan their daily missions in a coordinated manner. However, a goal of this thesis is to demonstrate the benefit that coordinated planning could provide, even in an asynchronous and hierarchical environment. Below, we present a diagram to illustrate the future operations framework:
2.2 The Coordination Planner within Larger Frameworks

2.2.1 Collection Management Process

According to the Open Geospatial Consortium (OGC), collection management is defined as "utilization and coordination of the resources involved in collecting information"[25]. Collection management is a process that extends beyond coordinating planning of assets. It involves clearly defining and refining information and collection needs, creating missions that can address those needs, planning to collect data to satisfy the needs, and translating that data into more useful information. The following subsections briefly describe how the coordination planner fits into this larger collection management process.

2.2.2 The Coordination Planner within a Web Service

The concept of coordinating the planning of sensor and imager activities and using the data they collect applies to many areas of interest. This includes the aforementioned Earth Science and IC realms. The data collection needs of each of these communities require very similar inputs and outputs. As such, it is logical to generate a standard procedure by which users may access sensors for any objective they wish. The Open Geospatial Consortium (OGC) has developed such standards for providing web service, one of which is the Sensor Planning Service (SPS), which allows web access to various sensors and the data collections they make.
The coordination planner could fit into an automated system that begins with a user approaching the coordinator in a web-based service as specified by the OGC.

When a user invokes an SPS, he might request information on a topic of interest from historical data, or he might try to acquire new information through submitting a request to various surveillance/reconnaissance assets through the web service. All requests made by the user are exactly that—requests—not guarantees of access to past information or of successful asset assignments. The user makes requests to the web service, and the planner within the web service chooses which requests to satisfy (i.e. plan for) such that the science or intelligence value of the decisions in each planning period is maximized. Initially, the user could search for information on any topic or use the web service to refine their request. The user accesses the web service, which then allows the user to choose one of the following three options:

Option 1: The user is asked to choose a time window in which to schedule a new request.
Option 2: The user is given a list of topics on which the system can find data, including categories for current events of interest (i.e. fires, floods, hurricanes, etc).
Option 3: The user has an idea for a request but only in general terms, and would therefore like an advanced request form that helps the user to refine their search.

For the remainder of this thesis, we assume users have chosen Option 1 and input their request information. When the coordinator has reached some threshold of requests received or time elapsed since the last planning cycle, the coordinator begins planning the collection of data to satisfy these requests. The coordination planner must choose which requests to allocate to the sub-planners, and must give feedback to the users to alert them when their requests have been scheduled.

2.2.3 The Coordination Planner within a Sensor Web

Another example of a collection management process that could be enabled by an SPS is a Sensor Web. The concept of a Sensor Web was first described by NASA to take advantage of improvements in sensor and communications capabilities. The system is “capable of automated reasoning for it can perform intelligent autonomous operations in uncertain environments, respond to changing environmental conditions, and carry out automated diagnosis and recovery” [26]. Thus, a Sensor Web is enabled by coordination of air, space, and ground sensors. It is important, however, to distinguish between a Sensor Web and “Distributed Sensors” or
"Sensor Networks." Distributed sensor networks simply gather the information obtained by multiple sensors and communicate it to a central node. A Sensor Web, however, seeks a closed-loop system in which the information gained by all assets is shared amongst the entire web, and new observation plans are dynamically created based on this additional information. As such, the stated goal of a Sensor Web “is to extract knowledge from the data collected and adapt and react accordingly”[26]. A Sensor Web, then, has three components:

- The sources of data, or sensor platforms that take observations
- The processing nodes, where data are transformed into more useful information
- The planning/tasking process, which uses the newly compiled information and creates new tasks for the sensors to carry out, completing the cycle

A diagram of a notional Sensor Web, with interaction between various distributed sensors, is shown in Figure 2-9 below:

![Sensor Web Diagram](image)

Figure 2-9: Sensor Web Diagram [26]

The benefits this type of system provides are clear. Tasks such as maintenance, calibration, and downloads that generally disrupt a mission’s planning cycle can be performed by a Sensor Web while the system remains intact. In current operations, missions are hindered by these tasks, but in a coordinated system this could be avoided. Moreover, the system can
react dynamically to information it processes to create new target lists to send to controlled assets, improving the timeliness and usefulness of observations. The following section describes scenarios that could find direct benefits from coordinated planning.

2.3 Real-World Scenarios

This section presents two real-world Earth Science scenarios in which the coordination planner, within a larger collection management process, could provide increased efficiency.

2.3.1 CLARREO Scenario

The Climate Absolute Radiance and Refractivity Observatory (CLARREO) Mission is headed by NASA and NOAA and is considered a key component of the future climate observing system. The mission is focused on taking climate observations, and considers improving the accuracy with which Earth-observing sensors take these climate observations to be an important goal. CLARREO seeks to provide a highly-accurate record of climate data that will be used to improve climate prediction models as well as calibrate other sensors that observe Earth and is considered “one of the 4 highest priority missions recommended in the National Research Council (NRC) Earth Science Decadal Survey”[27]. The mission is currently scheduled for launch in 2016 [28].

The 2007 NRC Decadal Survey report, “Earth Science and Applications from Space: National Imperatives for the Next Decade and Beyond,” outlined the mission objectives for CLARREO and suggested the mission/payload requirements. The Decadal Survey first estimated that three satellites would be needed, “two to obtain absolute, spectrally resolved radiance in the thermal IR and a 3rd to continue the IR absolute spectrally resolved radiance measurements,”[29]. The mission also adds CERES broadband instruments to the National Polar Orbiting Operational Environmental Satellite System (NPOESS) and NPOESS Preparatory Project (NPP). The first two satellites would require true 90° polar orbits to provide high latitude coverage from low Earth orbit (LEO) and require 100 km footprints. The third satellite would also be in a 90° polar orbit, but in an orbital plane 60° from that of the other two satellites. The NPP seeks to test new sensors that will eventually fly on the NPOESS, so their accurate calibration is imperative.

Since the Decadal Survey, however, CLARREO workshops (the latest in June 2009) have announced cutting the 3rd observatory to eliminate costs and to begin climate record keeping
sooner rather than wait multiple years for the observatories to launch. They also propose polar or near-polar orbits, with the two observatories having a 90° difference in Right Ascension of the Ascending Node (RAAN). The RAAN defines the point on the Earth’s equatorial plane at which the orbiting body crosses from the southern hemisphere to the northern hemisphere. The figure below more clearly depicts the proposed orbits:

A major goal of the CLARREO mission is to provide a calibration source for sensors already in orbit. The Decadal Survey specifically highlights the Visible/Infrared Imager Radiometer Suite (VIIRS) and the Crosstrack Infrared Sounder (CrIS) (sensors proposed for NPOESS), Landsat sensors, and CERES as sensors of interest. To complete their mission, the CLARREO sensors must take the same images/measurements at the same times/locations as the sensors of interest to ensure that each sensor is calibrated accurately. Moreover, there must be air and ground calibration sources that provide references of true values over the course of CLARREO’s mission life. It is foreseeable, then, that planning the collection of climate measurements over CLARREO’s mission life would involve coordination among assets on the ground, in the air, on the sea, in space, and those under the control of other organizations. Thus, the CLARREO mission fits the coordination planner’s data collection management framework quite well.

2.3.2 WRAP Scenario

The Wildfire Research and Applications Partnership (WRAP) is a joint project between the USFS and NASA that was initiated in 2003 “to facilitate and demonstrate evolved and evolving technologies for increasing the information content and timeliness of earth resource data collected for wildfires”[6]. The project has completed wildfire monitoring missions in the
western U.S. in 2003, 2006, 2007 and 2008 and totaled over 200 UAV flight hours. The data collected by these UAVs were combined in Google Earth with satellite data, including Geostationary Operational Environmental Satellites (GOES) weather data, satellite coverage maps for MODIS, and lightning detection data to monitor the fires' movement and strength. In this case, the synergy of data was done post facto, and scientists could have used coordinated observations with other independent mission planners to bolster the benefits of their research. NASA and the USFS planned similar missions for western U.S. wildfires in 2009, although they were not executed as planned. The demand for future wildfire studies in the western U.S., however, remains high as wildfire sizes have grown since the 1980's. Therefore, we consider the benefits of coordination in a notional wildfire scenario in this thesis.

2.4 Earth Phenomena Observation System (EPOS)

In this section, we present the context in which we formulate the CPP for this thesis. The coordination planner is implemented within the EPOS framework. EPOS includes a closed-loop planning and control testbed for the coordination of a system of sensors on satellites, UAVs, and USVs to collect data for the purpose of monitoring Earth phenomena. EPOS is Draper Laboratory's software environment for technology development funded under NASA ESTO (Earth Science Technology Office), AIST-99 (Advanced Information Systems Technology-1999), AIST-02, and AIST-05. While the EPOS concept was initially created as a way to coordinate the movements and observations of various sensor platforms to dynamically monitor the Earth's climate, it is foreseeable that such a system could be adapted for intelligence gathering or Intelligence, Surveillance, and Reconnaissance (ISR) battlespace management in real-time settings.

2.4.1 EPOS Functional Overview

This thesis focuses on the development of planning algorithms for operational scenarios in which climate monitoring and intelligence target monitoring are the primary missions. There are three levels: the coordination planner level, the sub-planner level, and the asset/sensor level. At the coordination level, the system's objectives are defined and target allocations are decided upon based on these objectives. The mid-level planners create their plans based on the contents of their target lists, which are influenced by the coordination planner. Assets then carry
out the plans uploaded to them by mid-level planners, and download the data to their ground stations. Figure 2-11 below more clearly depicts the EPOS architecture:

![EPOS Architecture Diagram](image)

Figure 2-11: EPOS Structure [30]

While this architecture originally assumed a centralized and synchronous world, we can easily adapt it here and apply it to the decentralized and asynchronous world we consider.

### 2.4.2 Role of the Coordination Planner

The coordination planner fits into the EPOS framework as the entity that coordinates the data collections made by satellites, UAVs, and USV's. It exists at the top-level in the EPOS hierarchy to efficiently coordinate the actions of the air, space, and ground sensors that observe targets and take measurements. The coordinator uses a system-level value function to allocate targets/requests to the various mid-level mission planners. The targets allocated to mid-level planners are added to the target lists for those planners (if the sub-planner accepts the request), and these single mission planners then dynamically create observation plans to maximize the value of the targets observed by their plans.

The decisions made by the coordination planner are determined by the constraints we assume to be present in the real-world. First, the coordinator must have clear objectives. The coordinator's objectives could range from observing a single target of extremely high-value multiple times, to observing as many different targets as possible, to satisfying high-priority
customers first no matter how difficult it is to schedule their requests. Each of these objectives requires careful valuation of each target that the coordinator receives. The Earth Science or intelligence value of each target is a function of both the priority of the user who submitted the request as well as the actual Earth Science (or intelligence) significance of the request.

The coordinator must also decide which sensors should observe which targets. This involves analysis of the properties of each sensor and the quality of the data collected by the sensor if tasked to observe a given request. For example, in general, an infrared (IR) sensor can track the movements of objects that generate heat well by identifying contrasts in heat levels amongst objects in its field-of-view (FOV). Moreover, the IR sensor will perform even better at night when the heat signature of the individual of interest is not masked by the radiation of the sun.

Temporal considerations are also of importance to the coordinator. Cases arise where a request that requires observation in the next 10 hours must be considered more valuable, at this moment, than a request of equal priority that must be observed sometime in the next week. Monitoring developing weather events require such constraints in time and space. Similarly, timely intelligence imagery can be pivotal in lending an advantage to a military force in a battlespace.

2.5 Coordination Planner Problem

The CPP can be stated in a similar fashion for each scenario presented in Section 2.3. Each scenario presents a problem where multiple resources are available to take observations/measurements of multiple targets. In the case of an Earth Science mission, users might invoke the coordination planner with targets requiring simultaneous observations at random locations on the Earth. A user might also desire as many observations as possible of a specific region of interest for, say, a developing hurricane. In either case, the coordination planner must make decisions on which requests to send to which sub-planners and communicate with those sub-planners efficiently to satisfy the customer’s objectives.

In an intelligence collection scenario, a JFC might require constant surveillance of his AOR for an upcoming operation. In this case, the user desires as many observations as possible in his AOR, with an emphasis on more important (higher priority) enemy targets at certain times.
In both scenarios, the problems are complicated by the inability to directly control single mission planners or the times at which new plans are uploaded to assets. Moreover, the problems have temporal constraints to ensure realism, as targets of interest in the Earth Science and IC realms often require observation at specific times and can emerge with little warning. We address this planning problem while preserving the features of the operational scenario.

The goals of this thesis, then, are as follows:

1) Formulate and address the CPP in a way that preserves the realisms of actual operations and yet is still tractable
2) Construct effective coordination planner algorithms for an asynchronous and distributed environment
3) Demonstrate the ability to balance different objectives in planning for many targets with heterogeneous resources
4) Quantify the benefit of coordination in the chosen real-world scenarios
3 Model Context and Development

This chapter explains the development of our model for the Coordination Planner Problem (CPP). The first section defines terminology necessary to understand the complexities of the problem and to define our model. The second section describes the key assumptions made for the model development. The third section provides an overview of the inputs, decisions, objectives, and constraints that we model for the CPP. The fourth section gives a description of the requirements of existing functional planning algorithms for the CPP’s sub-planners. A background on the actual sub-planners used in this thesis aids the reader’s understanding of how the coordinator interacts with sub-planners.

3.1 Problem Scope

The CPP presented in Chapter 2 has many sub-problems, and various solution approaches might be used to address each sub-problem. It is a high-dimensional problem with many opportunities for the application of Operations Research (OR) techniques to reduce the inefficiencies of current operations. This is due to the problem’s hierarchical and dynamic nature, and its mix of resource allocation, astrodynamics, and queuing problem features. Therefore, it is crucial to define the problem to be addressed in this thesis clearly.
3.1.1 **Key Terminology**

We begin by introducing terminology that defines how we model the problem and the level of fidelity at which we model it. This terminology is critical in extracting the realism of the actual problem while simultaneously scoping the problem to be addressed by this thesis. Some of the terms in this section are considered standard terminology, while others are specific to this thesis, and others define activities that do not exist in current practice but are necessary for realistic coordinated planning.

3.1.1.1 **Requests/Targets/Tasks**

Generally, in the Earth Science or Intelligence Collection (IC) worlds, a *request* refers to a desire for some type of data collection. A *target*, on the other hand, usually refers to some object or location that requires an action to be taken on it. For any asset, a *task* could entail anything from image capture, temperature measurement, or signal monitoring to takeoff, landing, orbit maneuvers, sensor slewing, upload, download, or other activities. We might say, then, that a user plus a target make a request, while a target plus a measurement type make a task. To the coordination planner, the terms request, target, and task are very similar in that they are all related to inputs to the coordinator that require observation or measurement. Each location input to the coordination planner represents a request, target, or task in some way. However, there is a difference between each term and for this thesis, we will refer to requests being *executed*, *serviced*, or *accepted*, targets being *observed* or *measured*, and tasks being *accomplished* in order to correspond with the intuition associated with each term.

A *user* is the human (or agent, in general) that sends a request to the coordinator. The coordination planner should accommodate a wide range of user types, from the person who knows little about aerospace-based observation, surveillance and reconnaissance missions or the planning of those missions, to the scientist who seeks a specific spectral band on a specific asset for his/her request. In addition, the coordinator must accommodate different types of requests with different priority levels and varying requirements. For example, a request can have very high inherent value if it relates to a developing natural disaster or a Time-Sensitive Target (TST). Other users might request that only a certain sensor type take an observation of a target. We choose a level of detail in the model that provides a useful model that is solvable in a practical amount of time. Thus, the following attributes are associated with each request:
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A name/number that uniquely identifies each request.</td>
</tr>
<tr>
<td>Priority</td>
<td>The science/intelligence value of the request. Can take on one of ten discrete levels. Higher numbers represent more important requests (higher priority/value).</td>
</tr>
<tr>
<td>Location</td>
<td>The target’s location on the Earth, defined by a latitude, longitude, and altitude.</td>
</tr>
<tr>
<td>Earliest Time</td>
<td>A single number indicating the earliest time at which the target must begin being observed.</td>
</tr>
<tr>
<td>Latest Time</td>
<td>A single number indicating the latest time at which the target must finish being observed.</td>
</tr>
<tr>
<td>Required Minimum Duration</td>
<td>A single number indicating the minimum amount of time a target must be observed for it to be considered serviced.</td>
</tr>
<tr>
<td>Collection Type</td>
<td>Indicates if the request is a single request or part of a dual collect.</td>
</tr>
<tr>
<td>Related Request</td>
<td>Indicates the request that it is related to, if it is part of a dual collect.</td>
</tr>
</tbody>
</table>

Table 3-1: Request Attributes

It is important to note that this table indicates that we model dual collects by creating two separate, related, requests. A user, however, would not think of their request for a dual collect as two separate requests, but rather as a single request for a dual collect. Thus, we assume the coordinator would pre-process this single request, as it is input by a user, by splitting it into two requests. We discuss additional modeling assumptions for dual collects in Section 3.2.5.

This table also states that we model all target locations as single locations defined by a latitude, longitude, and altitude. We develop this idea further in the following subsection.

*Point Targets*

For this thesis we consider point targets only. That is, only three fields are needed to define target locations: latitude, longitude, and altitude. However, it is often the case that scientists, or commanders, wish to monitor entire areas rather than single targets on the ground. These areas are usually defined by polygons composed of a finite number of straight line segments, or even “smooth” shapes such as circles or ellipses. Planning for the observation of an area target requires careful analysis, as they are often non-convex, and observing one might involve determining and tracking what percentage of an area a sensor has viewed and the varying pointing angles at which the area was viewed. We only consider point targets in this thesis to avoid these issues. This reduces the amount of computation required to measure
observation quality. To model a polygonal Area of Responsibility (AOR), then, we would discretize the area into a finite number of point targets and then create plans. This is shown in Figure 3-1 below on a notional non-convex area target in the western United States.

![Polygonal Target](image)

Figure 3-1: Using Point Targets to Define Areas of Interest

The number of point targets that we define within the region depends on the level of detail with which we wish to survey this area.

3.1.1.2 Simultaneous/Related Requests, Dual Collects

The terms simultaneous and related requests, or dual collects, all refer to the same idea: two requests that must be executed at the same time, or nearly the same time, with different sensors. The desire to model these requests is motivated by the discussion in Section 2.1.3.2. We consider related requests as a way to model high-value requests that require redundant viewings and calibration missions across different sensors. These requests relate to both the Earth Science and IC worlds in their own ways. We describe our assumptions for modeling dual collects in Section 3.2.5.

3.1.1.3 Planners

We refer to planners or sub-planners as those processes within the “stovepipe” systems that plan the use of satellites and/or Unmanned Aerial Vehicles (UAVs) to execute a set of requests. These planners operate independently of the coordinator, meaning that the coordinator cannot control which requests they include in their schedules, or when/how they
execute these schedules. It can only ask these planners which of its requests can be executed. Each planner controls the operations of one or more assets or platforms, e.g., satellites or UAVs. We refer to sensors as the devices that actually take observations and measurements. There can be multiple sensors on each asset.

For this thesis, we consider only two main types of assets, UAVs and satellites, but our methods are extensible to include surface vessels and ground-based assets.

3.1.1.3.1 Planner Management

Because assets are not under direct managerial control of the coordination planner, it is natural to associate some probability with the likelihood that requests will actually be executed given they were accepted during querying. In this line, we define two classes of assets, internally managed and externally managed, which correspond to how accessible information about the assets is. If the coordination planner has full knowledge of an asset’s planning processes (scheduling methodology, including algorithms, objective functions, constraints, etc.), then we call this asset internally managed. This corresponds to knowing with high probability whether or not a request that is allocated to this asset will actually be executed. It also implies a close relationship with the asset’s managers, which implies accurate feedback from the asset while interacting with it. An externally managed asset, then, is one for which the probability of having a request executed is not well known and likely low. Essentially, internally and externally managed classes are defined by the amount of knowledge the coordination planner has about the probability of an allocated request being executed.

For example, EO-1 is a National Aeronautics and Space Administration (NASA) satellite that accepts requests for new imagery from its user community. More specifically, EO-1 advertises a 10-15% chance that any image, or Data Acquisition Request (DAR), is actually executed [31]. The coordinator, then, could assume that if it allocates a task to EO-1, it will be accomplished 10-15% of the time. If, however, the coordination planner had a close relationship with EO-1 mission planning managers, this probability could be estimated more accurately, and possibly higher. As we discuss in Section 3.2.1, we relax this probabilistic aspect of the problem.

3.1.1.3.2 Taskable versus Non-Taskable Assets

Within the classes of planner management, all assets are either taskable or non-taskable. We define a taskable asset as one that can accept new requests as input in developing a schedule.
A non-taskable asset already has a flight plan (UAV) or series of pointing angles (satellite) set, but the coordination planner might be able to execute a request using these plans. This is referred to as “piggybacking.” We clarify the use of the taskable/non-taskable terms as follows: both of these subclasses denote assets that can accept requests (i.e., they are both useful types of assets for the coordination planner).

Most Earth Science satellite assets fall into the non-taskable category, as they are usually in “scanning” mode and are continuously collecting data if they are not undergoing maintenance. MODIS is an example of this type of sensor. MODIS has a very wide field-of-view that allows it to view the entire Earth every 1-2 days in 36 spectral bands, but cannot be tasked to look at a certain area at a certain time. On the other hand, it is more common for assets used for intelligence gathering to view smaller areas with better resolution, and are more likely to be pointable. A pointable asset is usually tasked to collect data when and where asset managers desire it. Thus, it is important to consider both of these types of assets. To more clearly define the difference in these types of assets, we present Example 3.1 below:

**Example 3.1:** If the coordination planner receives a request to image Location 1 in the western U.S. between 1200 and 1300 on September 22, there are three options for the coordination planner concerning this request. The coordinator has the following possible decisions:

a. Allocate the request to a taskable asset whose flight path/pointing angles is not yet determined (the request is explicitly planned for)

b. Allocate the request to a non-taskable asset whose flight path/pointing angles is predetermined and known and is such that the asset will observe Location 1 on September 22 between 1200 and 1300 (the request is implicitly planned for)

c. Not allocate the request to any assets

We present Figure 3-2 to aid our development of these different classes of assets and their relation to the coordination planner.
3.1.1.4 Planning Horizon

The planning horizon refers to the length of time for which the coordinator plans. A planning horizon of 10 hours means that the coordinator will try to schedule requests to satisfy user objectives on the available planners during a 10 hour time window. This assumes that all requests have time windows that lie within the planning horizon for any run of the coordination planner.

It is important to remember that we model the planning horizon for the coordination planner separately from those of the sub-planners. We model sub-planners as independent agents with their own planning horizons, algorithms, and target sets to more accurately model current operations.

3.1.1.5 Querying

Querying is the process of users approaching "stovepipe" systems and inquiring whether or not a system’s assets can execute a set of requests. This could be as simple as checking a database to see if an image/measurement has been taken in the past that satisfies the current information need (this type of capability is beyond the scope of this thesis). Or, it could mean interacting or iterating with an asset’s managers to determine which tasks the sub-planner is
willing to accommodate. The interaction with the asset’s managers could range from verbal negotiation with a human to automated, computer-based interaction. In either case, the sub-planner’s representatives could give feedback on whether or not each request can be executed at the desired time, given the resource’s current demand and constraints. Here, we define an iteration as one instance of the coordination planner querying the sub-planners with requests. We allow a finite number of iterations over the length of a planning horizon, dependent on some rate (iterations per hour).

We assume that each sub-planner can only respond to a finite number of queried requests at any one time. Constrained querying is reasonable given how constrained assets usually are. Satellites and UAVs provide valuable information and are thus in very high demand. Additionally, the assets themselves are physically constrained. Satellites view only a small portion of the Earth on every orbital revolution, while UAVs are constrained by fuel capacity, airspace restrictions, maximum climb rates, maximum descent rates, and other aircraft limits such as maximum speed.

In this thesis, we model the querying process as an automated process where we allow the number of queries per iteration, as well as the number of iterations per unit of time, to be entered as an input. Although today much of the querying process is manual, one could imagine a future concept of operations in which this querying procedure is automated.

A query asks a sub-planner whether or not it can include a request in its execution phase, to be explained in Section 3.1.1.9. Once the coordinator receives feedback indicating the sub-planner can execute a request, the request is sent to the sub-planner to be added to its target list. If the sub-planner then observes the target during the correct time window, the target has been successfully observed.

For any single query, the coordinator sends a sub-planner the request’s location, feasible early time window (ETW), late time window (LTW), and minimum required observation duration. We assume sub-planners respond by indicating which requests have been accepted (yes or no), and additional information if the request can be executed. This information includes the sensor with which the request will be executed and the time at which the request will be executed.
3.1.1.6 Planning Phases

Planning phases refer to the times at which sub-planners are creating plans for upcoming flights/orbits. Sometime before the start time of an upcoming flight/orbit, sub-planners cease receiving queries and use their most recent target lists to refine their final plans before uploading them to the assets.

It is important to note that we associate each planning phase with a single execution phase. Figure 3-3 demonstrates this. In this example, we show the planning cycles for a notional UAV sub-planner and a notional satellite sub-planner (EPOS 1). Planning phase 1 ("Planning 1") for the UAV sub-planner occurs from time $t_0$ to $t_1$. During this phase, all requests submitted as queries to the UAV sub-planner are submitted for execution during execution phase 1 ("Executing 1"), from time $t_3$ to $t_7$. Similarly, requests submitted to EPOS 1 from $t_0$ to $t_2$ are submitted for execution during time $t_4$ to $t_6$. Requests submitted to EPOS 1 during planning phase 2 ("Planning 2"), if accepted, could be executed between times $t_6$ and $t_8$.

This discussion implies that for this thesis, we are considering single orbit planning cycles (for satellites). This is the case if each execution phase is on the order of 100 minutes, a realistic orbital period for an Earth observing satellite. In reality, some planning cycles might create plans that extend well into the future. These plans provide a backbone for assets to use, which are then modified as the actual execution phase draws nearer. In our test runs, however, we usually model single orbit planning cycles for each satellite sub-planner.

3.1.1.7 Send/Upload Phases

We assume that each sub-planner has a sending/uploading phase for every planning/execution phase. This models the idea that sub-planners must actually upload tasks...
to their assets and we assume that during this time no new queries can be made. It would be reasonable to assume that planning for the next execution period continues during this send/upload time, but we assume that the sub-planner does not accept queries during this time. It can also be thought of as a maintenance or downtime period for the sub-planner, which often occurs in practice. Our inclusion of these different phases for each sub-planner allows the coordinator to carefully send requests to sub-planners and control in which planning cycles requests are serviced.

It is important to note the differences between the send/upload phases for satellites and UAVs; satellites are in orbit and receive new plans periodically to manage their actions, while UAV operators generally receive one flight plan for a single UAV sortie. This is because UAVs, as used in Earth Science and military operations, are currently piloted by humans. It is plausible that future operations will involve autonomous operation of UAVs, which will enable more flexible methods of uploading data. Even now, the development of long endurance UAVs used for persistent surveillance calls for more data upload options. Our approach can handle the current method and the future one by varying the length of send/upload phases.

3.1.1.8 Planning Periods

Planning periods refer to windows of time in which the sub-planners plan for a unique set of execution phases. A planning period is an artificial construct that is relevant only to the coordinator, and uses information about the planning cycles of sub-planners. We assume sub-planners are available for collecting data if they are in a planning phase and we assume they are unavailable for collecting data during a sending or uploading phase in which they upload tasks to the assets themselves. The additional assumption that sub-planners do not accept queries during a sending or uploading phase can be made if desired. These periods serve as windows of time in which the sub-planners accept queries for a unique set of execution phases. It is within planning periods that the coordination planner decides which requests it would like each sub-planner to give feedback about. The coordinator makes only a finite number of queries in each planning period. We use planning periods to allow the coordinator to organize its queries given knowledge of sub-planner planning cycles. Figure 3-4 depicts the decomposition of these asynchronous planning cycles into planning periods with discrete start and end times,
and finite request/resource sets. This allows us to clearly define which resources are available in each time period and use the approach we present in Chapter 4.

### Asynchronous Planning Cycles

Because we model planning periods this way, one issue that arises is how to handle requests that have already been input but need to be observed at a much later time. For example, in Figure 3-3, if we assume we have a target that must be observed between times $t_6$ and $t_7$, and the current time is $t_0$, one can see that there are two options for scheduling this request. First, the coordinator can query the UAV sub-planner until time $t_1$, and if it receives feedback that indicates the request can be executed between times $t_6$ and $t_7$, then we can allocate the request to the UAV sub-planner and consider the request serviced. We cannot benefit, however, from querying the satellite sub-planner because it is currently accepting requests to be executed between times $t_4$ and $t_6$, and an observation during this time would yield an infeasible solution due to the time window constraint. If we are unable to schedule the request on the UAV sub-planner, then we begin querying the satellite sub-planner at time $t_4$, for execution between times $t_6$ and $t_7$.

#### 3.1.1.9 Execution Phases

Execution phases are periods of time when an asset is carrying out an observation plan. Again, a query asks a sub-planner whether or not it can include a request in its execution phase.
Each execution phase includes a send/upload phase in which assets cannot collect data, and this is explicitly considered.

It is important to note that satellite assets are constantly executing a plan, and so satellite sub-planners create new schedules that are spliced into currently executing plans. Currently, UAVs rarely deviate from their flight plans, primarily due to federal regulations in the continental U.S., and thus multiple UAV execution phases in a scenario are less common. However, in future operations this might not be the case. We can model either framework by breaking down each planning cycle in the manner explained above.

3.1.2 Terminology in CLARREO Context

To make the use of these terms less ambiguous, we present them in relation to a hypothetical scenario of interest. In the CLARREO case, the coordination planner would receive the requests of NASA and NOAA scientists as inputs and create observation plans that coordinate the efforts of any accessible asset to satisfy these requests. We first note that assets might be of the two generic types we use in this thesis: UAVs and satellites. Among these assets, we must then distinguish between internally and externally managed assets; this is dependent on how involved NASA and NOAA would be in the coordinated planning. In this envisioned scenario, NASA and NOAA embrace a coordinated planning framework and thus we can consider NASA's and NOAA's assets internally managed. In this case, it might be true that there is a sub-planner for the CLARREO assets, a sub-planner for NPOESS, and a sub-planner for any other assets being considered for calibration purposes, whether they are NASA/NOAA assets or not. Targets would be on the ground, in the air, or in the sea, and many of them would require measurements at similar times. However, while NASA and NOAA have a close relationship, it is unclear whether or not the planners would be under the same control.

Next, we consider which assets are taskable and which are non-taskable. We will model the satellites directly relevant to CLARREO as defined in the Decadal Survey as taskable assets. We notionally define CLARREO sensors as taskable because a recent CLARREO workshop noted that adding a pointing capability to the CLARREO sensors could result in 3 or 4 times as many simultaneous viewing opportunities with other sensors of interest (Clouds and the Earth's Radiant Energy System (CERES) on Aqua, for example) [32].
3.2 Modeling Assumptions

This section outlines the key assumptions that allow us to create a tractable model. We describe our assumptions on the probabilistic nature of the problem, the amount of knowledge we have of sub-planner planning cycles, astrodynamics, dual collects, our model of time and space, and target-to-sensor valuation.

3.2.1 Stochastic Nature of the Real-World Problem

As discussed above, every time a request is sent to a sub-planner there is some probability with which the request will actually be executed, dependent on the results of querying the planner and on the performance of the asset once it is operating in an execution phase. When interacting with a sub-planner, the responses the coordination planner receives will not necessarily be correct. That is, there are a number of reasons an asset manager will incorrectly respond positively (indicating a target will be observed) to the coordinator. An incorrect response could result from the fact that the response was only an estimate of what tasks the assets could observe, or perhaps because the asset manager was unaware of higher priority targets entering the system that would have priority over the coordinator's target. For this thesis, however, we assume that the feedback we receive from sub-planners is accurate. This assumption allows us to relax some of the stochastic features of the problem and thus allows for a simpler model. Specifically, we do not distinguish between internally and externally managed sub-planners, as the primary difference between these classes of sub-planners, by definition, lies in the certainty with which each returns accurate feedback. The inclusion of probabilistic features is left for future work.

3.2.2 Simulation of the Real-World Problem

It is important to understand how we simulate the interaction between the coordinator and each sub-planner in this thesis. As stated above, we assume a finite number of queries during the planning horizon, based on the query limit (maximum number of requests that can be sent to a sub-planner at once) for each sub-planner and some constant query rate (per hour) that is input to the scenario and usually assumed identical for each sub-planner. Each instance in which the coordinator sends a batch of queries to its sub-planners is called an iteration. The coordinator can send a limited number of requests to each sub-planner at each iteration, and the number of iterations depends on the query rate of the system. We assume that each sub-planner
gives feedback to the coordination planner sometime between the last query and time until the next query is possible (based on the query rate), allowing enough time for the coordinator to incorporate the information it receives into its next set of queries.

### 3.2.3 Complete Knowledge of Planning Cycles

An additional assumption we make is that the coordination planner has full knowledge of the planning and execution cycles of the independent sub-planners. It is reasonable to assume this for a number of reasons. First, for UAVs, flight takeoff and landing times are known to the scientists (or military commanders) involved. This is generally publicized information and so we assume this data is known for this problem. As more unmanned aircraft are built and their use becomes more commonplace, however, it is important to note that it might become more difficult to determine flight times for all aircraft.

When not published, satellites require careful calculations to predict the timing of sending/uploading phases, yet we still assert it is reasonable to assume known send/upload times. Because spacecraft generally have unobstructed line-of-sight (LOS) access to ground stations for data downloads only at certain times, we can assume that send/upload times occur at times that are some multiple of the length of orbital periods.

For both UAVs and satellites, the coordination planner might have difficulty determining the times reserved for maintenance. UAVs might need to download the data they have collected at random times, or even cease all data collection when necessary. Similarly, satellites may perform maintenance tasks at any time regardless of their locations. As stated in section 3.1.1.7, however, we lump maintenance tasks into the send/upload phases for each sub-planners to simplify this potential problem.

### 3.2.4 Astrodynamics Modeling

In this section we describe how we incorporate astrodynamics into the problem. Specifically, we discuss how we model satellite movements (orbit propagation) and how we determine the ability of sensors to observe potential targets. Because the focus of this thesis is primarily on coordinated planning of target observations/measurements and less on precision orbit propagation and engineering realism, we use a simple model of orbit propagation called J2 secular theory.
3.2.4.1 J2 Secular Theory

To describe the position of an Earth-orbiting object in space, a 6-tuple called the Kepler element set is often used. The Kepler elements consist of the following 6 parameters presented in Table 3-2:

<table>
<thead>
<tr>
<th>Element</th>
<th>Symbol</th>
<th>Definition</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semi-major axis</td>
<td>a</td>
<td>One-half the longest diameter of an elliptical orbit</td>
<td>Kilometers</td>
</tr>
<tr>
<td>Orbit Eccentricity</td>
<td>e</td>
<td>A measure of how much the shape of an orbit deviates from a circle</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>Orbit Inclination</td>
<td>i</td>
<td>The angle between the orbital plane and the Earth’s equatorial plane</td>
<td>Radians (rad)</td>
</tr>
<tr>
<td>Right Ascension of the Ascending Node (RAAN)</td>
<td>Ω</td>
<td>The angle to the orbit ascending node measured from the vernal equinox</td>
<td>Radians (rad)</td>
</tr>
<tr>
<td>Argument of Perigee</td>
<td>ω</td>
<td>The angle measured from the ascending node to the direction of perigee (point on an elliptical orbit that is closest to the center of the Earth)</td>
<td>Radians (rad)</td>
</tr>
<tr>
<td>Mean Anomaly</td>
<td>M</td>
<td>The time since the object’s last pass through perigee (periapsis) times ( \frac{2\pi}{T} ), where ( T ) is the duration of one orbit</td>
<td>Radians (rad)</td>
</tr>
</tbody>
</table>

Table 3-2: Kepler Elements [33]

The first two elements \((a, e)\) describe the shape of the orbit while the next three \((i, \Omega, \omega)\) are angles that define the orientation of the orbit. The mean anomaly captures an orbiting object’s position on the orbit. When given these elements at any particular time for some object in orbit we can compute corresponding 3-dimensional position and velocity coordinates. With these position and velocity vectors we can compute the angle at which the orbiting object can view locations on the Earth.

Over time, orbit shapes and orientations change due to a number of perturbing forces. Primary perturbing forces include atmospheric drag, non-spherical Earth gravity (oblateness of the Earth), lunar and solar gravity, and solar radiation pressure. For this thesis, we model only perturbations due to the oblateness of the Earth and how those perturbations affect the orientation of an orbit. An object’s gravitational energy potential is not uniform in relation to latitude (or altitude) as a result of the Earth’s equatorial bulge. This can be accounted for by the \( J_2 \) perturbation term, a spherical harmonic of the Earth’s gravity field. The \( J_2 \) term is almost 1,000
times larger than the next largest coefficient, so it is necessary that we include this term. We apply what is called \( J_2 \) secular theory because we account only for linear changes in \( \Omega, \omega, \) and \( M \) over time, not periodic, or oscillating effects. We refer the reader to [33] for a derivation of the following rates of change of the elements for which we model perturbations:

\[
\frac{dQ}{dt} = \frac{3}{2} \sqrt{\frac{\mu}{r_E^3}} J_2 \left( \frac{r_E}{a} \right)^{3.5} \left( \frac{1}{\left(1 - e^2 \right)^{1.5}} \right) \cos(i)
\]

(3-1)

\[
\frac{d\omega}{dt} = \frac{3}{4} \sqrt{\frac{\mu}{r_E^3}} J_2 \left( \frac{r_E}{a} \right)^{3.5} \left( \frac{1}{\left(1 - e^2 \right)^{1.5}} \right) \left( 4 - 5 \sin^2(i) \right)
\]

(3-2)

\[
\frac{dM}{dt} = -\frac{3}{4} \sqrt{\frac{\mu}{r_E^3}} J_2 \left( \frac{r_E}{a} \right)^{3.5} \left( \frac{1}{\left(1 - e^2 \right)^{1.5}} \right) \left( 3 \sin^2(i) - 2 \right)
\]

(3-3)

where \( r_E \) is the radius of the Earth. We use these equations to perturb the orbits at each time step. \( J_2 \) secular theory is generally considered valid for analysis on the order of weeks, which, while insufficient for most satellite engineering applications, we deem sufficient for the problem we address because we plan for horizons on the order of hours or days.

3.2.4.2 Satellite Steering

In order to model what a space-based asset can view, including the pointing capabilities of the assets, we require some degree of knowledge of the way in which the instrument can be steered. For this thesis, we assume one spacecraft axis points to nadir and a second axis is normal to the plane spanned by nadir and the spacecraft’s velocity vector. The third axis results from the cross product of the other two. This allows us to compute the pitch and roll angles of the spacecraft required for viewing a target. We can think of these angles as the pitch and roll angles of the spacecraft or the \( x- \) and \( y- \) gimbal angles of the individual sensors.

Using this coordinate system, we can transform any LOS vector (vector from spacecraft to target) in the Geocentric Equatorial Coordinate System, or Earth-Centered Inertial (ECI) coordinate system, to the spacecraft’s coordinate system. Then we can calculate the pitch and roll angles needed to view a target. The ECI coordinate system uses the vernal equinox as one axis, and the vector \( 90^\circ \) to the east in the equatorial plane as a second axis. The third axis points through the North Pole, normal to the equatorial plane. We present the detailed calculations for space-based satellite viewings of targets on Earth in Chapter 4.
3.2.5 Dual Collects

Simultaneous, or related, requests are modeled as separate requests with identical attributes. The coordinator must observe the target associated with these requests multiple times with different sensors. Both requests are executed when both are executed within their time windows. Thus, the observations do not have to be made at the exact same time. We assume that a “near-simultaneous observation” occurs when the target is observed within its time window, but we do not distinguish between a “simultaneous observation” and a “near-simultaneous observation,” as both types of observations are valued equally.

However, we can generate data sets to ensure that dual collects are made within a certain amount of time of each other. If we use the “A-Train” as an example, we observe that OCO was supposed to follow 15 minutes behind Aqua in formation, collecting similar data. One could use 15 minutes, then, as a reasonable inter-observation time when generating the data for a single simultaneous collection. This would be modeled by creating two targets with a difference in time between their ETW and LTW of 15 minutes. If we consider the two targets \( i \) and \( i+1 \), we model them with the following data (\( ETW, LTW \) in units of hours since the start of the scenario):

\[
\begin{align*}
\text{Latitude}_i &= \text{Latitude}_{i+1} \\
\text{Longitude}_i &= \text{Longitude}_{i+1} \\
\text{Altitude}_i &= \text{Altitude}_{i+1} \\
ETW_i &= ETW_{i+1} \\
LTW_i &= LTW_{i+1} = ETW_i + 0.25 = ETW_{i+1} + 0.25
\end{align*}
\]

(3-4) (3-5) (3-6) (3-7) (3-8)

It is reasonable to assume that a 15 minute inter-observation time produces images suitable for calibration given the times between observations forced by orbit maintenance in the “A-Train.” We sometimes allow the time windows for related dual collects to be longer than 15 minutes, although for this thesis we generate some data sets containing related requests with very narrow time windows to ensure useful data for calibration purposes.

Modeling dual collects as separate requests is one way to incorporate relationships between various requests into the model. Another way would be to add another input field that denotes how many different sensors we wish to have the request executed by.

It is important to consider the synergistic effects of viewing dual collects as well. Sometimes, viewing one part of a dual collect but not the other is worthless while other times
the single image/measurement yields considerably more value than no collection at all. Thus, relationships between individual requests can mean that their valuation is best modeled non-linearly. Here, we assume that there is value in obtaining only one image of the specified location. We make this assumption based primarily on the fact that a single observation of a target of interest in a tactical scenario would still prove valuable to a commander. Viewing an important enemy facility only once before an ensuing operation almost certainly retains value even if the ultimate objective of a simultaneous viewing of the facility with multiple sensors was not completed. While this idea might not translate directly to the Earth Science realm because sensor cross-calibration usually requires more than one near-simultaneous viewing for comparison, this assumption allows a more tractable model and so we include it.

We also assume that servicing both parts of a dual collect yields more value than the sum of the individual values. The ability to include non-linear valuation is important when dealing with related observations where the second part of a dual collect, for example, contributes much more value than the first part by itself. We argue that this represents reality in both the Earth Science and IC worlds due to the reasons discussed in Section 2.1.3.2.

3.2.6 Continuous Time and Space

There are generally two ways to model spatial and temporal features of a problem. One approach involves discretizing space and time. This is usually done by transforming the real-world problem into a graphical structure that is easier to visualize and analyze. A static version of such a graph involves nodes for the starting locations of each asset, and nodes for each target being considered, with arcs connecting nodes that relate to each other. To account for temporal features of the problem, the graph is expanded by adding static versions of the graph for every discrete time period being modeled. The other approach involves maintaining continuous time and space. This usually requires more intensive computation and necessitates a different class of solution techniques, but yields better (more exact) results. Here, we allow time and space to take on a countable number of values, quantized to three decimal places.

Given the astrodynmic features of the problem, we allow assets, particularly satellites, to take on any location at any time. That is, we do not discretize three-dimensional space in regions, but rather we allow satellite locations to take on any value from a continuum. We also allow sensor pointing angles to take on arbitrary values. Similarly, target locations, time
windows, and required durations are arbitrary; that is, there is no finite set from which we choose locations, but rather a continuum of latitudes and longitudes to choose from, for example.

However, we discretize time when making astrodynatic calculations. We compute satellite locations every $\Delta t$ seconds and generally, $\Delta t$ is set at 60 seconds to find a high number of possible target viewings while keeping computation time relatively low. At each time step, we determine satellite locations, and compute line-of-sight accesses for each satellite to each target. Because we discretize time for space-based assets but allow time windows and required observation durations to take on arbitrary values, it is necessary to implement a rounding technique when performing feasibility checks. We round conservatively in that we consider $n$ consecutive time periods with access to a target to be less than or equal to $(n-1)\Delta t$ time units. This assumption is conservative because it is possible that the target was observed for nearly $(n+1)\Delta t$ consecutive time units. Given $n$ consecutive accesses to a target and time step $\Delta t$, the true duration of the viewing period, $d$, lies in the following interval

$$(n-1)\Delta t + \varepsilon \leq d \leq (n+1)\Delta t - \varepsilon \quad (3-9)$$

where $\varepsilon$ is an arbitrarily small positive quantity. Our estimate of the true duration, $\hat{d}$, is

$$\hat{d} = (n-1)\Delta t - \varepsilon, \quad n > 0 \quad (3-10)$$

and is thus strictly less than the true duration, $d$. Figure 3-5 helps to visualize this equation:

![Figure 3-5: Depiction of Rounding Heuristic for Computing Satellite-to-Target Accesses](image-url)
This figure depicts how we would calculate $\hat{d}$ for a given set of access values (0 or 1). Here, $n = 4$ which indicates that $\hat{d} = 3\Delta t - \varepsilon$. This rounding heuristic implies the following bound on our rounding error, $\varepsilon$:

$$\varepsilon < \frac{\varepsilon}{2} \leq \Delta t$$

We present more detailed astrodynatic calculations in Chapter 4.

### 3.2.7 Quality of Observations

One metric of interest for potential users (in the Earth Science and IC communities) of the coordination planner is the quality of the images/measurements that the coordinator is able to obtain. The IC community has developed a standard scale for measuring the quality of aerial images based primarily on the interpretability images. The National Imagery Interpretability Rating Scale (NIIRS) ranges from 0 (worst) – 9 (best) based on the ability of imagery analysts to clearly identify specified objects of interest for a given image [34]. NIIRS quality is defined for four major imaging types (visible, Radar, infrared, and multispectral) and the different quality levels are defined by listing specific objects that can be resolved at each level. The quality of an image is a function of at least the following: target type, sensor type, image resolution, length of the observation (a longer duration implies the asset had more time to settle and take a clearer image), cloud cover, angle off-nadir, and the time of day at which the observation was made.

We model the quality of an observation by considering only the target type, sensor type and length of the observation. We do this by determining relative values for each target type/sensor type pair that we define. While a detailed analysis of each target type – to – sensor type pair is out of the scope of this thesis, we briefly present reasons for our valuations below. The first three sensors model generic sensors in each sensor class, and we then develop valuation for a handful of specific, real-world sensors.

#### 3.2.7.1 Generic Sensors

**Electro-optical (EO) sensors** – Used to gather information on phenomena that emit or reflect electromagnetic energy in visible, infrared, or ultraviolet spectra. Therefore, EO sensors are useful in identifying activity in volcanoes and wildfires, and less useful for phenomena such as hurricanes. For intelligence data collection, EO sensors contribute valuable Measurement and Signature Intelligence (MASINT) that detect target objects and sources [35]. EO sensors do not
lose their effectiveness when observing targets not in view of the sun, as these measurements are not greatly affected by solar radiation.

*Infrared (IR) sensors* – Used to image targets by identifying heat signatures. These sensors are used to monitor atmospheric infrared radiation and how energy is exchanged between the earth and the atmosphere. IR sensors are quite useful for intelligence collection in that they can identify targets that generate heat but might be invisible to unassisted human eye. IR sensors become more valuable when targets are not in view of the sun, as solar radiation masks the true heat signature of targets.

*Synthetic Aperture Radar (SAR)* – SAR uses multiple Radar images to build better images than is possible with conventional methods. These sensors provide valuable data for both Earth Science and Intelligence targets.

### 3.2.7.2 Real-World Sensors

*CERES* – CERES instruments are part of the Earth Observing System (EOS) and contribute to the tracking cloud properties by measuring solar-reflected and Earth-emitted radiation in the Earth’s atmosphere. The instruments are on board Aqua, Terra, and TRMM and provide valuable Earth climate monitoring measurements.

*Advanced Spaceborne Thermal Emission and Reflection Radiometer (ASTER)* – ASTER is a taskable sensor on-board NASA’s Terra satellite that accepts requests to take data at new locations on the Earth. It is primarily used to map land surface temperatures, reflectances, and elevations, providing valuable Earth Science data to its users [36].

*Notional CLARREO sensors* – The CLARREO mission proposes one IR sensor and three spectrometers per satellite [28]. For simplicity, we model these sensors as improved versions of our generic IR sensors.

When applying these valuations, we do so in an offline manner using a lookup table that has been generated *a priori*. We present the actual values used in the lookup table in Appendix B.

### 3.3 Coordination Planner Model Overview

In this section we state, in words, the inputs, decisions, objectives, and constraints for the coordination planner. This allows us to clearly summarize the coordination problem
described in this thesis, and gives the reader background necessary to understand the planning algorithms in Chapter 4.

3.3.1 Coordination Planner

The coordination planner is tasked with the job of managing the requests of many users by allocating them to various sub-planners. We have described the level of "coordination" that is possible in our problem through our descriptions of the operational problem in Chapter 2 and assumptions in Chapter 3. The level of coordination is a function of the coordinator’s knowledge of sub-planner capabilities (extensive knowledge assumed), the degree to which sub-planners interact with each other (essentially no interaction), and the degree to which the coordination planner’s objectives can influence sub-planner actions (assumed little). If we examine the possible extremes of coordination, at one extreme is no coordination where each sub-planner receives the same information from the coordinator and the actions of sub-planners are never communicated to any of the other agents in the system. The other extreme is full coordination, which can often be difficult to define as noted in [37]. We imagine full coordination allows all sub-planners to operate in a synchronous fashion, and allows the coordinator to force sub-planners to value requests as it does. The following sections further define, in words, our notion of "coordination" by highlighting the inputs, constraints, objectives, decisions, and outputs of the coordination planner.

3.3.1.1 Inputs

The inputs to the coordinator originate from the scenario and from the user of the coordinator. The inputs can be broken into four categories:

1) System Data
2) Sensor Data
3) User Objectives Data
4) Request Data

System Data: The scenario data defines which assets are included in the scenario, as well as how many coordinator requests there are, and how many requests are internal to each sub-planner. These data also include the length of planning, sending/uploading, and execution phases for each sub-planner. We also include the limit on the number of queries allowed per iteration with each sub-planner in these data files.

68
**Sensor Data:** The sensor data provides a table of data for each sensor involved in the scenario. It includes a brief description of the sensors themselves, as well as the features of each sensor that are modeled, such as the slew rates/limits, FOV, and the cost of slewing the sensor.

**User Objectives Data:** These data pertain to the weights applied to the value function used for the formulation. The user can input this data to force the coordination planner to focus its efforts on achieving various objectives.

**Request Data:** We include both requests that the coordination planner must handle, as well as requests that are internal to each sub-planner, representing the idea that some users will still approach the sub-planners individually and that those sub-planners still have their own missions to carry out. For coordinator requests, we include the data described in Section 3.1.1.1. Requests internal to the sub-planners themselves have their own data formats, and we briefly describe this input in Sections 3.4.1 and 3.4.2.

### 3.3.1.2 Constraints

We established that the CPP has constraints that originate from the many different features of the problem. In this thesis, we consider the coordinator to be affected by physical, temporal, and communication constraints. In some instances, the coordinator is constrained by simultaneous viewing requests (dual collects), which require that the requests be accepted by different sensors, if not different sub-planners altogether.

Some low-level constraints are accounted for in the sub-planners themselves and while they are certainly constraints on the CPP, there are no mathematical equations that correspond to these constraints. One example of this is the data storage capacity on-board each satellite. It would be difficult for the coordination planner to be able to estimate the data storage capacity remaining at any time step on an asset given the nature of current operations. This is an example of a constraint that is fully modeled in the satellite sub-planner, but not explicitly modeled at the coordination level. Similarly, the coordinator’s requests can be rejected by sub-planners due to the current demand on the sub-planner request lists and the fact that the sub-planners operate independently of the coordination planner. This is another constraint on the coordination planner that, while not modeled explicitly in the mathematical formulation presented in Chapter 4, is implicitly included in our model.
3.3.1.3 Objectives

It is important to note that the user of the coordination planner will have his own objectives. We identify four objectives for coordination that align with the operational concepts from Chapter 2:

1) Maximize the average priority of the requests executed (execute high-value requests)
2) Maximize the number of requests executed
3) Maximize the number of dual collects made
4) Maximize the quality of the observations taken (maximize the value of target-to-sensor matching)

Maximize the average priority of the requests executed (avgPriority): This objective corresponds to a situation where it is most important to execute requests that correspond to high-value targets. This objective would be useful in an AOR where TST’s are rapidly emerging, or when a developing hurricane threatens a coastal city.

Maximize the number of requests executed (numRequests): This objective seeks to take as many observations as possible, and could correspond to a long-term Earth Science monitoring campaign, or to a routine intelligence operation.

Maximize the number of dual collects made (numDualCollects): This objective corresponds to the Earth Science cross-calibration missions, or to the need for highly accurate surveillance on a high-value target in an AOR.

Maximize the quality of the observations taken (avgTSV): This last objective forces the coordinator to choose its queries more carefully; here, observation quality is of paramount importance, so the coordinator is forced to make decisions that allocate targets to sensors in a way that maximizes the average quality of the observations.

It is important to note that these objectives compete with each other. For example, trying to maximize the average priority of the requests executed clearly competes with the objective of maximizing the number of requests executed. Some users will emphasize only one of these objectives, while others will seek to balance all four. This is important to understand the inclusion of each objective as a Measure of Performance (MOP) and how we formulate our value function in Chapter 4.

We also note that a fifth objective could relate to executing requests as fast as possible. This objective would be important in the IC world, especially in an AOR where identifying
enemy targets as quickly as possible is imperative. However, we do not include it as a primary coordination objective and analysis of the coordination planner’s performance for this metric is left for future work.

3.3.1.4 Decisions

The coordination planner sends as many targets to the sub-planners as often as it can, subject to the communication constraints stated in Section 3.3.1.2. The planner must decide which requests to query, and when. This involves sending target locations and the times at which the requests should be executed. We do not allow the coordinator to send relative values of requests to the sub-planners. This is because we assume each sub-planner operates independently, which means in addition to having their own constraints, request sets, and planning cycles, they also have their own value functions. This assumption also results from the fact that we do not include cost in the problem, which could be used to influence the sub-planner’s valuation of requests. Thus, the values placed on individual requests in the sub-planners do not necessarily match those of the coordination planner. It is important to note that in the IC world, there is an established priority system that all users adhere to, and this can be modeled in our formulation.

3.3.1.5 Outputs

The coordination planner must make decisions and interact with the sub-planners to produce, in the end, an observation plan. The plan consists of a list of targets, the platforms by which they are observed/measured, and the start and end times of the observations/measurements. From these outputs, we can obtain the four MOPs that we deem important: the average priority of the targets observed, the total number of targets observed, the number of dual collects made, and the average quality of the observations. We also have access to the full-length plans created by the sub-planners, which include the coordinator’s requests as well as those internal to the sub-planners themselves.

3.4 Existing Functional Pieces

In this section we briefly outline the solution methodologies for each type of sub-planner with which the coordination planner interacts. We include this description to give the reader a sense of how the coordinator interacts with sub-planners and to establish that we have
successfully integrated the coordination planner with complex planning mechanisms (sub-planners) for this thesis.

### 3.4.1 UAV Sub-Planners

The coordination planner is currently integrated with one aircraft routing planner. The planner for which integration is complete uses a composite variable approach to the heterogeneous UAV planning problem in three dimensions. In [38], composite variables are generated heuristically to intelligently shrink the number of variables needed for the model. By definition, composite variables capture multiple decisions in one variable. Moreover, composite variable approaches usually have strong Linear Programming (LP) relaxations that allow large problem instances to be solved quickly. In [38], Negron intelligently assigns subsets of tasks to each UAV, uses a three-phase heuristic algorithm to create and improve UAV routes, and solves an LP problem to choose which routes to include in the plan. It is important to note for analysis purposes that the subset allocation step involves generating random subsets, which makes the solution method non-deterministic. We refer the reader to [38] for a detailed explanation of the Composite Operations Planning Algorithm (COPA).

While we are careful to create a model for the coordination planner that can be integrated with any type of sub-planner, for the purposes of integration with actual sub-planners it is important to understand what the coordination planner must send the UAV planner at a minimum. The UAV sub-planner we use requires the following inputs:

- Length of planning horizon
- UAV characteristics
  - Number of UAVs
  - Start location of UAVs
    - Latitude
    - Longitude
    - Altitude
  - Attributes of each type of UAV
    - Speed
    - Endurance
    - Floor
    - Ceiling
    - Sink Rate
    - Climb Rate
- Target locations

72
The UAV planner also allows for multiple observation locations for a single target, simulating the idea that the UAVs could trade total quality of observations (off-nadir viewing) for an increase in the total number of targets observed. We fix the number of locations available per target at one for this thesis. Also, we fix the number of composites generated (feasible paths generated) per run of the planner at a high value to reduce the variation in the solutions found.

3.4.2 Satellite Sub-Planners

The coordination planner is also integrated with a sub-planner that creates observation plans for space-based sensors. This sub-planner is adapted from [39] in which the authors present planning algorithms based around a future Earth monitoring system with large numbers of inexpensive small satellites. The problem involves tasking individual pointable sensors to maximize the total science value of the observations made. The authors approach this complex problem by hierarchically decomposing it into two levels: a top-level optimization problem and a bottom-level optimization problem. At each time step, the top-level problem is solved and chooses a target for each sensor. The bottom-level problem takes the results of the top-level solutions as input and chooses pointing angles for each sensor. The solution approach accounts for slewing constraints as well as data storage capacity constraints and includes the ability to allow sensors to download data to ground stations when possible. The satellite sub-planner, as it was created in [39] requires the following as inputs:

- Scenario start time
- Length of planning horizon
- Time step
- Satellite data
  - Kepler elements at start time of scenario
- Sensor data
  - Sensor type
  - Field-of-View
We have modified this sub-planner by adding time windows to input files. This adds another level of fidelity to the planner and only requires modifying the way in which data are generated for the optimization problems.
4 Algorithmic Approaches

Having established an operational background for the problem and explained how we model the various features of the problem, this chapter presents the planning algorithms we implement to address the problem. The first section compares the problem to problems found in the literature. It outlines mathematical models for satellite and Unmanned Aerial Vehicle (UAV) scheduling problems that relate to the Coordination Planner Problem (CPP), discusses a number of dynamic planning models, reviews past literature on coordinated planning, and explains the choice of our mathematical model. The second section details the value function that we use for the optimization problems we solve over time. The third section describes the optimization problem we formulate to decide which requests to query for tasking on each sub-planner at any time step. The fourth section explains how our models are incorporated into software.

4.1 Literature Review

This section reviews the literature that addresses problems related to the CPP. We first review a number of mathematical models that address planning and scheduling for space-based observation missions. Next, we review some techniques for solving large planning problems involving unmanned vehicles, especially UAVs. We then present dynamic models whose solutions depend on the outcome of future events. We follow our discussion of dynamic models with a review of literature that addresses coordinated planning problems. The final class of
problems we discuss relates to the classic assignment problem, an optimization problem that is well-established in the literature. The last subsection explains the choice of the mathematical model that we present in the later sections of Chapter 4.

4.1.1 Space-Based Earth-Observation Mission Planning

We begin the literature review by summarizing how space-based Earth-observation missions have been planned in the past and discussing how these methods apply to the CPP in this thesis. Satellite mission scheduling problems are highly constrained problems with many objectives that need to be balanced. Moreover, they are high-dimensional problems involving a great deal of stochasticity. This motivates the hierarchical decomposition approach that Abramson, et. al. used, as discussed in Section 3.4.2 [39]. Other satellite mission planning formulations address different variations of the problem by including dynamically arriving targets, cloud cover prediction, and precedence constraints on the order of observations, among other issues.

In [40], the authors address the problem of managing the daily operations of a single satellite (SPOT). The strength of their model is the great detail with which they model the SPOT sensors. They formulate an Integer Programming (IP) problem and a Valued Variable Constraint Satisfaction Problem (VVCSP) model using their high-fidelity model of the three SPOT sensors. They test the performance of the IP formulation, multiple solution methods for the VVCSP, and a number of approximate methods including greedy algorithms and a Tabu Search method. The problem does not include time windows on requests and considers only one spacecraft. Because the model was constructed specifically for the SPOT satellite, it bears very little relation to a UAV planning problem. However, the approach could be modified for planning operations of a single imaging satellite.

In [41], the author presents a number of different scheduling algorithms for satellite mission planning with multiple spacecraft and targets with time windows. He first presents the one-pass scheduler (OPS) algorithm that assigns each target to its earliest possible time slot in priority order. The sequential algorithm (Seq) assigns targets in a similar fashion (one target at a time) but chooses to schedule those tasks with the earliest time windows first. In addition to these greedy algorithms, the author presents a Neural Network-based algorithm that iteratively assigns tasks to resources. The author evaluates each solution method using a Figure of Merit
(FOM) statistic. The FOM is a weighted linear function of multiple criteria, including the
target’s priority, the fraction of the desired observation duration for the target that is actually
scheduled, and the value of the temporal position of the observations within each target’s time
window. This function demonstrates how a planner can balance multiple objectives dealing
primarily with the temporal constraints of the problem.

The authors of [42] use Markov Decision Processes (MDP) to incorporate uncertainty
into the operational planning of a single satellite through a complex value function. They
calculate the number of Remaining Feasible Opportunities (RFO’s) for each image and use the
RFO’s to deem certain images more urgent than others. The authors also incorporate cloud
prediction data into the solution method. They select a feasible set of photographs for the
current day, \(d_c\), that maximizes the sum of the weights, \(w\), of the photographs, \(p\), where the
weights are calculated as follows:

\[
w(p, d_c, \pi^*) = g(p) \cdot p_r(p, d_c) \cdot P_{ef}(p, d_c, \pi^*)
\]

where

\[
P_{ef}(p, d_c, \pi^*) = \begin{cases} 
1, & \text{if } RFO(p, d_c) = \emptyset \\
\prod_{d \in RFO(p, d_c)} (1 - p_r(p, d)) \cdot p_s(p, d, \pi^*), & \text{otherwise}
\end{cases}
\]

The authors define \(\pi^*\) as the optimal policy, \(g(p)\) as the actual gain of the realized photograph \(p\),
\(p_r(p, d)\) as the realization probability of photograph \(p\) on day \(d\), \(p_s(p, d, \pi)\) as the selection
probability of photograph \(p\) on day \(d\) under policy \(\pi\), \(P_{ef}(p, d, \pi)\) as the non-realization
probability of photograph \(p\) after day \(d\) under policy \(\pi\), and \(RFO(p, d)\) as the set of days after
day \(d\) with RFO’s for photograph \(p\). The selection probability \(p_s\) is a function of the
photograph’s weight \(w\), location (relative to high demand areas), and type (e.g. stereo versus
mono). The selection probability increases for photographs with higher weights, lower demand
in their area (smaller probability of conflict with other photographs), and photograph types that
consume fewer resources. To determine the exact function for the selection probability, the
authors suggest an on-line learning approach that modifies the function as the planner receives
feedback on past policies.

The methodology finds an optimal policy by first examining the last opportunity
(photograph such that \(RFO(p, d) = \emptyset\)), and ending with the current one, alternating
computations of selection probabilities and weights. Key insights of this methodology include
the use of RFO's and a complex value function to incorporate a wide range of data into the solution methodology. We adopt these concepts for our approach as well.

In [43], the authors address the more complex problem of satellite scheduling involving polygonal targets. They address a problem with time windows and required observation durations, as well as non-convex polygonal target areas. They model the value of observing portions of the target areas with a partial reward function, \( P(x) \), where \( x \) is the fraction of the target area observed. \( P(x) \) can be linear or non-linear, but is usually a piecewise linear function that allows the authors to favor the complete viewing of a single polygon more than partial viewings of multiple polygons. They preprocess the known request set \( R \) to generate a set of images, \( I \), for the upcoming planning horizon based on their assumptions of the geometry of possible images. They then choose which images to observe by maximizing the objective \( Q \), where

\[
Q = \sum_{i \in I} W_i P \left( \sum_{i \in I_r} \frac{A_i}{A_r} x_i \right)
\]  

(4-3)

Image \( i \) is associated with request \( r \), \( W_i \) is request \( r \)'s weight, \( I_r \) is the set of images associated with request \( r \), \( A_r \) is the surface area of request \( r \), \( A_i \) is the surface area of image \( i \), and \( x_i \) is the acquired fraction of the image \( i \). They maximize this objective (4-3) subject to various constraints. The constraints force sensible paths, meaning there is an image preceding and succeeding each selected image, ensure that each image begins within its feasible time window, ensure only one image can be taken per strip, and force stereoscopic images for the pairs of stereoscopic images in the set of all images. The authors note that solving their Mixed-Integer Programming (MIP) formulation gives poor results in practice due to the fact that it only chooses starting times for each image, and if the time windows are relatively wide, the order of the images and the duration of each of the observations are unclear. So, they also solve the problem with a greedy algorithm, a dynamic programming algorithm, and local search heuristics. However, none of these methodologies can utilize a non-linear reward function, and, except for the local search heuristics, each has trouble incorporating the constraints that relate images, such as those required for stereoscopic images. Nonetheless, the authors find that local search heuristics provide the best approach to create reasonable solutions while accounting for the complex imaging constraints. We present this formulation to demonstrate how one might incorporate polygonal targets into a formulation, and also to highlight the difficulties inherent...
in solving problems with polygonal targets. It also shows the utility of local search heuristics in solving an operational problem with many complex constraints.

Another version of the satellite mission planning problem addresses targets with precedence constraints as discussed in [44]. The authors first formulate a stochastic IP problem, and use a rolling horizon approach to effectively remove random variables from the formulation. Using Lagrangian relaxation, they decompose the problem into sub-problems that are solved by a linear search method, and update the multipliers using a standard subgradient method. The model lacks direct applicability to the CPP but this formulation is useful for demonstrating the use of a Bernoulli random variable to model the probability that an image is successfully obtained, given uncertainty in the problem.

The authors of [45] formulate the satellite mission planning problem as a network-based shortest path problem. A directed acyclic digraph $G^k$ is defined for each satellite $k$, where each node in the graph represents a Data Take Opportunity (DTO) for the satellite and each arc represents a movement the spacecraft must make or set-up process it must complete. For each graph $G^k$, the authors define the sets $K$ of all satellites, $W$ of all images, $V^k$ of all nodes, $A^k$ of all feasible arcs, $\Gamma^k_i$ and $\Psi^k_i$ of all successor and predecessor nodes for node $i$, and $X^k$ of feasible flow variables given data storage and other operational constraints. The decision variables $x^k_{ij}$ indicate flow on arc $(i, j)$, from DTO $i$ to DTO $j$, where nodes $o^k$ and $d^k$ are source and sink nodes for satellite $k$, respectively, and $v^k_{ij}$ indicate values associated with each image. They create the following IP problem:

$$\text{max } \sum_{k \in K} \sum_{(i,j) \in A^k} v^k_{ij} x^k_{ij}$$

subject to

$$\sum_{k \in K} \sum_{(i,j) \in A^k} x^k_{ij} \leq 1 \quad \forall w \in W$$

$$\sum_{j \in \Gamma^k_i} x^k_{ij} = 1 \quad \forall k \in K$$

$$\sum_{j \in \Psi^k_i} x^k_{ij} - \sum_{j \in \Psi^k_i} x^k_{ji} = 1 \quad \forall k \in K, \forall i \in V^k$$

$$\sum_{j \in \Psi^d_k} x^k_{jd} = 1 \quad \forall k \in K$$
The objective function (4-4) seeks to maximize the value of the images observed by choosing which spacecraft actions, $x^k$, to carry out. Constraints (4-5) ensure each request is executed at most once. Constraints (4-6) - (4-8) state the conservation of flow requirements for the graph. Constraints (4-9) indicate that only decision variables that represent feasible operational paths are included, while constraints (4-10) reflect the fact that decision variables are binary decisions.

Using Lagrangian relaxation on constraints (4-5), the authors exploit the structure of their formulation to decompose the problem into sub-problems that are Shortest Path Problems with Resource Constraints (SPPRC) for each satellite. They solve the SPPRC Lagrangian relaxation problems with a dynamic programming algorithm, and solve the Lagrangian dual problem using a standard subgradient method.

In [46], the authors reformulate this problem and evaluate the LP relaxation of the new formulation, which they show to provide a much smaller integrality gap. The formulations by Gabrel, et. al., solve the multi-satellite multi-image assignment problem with time windows, data storage constraints, and image viewing types (“spotlight” versus “widefield” images) through an intelligent network formulation. One could imagine how UAV routes might be incorporated into this network as well. However, the formulation fails to deal with the asynchronous nature of the problem and, because it was not created within a coordinated planning construct, it directly creates the viewing schedules of each sensor.

Other published approaches include [47], where the authors study the application of thirteen local search heuristics to a satellite-based Earth observation planning problem, concluding that a “simulating annealing” approach with a carefully chosen cooling procedure works well for these types of problems. In [48], the authors apply three algorithms to the satellite scheduling problem, including a greedy dispatch algorithm and a look-ahead algorithm, as well as a complex Genetic Algorithm (GA). These publications further demonstrate the need for flexible algorithms when dealing with Earth-observation planning involving space-based assets.

The main difference between the problems these studies address and the problem we address in this thesis is the fact that our problem deals with both spacecraft and aircraft. Many
of these formulations are closely tied to astrodynamics and thus cannot be directly applied to a problem with both satellites and UAVs. Moreover, we do not assume the coordinator has the tactical control of assets that is assumed in the literature discussed above. That is, the coordination planner does not solve the problem of choosing efficient routes for individual UAVs and gimbal angles for individual sensors. The existence of asynchronous planning cycles present additional complications for the CPP that remain unaddressed by the examples above. However, there are portions of each solution approach that are useful for the problem we address. Most notably, the ways that complex value functions can account for competing objectives, uncertainty, and even constraints in satellite mission scheduling problems seem to be applicable to the CPP.

4.1.2 UAV Mission Planning

We discussed in Section 3.4.1 the composite-variable approach implemented by Negron [38] to solve the large-scale heterogeneous UAV planning problem in three dimensions. The advantage of this formulation is the speed with which it generates efficient plans. The remainder of this section reviews a number of other planning methods applied to UAV planning problems.

In [49], the authors decompose a complex UAV planning problem into a hierarchy with three levels. The top-level problem deals with how to organize individual aircraft into teams of aircraft that can be allocated to a cluster of targets of interest. The clusters of targets are organized by considering the relative geographical distances between targets (by solving shortest-path problems) as well as the precedence of actions to be taken against the targets, based on the target types (enemy air defense systems, communications centers, etc.). The next stage involves heuristically creating teams of aircraft to address each cluster of targets. The solution to this mid-level problem provides routes for the aircraft and creates detailed action sequences for each aircraft. Finally, the solution to the bottom-level problem refines the routes created by the mid-level plan, creates detailed sensor and weapon planning, and generates the constraint matrix used for the top-level composite-variable program. The composite-variable optimization problem chooses the options for each UAV team, where an option includes routing, timing, and sensor/weapon usage decisions. Although the UAV planning problem in this paper does not deal with many of the constraints of the CPP, we discuss this work to
demonstrate the use of a hierarchical decomposition for a complex problem that is similar to the framework in which we formulate the CPP.

In Philemon Sakamoto's Master's thesis [50], he addresses a UAV planning problem with uncertainty. He formulates a network-based model that chooses routes and actions for UAVs of different types. The uncertainty lies in three sets of constraints. The first constraints correspond to set-up time constraints. There is also uncertainty in the time windows of each target, representing the idea that enemy movements are uncertain. Finally, the author assumes uncertainty in the constraints on the ranges of UAVs. Sakamoto applies the Robust Optimization (RO) methodology proposed by Bertsimas and Sim [51] to this Vehicle Routing Problem with Time Windows (VRPTW). While the centralized, synchronized version of the CPP may be thought of as a VRPTW, the operational problem we address is neither centralized nor synchronized, making it difficult to formulate the problem in terms of a network graph. This approach is, nonetheless, worth considering because it suggests an approach to the completely centralized problem, and does so in a robust fashion. However, we deem the solution approach we present later in this chapter more suitable to the CPP given the operational constraints of the problem.

4.1.3 Dynamic Models

This subsection discusses three solution approaches that are dynamic in nature. They create solutions with decisions contingent on the outcome of the system over time. The first technique we discuss is Stochastic Programming (SP). SP allows one to break the original mathematical programming problem into stages. It assumes uncertain data and at each stage, some new information becomes available to the decision-maker. However, SP solves the problem deterministically. That is, it obtains optimal decisions for each stage, for each possible scenario that could occur given the uncertainties in the data. Thus, the solution provides initial decisions as well as a policy for future decisions contingent on actual realizations of data [52]. At first, SP appears to provide a reasonable approach to solve a problem where the coordination planner receives feedback after each query. However, the following example shows that this is not the case.

One could imagine an SP formulation for the CPP with binary integer decision variables for each request-to-sub-planner pair, for each communication iteration with the sub-planners. A
decision variable would take value 1 if the coordinator sends that request as a query to that sub-planner for a single communication iteration, and 0 otherwise. A solution to the stochastic program would provide initial assignments of requests to sub-planners and would also provide assignments for later stages for every case of feedback that could possibly be received from the sub-planners.

In formulating an SP problem, it is important to define the stages carefully. Here, we could imagine solving one SP for every one of the coordinator’s planning periods. That is, stages correspond to different communication iterations with sub-planners in the same planning period. This represents the idea that we could run some iterative assignment algorithm that allocates requests to sub-planners, gets feedback from the asset managers as to whether or not the request will be executed, and then uses the feedback information to reassign requests to assets based on the scenario observed. Formulating the problem in this way, however, would require an enormous number of decision variables. To illustrate, we consider a two-stage formulation with a known set of requests. This example would entail $(2P+1)^R$ possible scenarios in stage 2, where $P$ and $R$ are the number of sub-planners and requests, respectively. This term is derived from the fact that we would have to consider all combinations of assignments of requests to assets, as well as the feedback we receive (if we assume a binary yes/no response). A request could be assigned to a sub-planner and have the request executed, or not executed $(2P)$, or a request could not be assigned to any assets (hence the additional scenario). With 500 requests and 10 assets (a very reasonable problem size), this problem has $9.37 \times 10^{556}$ possible scenarios in stage 2. Even Bender’s decomposition, a method for solving problems with a “block-ladder” structure such as those in SP [52], would struggle to solve problems of this size. Thus, the solution method would need to intelligently generate a subset of all scenarios, or else the problem is intractable. We do not pursue SP further in this thesis.

Another example of a dynamic formulation combines linear programming (LP) with Partially Observable Markov Decision Processes (POMDP). This methodology is used by Eric Zarybnisky in [53] to address an intelligence, surveillance, and reconnaissance (ISR)/strike problem. POMDPs are useful for determining courses of action while modeling the probabilistic nature of a battlefield environment. They differ from the more standard MDP in that the state of the system is not precisely known after each action is taken. Zarybnisky uses POMDPs to generate new columns, or policies, for each target type, where each policy contains a series of
decisions contingent on observations of the true state of the system. A master LP problem chooses which policies to undertake for each target type. He alternates between generating columns and solving LP problems until the solution quality of the LP is within a tolerance level. Zarybnisky’s approach could provide a way to address a stochastic version of the CPP. However, we deem the inclusion of the probabilistic nature of interaction among the coordinator and the sub-planners beyond the scope of this thesis, as discussed in Section 3.2.1, and leave it for future work.

4.1.4 Coordinated Planning

In this section we review literature pertaining to the coordination of multiple agents. It is important to note that most of the following formulations of coordination planners assume sub-planners, or agents, are entirely subservient to the coordinating agent. This is a fundamental difference between the CPP and more common coordinated planning problems. Also, coordinated planning is a problem rarely addressed, especially when dealing with both air- and space-based assets. In fact, to our knowledge, only in [37] do we observe a similar problem where the author presents algorithms for coordinating the actions of aircraft and space-based sensors and weapons.

We begin the review of past coordinated planning efforts with [54]. In this paper, the authors address the coordinated planning problem in a manner similar to our approach. That is, they highlight the need for a coordination planner for future missions where the level of coordination lies somewhere between centralized, synchronous planning and fully distributed planning. The authors foresee the need for coordinated planning for future Earth Science campaigns where pointable sensors are more plentiful. They formulate a Constraint Programming (CP) model for collection plans, allowing users to relax constraints as time progresses to allow previously infeasible plans to be executed. CP is useful for problems with many complex constraints, as can exist in satellite-based observation problems. We acknowledge that the use of CP is one possible solution approach to a version of the CPP. However, we address a problem where the coordinator interacts with the sub-planners frequently. Nonetheless, to our knowledge this publication is the only literature that addresses the problem of Earth-observation in a coordinated and decentralized sense.
In [55], Freuder and Wallace use a CP approach to coordinated planning for a satellite scheduling problem. They model complex requests by creating a variable for each request and associated support variables for the event windows, required resources, request execution times, and durations for the requests. The authors generally create multiple request/support variables to model a single request with complex observation requirements. They use heuristic search and constraint propagation methods to create feasible schedules in a timely manner. The authors in [56] address the problem of dynamically updating CP solutions in a real-time setting. However, we leave the use of CP for the CPP as future work in a centralized version of the problem.

In [37], Wroten addresses the problem of dynamic, coordinated planning of ISR and strike operations for air- and space-based assets in a military Area of Responsibility (AOR). Wroten creates a greedy, heuristic algorithm for coordination, as well as a stochastic integer optimization-based method that was never implemented due to software limitations. The heuristic algorithm, the Heuristic Air and Space Operations Coordinator (HASOC), has knowledge of sub-planner capabilities and sequentially assigns targets to sub-planners based on the weights given to each target. The weights are a function of the target's inherent value, the risk level for the asset being considered, and the amount of time remaining in the scenario. The HASOC only considers each target's value to a single sub-planner at a time and only allocates targets to the next sub-planner once the previous sub-planner has been fully utilized. Wroten's approach demonstrates the use of heuristics in generating target weights and the importance of creating an effective value function.

A UAV coordination problem is presented in [57], in which the authors decompose a very complex problem into five sub-problems, two of which correspond roughly to the coordination level in which the CPP resides. The authors solve the coordination problem in the first two stages of the problem, called the cooperative target assignment and coordinated UAV intercept stages, and use the last three stages to construct detailed UAV paths subject to various constraints. In the first phase, the target assignment phase, each UAV is assigned to a target in such a way that each target has, if possible, multiple UAVs assigned to it. Preference is given to higher-valued targets. Using an objective function that is a careful blend of the number of targets assigned, threat exposure, and fuel consumption, the algorithm chooses a set of targets that yields maximum value for each individual UAV. The coordinated UAV intercept step tries
to ensure that teams of UAVs assigned to the same target survey or strike that target at similar times. The algorithm takes the $n$ known target locations and creates a Voronoi diagram in which a space is decomposed into sectors whose boundaries are equidistant from a discrete set of points. The segments separating the $n$ convex regions in the diagram correspond to arcs that UAVs could potentially travel on. Each arc has a cost associated with it that is a function of the distance travelled along the arc as well as the threat level faced on the arc. The threat level on an arc is a function of the distance to nearby threats at discrete locations on that arc. The authors then implement Eppstein's $k$-best paths algorithm to search the graph for the best paths for each UAV. The coordinated intercept step chooses waypoints and velocities for each UAV that determine the times at which UAVs must observe the same target such that the threat exposure and distance travelled is minimized. The paper presents a very high-fidelity model (including details such as UAV turning rate constraints) that the authors show performs well in practical examples. This paper is yet another example of hierarchical decomposition for a coordination problem and incorporates simultaneous actions by multiple assets against targets. While we could not implement these exact methodologies in the CPP because we have more restrictive operational constraints, it is important to note how the authors combine a carefully constructed value function with local search heuristics for the coordination phases.

A study involving both centralized and distributed coordinated planning is presented in [58]. This paper compares three different methods for coordinating the activities of rovers in a geological science scenario. The authors test a fully centralized system, where the centralized planner assigns targets to and creates detailed routes for each rover to carry out, as well as a decentralized system, and a contract net protocol that conducts an auction amongst the rovers for each target. In the decentralized system, the coordination planner assigns targets to each rover but a planner for each rover creates the detailed routes and timing of the activities. In the contract net protocol, the central planner coordinates an auction among the rovers for each target. The authors use a CP modeling language called the Automated Scheduling and Planning Environment (ASPEN [59]) to create initial routes for the rovers and use an insertion heuristic to improve the existing routes. The auction approach allows the rovers to complete the most tasks (with the highest computation time), while the centralized approach performs slightly worse. The decentralized approach performs worse than the centralized approach but manages to keep computation time very low compared to the other two planning methods. We note the use of an
auction mechanism to model interaction between the coordinator and its sub-planners as an interesting solution method. It is computationally expensive but yields strong results. However, because we do not include the ability to purchase an image/measurement in our model, the sub-planners have no incentive to bid for incoming requests. Yet, it could be important to consider the ability to purchase the right to collect data, and we leave this concept’s inclusion for future work.

The final paper we review on coordinated planning deals with the MIP formulation of a commercial air traffic routing problem. In [60], the authors introduce a binary integer variable that results in a very strong LP relaxation and thus allows the formulation to be applied to large-scale, realistically-sized problems. The method creates a binary integer variable for every sector that each flight passes through for every time step. The variable takes on a value of 1 if flight $f$ has entered sector $i$ by time $t$, rather than the standard at time $t$. The model performs extremely well at organizing the temporal movements of a large number of aircraft. We mention it in this section as a possible starting point for an IP formulation for a centralized, synchronous version of the problem, and perhaps as a route modification mechanism, given initial routes.

4.1.5 Assignment Problems

The final type of problem that we review is the assignment problem. The assignment problem can be thought of as a problem with $n$ people and $n$ projects, where the objective is to assign the people to projects in a way to maximize the value gained by the assignments. We adapt the following assignment problem from [52]:

\[
\begin{align*}
\text{maximize} & \quad \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} f_{ij} \\
\text{subject to} & \quad \sum_{i=1}^{n} f_{ij} = 1, \quad \forall j = 1..n \\
& \quad \sum_{j=1}^{n} f_{ij} = 1, \quad \forall i = 1..n \\
& \quad f_{ij} \geq 0, \quad \forall i,j
\end{align*}
\]

Here, the objective (4-11) maximizes some linear cost function, where $f_{ij} = 1$ if the $i$th person is assigned to the $j$th project, and $f_{ij} = 0$ otherwise. Constraints (4-12) and (4-13) state that each project must be assigned to one person, and each person must be assigned to one project,
respectively. Constraints (4-14) enforce non-negativity in the decision variables. The nature of the problem suggests that a binary integer constraint on the \( f_{ij} \) variables is needed. However, due to the structure of the constraint matrix and the fact that constraints (4-12), (4-13), and (4-14) imply \( f_{ij} \leq 1 \), the binary integer constraints can be relaxed. Specifically, the constraint matrix for the assignment problem shown above is \textit{totally unimodular} (TU), meaning if we relax the binary integer constraints we still obtain an integer optimal solution [61]. This property is desirable in the constraint matrix of an optimization problem because it means the problem can be solved as an LP (i.e., quickly).

The classical assignment problem deals with the case where the number of people equals the number of projects, but often this is not the case, and this is called the asymmetric assignment problem. There are many other variants of the assignment problem, including those with: maximum capacities for each project, a single capacitated project (knapsack problem [52]), no constraints on whether or not each person is assigned, non-linear objective functions (e.g., the quadratic assignment problem), and objective functions that seek to maximize the minimum value of assignments (linear bottleneck assignment problem, [62]). One variant of interest is the dynamic assignment problem, studied by Spivey and Powell in [63].

The dynamic assignment problem considers tasks and resources that arrive at and depart from the system dynamically over time. Spivey and Powell address the problem in the special case where each resource can service only one task at a time. They use the Markov property, which says that the state of the system at the next time-step is only dependent on the state at the current time step. They formulate the problem as a dynamic programming (DP) problem. To solve realistically-sized problems, Spivey and Powell approximate the cost function of the dynamic program linearly. The authors show that their dynamic methods outperform myopic methods in most cases, and they conduct extensive analysis on the value of future information.

This section has discussed various types of assignment problems seen in the literature. In subsequent sections, we explain how we solve a variant of the assignment problem at each iteration of communication with sub-planners.
4.1.6 Choosing a Model

Having reviewed literature related to the CPP, this section discusses how we choose a mathematical model for the problem in this thesis. We choose a solution method based on the applicability of past techniques, and other factors such as computer software restrictions and time, but primarily on how well the method deals with the operational constraints of the problem we address.

In [39], the authors state that a complex problem like the multi-satellite problem they deal with warrants hierarchical decomposition. We have noted already that the operational problem we are addressing in this thesis lends itself to a hierarchical decomposition. This is depicted in Figure 4-1:

We see that if we allow sub-planners to independently route their assets, it is natural to construct the coordination planner as a top-level planner in the hierarchy. The authors also note that it is unwise for the highest level in the hierarchy to create detailed, long-term plans for a stochastic problem such as theirs because “detailed actions planned on the basis of a specific prediction of the future may become obsolete well before they are to be executed due to an inability to accurately predict the future” [39]. While we relax some of the probabilistic aspects of the real-world problem for this thesis, we do allow uncertain information (whether or not sub-planners will execute queried requests) to become realized over time. Thus, we seek a solution method that is flexible to this feedback and that intelligently determines which assets are suitable for each request.
However, in [58], in attempting to coordinate the operations of rovers, the authors note that a decentralized approach can make coordination of activities more difficult. This is complicated in the CPP by the fact that sub-planners are not entirely subservient to the coordinator.

Finally, we note the utility of iterative algorithms and heuristics in solving past spacecraft operations planning problems, which are featured in the CPP. In [59], the authors use “early-commitment, local, heuristic, iterative algorithms” and declare that “using an iterative algorithm allows automated planning to be utilized at any time and on any given initial plan.” The authors also state that their local algorithm does not need to “maintain intermediate plans or past [solution] attempts.” However, our solution approach must incorporate the information gained from past iterations to account for the feedback received from sub-planners. We intelligently do so in our value function, which is described in the Section 4.3.

4.2 Coordination Planner Functional Architecture

Our solution approach embeds a carefully constructed value function within a modified version of an assignment problem to intelligently query sub-planners. The value function is forward-looking, and accounts for the physical constraints in the problem as well. We alter sub-planner target lists based on the assignments produced by the optimization problem and call the sub-planners to generate realistic and actionable feedback. Moreover, we use intuitive heuristics to incorporate feedback data into the value function and use a rolling horizon-type approach to accommodate interaction with sub-planners over time. To give the reader an idea of the functions, interactions and corresponding information flow in the coordination planner, we present the functional architecture of the coordination planner in Figure 4-2:
The inputs to the coordination planner include data from the four types of inputs discussed in Section 3.3.1.1. The opportunity finder finds feasible opportunities given the request and sub-planner sets, while the target-to-sensor value lookup provides additional input to the quality of those opportunities. Together, these two functions provide situation assessment for the coordinator, in the form of inputs to the value function. This value function is used in the objective function for the optimization problems (assigner) we solve over time, which assign requests to sub-planners as queries. The sub-planners receive these requests, as well as those from external sources, generate their observation plans, and provide feedback to the coordination planner on which requests are accepted/rejected. The final function in the coordination planner is the plan comparer, which interprets the feedback from the sub-planners and updates the value function based on its findings. We explicitly discuss the value function, assigner, and plan comparer functions in the remainder of this chapter, while the opportunity finder and target-to-sensor lookup functions are covered implicitly in the discussion of the value function.
4.3 Value Function Construction

This section describes each component of the value function we use and how we combine the components to address the problem. We discuss reasons for including each component of the value function, as well as how each component is computed and appropriately scaled. The scaling of each component is important because we use a convex combination of the value function components for the static assignment problems we solve.

4.3.1 Convex Combination of Value Function Components

To provide a background for the sections that detail the value function, we first describe the equation that calculates the value for each request to sub-planner pairing. The function is a linear, convex combination of seven components and is the objective function for the optimization problems we solve over time.

For any vectors \( x^1, \ldots, x^m \) in \( \mathbb{R}^n \) and \( \lambda_1, \ldots, \lambda_k \) nonnegative scalars whose sum is unity, the vector \( \sum_{i=1}^{m} \lambda_i x^i \) is a convex combination of \( x^1, \ldots, x^m \) [52]. In the case of our value function, \( n = 1 \) and \( m = 7 \), implying that each \( x \) is a scalar. We introduce the weights \( w_1, \ldots, w_7 \) in place of the \( \lambda_i \)'s and proceed as follows: for any request \( r \) and sub-planner \( p \), the scaled value, \( \tilde{v}_{rp}^{ka} \), is a convex combination of the scalar values and the weights \( w_i \), where \( \sum_i w_i = 1 \). Therefore, each value computed lies in the range \([0, 1]\). Algebraically, we calculate the value for any request, \( r \), to sub-planner, \( p \), pair as follows:

\[
\tilde{v}_{rp}^{ka} = \begin{cases} 
  \sum_{a} w_p \tilde{p}_r + w_{TS} \tilde{TSV}_r + w_o \tilde{a}_r + w_d \tilde{d}_r + w_{RF} \tilde{RF}_{a} + w_{simult} \tilde{simult}_{a} + w_{TW} \tilde{TW}_r, & \text{if } \tilde{a}_r > 0 \\
  0, & \text{otherwise}
\end{cases}
\] (4-15)

We include the superscript \( k \), corresponding to the iteration number, because it allows the value to change based on the feedback received from the various sub-planners in past queries. While, in general, a convex combination has an important geometric interpretation, here it provides a construct for relative valuation of request-to-sub-planner pairs. Subsequent sections describe the notation used in this function and how each component is calculated.

4.3.2 Notation and Definitions

This subsection defines the sets, data inputs, and decision variables for our mathematical formulation. Table 4-1 defines the sets used to define the model:
Set Description

- \( A \): Set of planning periods (periods sub-planners are available for querying for a unique set of execution phases)
- \( R \): Set of all requests
- \( \bar{R}^k_a \): Set of requests that have not yet been serviced and have Remaining Feasible Opportunities (RFO's) at iteration \( k \) of planning period \( a \), \( \bar{R}^k_a \subseteq R \)
- \( G \): Set of target types
- \( D \): Set of pairs of dual collects
- \( D' \): Set of all single requests that are associated with dual collects, \( |D'| = 2|D|, D' \subseteq R \)
- \( P \): Set of all sub-planners
- \( P^U \): Set of UAV sub-planners, \( P^U \subseteq P \)
- \( P^S \): Set of satellite sub-planners, \( P^S \subseteq P \)
- \( S \): Set of all sensors (assets) in the system
- \( S_p \): Set of sensors (assets) on sub-planner \( p \in P \)
- \( U_p \): Set of UAVs on sub-planner \( p \in P^U \)
- \( Sat_p \): Set of satellites on sub-planner \( p \in P^S \)
- \( V_{r_p}^a \): Set of assets on sub-planner \( p \) that can view request \( r \) in planning period \( a \)

| Table 4-1: Sets |

Next, Table 4-2 defines the inputs associated with request data:

<table>
<thead>
<tr>
<th>Request Data Inputs</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( TgtType_r \in G )</td>
<td>The target type associated with request ( r \in R )</td>
</tr>
<tr>
<td>( Loc_r \equiv (lat_r, lon_r, alt_r) )</td>
<td>The location of the target associated with request ( r ), as defined by its latitude (degrees), longitude (degrees), and altitude (meters), ( \forall r \in R )</td>
</tr>
<tr>
<td>( ETW_r \in [0, H] )</td>
<td>The early time window for the observation associated with request ( r \in R ) (in hours since the start of the scenario)</td>
</tr>
<tr>
<td>( LTW_r \in [ETW_r, H] )</td>
<td>The late time window for the observation associated with request ( r \in R ) (in hours since the start of the scenario)</td>
</tr>
<tr>
<td>( Dur_r \in [0, LTW_r - ETW_r] )</td>
<td>The minimum required duration of the observation associated with request ( r ), ( \forall r \in R ) (in hours)</td>
</tr>
<tr>
<td>( Priority_r )</td>
<td>The priority level of request ( r \in R )</td>
</tr>
<tr>
<td>( Dual_r = \begin{cases} 1, &amp; \forall r \in D' \ 0, &amp; \forall r \in R \setminus D' \end{cases} )</td>
<td>Takes value 1, if request ( r ) is part of a dual collect, ( \forall r \in D' ) \newline Takes value 0, if request ( r ) only requires a single observation, ( \forall r \in R \setminus D' )</td>
</tr>
<tr>
<td>( Related_r = \begin{cases} j, &amp; \forall r \mid (r, j) \in D \ 0, &amp; \forall r \in R \setminus D' \end{cases} )</td>
<td>Takes value ( j ) if ( (r, j) \in D ), defining the other part of a dual collect associated with request ( r ), ( \forall r \in D' ) \newline Takes value 0 if request ( r ) only requires a single observation, ( \forall r \in R \setminus D' )</td>
</tr>
</tbody>
</table>

| Table 4-2: Request Data |
Table 4-3 presents the data inputs for each planner:

<table>
<thead>
<tr>
<th>Planner Data Inputs</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$W_p$</td>
<td>The planner type of sub-planner $p$, $\forall \ p \in P$</td>
</tr>
<tr>
<td>$PL_p$</td>
<td>The length of each planning phase of sub-planner $p \in P$ (in hours)</td>
</tr>
<tr>
<td>$UL_p$</td>
<td>The length of each send/upload phase of sub-planner $p \in P$ (in hours)</td>
</tr>
<tr>
<td>$\theta_p$</td>
<td>The number of requests sub-planner $p$ will accept as queries at any iteration, $\forall \ p \in P$</td>
</tr>
<tr>
<td>$ERP_p$</td>
<td>The number of requests internal to sub-planner $p \in P$ (external to the coordinator)</td>
</tr>
<tr>
<td>$\tau_p$</td>
<td>Mean value of a request allocated to sub-planner $p$ (to the sub-planner)</td>
</tr>
</tbody>
</table>

Table 4-3: Planner Data

Here, we assume that the length of each planning phase and upload phase is always the same for any one planner. That is, planners operate on repetitive planning/uploading phases. Calculations for the start and end times of each planning, sending/uploading, and execution phases are shown in Appendix C. We then determine the start and end times of each artificial planning period based on these data. This data generation process amounts to nothing more than bookkeeping, but tracking these planning cycles is important and provides a clear benefit. Here, we give a brief demonstration of how sub-planners and their execution phases correspond to planning periods.

We let each entry $b_{pa}$ of the $|P| \times |A|$ matrix $B$ contain the execution phase for which a planner $p$ is planning in period $a$. That is, for any planning period $a$, the set of execution phases for which the coordinator can query sub-planners is $\{b_{1a}, b_{2a}, ..., b_{|P|a}\}$. Because planning periods allow us to determine which execution phases each sub-planner is currently creating plans for, the matrix $B$ provides a construct for mapping the planning cycles of individual sub-planners to our artificial planning periods. For planning periods 1-6 in the example in Figure 4-3, the matrix $B$ is shown below:

$$B = \begin{pmatrix} 1 & 1 & 2 & 2 & 2 & 2 \\ 1 & 1 & 1 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 2 & 2 \\ 1 & 2 & 2 & 2 & 2 & 3 \end{pmatrix}$$  (4-16)
Thus, given a planner \( p \) and a planning period \( a \), we can determine the period of time for which this sub-planner is currently creating observation schedules by accessing \( b_{pa} \).

Additionally, some system-level data are required to specify the locations of satellites at certain times based on the orbit propagation methods described in 3.2.4.1, and how often we calculate these locations. Other data are required to specify the length of the planning horizon we are considering. The following table specifies the data inputs for the system's characteristics:

<table>
<thead>
<tr>
<th>System Data Inputs</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( H )</td>
<td>The length of the planning horizon (hours)</td>
</tr>
<tr>
<td>( itrsPerHour )</td>
<td>The number of iterations with sub-planners (per hour)</td>
</tr>
<tr>
<td>( startTime )</td>
<td>Defines the start time of the scenario</td>
</tr>
<tr>
<td>( \Delta t )</td>
<td>The time step for satellite orbit propagation (seconds)</td>
</tr>
</tbody>
</table>

Table 4-4: System Data

Table 4-5 presents the data inputs for each sensor in the system:
Sensor Data

- \( MaxSlewX_s \in [0, \infty) \): The maximum amount that sensor \( s \) can slew in the x-direction, \( \forall s \in S \) (in degrees)
- \( MaxSlewY_s \in [0, \infty) \): The maximum amount that sensor \( s \) can slew in the y-direction, \( \forall s \in S \) (in degrees)
- \( FOVx_s \in [0, \infty) \): The FOV of sensor \( s \) in the x-direction, \( \forall s \in S \) (in degrees)
- \( FOVy_s \in [0, \infty) \): The FOV of sensor \( s \) in the y-direction, \( \forall s \in S \) (in degrees)
- \( SlewRateX_s \in [0, \infty) \): The maximum rate at which sensor \( s \) can slew in the x-direction, \( \forall s \in S \) (in degrees/sec)
- \( SlewRateY_s \in [0, \infty) \): The maximum rate at which sensor \( s \) can slew in the y-direction, \( \forall s \in S \) (in degrees/sec)
- \( \gamma_s \in [0,1] \): The cost of slewing sensor \( s \), \( \forall s \in S \) (per degree)

<table>
<thead>
<tr>
<th>UAV Sub-planner Data</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BaseLoc&lt;sub&gt;p&lt;/sub&gt; = (BaseLat&lt;sub&gt;p&lt;/sub&gt;, BaseLon&lt;sub&gt;p&lt;/sub&gt;)</td>
<td>The base location of the UAVs on planner ( p ), ( \forall p \in P^U )</td>
</tr>
<tr>
<td>( sensor_{up} \in S_p )</td>
<td>The type of sensor on UAV asset ( u ) on planner ( p ), ( \forall u \in U_p, p \in P^U )</td>
</tr>
<tr>
<td>Endurance&lt;sub&gt;up&lt;/sub&gt;</td>
<td>The endurance (in hours) of UAV asset ( u ) on planner ( p ), ( \forall u \in U_p, p \in P^U )</td>
</tr>
<tr>
<td>MaxSpeed&lt;sub&gt;up&lt;/sub&gt;</td>
<td>The maximum speed (in knots) of UAV asset ( u ) on planner ( p ), ( \forall u \in U_p, p \in P^U )</td>
</tr>
</tbody>
</table>

Additional data, such as the minimum and maximum cruise altitude, and maximum climb and sink rates for each UAV are input to the UAV sub-planners but are not used by the coordination planner so they are not shown in Table 4-6.

The following table shows the data inputs for each satellite in the scenario. For real-world satellites, we use the satellite Kepler elements from the Two Line Element (TLE) data available on www.space-track.org every few weeks.
Satellite Sub-planner Data Description

| numGS_p | The number of ground stations accessible for plan upload/data download by the satellites of sub-planner p ∈ P^S |
| dlt_sat_p | The time at which TLE data was downloaded from www.spacetrack.org for satellite sat on sub-planner p, ∀ sat ∈ Sat_p, p ∈ P^S |
| sensor_sat_p ∈ S_p | The type of sensor on satellite sat on sub-planner p, ∀ sat ∈ Sat_p, p ∈ P^S |
| a_sat_p | The semi-major axis of the orbit of satellite sat on sub-planner p, ∀ sat ∈ Sat_p, p ∈ P^S |
| e_sat_p | The eccentricity of the orbit of satellite sat on sub-planner p, ∀ sat ∈ Sat_p, p ∈ P^S |
| i_sat_p | The inclination of the orbit of satellite sat on sub-planner p, ∀ sat ∈ Sat_p, p ∈ P^S |
| Ω_sat_p | The RAAN of the orbit of satellite sat on sub-planner p at dlt_up, ∀ sat ∈ Sat_p, p ∈ P^S |
| ω_sat_p | The argument of perigee of the orbit of satellite sat on sub-planner p at dlt_sat, ∀ sat ∈ Sat_p, p ∈ P^S |
| M_sat_p | The mean anomaly of the orbit of satellite sat on sub-planner p at dlt_sat, ∀ sat ∈ Sat_p, p ∈ P^S |

Table 4.7: Satellite Sub-planner Data

Much data can be generated a priori given the input data for each sub-planner, sensor and request. Thus, it is important to understand how we use the available data for our value function, within the scope of our model.

4.3.3 Value Function Components

In this section, we describe each component of the value function that we use for the coordination planner. It is important to note that the values we generate using this function are values to the coordination planner, not to the users that input the requests or to the sub-planner managers. At each instance it interacts with the sub-planners, the coordination planner chooses queries to improve some subset of the four Measures of Performance (MOPs) discussed in Section 3.3.1.5. Thus, the values inherent to each request (i.e., priorities) and the values of each request to each sub-planner (assumed unknown) are not the coordinator’s only concern.
Also, because we formulate the problem as how to allocate requests to sub-planners, and not to individual assets, each value we generate is for a request-to-sub-planner pair. We do this because of the coordinator’s lack of direct control over individual assets, as discussed in Section 2.1.3.3. Although it is currently more common for a sub-planner to manage only one asset, it is plausible that sub-planner control of multiple assets could become the norm in the future. In either case, we account for multiple assets on a single sub-planner, but compute each value for a request-to-sub-planner pair.

4.3.3.1 Priority

The first component of the value function is priority. This is the priority of the request, \( r \), we are considering, \( \text{Priority}_r \). It is included because it contributes to the \( \text{avgPriority} \) MOP. We allow priority to take on ten discrete levels, from 100 to 1,000\(^1\).

The priority component can be computed directly from the request data. To scale this portion of the value function, we divide each priority level by the maximum priority (1,000). So, the scaled priority value of request \( r \), \( \bar{p}_r \), takes values between 0 and 1. Mathematically,

\[
\bar{p}_r = \frac{\text{Priority}_r}{1000}
\] (4-17)

This portion of the value function is a function only of the request we are considering and is independent of the sub-planner we are considering.

4.3.3.2 Observation Quality

The observation quality component of the value function is a function of the request we are considering as well as the sensors available on the sub-planner we are considering, and the time at which we are considering the pairing. As discussed in Section 3.2.7, we obtain values for this component using the lookup table in Appendix B. We include this component because of its relationship to the \( \text{avgTSV} \) MOP. While this component can be made to include other contributing factors to the quality of a potential observation, here we consider it a function only of the type of target we are considering and the type of sensors on the sub-planner we are considering. We scale the component by dividing by the maximum target-to-sensor value for

---

\(^1\) We chose these seemingly arbitrary numbers to align with the scale used by EO-1’s planning methods. However, we assume that larger integers correspond to requests with higher priority, which is not true for EO-1. In either case, it is easy to transition from one priority system to the other.

98
that target type across all sensors in the system. These values can also be found in the table in Appendix B. Again, this means the component can only take values between 0 and 1, where a value of 0 corresponds to a sub-planner that does not control a sensor with the capability to take the desired measurement/observation, and a value of 1 corresponds to the highest quality observation possible for that target type across all sensor types. An example of where a target type-to-sensor type matching would yield zero value would be when the request involves measuring the level of some electrical signal emanating from an enemy target, and the candidate sensor is a camera that can only take images in the visible spectrum. Since this type of pairing would have no value to the user, we use this component to screen potential pairings. That is, if $T_{SV}^{a} = 0$, we set $v^{ka}_{rp} = 0$, to ensure the coordinator does not try to query this request on the associated sub-planner when it represents an infeasible pairing.

We define the matrix $Q$, whose rows correspond to target types and whose columns correspond to sensor types, as a representation of the lookup table in Appendix B. We let $q_{gs}$ represent a single entry from this matrix, which stores the benefit of pairing a target of type $g$ with a sensor of type $s$. Mathematically, we calculate the scaled observation quality parameter for request $r$ with $TgtType_r = g$ and sub-planner $p$ in planning period $a$, $TSV^a_{rp}$, as follows:

$$TSV^a_{rp} = \frac{\sum_{s \in V^a_{rp}} q_{gs}}{|V^a_{rp}| \max_{s \in S} q_{gs}}$$

(4-18)

Thus, the observation quality parameter is an average of the quality of possible observations we could obtain for request $r$ on sub-planner $p$.

4.3.3.3 Physical Feasibility

This section describes how we incorporate physical constraints into the value function. We also discuss how we alter the value for any request to sub-planner pair based on how "difficult" it would be for the sub-planner to service the request, as well as how we incorporate future feasible opportunities into the valuation.

4.3.3.3.1 Feasibility Check

One major purpose of the value function in our formulation is to account for the physical constraints of the problem. For this thesis, we consider geographical and temporal feasibility as the only limiting factor for assignment potential. That is, as long as the sub-planner
has one asset that can be oriented to view the target within its time window, this request-to-sub-
planner assignment will have some value, assuming the input weights make this possible.

The value function component \( \tilde{a}_{rp} \) is the scaled number of opportunities on sub-planner 
\( p \) to view the location associated with request \( r \) during the execution phase associated with 
planning period \( a \). As discussed in Section 3.1.1.8, we assume knowledge of each sub-planner’s 
planning cycles. So, in any planning period \( a \), the coordination planner knows for which 
execution phase planner \( p \) is currently planning. Thus, the coordinator can calculate whether or 
not the assets on planner \( p \) can view request \( r \) in that execution phase. This methodology is 
particularly important for satellite sub-planners because we can predict the locations of 
satellites with some accuracy. The coordination planner does not know UAV locations at any 
time, however, because we do not allow the coordinator knowledge of existing UAV sub-
planner routes. Thus, we assume taskable UAVs in all cases.

4.3.3.3.1 Satellite Feasibility Calculations

If the sub-planner contains satellites, the feasibility of an observation is dependent on 
four things:

1) Is the location visible from some point in the orbit, and, if so, is the satellite at that 
location?

2) Can the satellite’s instrument be pointed such that the location is within view?

3) Is the time at which the viewing is possible within the request’s time window?

4) Is the length of the feasible observation at least as long as the minimum required 
observation duration?

For planner \( p \)’s execution phase associated with period \( a \), we propagate each satellite’s orbit 
over the entire execution phase, and check physical feasibility at each time step using J2 
perturbation theory. If the vectors \( s(t) \) and \( v(t) \) are the satellite’s Earth Centered Inertial (ECI) 
position and velocity vectors at time \( t \), and the vector \( r(t) \) is the target location’s ECI position 
vecrors at time \( t \) (easily converted from latitudes, longitudes, and altitudes), then we compute 
the line of sight (LOS) vector, \( \rho_{ru}(t) \), from satellite \( s \) to target \( r \) as follows:

\[
\rho_{ru}(t) = r(t) - s(t)
\]

\[ (4-19) \]

\[ ^2 \text{The satellite feasibility calculations are based on the methods described in [33].} \]

100
The LOS from satellite to target, using the law of cosines, is unobstructed if the inequalities (4-20) and (4-21) hold:

\[
\frac{\rho_{ru}(t) \cdot s(t)}{\|\rho_{ru}(t)\| \|s(t)\|} \leq 0 \quad (4-20)
\]

\[
\frac{\rho_{ru}(t) \cdot r(t)}{\|\rho_{ru}(t)\| \|r(t)\|} \leq 0 \quad (4-21)
\]

If inequalities (4-20) and (4-21) hold, then we have satisfied the first condition for satellite-to-target feasibility. Next, we must determine if the sensor we are considering can be slewed so that the target lies in its field of view (FOV). To do this, we first determine the roll and pitch angles, \( \varphi(t) \) and \( \theta(t) \) respectively, at which the sensor must be slewed to place the target at the center of its FOV. Orienting the sensor can be thought of as rolling/pitching the spacecraft or actually slewing the sensor, so we refer to \( \varphi(t) \) and \( \theta(t) \) as roll and pitch angles, respectively.

Let \( \hat{h} = \frac{h}{\|h\|} \) be the unit vector of some vector \( h \). We first create the satellite-to-ECI transformation matrix, \( T(t) \), as follows:

\[
T(t) = [\hat{x} \hat{y} \hat{z}] \quad (4-22)
\]

where

\[
x = \hat{y} \times \hat{z} \\
y = -(\nu(t) \times \hat{z}) \\
z = -s(t)
\]

The ECI-to-satellite transformation matrix, \( \hat{T}(t) \), is

\[
\hat{T}(t) = T^T(t) = \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{bmatrix} \quad (4-23)
\]

We note that we may use \( T^T(t) \) because \( T(t) \) is an orthogonal matrix; that is, its columns are orthogonal unit vectors, so \( T^{-1}(t) = T^T(t) \). We find the unit LOS vector in the satellite’s coordinate frame, \( \hat{\rho}' \), as

\[
\hat{\rho}'_{ru}(t) = \hat{T}(t)\hat{\rho}^T
\]

We obtain the roll and pitch angles, \( \varphi(t) \) and \( \theta(t) \), by computing the right-hand-side of the following equations:

\[
\varphi(t) = \sin^{-1}(-\hat{y} \cdot \hat{\rho}^T) \quad (4-25)
\]

\[
\theta(t) = \tan^{-1}\left(\frac{\hat{x} \cdot \hat{\rho}^T}{\hat{z} \cdot \hat{\rho}^T}\right) \quad (4-26)
\]
The equations (4-25) and (4-26) use projections of the ECI unit LOS vector, \( \hat{\rho}_{ru}(t) \), in the satellite’s coordinate frame to compute the angles at which the sensor must be gimbaled to place the target at the center of its FOV.

Given the angles \( \varphi(t) \) and \( \theta(t) \), the target remains somewhere in the sensor’s FOV if the range of angles for which the target is observable intersects with the range of feasible slewing angles for the sensor. This can be written as follows:

\[
\left[ \varphi(t) - \frac{FOV_x}{2}, \varphi(t) + \frac{FOV_x}{2} \right] \cap \left[ -MaxSlewX_s, MaxSlewX_s \right] \neq \emptyset
\]  
(4-27)

\[
\left[ \theta(t) - \frac{FOV_y}{2}, \theta(t) + \frac{FOV_y}{2} \right] \cap \left[ -MaxSlewY_s, MaxSlewY_s \right] \neq \emptyset
\]  
(4-28)

If conditions (4-27) and (4-28) hold, then the second condition is also met.

The third condition is that the potential observation times lie in the time window of the associated request. For satellites, then, the parameter \( o_{rs}^e \) for a satellite-based sensor \( s \) capable of viewing target \( r \) during execution phase \( e \) is 1 if conditions (4-20), (4-21), (4-27), and (4-28) hold, and this observation possibility is temporally feasible. That is, for each \( t \) that meets the four conditions listed, these opportunities are only feasible if

\[
ETW_r \leq t \leq LTW_r
\]  
(4-29)

The fourth condition is that any consecutive set of feasible opportunities last at least as long as the required duration for the observation. That is, for a consecutive set of feasible opportunities with \( n \) discrete feasible instances,

\[
n \cdot \Delta t \geq Dur_r
\]  
(4-30)

is the observation duration requirement. These four conditions represent only one possible observation, however. Thus, the parameter \( o_{rs}^e \) is defined as the number of disjoint opportunities (in time) that an asset has to observe request \( r \) in execution phase \( e \).

4.3.3.3.1.2 UAV Feasibility Calculations

The computation for a UAVs access to a potential target is simpler. Because the coordination planner does not have knowledge of existing UAV routes, we use the UAVs travel distance to the exact latitude and longitude of the target. That is, we assume the UAV will view the target at nadir. We could compute each UAVs minimum travel distance to the target such that the target is in its FOV, but without knowledge of the UAVs pre-existing routes we predict
little marginal benefit from this calculation. Thus, we calculate the great circle distance, \( M_{rl} \),
from UAV sub-planner \( p \)'s (\( p \in P^U \)) base \( l \) to target \( r \)'s location \((lat_r, lon_r)\) using the Haversine formula [64], which is very accurate even for small distances. If we let \( dlat = lat_r - BaseLat_p \) and \( dlon = lon_r - BaseLon_p \), where \( lat_r, lon_r, BaseLat_p \), and \( BaseLon_p \) are first converted to radians, then we calculate \( M_{rl} \) as follows:

\[
\begin{align*}
    a_{rl} &= \sin^2\left(\frac{dlat}{2}\right) + \cos(lat_r) \cos(BaseLat_p) \sin^2\left(\frac{dlon}{2}\right) \\
    c_{rl} &= \tan^{-1}\left(\frac{\sqrt{a_{rl}}}{\sqrt{1 - a_{rl}}}\right) \\
    M_{rl} &= r_E c_{rl}
\end{align*}
\] (4-31)

Equation (4-31) calculates \( a_{rl} \), the square of half the chord between the two locations of interest. Equation (4-32) computes the central angle, or spherical angle between the two points, in radians. To find the great circle distance between the two locations we simply multiply \( c_{rl} \) by the radius of the Earth, as shown in (4-33).

Using this distance \( M_{rl} \), we deem a sensor \( s \) on UAV \( u \) located at \( l \) on sub-planner \( p \) capable of viewing a target \( r \) if the UAV can travel to this location, observe the target, and travel back to its base, given its endurance, and if the UAV can travel to this location and observe the target before the target's late time window:

\[
2M_{rl} \leq (Endurance_{up} - Dur_r) * \text{MaxSpeed}_{up} \tag{4-34}
\]

\[
\frac{M_{rl}}{\text{MaxSpeed}_{up}} \cdot \text{StartTime}^e + Dur_r \leq LTW_r \tag{4-35}
\]

That is, \( o_{ru}^e = 1 \) if conditions (4-34) and (4-35) are met, and 0 otherwise, where \( \text{StartTime}^e \) is the start time of execution phase \( e \).

4.3.3.3.1.3 Total Feasible Opportunities

We sum the feasible opportunities over all satellites/UAVs on sub-planner \( p \) to obtain the number of feasible opportunities available in this system in execution phase \( e \):

\[
\delta_{rp}^e = \begin{cases} 
\sum_{u \in U_p} o_{ru}^e, & \text{if } p \in P^U, \\
\sum_{s \in S_p} o_{rs}^e, & \text{if } p \in P^S, 
\end{cases} \quad \forall \ e \in E^p 
\] (4-36)
We map the value $\tilde{a}_{rp}^e$ to each planning period according to the matrix B. Thus, $\tilde{a}_{rp}^a$ represents the number of feasible opportunities in the system for request $r$ if it is assigned to sub-planner $p$, given that the sub-planner is planning for the execution phase associated with period $a$.

In both the satellite and UAV cases, this component is scaled by dividing by the maximum number of feasible opportunities available in the system for the current planning period $a$, so that $\tilde{a}_{rp}^a \in [0, 1]$. That is,

$$\tilde{a}_{rp}^a = \frac{\tilde{a}_{rp}^a}{\max_{r \in R^a, p \in P} \tilde{a}_{rp}^a}$$  \hspace{1cm} (4-37)

The parameters $\tilde{a}_{rp}^e$ and $\tilde{a}_{rp}^a$ will be useful for the calculations for the Remaining Feasible Opportunities (RFO) component.

We use this component in two ways: first, it is used at the current iteration to screen all requests for physical feasibility. If $\tilde{a}_{rp}^a = 0$, then this request to sub-planner pair has no value to the coordinator for querying and the values of the other value function components are not used for computing the value. If, however, this component has non-zero value, then we compute the value for this request-to-sub-planner pair using $\tilde{a}_{rp}^a$ and the other value function components.

4.3.3.3.2 Measure of Feasibility

We use the scaled distance component, $\tilde{d}_{rp}^a$, to measure “how feasible” it is for an asset to view a target. This component is included in an effort to improve the coordinator’s ability to intelligently use its queries. Mathematically, the component $\tilde{d}_{rp}^a$ can be written as

$$\tilde{d}_{rp}^a = \begin{cases} 1 - (G_{rp}^a)^2, & \forall p \in P^S \\ 1 - (M_{rp}^a)^2, & \forall p \in P^U \end{cases}$$  \hspace{1cm} (4-38)

We square the gimbal angle and UAV travel distance terms to ensure the coordinator grants even more value to “more feasible” targets than to those that are very far from nadir/base. The methodology for computing this component depends on the type of sub-planner we are considering, as described in the following subsections.

4.3.3.3.2.1 Satellite Distance Component Calculations

If we are considering a satellite sub-planner $p \in P^S$, then the measure of infeasibility for viewing request $r$ with sensor $s$ in execution phase $e$ is a function of the absolute value of the gimbal angle required for the sensor to place the request location at the center of its FOV. We
use an average of the pitch and roll angles required over all sensors available on the sub-
planner that can view the target. If we let \( V_{rp}^e \) be the set of sensors on planner \( p \) that can possibly view target \( r \) during execution phase \( e \), then

\[
G_{Ars}^e = \frac{\varphi_{rs}^e(t) + |\theta_{rs}^e(t)|}{2}, \forall \ s \in V_{rp}^e \tag{4-39}
\]

\[
\bar{G}_{Ars}^e = \frac{\sum_{s \in V_{rp}^e} G_{Ars}^e}{|V_{rp}^e|} \tag{4-40}
\]

Equation (4-39) computes the mean gimbal angle required to view target \( r \) with sensor \( s \) on sub-
planner \( p \) in execution phase \( e \). Equation (4-40) averages these mean gimbal angles over all the
sensors on sub-planner \( p \) that can possibly view target \( r \) during execution phase \( e \). We scale the
average gimbal angle by dividing it by the absolute value of the maximum average gimbal
angle possible across all sensors in the system:

\[
\bar{G}_{Ars}^e = \frac{\bar{G}_{Ars}^e}{\max_{s \in S} \{\text{maxSlewXs} + \text{maxSlewYs}\}} \tag{4-41}
\]

Again, to map execution phases to planning periods, we utilize the matrix \( B \). Because smaller
gimbal angles imply an increased likelihood of feasibility, we subtract this scaled value from 1
so that targets that require no slewing (that can be viewed at nadir) are given the highest value.

**4.3.3.3.2.2 UAV Distance Component Calculations**

In the UAV case, we average the distances that each UAV on sub-planner \( p \) must travel
to view the target (at nadir). Again, if we let \( V_{rp}^e \) represent the set of UAVs on sub-planner \( p \) that
can possibly view target \( r \) during execution phase \( e \), then we compute the average travel
distance required, \( \bar{M}_{rp}^a \), as follows:

\[
\bar{M}_{rp}^a = \frac{\sum_{u \in V_{rp}^e} M_{ru}^a}{|V_{rp}^e|} \tag{4-42}
\]

where \( l_u \) is the base location of UAV \( u \). The distance component for a request-to-UAV sub-
planner pair is scaled by dividing \( \bar{M}_{rp}^a \) by the maximum distance a UAV in the system can travel
before having to return to base.

\[
\bar{M}_{rp}^a = \frac{2\bar{M}_{rp}^a}{\max_{u \in U_p, p \in PU} \text{MaxSpeed}_u \cdot \text{Endurance}_u} \tag{4-43}
\]
4.3.3.3 Remaining Feasible Opportunities

This component is adapted from [42] and makes the value function forward-looking. It encourages the coordination planner to value those requests with fewer feasible opportunities remaining more than those with many opportunities remaining. For each execution phase, we compute, \textit{a priori}, the number of feasible opportunities for each target over the planning horizon. Here, we must distinguish between planning periods and execution phases because RFO's are dependent on execution phases only. That is, a change in planning period does not necessarily correspond to a change in the number of RFO's for any request. We denote $RFO_{rp}^a$ as the number of feasible opportunities remaining for request $r$ on sub-planner $p$ strictly after planning period $a$. We define $RFO_{c}^a$ as the number of remaining feasible opportunities for observing a location associated with request $r$ across all sub-planners after planning period $a$. We use the parameters $\delta_{rp}^c$, defined in Section 4.2.3.3.1.3, to efficiently compute the RFO component. In any planning period, a unique set of execution phases are being planned for, so after each period $a$, the coordinator loses one execution period of feasible opportunities. Thus,

$$RFO_{rp}^a = \sum_{e=(bpa+1)}^{\left|E^p\right|} o_{rp}^e$$

$$RFO_c^a = \sum_{p \in P} RFO_{rp}^a$$

(4-44)

(4-45)

for the planning periods associated with final execution phases, $RFO_{rp}^a = 0$.

We use the maximum number of feasible opportunities for any single request in planning period $a$ as part of the scaling factor for the RFO component. We scale this parameter non-linearly in order to obtain more value for a request that has few feasible opportunities remaining after the current set of communication iterations with sub-planners is complete. Similarly, this means we give little value to the request that has the most remaining opportunities for observation after this planning period. The scaling method is as follows:

$$\overline{RFO}_p^a = 1 - \left(\frac{RFO_{r}^a}{\max\{RFO_{r}^a\}}\right)^2$$

(4-46)

Thus, the scaling factor changes with each planning period. This is necessary to account for the fact that viewing opportunities are not necessarily evenly spaced over the planning horizon.
4.3.3.4 Simultaneous Viewings

This portion of the value function grants additional value to requests that are part of a dual collect. We include this component for two reasons: first, we assume that dual collects are extremely valuable to their users. Second, we acknowledge the difficulty of obtaining these dual collects, particularly in a distributed framework. Thus, we give each target $r \in R$ additional value if $\text{Dual}_r = 1$.

Moreover, if we calculate that both requests in a dual collect are feasible at a particular location in the upcoming execution phases, the associated requests receive a still higher value. These data can be expressed in terms of $V_r^a$, the set of assets on sub-planner $p$ that can observe request $r$ in planning period $a$. Specifically,

$$\text{simult}_r^a = \begin{cases} 
0.5, & \text{if } \sum_{p \in P} |V_r^a| = 1, r \in D' \\
1, & \text{if } \sum_{p \in P} |V_r^a| > 1, r \in D' \\
0, & \text{otherwise}
\end{cases} \quad (4-47)$$

Thus, this component takes on value 0.5 if target $r$ is part of a dual collect and is potentially visible on a single asset in the execution phase associated with planning period $a$ but no dual collects are possible. The component takes on value 1 if the target is part of a dual collect and can potentially be viewed simultaneously (or nearly simultaneously) by different sensors during period $a$. We allocate value to a request associated with a dual collect even if a dual collect does not appear possible in the upcoming execution phases because it is possible that opportunities are available in future planning periods, especially if the associated time windows are very long.

4.3.3.5 Time Windows

The value function includes a component related to the size of the time windows of the requests, denoted $TW_r$. In a system inhibited by communication constraints, it becomes especially important to consider the usefulness of each query. In that line, the time window component discourages the coordination planner from including requests that have a small chance of being accepted by the planners in its queries.

We create this component based on the idea that requests with very small time windows constrain the solution space of a routing problem more than requests with large time windows. This is especially true if the required duration for each observation is nearly equal. This is clear
when we consider the extreme cases; if we lengthen the time window of a request to the length of the planning horizon, we have made the inclusion of that request in an operations plan much easier, but if we shorten the time window to a very small but nonzero value, it becomes far more difficult to insert into existing routes.

While this component will discount the value of some high-priority requests (those with narrow time windows), it will do so to all requests with small time windows. The other value function components exist so that the coordination planner can balance its multiple objectives, so we do not believe this component will hurt the coordinator’s ability to have high-priority requests serviced. Instead, the time window component exists to improve the efficiency of the coordinator’s planning algorithms.

To compute this component’s contribution to the value function, we determine the length of each request’s time window, and scale it by the size of the planning horizon itself, as shown in Equation (4-48):

$$TW_r = \frac{LTW_r - ETW_r}{H}$$

Thus, the time window component is dependent only on the request being considered. It is independent of each sub-planner and the time at which it is being computed.

4.3.3.6 Summary

Given that constructing the value function requires many calculations, we provide this section to summarize this process. Table 4-8 explains the purpose for including each component and the appropriate interpretation of the scaling method used for that component:
<table>
<thead>
<tr>
<th>Component</th>
<th>Purpose of Inclusion</th>
<th>Scaling</th>
</tr>
</thead>
</table>
| $\bar{p}_r$ | To encourage the planner to query high-priority requests more frequently | $\bar{p}_r = 0 \rightarrow$ Corresponds to a request with no value  
$\bar{p}_r = 1 \rightarrow$ Corresponds to a request of highest value |
| $\ast TSV^a_{rp}$ | To encourage the planner to query sub-planners that could provide the best quality observation possible | $TSV^a_{rp} = 0 \rightarrow$ Corresponds to a pairing with no benefit for that target type  
$TSV^a_{rp} = 1 \rightarrow$ Corresponds to a pairing with the best quality possible for that target type |
| $\ast \bar{d}^a_{rp}$ | To incorporate physical constraints into the optimization problem without formulating the constraints explicitly | $\bar{d}^a_{rp} = 0 \rightarrow$ Corresponds to a request with no feasible opportunities on this planner in the next execution phase  
$\bar{d}^a_{rp} = 1 \rightarrow$ Corresponds to a request with feasible opportunities on every asset in the upcoming execution phase |
| $\bar{d}^a_{rp}$ | To encourage the planner to query requests on sub-planners that are most likely to service them | $\bar{d}^a_{rp} = 0 \rightarrow$ Corresponds to a request whose viewing requires  
- the maximum gimbal angle possible  
OR  
- the maximum flight distance possible  

$\bar{d}^a_{rp} = 1 \rightarrow$ Corresponds to a request that can be viewed  
- at nadir by a satellite sensor  
OR  
- at the UAVs base |
| $\bar{RFO}^a_r$ | To efficiently utilize query opportunities over the length of the planning horizon | $\bar{RFO}^a_r = 0 \rightarrow$ Corresponds to a request with the most feasible opportunities remaining of all requests  
$\bar{RFO}^a_r = 1 \rightarrow$ Corresponds to a request with no opportunities remaining after this period |
| $\bar{simult}^a_r$ | To improve the planner's ability to coordinate the taking of dual collects | $\bar{simult}^a_r = 0 \rightarrow$ Corresponds to a request that requires a single collection  
$\bar{simult}^a_r = 1 \rightarrow$ Corresponds to a request that requires dual collection, and there exists a feasible opportunity for dual collection at this time |
| $\bar{TW}_r$ | To improve the efficiency with which the coordinator uses queries | $\bar{TW}_r = 0 \rightarrow$ Corresponds to a request with a time window that is essentially zero minutes long  
$\bar{TW}_r = 1 \rightarrow$ Corresponds to a request without a time window (feasible during the entire planning horizon) |

Table 4-8: Value Function Summary

*Signifies parameters used to screen for potential pairings. If either of these parameters have value equal to 0, then the pairing has no value.

### 4.4 Assignment Problem Construction

In this section we present the IP formulation of the assignment problem we wish to solve. We use the value function established in Section 4.3.3 and solve a variation of a static assignment problem with side constraints.
### 4.4.1 Static Formulation

In Section 4.1.5, we discussed the classical assignment problem. Here, we present the modified version of the assignment problem we solve at each iteration over the course of the planning horizon.

The objective function we use is a sum of the values for all chosen request/sub-planner pairings. The individual values are obtained from the value function described in the previous section. We modify the constraints described in the classical assignment problem to allow each project (or sub-planner) to potentially have more than one person (request) assigned to it, and we do not force each person (request) to be scheduled on a project (sub-planner). We present the following *Binary Integer Assignment Formulation* (BIAF) for the problem of intelligently assigning requests to sub-planners for querying, at iteration $k$ and planning period $a$:

**BIAF:**

Maximize

$$\max \sum_{r=1}^{n(a)} \sum_{p=1}^{m} \bar{v}_{rp} x_{rp}$$

Subject to

$$\sum_{r=1}^{n(a)} x_{rp} \leq \theta_p, \quad \forall p \in P$$

$$(1 - S_{(r_1 r_2)p}^a)(x_{r_1 p} + x_{r_2 p}) \leq 1, \quad \forall (r_1, r_2) \in D, p \in P$$

$$x_{rp} \in \{0, 1\}, \quad \forall r \in \bar{R}_{ka}, p \in P$$

We introduce the decision variables $x_{rp}$, where each variable takes value 1 if we assign request $r$ to sub-planner $p$ at iteration $k$ in planning period $a$, and takes value 0 otherwise. Note that the superscripts corresponding to the iteration numbers and planning periods are necessary because we use the values of past variables for updating the future values. However, we solve for only $|\bar{R}_{ka}| \cdot |P|$ variables at any one time. The objective (4-49) states that we seek to maximize the value of the assignments of requests to sub-planners. Constraints (4-50) are communication constraints on the sub-planners; namely, sub-planner $p$ can receive no more than $\theta_p$ requests as queries at any iteration. Constraints (4-51) ensure that related dual collects should not be sent to the same sub-planner if it provides no benefit. These constraints are included in order to prevent the coordinator from sending the same locations to the same sub-planner when a dual collect is infeasible. The parameter $S_{(r_1 r_2)p}^a$ takes value 1 if a dual collect (defined by requests $r_1$ and $r_2$) is feasible on sub-planner $p$ in period $a$, and 0 otherwise. Thus, the constraints...
are only imposed if the dual collect is infeasible. Constraints (4-52) enforce binary integer constraints on the decision variables $x_{r_{1}p}^{ka}$.

As discussed in Section 4.1.5, the constraint matrices of classical assignment problems are Totally Unimodular (TU), which allows the problems to be solved to optimality, with integer solutions, by relaxing the binary integer constraints. The total unimodularity property is a desirable one, as it allows for low computation time for an optimization problem. In Appendix D, we prove that our constraint matrix is TU, so that we can relax the binary integer constraints and still find integer optimal solutions to our optimization problems. Thus, we solve the Linear Assignment Formulation (LAF) at each communication iteration:

\[
\text{LAF:} \quad \max \sum_{r=1}^{|\bar{R}^{ka}|} \sum_{p=1}^{|P|} \overline{v}_{r_{1}p} x_{r_{1}p}^{ka} \quad \text{subject to} \quad \sum_{r=1}^{|\bar{R}^{ka}|} x_{r_{1}p}^{ka} \leq \theta_{p}, \quad \forall \ p \in P
\]  
\[
(1 - s_{(r_{1}, r_{2})p}) (x_{r_{2}p}^{ka} + x_{r_{2}p}^{ka}) \leq 1, \quad \forall \ (r_{1}, r_{2}) \in D, p \in P
\]  
\[
0 \leq x_{r_{1}p}^{ka} \leq 1, \quad \forall \ r \in \bar{R}^{ka}, p \in P
\]

We relax the equality constraints that are usually present in an assignment problem for two reasons; first, we do not wish to overload sub-planners with unnecessary queries. Second, there is no requirement that each request be queried on any sub-planner. It would not be sensible to query sub-planners with requests even if no corresponding feasible opportunity exists, or if the request has been rejected by this sub-planner before. The LAF can be thought of as a version of an assignment problem for two reasons: first, the problem is, in its nature, bipartite (a request set, and a sub-planner set). Second, we could create dummy nodes for the \( \theta_{p} \) query opportunities for each sub-planner \( p \). That is, we could create a graph with \( \sum_{p \in P} \theta_{p} \) "project", or sub-planner, nodes, and formulate the problem as an asymmetric assignment problem ("asymmetric" assuming \( |\bar{R}^{ka}| \neq |\sum_{p \in P} \theta_{p}| \)) where each sub-planner node could receive no more than one request. Then, we could add additional dummy nodes to equate the cardinality of the two sets of nodes. This would allow us to solve a classical assignment problem with the side constraints (4-56) (i.e., with equality constraints and right-hand-sides equal to 1). However, for this thesis, we solve the problem using CPLEX without concern for 111
solving a classical assignment problem. We do this because the computation time for any realistically-sized problem is not long.

We can better understand this optimization problem by visualizing it. At any iteration $k$ of any planning period $a$, we solve the problem of assigning requests to sub-planners such that the value of the assignments is maximized. This is depicted in Figure 4-4:

![Diagram](image)

For a single iteration $k$ in planning period $a$, each arc $(r,p)$ has profit $\phi_{rp}$ and flow must take value 0 or 1.

Figure 4-4: Assignments at a Single Iteration

We then receive feedback from the sub-planners, incorporate this feedback into the value function, and solve the optimization problem again. Figure 4-5 depicts an example of this process and includes three different assignment problems:
In this figure, the first iteration \((k')\) in planning period \(a'\) contains five queries, one of which returns feedback indicating the request can be serviced by the sub-planner. The second iteration yields one query with positive feedback out of two total queries. After this iteration, the system enters a new planning period \((a'+1)\) because one sub-planner beings planning for a new execution phase. The first iteration in this new planning period (the third iteration so far) results in one accepted request. Thus, a total of three of the four requests were successfully serviced in this example.

The problem of how to interpret feedback from sub-planners is the subject of the following two sections.

### 4.4.2 Dynamic Value Updating

It is important for the coordination planner to “remember” which requests were accepted/rejected on which planners in order to efficiently query the sub-planners over time.
This section describes how we dynamically update the value function after each iteration to incorporate past feedback into future queries.

In each planning period $a$, a unique set of execution phases is considered. For the sub-planner whose execution phase has just been changed (only one sub-planner at a time experiences this change in the asynchronous case), we compute each request-to-sub-planner pair value using the full value function described in Section 4.3.3. Upon receiving feedback on this iteration, however, the value should change depending on the nature of the feedback. If we let $f_{b_{rp}}$ store data concerning the result of querying request $r$ on sub-planner $p$ at iteration $k$ of planning period $a$, then we define

$$f_{b_{rp}}^{ka} = \begin{cases} -1, & \text{if accepted but new routes were not an improvement} \\ 0, & \text{if rejected, (or never allocated)} \\ 1, & \text{if accepted and new routes were an improvement} \end{cases}$$ (4-58)

Note that an “improvement” will be defined later in Section 4.4.3. Given this definition, we note that the values computed at each iteration, $\tilde{v}_{rp}^{ka}$, can be described in the following general forms:

$$\tilde{v}_{rp}^{ka} = 0, \quad (4-59)$$

We base each update decision on a natural interpretation of the feedback the coordinator receives. For each possible outcome, we summarize the coordination planner’s actions, and the reasons for those actions in the following table:

<table>
<thead>
<tr>
<th>Feedback Data</th>
<th>Action(s)</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_{rp}^{(k-1)a} = 0$ and $\exists x_{rp'}^{(k-1)a} = 1$ for which $f_{b_{rp}}^{ka} = 0$</td>
<td>$\tilde{v}<em>{rp}^{ka} = \tilde{v}</em>{rp}^{(k-1)a}$</td>
<td>By default, make no change.</td>
</tr>
<tr>
<td>$x_{rp}^{(k-1)a} = 1$ and $f_{b_{rp}}^{ka} = 0$</td>
<td>$\tilde{v}_{rp}^{ka} = 0$</td>
<td>If the request was rejected by the sub-planner we reduce its value to zero because this pair no longer has value to the coordinator.</td>
</tr>
<tr>
<td>$x_{rp}^{(k-1)a} = 1$ and $f_{b_{rp}}^{ka} = 0$</td>
<td>$\tilde{v}_{rp}^{ka} = 0, (r, r') \in D$</td>
<td>If the request was rejected by the sub-planner, all requests related to it (with the same physical characteristics) can be assumed to have been rejected as well.</td>
</tr>
</tbody>
</table>
If the coordinator receives confirmation from a sub-planner that a request will be accepted, and its acceptance results in a system-wide allocation improvement, the request is sent to the sub-planner at this point and eliminated from the request set.

If a request \( r \) that was not allocated to sub-planner \( p \) is rejected on sub-planner \( p' \), then it becomes more important to attempt having the request serviced on other feasible sub-planners and so we increase the value for those assignments. This action is only taken if the pair had value last iteration (i.e. a feasible opportunity exists). \( \delta \) is the amount by which we increase \( \bar{v}_{rp} \), and lies between 0 and 1.

If the request was accepted by the sub-planner but resulted in a system-wide allocation with less value than the previous allocation, we do not want to eliminate the request's value altogether, but it should have lesser value to the coordinator. \( \delta \) is the amount by which we decrease \( \bar{v}_{rp} \), and lies between 0 and 1.

Table 4-9: Dynamic Value Update Decisions

An additional value update relates to the coordinator's ability to complete dual collections. Because sub-planner planning cycles are asynchronous and we assume sub-planners accept requests for only one execution phase at a time, it is important to consider which portions of dual collects have been accepted by sub-planners in past iterations. We increase the value \( \bar{v}_{rp} \) of the remaining unscheduled portion of a dual collect to ensure it is sent as a query to the sub-planners. The amount by which we increase \( \bar{v}_{rp} \) is dependent on the input weight vector, \( w \). The subject of measuring system-wide improvements is the subject of Section 4.4.3.

4.4.3 Plan Comparisons

As discussed above, queries can affect the system-wide value of allocations. In some cases, sub-planners agree to execute new requests they are given by replacing old coordinator requests that were accepted at previous iterations. This represents the idea that some requests will be unexpectedly "bumped" from existing observation plans and will thus no longer be serviced. To deal with this possibility, the coordination planner must check for conflicts of current plans with existing plans. In this section, we describe the simple heuristics implemented to address this issue.
First, we must decide how to evaluate and choose between two different plans. Two options are available:

1) Evaluate plans in terms of the total value $\bar{\psi}_{rp}^{k}$, at the time they were last queried, for all requests that have been scheduled in each plan.

2) Evaluate plans in terms of individual MOPs. This involves choosing which MOPs to value most and how to value them, based on user weighting inputs.

We have implemented both of these plan evaluation methods. For the latter case, we evaluate plans based on user input weights. Specifically, we use the following logic to decide how we choose the MOP to emphasize:

<table>
<thead>
<tr>
<th>Maximum Weight</th>
<th>MOP Emphasized</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_p$</td>
<td>avgPriority</td>
</tr>
<tr>
<td>$w_{TS}$</td>
<td>avgTSV</td>
</tr>
<tr>
<td>$w_D$</td>
<td>numRequests</td>
</tr>
<tr>
<td>$w_S$</td>
<td>numDualCollects</td>
</tr>
</tbody>
</table>

Table 4-10: Pairing Input Weights with MOPs

Next, we must consider whether to evaluate plans locally (that is, for each sub-planner) or globally (system-wide). Comparing plans between two iterations at a local level could allow for improved use of individual sub-planners, but we hypothesize that considering the global, or system-wide, value of plans should yield better results across all sub-planners over the course of a planning horizon. Thus, we choose to evaluate plans at a global level.

If we let $systemValue^k$ be the system-wide value of plans after iteration $k$, and we let $totalV_p^k$ be the value of the plans created by sub-planner $p$ after iteration $k$, then we decide that the decisions made at iteration $k+1$ provide an improvement in value if

$$systemValue^{k+1} = \sum_{p\in P} totalV_p^{k+1} \geq systemValue^k = \sum_{p\in P} totalV_p^k$$

(4-61)

The "value" of the plans is determined by which of the valuation methods we use, as discussed above. If this condition holds, then we accept the current set of assignments and update the values $\bar{\psi}_{rp}^{(k+2)a}$ $\forall r, p$ accordingly.

116
4.5 Implementation of Iterative Assignment Problem Formulation

This section describes our implementation of the above algorithms in software. We first describe the software architecture and algorithmic flow of the coordination planner. Next, we discuss the format of the input data and show sample inputs. Then, we describe the development of each test. Finally, we show output from a sample run of the coordination planner software.

4.5.1 Software Architecture

To test our algorithm’s ability to generate efficient observation plans, we implement the algorithm in software that is integrated with the two types of sub-planners. We generate the data for the problem and control the software using MATLAB. To solve the assignment problems, we call ILOG’s OPL Studio, an optimization modeling system which uses the optimization software CPLEX 11.1. The satellite sub-planner MIP is coded in OPL Studio as well, and is solved using CPLEX 11.1, while the UAV sub-planner is implemented in Java and calls CPLEX 11.0 to solve a composite variable linear program. Figure 4-6 shows the software architecture constructed to implement the coordination algorithms:
Additionally, we present a more detailed diagram in Figure 4-7 demonstrating the algorithmic flow of the coordination planner, to give the reader an idea of how these algorithms work within a relatively complex software system with extensive communication between planning systems. Table 4-11 refers the reader back to the descriptions of individual functions indicated in the diagram.
Figure 4-7: Algorithmic Flow

<table>
<thead>
<tr>
<th>Function</th>
<th>Described in section(s):</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generate value function parameters</td>
<td>4.2</td>
</tr>
<tr>
<td>Target-sensor compatibility lookup</td>
<td>4.2.3.2</td>
</tr>
<tr>
<td>Opportunity finder</td>
<td>4.2.3.3 - 4.2.3.4</td>
</tr>
<tr>
<td>Compute/update values</td>
<td>4.2.3, 4.3.2</td>
</tr>
<tr>
<td>Optimization problem</td>
<td>4.3.1</td>
</tr>
<tr>
<td>Randomized value increases*</td>
<td>5.4</td>
</tr>
<tr>
<td>Plan comparisons, conflict check, improvement?</td>
<td>4.3.3</td>
</tr>
</tbody>
</table>

Table 4-11: Algorithmic Functions by Section

*Discussed in Chapter 5, Results and Analysis, as an additional function
4.5.2 Sample Input Data

The data for requests being considered by the coordination planner are input in an XML file and read by MATLAB. Figure 4-8 shows a sample input file of request data for two requests, for a test set with 40 requests over a horizon of 4 hours. The file contains the data from Table 4-2:

```
<?xml version="1.0" encoding="utf-8"?>
<Requests40Horizon4Set3>
  <Request>
    <Index>1</Index>
    <Name>CRequest1</Name>
    <Type(ff)</Type>
    <Priority>300</Priority>
    <Long>-115.677</Long>
    <Lat>44.695</Lat>
    <Alt>0</Alt>
    <ETW>2.5</ETW>
    <LTW>3.9</LTW>
    <Dur>0.028</Dur>
    <Simult>0</Simult>
    <RelatedTo>0</RelatedTo>
  </Request>
  <Request>
    <Index>2</Index>
    <Name>CRequest2</Name>
    <Type(ff)</Type>
    <Priority>600</Priority>
    <Long>-113.802</Long>
    <Lat>39.045</Lat>
    <Alt>126</Alt>
    <ETW>2.2</ETW>
    <LTW>2.8</LTW>
    <Dur>0.037</Dur>
    <Simult>1</Simult>
    <RelatedTo>3</RelatedTo>
  </Request>
</Requests40Horizon4Set3>
```

Figure 4-8: Sample Request Data

Sub-planner data are input in a text file and also read in by MATLAB. The sub-planner data file corresponding to the request data in Figure 4-8 is shown below:
Note that the Kepler elements are not explicitly input into the coordination planner. Instead, the input to the coordination planner for individual satellite assets is in a text file that contains all of the data on the asset’s orbital elements and their download times (from www.spacetrack.org). These text files are identified in the sub-planner data input files. System data are input to the coordination planner software as arguments, and sensor data are stored in a master text file that is read in separately from request/sub-planner inputs. The weights \( w \) are read in from a text file as well.

### 4.5.3 Test Set Development

Each run of the coordination planner requires a significant amount of data. To automate the process of generating test sets, we developed a program in MATLAB to randomly generate request data.

Some data were chosen manually based on realistic operational values, such as UAV speeds, endurances, locations, and operational floors/ceilings. Satellite data were extracted from www.spacetrack.org and thus also represent realistic data. We usually choose the start time of the scenario, planning cycle data, \( \theta_p \), and the number of external requests on each sub-planner manually as well.

121
Random number generators are used to generate the request data for requests internal to the coordination planner and those internal to each sub-planner. The request priorities are chosen from a discrete uniform distribution, while the latitudes, longitudes, altitudes, time windows, and required durations are chosen from continuous uniform distributions with reasonable lower and upper bounds. We generate required observation durations after generating time windows to ensure the required durations are no longer than the time windows. Finally, we generate the priorities for requests associated with dual collects randomly but from uniform distributions with higher lower bounds than for a single request.

4.5.4 Output

The final output of the program provides the assignments of requests to assets and the planned start and end times of each observation. The final output also notes the types of sensors that were used for the observations. Each of the four primary MOPs are reported when the program is complete. The figure below shows example output from a run of the coordination planner software:

![Sample Output](image-url)
Intermediate output includes which requests were allocated as queries to which sub-planners at each iteration. The program also outputs the feedback on request acceptance/rejection from each interaction with the sub-planners. We have also created KML output that can be displayed in Google Earth. This output provides routes for each execution phase of each asset in the system.
5 Results and Analysis

In this section, we present the results of coordinated observation planning using the methodologies from Chapter 4 on various test scenarios. The first section describes analysis conducted on the value function to ensure it performs as we intend, including empirical and statistical analysis. The second section demonstrates the use of our statistical analysis in conjunction with non-linear optimization to enable a user of the coordination planner to choose input weights to meet his objectives. The third section conducts analysis on the impact of additional communication with sub-planners. The fourth section shows the benefit of querying those requests that were previously rejected on sub-planners multiple times. The fifth section compares the performance of a coordination planner versus three notional baseline scenarios. The sixth section demonstrates how we can analyze the impact of varying orbital parameters in the context of the CLARREO scenario. The seventh section emphasizes dual collections. The eighth and final section presents the results of coordinating amongst air and space assets in a simulated Wildfire Research and Applications Partnership (WRAP) scenario. This section focuses on the benefit of additional UAVs in the scenario.

5.1 Value Function Testing

We use the value function discussed in Chapter 4 as an intermediate construct to create observation plans that align with user objectives. In this section, we show that the value function can be used to do exactly that through empirical and statistical analysis.
5.1.1 Empirical Analysis

In this section, we show that the value function can be tuned to align with user objectives by direct experimentation on a dataset. In particular, we show how the \( \text{avgPriority} \), \( \text{avgTSV} \), \( \text{numRequests} \), and \( \text{numDualCollects} \) Measures of Performance (MOPs) can each be emphasized by varying the input weight vector \( w \).

For each of the following subsections, we use a test set with 1,000 requests, 10 sub-planners, and a 24 hour planning horizon. Specifically, there are 2 Unmanned Aerial Vehicles (UAVs), and 11 satellite-based sensors involved in the scenario.

5.1.1.1 Priority

Here, we show the effect of increasing the priority weight, \( w_p \), while keeping all other parameters at an equal level. Specifically, we hold all input weights at 0 except for the dual collection parameter, which is equal to \( 1 - w_p \). We average the output values over 3 runs for each of 10 values of \( w_p \).

![Figure 5-1: Varying the Priority Input Weight](image)

We note that, for this scenario, the curve depicting the average priority of the requests observed for each input value is increasing with diminishing marginal returns. When the priority of requests is not considered important \( (w_p = 0) \), the average priority of the requests observed is around 500, and when we consider the priority of requests to be the most important MOP, the metric increases to slightly over 700. This curve is not guaranteed to be monotonically
increasing, however, as we are still operating in a decentralized planning environment in which sub-planners are not entirely subservient to the coordinating agent.

5.1.1.2 Dual Collections

In this subsection, we demonstrate how the simultaneous collection input weight, \( w_s \), can be used to emphasize dual collections in final observation plans. We increase the dual collect input weight from 0 to 1, and record the number of dual collections obtained in the final observation plan. For this scenario, requests were submitted for dual collections at 125 distinct locations, meaning 250 of the 1000 total requests in the system were parts of dual collects.

![Figure 5-2: Varying the Dual Collection Input Weight](image)

For dual collections, we see a roughly linear increase in dual collections as we increase the corresponding input weight. Initially, only around 36 dual collections were made, and in the case where the dual collection component is the only value function component given positive weight, nearly 60 dual collections were made. The coordination planner cannot plan observations for all 125 dual collection requests because not all of the requests have feasible opportunities for dual collections and because of the request traffic (of external requests) on individual sub-planners.

5.1.1.3 Quality of Observations

Here, we analyze the effects of increasing the value function component relating to the quality of observations, \( w_{TSV} \). We hypothesize that increasing this parameter relative to the
other input weights should encourage the coordination planner to choose request-to-sub-planner pairings with opportunities higher quality observations. To test this hypothesis, we increase the observation quality input weight from 0 to 1, and record the total target-to-sensor value of all observations planned for. For this scenario, 1000 requests were in the system, with 10 sub-planners (13 sensors). To control for the number of feasible opportunities for each sensor type, we built the scenario such that the different sensor types were available on the same satellite asset.

This output indicates that the relationship between the $w_{TSV}$ input weight and the quality of observations is roughly linear, although the curve is certainly not monotonically increasing. This is likely data-driven; that is, 13 sensors is a relatively low number of sensors (although it is a very realistic problem size), and we only have data for 6 different types of sensors. We hypothesize that the results would be more pronounced in a longer scenario and if we had models of additional sensor types. Nonetheless, we do notice a positive trend between the $w_{TSV}$ input weight and the quality of the observations included in final plans.

5.1.2 Statistical Analysis

In this section, we discuss how we use statistical analysis to verify that the value function has the effects on final observation plans that it should. We use the value function as an intermediary construct whose objective function does not directly align with those of a user of the coordination planner. For example, the Remaining Feasible Opportunities (RFO)
component of the value function might not seem to directly correspond to the objective of maximizing the number of dual collects obtained. Thus, in order to ensure that the value function is having desired effects on user end objectives, we use empirical and statistical analysis, including three regression techniques, to evaluate its performance.

A well-formed regression model provides two things: an understanding of the relationship between the independent and dependent variables, and the ability to accurately predict the values of dependent variables given a vector of independent variables. The regression analysis in this section is motivated by the former, and Section 5.2 uses the latter.

This section tests the following hypotheses:

1) The priority weight is most positively correlated with the average priority of the observations in final plans output by the planner.

2) The Target-to-Sensor Value (TSV) weight is most positively correlated with the average target-to-sensor value of the observations in final plans output by the planner.

3) The observation weight is positively correlated with the number of targets observed and the number of dual collects obtained. However, its main contribution is that the value for each target-to-sub-planner pair is zero unless this component is nonzero, so an insignificant correlation would not be alarming.

4) The distance weight is positively correlated with the number of targets observed, and possibly with the number of dual collects made, because it is a measure of “how feasible” each request is on each planner (i.e. it is associated with physical constraints).

5) The RFO component is positively correlated with the number of targets obtained and the number of dual collects made. This is because the RFO component is calculated as the number of remaining physically feasible opportunities, and does not take into account the quality of those potential observations.

6) The simultaneous observation component is positively correlated with the number of dual collects made in a plan, as it only gives value to dual collects.

7) The time window component is positively correlated with the number of targets obtained and the number of dual collects made.

The first technique we use is standard linear regression, the second technique is stepwise regression, and the third technique is ridge regression. We motivate the use of each technique,
briefly discuss each methodology, and present results for a sample system in the following sections.

5.1.2.1 Notation

For this analysis, we use notation adapted from [65]. We denote an input, or independent variable, with $X$. If $X$ is a vector, we denote $X_j$ to access the $j^{th}$ element of $X$, and use $p$ to denote the number of independent variables. We use $Y$ to signify an output, or dependent variable (scalar or vector). An observed value of $X$ is denoted $x_i$, which can be a scalar or a vector. A set of $N$ input $p$-column vectors $x_i, i = 1..N$, is represented by the $N \times p$ matrix $X$. The $i^{th}$ row of $X$, then, is $x_i^T$. Given this notation, we can roughly define the task of learning as making a good prediction of the output $Y$, denoted $\hat{Y}$, given an input vector $X$. In the context of our problem, we seek a vector of input weights $X$ that optimize a vector $Y$ of MOPs.

5.1.2.2 Ordinary Least Squares (OLS) Linear Regression

Given an input vector $X$, Ordinary least squares (OLS) regression predicts the output $Y$ using the model

$$\hat{Y} = \hat{\beta}_0 + \sum_{j=1}^{p} X_j \beta_j + \epsilon = X^T \hat{\beta} + \epsilon$$

(5-1)

where $\hat{\beta}_0$ is the intercept of the model, which requires adding a column of 1's to $X$ if it is included, and $\epsilon$ is an $n$-vector error term. The error term captures measurement errors, omitted variable bias, and inherent randomness in realized data. It is treated as a random variable and must meet certain assumptions, which we briefly discuss below. To choose the vector $\hat{\beta}$, OLS regression solves the following optimization problem:

$$\min_{\hat{\beta}} \sum_{i=1}^{N} (y_i - x_i^T \hat{\beta})^2$$

(5-2)

Thus, the OLS model chooses a predictor that minimizes the residual sum of squares (RSS).

The OLS model makes many assumptions, primarily concerning the normality of the underlying data. It assumes a linear model with linearly independent columns, independent observations, and normally distributed error terms with a mean of 0.

5.1.2.3 Stepwise Regression

Stepwise regression is a method of subset selection. If too many variables are included in the model, collinearity and variance inflation can occur. These problems lead to incorrect signs
on the coefficients for some variables, and insignificant coefficients on variables that might in fact be significant. On the other hand, having too few variables in the model can cause bias and larger prediction intervals. Thus, it is important to intelligently choose a subset of the independent variables.

Forward stepwise regression begins with no independent variables in the model and adds one variable at a time. It checks the statistical significance of each model after adding an additional variable, and chooses a variable to include based on which variable improves the statistical significance of the model the most. The algorithm continues until adding another variable does not produce a model with statistical significance over some pre-determined threshold. Formally, the algorithm proceeds as follows:

1) Start with only the constant term in subset $S$

2) Compute the Residual Sum of Squares (RSS) by including each variable not in $S$

Let

$$F_i = \max_{i \not\in S} \frac{RSS(S) - RSS(S + \{i\})}{\hat{\sigma}^2(S + \{i\})}$$

(5-3)

If $F_i > F(\text{threshold})$, $S = S \cup \{i\}$, where $F(\text{threshold})$ is usually between 2 and 4.

Else, terminate algorithm.

3) Repeat Step 2.

Backward stepwise regression proceeds in a similar manner but instead of selecting new variables for the model, it eliminates variables that are already in the model. It begins with every independent variable in the model and eliminates one variable at a time if the elimination of that variable causes little change in the statistical significance of the model. We present an algorithm for backward stepwise regression below:

1) Start with all variables in $S$

2) Compute the increase in RSS by excluding each variable present in $S$

Let

$$F_i = \min_{i \in S} \frac{RSS(S - \{i\}) - RSS(S)}{\hat{\sigma}^2(S)}$$

(5-4)

If $F_i < F(\text{threshold})$, $S = S \setminus \{i\}$, where $F(\text{threshold})$ is usually between 2 and 4.

Else, terminate algorithm.

3) Repeat Step 2.
Forward stepwise regression usually chooses a smaller subset of independent variables in its models than backward stepwise regression. In both methods, it is rare that all independent variables are kept in the model. Here, while we are concerned with selecting significant coefficients, we also desire a model that gives insight to the relationships between each independent variable and each MOP. Also, we note that both algorithms are greedy and run the risk of getting stuck at a local optimum. However, these methods can be effective at reducing the number of variables in the model, thus potentially reducing collinearity and allowing for sensible coefficients.

5.1.2.4 Ridge Regression

Ridge regression is a form of coefficient shrinkage that works well for datasets with collinearity. When collinearity exists, we often see very large coefficients on independent variables, some positive and others negative, many of which imply relationships between independent and dependent variables that may be illusory. Ridge regression penalizes the model for having very high coefficient values, and thus discourages this behavior. Ridge regression solves the following optimization problem:

\[
\hat{\beta}_{\text{ridge}} = \arg\min_{\beta} \left\{ \sum_{i=1}^{N} \left( y_i - \beta_0 - \sum_{j=1}^{P} x_{ij} \beta_j \right)^2 + \lambda \sum_{j=1}^{P} \beta_j^2 \right\}
\]  

(5-5)

where \( \lambda \) is the shrinkage parameter. Increasing \( \lambda \) introduces a greater amount of shrinkage in the coefficients (towards zero).

5.1.2.5 Model Setup and Results

This section discusses the application of these techniques to our problem and shows the results of each method. The data set consists of 293 data points corresponding to 293 runs of the coordination planner on various datasets and for many different value function combinations. Each row of data corresponds to one run of the coordination planner. Each row consists of the input for each of the seven value function components, the number of requests in the dataset, the length of the planning horizon, and the final values of each of the four MOPs. Our independent variables, then, consist of each value function component, plus each second-order interaction term of the value function components, the number of requests in the dataset, and the length of the planning horizon. Thus, \( X \), the matrix of inputs, is a 293 x 9 matrix. The number of columns grows beyond 9 for the models that use the second-order interaction terms.
This means we hold constant all other inputs (sub-planners in the system, planning cycle data, communication data, external requests on sub-planners, etc.). That is, we treat all other possible variables as exogenous to our model. This is necessary for the following reasons:

1) There are many, many factors affecting the MOPs, and allowing each factor to be an independent variable in the regression model creates an overly complex model.

2) If we include a large number of factors as independent variables, we would require a sample size too large to generate in a reasonable amount of time to build an effective model.

3) The benefit of creating models for a single system far outweighs the costs of building such a model. This is especially true relative to building a model for all systems.

To show this idea, we present the figures showing the process of using regression to build a relationship between the inputs and outputs for a system.

![Figure 5-4: Regression Model Inputs/Outputs](image-url)
The top of this figure depicts the process of finding some function that maps the inputs to the system to the outputs of the system. The bottom half of the figure lists a number of inputs that affect the outputs of the system, which we consider to be the four MOPs of interest. In addition to these inputs, every model requires a term that represents the bias, constants, noise, and variables that are not considered to influence the system (exogenous variables). As stated above, however, it is difficult to generate a test set large enough to account for all of these different variables for a single system. So, we hold many of the inputs constant and build a simpler model for simplification purposes. This is shown in Figure 5-5:

We use the four MOPs \((\text{avgPriority}, \text{avgTSV}, \text{numRequests}, \text{numDualCollects})\) as the dependent variables. We create a model for each MOP using the same set of independent variables.

To evaluate each model, we consider three measures:

1) Root Mean Squared Error (RMSE) – the average error on a prediction
2) Overestimation error (Avg OE) – average amount by which the model overestimated a prediction (because a user wants each MOP as high as possible, overestimating the prediction is problematic)

3) Percentage of predictions that are overestimated (% OE)

To create the models, we first split the dataset into a training set (75% of the data points) and a test set (25% of the data points). We do this to avoid overfitting the model to this particular dataset. For the OLS models, we create models with and without an intercept, and with and without 2nd order interaction terms.

For the ridge regression models, we first scale the data because ridge coefficients are known to be sensitive to scale [65]. We then transform the scaled coefficients to obtain coefficients that are on the scale of the original data. To choose the ridge parameter $\lambda$ we use 5-fold cross-validation on the training data and choose $\lambda$ such that the average RMSE, Avg OE, and %OE across all folds are low. We also consider the ridge trace, ensuring we choose $\lambda$ such that each coefficient is sufficiently stable as the ridge parameter is changed near $\lambda$.

Stepwise regression results were not included in the output tables below due to their poor performance. Forward stepwise regression was overly selective. The algorithm terminated after only two, three, or four of the independent variables were included. Because of this, the models lacked important variables and the coefficients were misleading. The $R^2$ and adjusted $R^2$ values were quite low as well, indicating insignificant explanatory power.

We present the regression analysis for the average priority of observations (avgPriority) MOP as a sample of our findings.

<table>
<thead>
<tr>
<th>Ridge, Interactions $(\lambda = 8.95)$</th>
<th>Ridge, No Interactions $(\lambda = 0.5)$</th>
<th>Best OLS Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMSE</td>
<td>53.56</td>
<td>46.46</td>
</tr>
<tr>
<td>Avg OE</td>
<td>36.7</td>
<td>29.9</td>
</tr>
<tr>
<td>% OE</td>
<td>48.81</td>
<td>51.55</td>
</tr>
</tbody>
</table>

Table 5-1: Training Set Regression Results

We highlight in blue the methods that performed the best in each category of evaluation. The OLS model had the lowest average error (RMSE) (as expected) and average overestimation error (Avg OE), but highest percentage of overestimated predictions, although each model
overestimates nearly 50% of the training data points. This model had no interaction terms, and each value function component was highly significant in the model. The coefficient on the priority weight, \( w_p \), had the highest absolute value of all positive coefficients, reaffirming our hypothesis that this weight would have the most significant impact on the average priority observed MOP. Coefficients are shown in Appendix E.

We build ridge regression models with and without second-order interaction terms, and with and without intercept terms. The models with intercepts performed far better than those without intercepts. Also, the ridge regression model with interaction terms overestimated its predictions least often compared to the other models, but had a significantly higher RMSE than the best OLS model. This was true on the test set as well. The test set results are shown below:

<table>
<thead>
<tr>
<th></th>
<th>Ridge, Interactions ((\lambda = 8.95))</th>
<th>Ridge, No Interactions ((\lambda = 0.5))</th>
<th>Best OLS Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMSE</td>
<td>45.14</td>
<td>37.79</td>
<td>37.64</td>
</tr>
<tr>
<td>Avg OE</td>
<td>34.61</td>
<td>29.33</td>
<td>29.15</td>
</tr>
<tr>
<td>% OE</td>
<td>52.70</td>
<td>54.05</td>
<td>54.05</td>
</tr>
</tbody>
</table>

Table 5-2: Test Set Regression Results

The OLS model outperforms the other models in terms of RMSE and Avg OE. Each of the models creates predictions that overestimate the actual responses over 50% of the time. While the OLS model overestimated actual responses more often than our training error would indicate it should, these overestimations were low in magnitude because the RMSE is better (lower) in the test set than it was in the training set.

At this point, it is important to note that there is, by definition, multicollinearity in the data. This is true due to the fact that the weight input terms \( w_i \) sum to 1. Each input term can be written as a linear combination of the others and is thus dependent on the others. This means that our estimates of the coefficients, while still unbiased, suffer from high variance and are sensitive to even small changes in the data. Nonetheless, it is evident from our training and test set results that the OLS model for \( \text{avgPriority} \) outperforms the other models. In fact, this was the case for the other MOPs as well. We included the ridge regression models specifically to address this issue of multicollinearity but saw no significant improvement in model performance. However, the ridge regression models do produce more sensible coefficients. That
is, the ridge models more accurately capture the fact that some input weights are negatively correlated with MOPs. For example, forcing the coordinator to take more dual collections will hinder its ability to take as many observations. This is because the coordinator attempts to have more dual collections completed rather than simply schedule observations for as many targets (whether single collections or dual collections) as possible. The ridge model is able to capture this by generating a negative coefficient on the dual collection input weight for the numRequests model. Although the ridge models offer better explanations for the relationship between independent and dependent variables, the OLS model performs better for prediction purposes. Thus, we use the OLS model with no interaction terms as the predictor for the average priority of the requests observed.

Backward stepwise regression worked well in some cases but not in others. The method chose to keep each value function component in each model, along with a handful of 2nd order interaction terms. However, at times, the interaction terms that it kept had highly negative coefficients, which could be an indication of a poor model and collinearity in the data. For these reasons, we do not present the detailed output from these regression models but rather explain their performance at a high level.

We present the results of each regression model in Appendix E, and we summarize how the models quantify the relationships between input weights and output statistics in the following table:

<table>
<thead>
<tr>
<th>Input Weight</th>
<th>Relationships Found</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_p$</td>
<td>Highest positive coefficient of all components in avgPriority model</td>
</tr>
<tr>
<td>$w_{TSV}$</td>
<td>Second highest positive coefficient of all components in avgTSV model (coefficient is only slightly smaller than the simult component, which takes a high value due to the chosen datasets)</td>
</tr>
<tr>
<td>$w_o$</td>
<td>High positive coefficient in numRequests, numDualCollects models</td>
</tr>
<tr>
<td>$w_d$</td>
<td>Highest positive coefficient of all components in numRequests model</td>
</tr>
<tr>
<td>$w_{RFO}$</td>
<td>High positive coefficient in numRequests model</td>
</tr>
<tr>
<td></td>
<td>Interaction with Time Window component in avgTSV model</td>
</tr>
<tr>
<td>$w_s$</td>
<td>Highest positive coefficient of all components in numDualCollects model</td>
</tr>
<tr>
<td>$w_{TW}$</td>
<td>High positive coefficients in numRequests, numDualCollects models</td>
</tr>
<tr>
<td></td>
<td>Interaction with RFO component in avgTSV model</td>
</tr>
</tbody>
</table>

Table 5-3: Relationships Found Between Variables Through Regression

137
These results support our hypotheses in most cases. For the OLS models, we note that the coefficients on the input weights we predict to be highest for each model are indeed the highest in absolute value. The ridge models, which we believe more clearly quantify the relationships between the independent variables and the dependent variables, also yield sensible coefficients in most cases. Only for the RFO component do we see confusing results. The ridge model suggests that increasing the RFO component hinders the coordinator’s ability to take dual collections. This is likely a either a function of the specific datasets used to produce the rows of data for the regression analysis, or a result of the fact that the dual collection input weight is so strongly correlated with the dual collection output that the RFO component forced to be negative. This is possible because the ridge parameter for this model was quite low (0.6), indicating that the model is nearly an OLS model with an intercept, and hence is still experiencing collinearity. It is important to note that the ridge parameter is chosen based on the model’s average cross-validation error, and not on the sensibility of each coefficient.

Two other relationships of interest are the surprisingly high positive correlations between the time window component and the number of requests accepted/dual collections made, as well as the interaction between the RFO and time window components for the avgTSV MOP. The high correlation between the time window component and the number of observations indicates that this component is affecting the coordinator’s performance more than we anticipated it would. This relationship suggests that it is quite useful to consider sub-planner objectives and constraints.

The interaction term between the time window and RFO components for the quality of observations MOP is also interesting to consider. This term indicates a relationship between the difficulty of observing a target and the urgency with which we consider its inclusion in a batch of queries. The relationship is different, however for the OLS and ridge models. Equations (5-6) and (5-7) define the rates of change of the avgTSV MOP with respect to each of these value function components:

\[
\frac{\partial \text{avgTSV}}{\partial w_{\text{RFO}}} = \beta_{w_{\text{RFO}}} + \beta_{w_{\text{RFO}}w_{\text{TW}}} w_{\text{TW}} \quad (5-6)
\]

\[
\frac{\partial \text{avgTSV}}{\partial w_{\text{TW}}} = \beta_{w_{\text{TW}}} + \beta_{w_{\text{RFO}}w_{\text{TW}}} w_{\text{RFO}} \quad (5-7)
\]
For the ridge models, $\hat{\beta}_{WRFO} < 0$, $\hat{\beta}_{WTW} < 0$, and $\hat{\beta}_{WRFO^TW} > 0$, indicating that a marginal increase in either input weight alone will decrease the average observation quality MOP. However, this effect will be less pronounced when the related input weight is higher. That is, the more the coordinator values targets with few RFO’s highly (the higher $w_{RFO}$ is), the model suggests that valuing “easy” requests (increasing $w_{TW}$) will result in a smaller decrease in the observation quality MOP. Thus, if the coordinator is using future information ($w_{RFO}$ is high), it is beneficial (in terms of $avgTSV$) to use queries efficiently. To demonstrate this relationship, we consider two targets of equal priority having an equally small number of RFO’s remaining: one with a narrow time window and the other with a much wider time window. If the coordinator is operating with a sense of urgency for these requests, it is making the quality of the observations it can obtain a secondary objective, and the model suggests choosing to query the request with a higher chance of being scheduled will lessen the hurt to the observation quality MOP. We hypothesize that this is the case for the following reason: if the coordination planner is able to eliminate this request sooner rather than later, it has a greater ability to match targets with sensors such that observation quality is maximized. If it fails to schedule the urgent requests at the current time, these requests only become more important to the coordinator in future time steps, perpetuating the coordinator’s inability to make observation quality a primary objective. If the coordinator is not as concerned with RFO’s, then valuing “easier” requests is not as useful. Instead, the coordinator is better off simply considering which decision has the potential to yield a higher quality observation without regarding the chances it is accepted. It is important to note that we would expect other interaction terms to be statistically significant due to the nature of the seven value function components, and the fact that the MOPs are competing objectives; however, this was the only significant interaction term that we identified.

Finally, based on empirical as well as statistical analysis, we conclude that each component contributes to the coordination planner’s performance significantly in some way, and in the manner we intended.

5.1.2.6 Regression Summary

The previous subsections described the use of three regression models to quantify the relationship between the value function components and each MOP. Linear regression and
ridge regression performed similarly, while stepwise regression performed poorly. Although they suffer from collinearity, the linear regression models performed slightly better than the ridge regression models. Table 5-4 summarizes the findings of our statistical analysis of the value function:

<table>
<thead>
<tr>
<th></th>
<th>OLS</th>
<th>Forward Stepwise</th>
<th>Backward Stepwise</th>
<th>Ridge Regression</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>avgPrior</strong></td>
<td>Highly significant model and coefficients if we use only the 7 VF components; all interaction terms insignificant.</td>
<td>Very low R-squared</td>
<td>Highly negative coefficients on interaction terms, insignificant p-values on these terms.</td>
<td>Models with interaction terms performed poorly, those with no interaction terms performed slightly better but no better than OLS, even on test sets. Generates more sensible coefficients.</td>
</tr>
<tr>
<td><strong>avgTSV</strong></td>
<td>Highly significant model and coefficients if we use only the 7 VF components.</td>
<td>Very low R-squared</td>
<td>High R-squared, significant coefficients.</td>
<td>Models with interaction terms performed poorly, those with no interaction terms performed slightly better but no better than OLS, even on test sets. Ridge model with single RFO*TW interaction term performed better than other ridge models, but no better than OLS. Generates more sensible coefficients.</td>
</tr>
<tr>
<td><strong>numRequests</strong></td>
<td>Highly significant model and coefficients if we use only the 7 VF components.</td>
<td>Very low R-squared</td>
<td>Highly negative coefficients on interaction terms, insignificant p-values on these terms.</td>
<td>Models with interaction terms performed poorly, those with no interaction terms performed slightly better but no better than OLS, even on test sets. Generates more sensible coefficients.</td>
</tr>
<tr>
<td><strong>numDualCollects</strong></td>
<td>Highly significant model and coefficients if we use only the 7 VF components.</td>
<td>Very low R-squared</td>
<td>Some negative coefficients on interaction terms, insignificant p-values on some of these terms.</td>
<td>Models with interaction terms performed poorly, those with no interaction terms performed slightly better but no better than OLS, even on test sets. Generates more sensible coefficients.</td>
</tr>
</tbody>
</table>

Table 5-4: Summary of Regression Model Performance

While this table summarizes the performance of each of the regression models, we note that Table 5-3 describes the relationships found between the independent variables and each MOP. Based on these relationships, we recommend a user choose weights as follows:
To emphasize the priority of the requests serviced, increase the priority weight, \( w_p \), to a value greater than the other weights. To emphasize the number of targets observed, increase the distance weight, \( w_d \). To encourage the coordinator to choose queries such that the quality of the
observations is maximized, increase \( w_{TSV} \). Finally, to emphasize dual collections, increase the dual collection weight, \( w_S \). While these are the primary weights that should be altered, the remaining weights, \( w_D, w_{RFO}, \) and \( w_{TW} \) should be kept at nonzero levels, as each improves the final MOPs in their own ways. The following section describes methodologies that can allow a user to more optimally choose input weights for a set of desired MOPs, if the resources are available.

5.2 Objective Function Tuning

This section addresses how to most intelligently use the value function to meet specific objectives using the statistical analysis of Section 5.1.2 as a basis. The value function we use in this thesis can be weighted to balance the many competing objectives present in the problem, and, as shown in the previous section, some non-linearities exist in the value function that make the most efficient use of the value function less than obvious. Moreover, the models created in the previous section were relatively simple and did not account for all of the variables that might influence final plans in an operational setting. Thus, we hypothesize that the model that best predicts the value of the final MOPs would likely be highly non-linear in a real-world application. Given this complexity, in this section we present small optimization problems that can be used to optimize the choice of value function weights to meet a user's specific objectives. This type of analysis can be thought of as a form of isoperformance in that we choose inputs such that our vector of objectives lies in a specified region. However, we do not utilize the algorithms specifically proposed for isoperformance (exhaustive search, vector-spline approximation, tangential front following, etc.) [66].

5.2.1 Non-Linear Formulations

Because we found that a two-way interaction term was significant in determining the output of the coordination planner, the following optimization problems are non-linear, either in the objective function or in the constraints.

Each optimization problem seeks a weight vector \( w \) that optimizes the predicted objectives using the models derived by regression. It is important to note that in the regression models, the vector \( w \) was input; however, in the optimization problems that follow, \( w \) is a vector of decision variables. We also include constraints on the other objectives. For instance, if we seek an objective function that maximizes the average priority of the requests obtained
(avgPriority), we solve the following non-linear optimization problem, where our decision variables are the $w_i$'s:

**AP-NL:**

\[
\text{max} \quad \text{avgPriority}
\]

subject to \[
\text{avgTSV} \geq \text{avgTSVDesired} \\
\text{numRequests} \geq \text{numRequestsDesired} \\
\text{numDualCollects} \geq \text{numDualCollectsDesired} \\
\sum_{i=1}^{7} w_i = 1 \\
0 \leq w_i \leq 1, \forall i \in \{1..7\}
\]

Each of the right-hand side values are constants chosen by the user, while the left-hand side and objective function are defined by equations corresponding to the regression equations for each MOP. In addition to the **avgPriority** formulation (AP-NL), we have the following formulations for the other MOPs:

**NR-NL:**

\[
\text{max} \quad \text{numRequests}
\]

subject to \[
\text{avgTSV} \geq \text{avgTSVDesired} \\
\text{avgPriority} \geq \text{avgPriorityDesired} \\
\text{numDualCollects} \geq \text{numDualCollectsDesired} \\
\sum_{i=1}^{7} w_i = 1 \\
0 \leq w_i \leq 1, \forall i \in \{1..7\}
\]

**ATSv-NL:**

\[
\text{max} \quad \text{avgTSV}
\]

subject to \[
\text{numRequests} \geq \text{numRequestsDesired} \\
\text{avgPriority} \geq \text{avgPriorityDesired} \\
\text{numDualCollects} \geq \text{numDualCollectsDesired} \\
\sum_{i=1}^{7} w_i = 1 \\
0 \leq w_i \leq 1, \forall i \in \{1..7\}
\]

**DC-NL:**

\[
\text{max} \quad \text{numDualCollects}
\]

subject to \[
\text{numRequests} \geq \text{numRequestsDesired} \\
\text{avgPriority} \geq \text{avgPriorityDesired} \\
\text{avgTSV} \geq \text{avgTSVDesired} \\
\sum_{i=1}^{7} w_i = 1 \\
0 \leq w_i \leq 1, \forall i \in \{1..7\}
\]
5.2.1.1 Implementation and Results

We implement these formulations in AMPL using the solver LOQO, which uses a barrier function to solve non-linear optimization problems. By changing the right-hand side for each of these formulations, the problem becomes infeasible fairly quickly, but it can still be tuned to some degree. This issue points to the need to formulate these problems using Robust Optimization (RO), which is left for future work. However, we present some results showing our ability to directly embed regression models in non-linear optimization problems to adjust the performance of the coordination planner according to some input objectives.

Again, we present sample output using the AP-NL formulation. Table 5-5 shows three separate sets of weight inputs based on notional user objectives.

<table>
<thead>
<tr>
<th>Purpose</th>
<th>numRequests Desired</th>
<th>avgTSV Desired</th>
<th>numDC Desired</th>
<th>w_p</th>
<th>w_TSV</th>
<th>w_o</th>
<th>w_D</th>
<th>w_RF0</th>
<th>w_S</th>
<th>w_TW</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case 1</td>
<td>Unconstrained problem</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Case 2</td>
<td>Make observing some number of targets a constraint</td>
<td>20</td>
<td>0</td>
<td>0</td>
<td>0.2622</td>
<td>0</td>
<td>0</td>
<td>0.7378</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Case 3</td>
<td>Ensure we view 3 dual collects</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0.2059</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.794</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 5-5: Results from Examples of Weight Vector Optimization

We note that these three runs are actually linear optimization problems because we have effectively relaxed the non-linear constraints related to the quality of observations desired (avgTSVDesired).

In the first case, we notice that the priority weight is increased to maximum value, which we expect due to the linearity of the regression function and the fact that the priority weight coefficient was highest. The second case represents a user who is concerned with both the priority of and the number of requests that are executed. The optimization problem yields a distance weight that is almost three times that of the priority weight, indicating that although the objective function seeks to maximize the average priority of requests observed, the coordination planner cannot ignore low priority requests if it will meet the constraint on the numRequestsDesired MOP. The third case requires that the coordination planner collects a certain
number of observations simultaneously, and indicates that the coordination planner will have to place almost four times as much emphasis on the dual collect component of the value function as the priority component. The results from the weighting schemes are shown below (averaged over 3 runs for each case):

<table>
<thead>
<tr>
<th></th>
<th>Predicted avgPriority</th>
<th>Actual avgPriority</th>
<th>Predicted avgTSV</th>
<th>Actual avgTSV</th>
<th>Predicted numRequests</th>
<th>Actual numRequests</th>
<th>Predicted numDC</th>
<th>Actual numDC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case 1</td>
<td>694.79</td>
<td>672.20</td>
<td>43.24</td>
<td>43.14</td>
<td>17.86</td>
<td>17.5</td>
<td>0.374</td>
<td>1</td>
</tr>
<tr>
<td>Case 2</td>
<td>634.94</td>
<td>613.35</td>
<td>43.86</td>
<td>42.69</td>
<td>20</td>
<td>21.33</td>
<td>0.413</td>
<td>2</td>
</tr>
<tr>
<td>Case 3</td>
<td>615.49</td>
<td>604.55</td>
<td>44.95</td>
<td>44.77</td>
<td>15.06</td>
<td>13.5</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 5-6: Actual versus Predicted MOPs

The table’s blue entries correspond to the actual average priority of the requests observed over all runs. The green entries correspond to the actual value of the right-hand side of the constraints for the respective cases. We can see from this output that the predicted values satisfy the constraints for each case, and that the objective function values were close to, albeit below, their predicted values. We present this example to demonstrate the ability to use our regression models to effectively tune the objective function. The optimization problems provided us with a vector of weight inputs that was arguably not immediately obvious even though the formulations involved were linear. One could argue that the choice of weight inputs is, in general, obvious enough that mathematical programming is unnecessary, although this may not be the case for systems with a highly non-linear relationship between the weighting inputs and the MOPs of the resulting plans. Using the inputs derived by the optimization models, the coordination planner created plans that aligned with the original user objectives. We must be careful to note that the coordination planner will not always meet all constraints, and that the optimization problems are sometimes infeasible. Also, this methodology requires extensive statistical analysis of past observation plans. Nonetheless, we conclude that the process of optimization-based value function tuning provides a benefit.

5.2.2 Robust Optimization (RO) Approaches

This section presents a non-linear optimization formulation for value function tuning based on the work of Bertsimas and Sim [51]. RO is well-suited for regression-based optimization problems given the uncertainty inherent in statistical models.

In the previous section, we formulated small non-linear optimization problems to meet constraints on certain MOPs while maximizing another. However, the coefficients from the
regression models that form the objective function and the constraint matrices for these problems are uncertain. To address this, we adapt a simple, non-linear RO formulation from the work by Bertsimas and Sim.

If we use the OLS regression models, we have an estimate of the mean and of the standard deviation (standard error) of the true coefficients, and this provides a basis for creating the RO formulation. Thus, there is uncertainty in the estimate $\hat{\beta}_i$, the regression coefficient on the $i^{th}$ component of the value function.

From the OLS models, we obtain an estimate of the mean for each $\beta_i (\mu_i)$, and their standard errors ($s_i$). Let $\hat{\beta}$ be the vector of regression coefficients for the regression equation obtained for the average priority of the targets observed. Then we have uncertainty in the realizations of the coefficients as defined by

$$\hat{\beta}_i = \mu_i + s_i z_i, \quad \forall \ i \in 1..7$$

where $z_i$ is a decision variable to be chosen, and helps to represent realizations of the coefficients. If we imagine a problem where we wish to maximize the average priority of the targets obtained with uncertain regression coefficients (and we relax the constraints due to the other measure of performance) the problem becomes

$$\text{RO-NL:} \quad \max \min_w \sum_{i=1}^{7} (\mu_i + s_i z_i) w_i$$

subject to

$$\sum_{i=1}^{7} w_i = 1$$

$$0 \leq w_i \leq 1, \quad \forall \ i$$

$$-1 \leq z_i \leq 1, \quad \forall \ i$$

$$\sum_{i=1}^{7} |z_i| \leq \Gamma$$

This problem seeks to choose $z$ and $w$ to maximize the average priority obtained in the worst case. Minimizing over $z$ is what causes the model to act conservatively. Constraint (5-10) ensures that our weights sum to 1, as is required by the value function, while Constraints (5-11) ensures all weights are within their correct bounds. Constraints (5-12) dictate that our regression coefficients deviate by no more than 1 standard deviation (this can be altered). Constraint (5-13) enables a Budget of Uncertainty ($\Gamma$) that dictates how much deviation we allow in the problem and is generally chosen by some mild assumptions on the distribution of the uncertain data.
This problem can be easily solved for some fixed $z$ or $w$ with subgradient optimization. The steps for the subgradient optimization algorithm are as follows:

Step 0: Initialize with some feasible $w$.

Step 1: Solve the sub-problem

\begin{equation}
\min_z \sum_{i=1}^{7} (\mu_i + s_i z_i) w_i
\end{equation}

\begin{equation}
-1 \leq z_i \leq 1, \ \forall \ i
\end{equation}

\begin{equation}
\sum_{i=1}^{7} |z_i| \leq \Gamma
\end{equation}

Step 2: Update $w$ at iteration $t$ by $w_{t+1} = w_t + \alpha \frac{\gamma}{\|\gamma\|}$ where $\gamma = \mu + sz^*$ and $\alpha$ is some intelligently chosen step-size, subject to Constraints (5-10) and (5-11).

Step 3: Return to Step 1.

We note that given the small size of the problems being considered, a non-linear solver such as LOQO could easily solve this problem as well.

Although we have not implemented this formulation, we present it as a way of diminishing the probability of obtaining infeasible solutions, which is a potential issue for the prior formulations. Moreover, we could expand the RO to this process of value function tuning to meet user objectives by including constraints on the other MOPs based on their respective regression models. However, this is also left for future work.

5.3 Sub-Planner Query Flexibility

In this section, we analyze the impact of varying the communication constraints imposed on the system. That is, we vary the number of queries allowed per iteration ($\theta_p$), and we vary the number of iterations allowed per hour. For this analysis, we use a product of the $avgPriority$ and $numRequests$ MOPs as the metric by which we compare plans.

We measure the effect of varying $\theta_p$, the number of queries allowed per iteration. This parameter controls how many requests the sub-planners are willing to provide feedback on at any one iteration. The goal of this analysis is to assess the increase in coordinator performance at various levels of sub-planner flexibility. We hypothesize that as $\theta_p$ increases, so should the coordinator's ability to have targets observed on sub-planners, to a point. Clearly, increasing $\theta_p$ provides the coordination planner with more information on targets than can be observed. As
\( \theta_p \) becomes larger, however, there is a greater chance that the sub-planners choose lesser value targets, because the sub-planners and the coordinator do not value targets equally. This means the quality of the plans is not guaranteed to increase as we increase \( \theta_p \). Thus, the value of plans created should increase to a point. We test this hypothesis by varying \( \theta_p \) on 15 discrete levels, from 1 to 40, and running the coordination planner on 15 test sets, with 3-4 runs per test set due to the stochastic nature of the UAV sub-planner. Thus, each bar on the graph below is the result of 45-60 runs. We assume \( \theta_p \) is equal for each sub-planner for simplification purposes.

We also analyze the effect of changing the number of iterations allowed per hour. The purpose of analyzing the sensitivity of final observation plans to changes in this parameter is to quantify the benefit of having sub-planners send feedback to the coordination planner. We hypothesize that we should see an increase in plan quality when we increase the amount of feedback the coordinator receives from the sub-planners. To test this hypothesis, we conduct runs at each of three levels: 4 iterations per hour, 10 iterations per hour, and the case where the coordinator does not receive any feedback from the sub-planners. We include this last case to quantify the value of feedback information.

We present analysis for a day-long scenario, with 100 requests and 6 sub-planners. We present the results from these runs in Figure 5-6:

![Theta vs. Total Priority, 100 Requests, 24 Hour Horizon](image)

Figure 5-6: Varying Communication Constraints
These results confirm our hypothesis that increasing $\theta_p$ will result in an increase in the value of plans, to a point. That is, we observe a rapid increase in performance until $\theta_p \approx 10$, at which point the total priority of the plans we create increases more gradually, from 30,000 to 35,000. This is the case for each communication rate level. We do see that the graph is not monotonically increasing because of decreases in total priority of final plans near $\theta_p = 7$ and $\theta_p = 9$. This is a result of the fact that at these values, the nature of these datasets are such that the sub-planners receive subsets of requests that are high in value and feasible but at $\theta_p > 9$ the sub-planners are exposed to additional requests they are more likely to schedule. This is the reason that the total priority is not monotonically increasing in $\theta_p$.

These results also confirm our hypothesis concerning the value of feedback information. That is, we observe a great deal of benefit gained by getting feedback from the sub-planners, especially when $\theta_p$ is small. It is sensible that the value of receiving this information is higher in a more constrained setting (i.e., when $\theta_p$ is small). In particular, the following chart shows the percent difference in total priority of the observation plans as a result of increasing the communication rate:

<table>
<thead>
<tr>
<th>% Increase, 4 ltrs/Hr</th>
<th>% Increase, 10 ltrs/Hr</th>
</tr>
</thead>
<tbody>
<tr>
<td>60</td>
<td>50</td>
</tr>
<tr>
<td>50</td>
<td>40</td>
</tr>
<tr>
<td>40</td>
<td>30</td>
</tr>
<tr>
<td>30</td>
<td>20</td>
</tr>
<tr>
<td>20</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 5-7: Percent Increase in Total Priority Due to Feedback

Above, we see the percent increase in total priority observed decrease drastically as $\theta_p$ reaches 7. However, we still see a benefit of receiving feedback on the order of 5-12% beyond $\theta_p = 7$ for a communication rate of 4 iterations per hour, and a benefit of 9-18% for 10 iterations per hour.
We note that the percent increase begins to stabilize at around 8% (4 iterations per hour) and 10% (10 iterations per hour) for $\theta_p > 10$.

5.4 Rejected Queries

It is possible at any iteration that there may not be $\theta_p$ requests with non-zero value when paired with sub-planner $p$. This could occur if less than $\theta_p$ feasible opportunities for unique requests exist, or if feedback from previous iterations has indicated that fewer than $\theta_p$ requests should be given positive value. However, if we are given $\theta_p$ slots for requests, and we relax the assumption that we try not to overload sub-planners, the coordination planner should presumably use as many as possible. This section discusses a stochastic technique that attempts to make better use of these opportunities.

One way to think about how the coordination planner operates is that it attempts to insert targets into the existing routes of each sub-planner, according to its value function. This methodology is necessary given the framework in which we assume the coordinator must operate. A greedy insertion technique, while effective, often leads to a solution near a local optimum. To expand the neighborhood of possible solutions at its next iteration, insertion techniques sometimes introduce randomness. Randomness can allow a solution methodology to get closer to the global optimum. While we do not allow "worse" solutions to replace better solutions, we introduce randomness to randomly increase previously zeroed-out values to non-zero levels once again in order to widen the search space used to find better solutions. We hypothesize that requests that were previously rejected by a sub-planner could possibly be inserted into the routes at a later time because of changes in the existing routes since their last rejection.

To test this hypothesis, we run the planner with and without the randomized value increase on the same datasets. We test the concept on 15 datasets, with three runs for each case. We fix the weights of the value function for each run so that $w_i = \frac{1}{7}$ $\forall i$, and we fix the parameters associated with system communication constraints. The results shown in the table below are averages over all datasets:
To demonstrate that the two methods produce plans with differences in output that are statistically significant, we present a paired t-test for these data in Appendix F. We use a paired t-test because we run the planner twice on each dataset, once with the randomization module, and once without it. Thus, the two runs are dependent on each other (based on the same dataset). The tests show that our hypotheses are indeed confirmed.

An important corollary to this analysis is that it is useful to maintain some value for requests that were previously rejected and try querying these requests multiple times. This corollary could have an especially strong impact if the coordination planner was given a reason for the rejection of each request. One could imagine a process that learns a function that appropriately discounts the value of request-to-sub-planner pairs upon receiving a certain type of feedback. This could help the coordinator accurately retain value for requests that have been rejected but may be accepted at a later time.

### 5.5 Coordination Planner versus Baseline Scenarios

In this section, we quantify the benefit of coordination in a notional CLARREO scenario. We build a scenario with 1 UAV sub-planner, and 5 satellite sub-planners. The 5 satellite sub-planners notionally represent the Aqua, Terra, CLARREO, NPOESS, and TRMM missions, while the UAV sub-planner represents a notional UAV mission based at Edwards Air Force Base (AFB), California. In the past, UAVs have been flown out of Edwards AFB for Earth Science missions under the control of the Dryden Flight Research Center.

The purpose of this analysis is to quantify the benefit of having users coordinate their request submittals versus current operations. We hypothesize that coordination should allow for more intelligent use of queries, and for an overall more efficient allocation of requests to
sub-planners. Thus, we foresee a significant increase in each MOP. To test this hypothesis, we build a simulation environment in which we model increasing levels of coordination, using the distribution of the priority of the targets observed to evaluate performance.

The simulation environment, implemented in MATLAB, can test any of the following levels of coordination on the same set of inputs:

Case 1: Requests randomly assigned to sub-planners ("blind" approach)
Case 2: Users intelligently query 1 sub-planner; choose randomly from other sub-planners if returned value is zero
Case 3: Users intelligently query all sub-planners; choose sub-planner returning highest value
Case 4: Coordination Planner allocates requests to sub-planners

We depict each case in Figure 5-8 below:

![Levels of Coordination](image)

Figure 5-8: Simulation Environment Cases

The difference between Cases 3 and 4, then, is the lack of a coordinating agent to organize which queries to send to each sub-planner in Case 3. The resulting inefficiency is that if a sub-planner \( p \) is sent \( > \) requests, the sub-planner randomly chooses a subset (of size \( \) ) of those requests it is sent to evaluate for insertion into its plans.
To control the experiments, we allow users to use the value function created for the coordination planner. We do this to simulate how an “intelligent user” would query sub-planners. We also subject each case to the same communication constraints present in the coordination case. The software supports multiple planning periods and multiple queries per period and interfaces with the UAV and satellite planners just as the coordination planner does.

We note that current operations are most accurately described by Case 2, in which each user intelligently queries one sub-planner. That is, this case most accurately represents that, in current operations, assets usually have a user community and users become attached to a single asset. While it is important to observe that in current operations, the lack of a feasible opportunity in the near future on the user’s preferred asset does not usually prevent the user from obtaining data for his request. Instead, in this case, the user will likely experience a significant time delay in obtaining the data he requires, since he is unaware of the opportunities available on other assets at the time he approaches his preferred asset. However, we leave analysis of the time delay MOP as future work.

To demonstrate the benefit of coordination, we use a scenario with 496 targets (95 requiring dual collection) distributed in a grid around the Earth. We use 6 sub-planners (1 UAV sub-planner and 5 satellite sub-planners), and consider a 10-hour planning horizon. For this scenario, we obtain results averaged over 3 runs, by case, as shown in Figure 5-9:
We note the number of targets in the system at each priority level atop each x-axis entry. Again, the priority system is such that a target with 1,000 priority is more important than one with 100 priority. The dataset was created with fewer targets of high priority, and more of lesser priority.

The output shows that the coordinated case (Case 4) outperforms each of the other cases substantially. Case 2, which most closely resembles current operations, performs quite poorly relative to the coordinated case. This is because of the coordination planner’s enhanced situational awareness relative to the other cases, which enhances its ability to organize its queries to obtain the most useful information from sub-planners at any time. Case 3 outperforms Case 2 substantially, primarily because allowing users to query all of the sub-planners allows for more feasible opportunities. We also see a major increase in final plan quality from Case 3 to Case 4. Specifically, coordination shifts the distribution of the priority of the targets observed far to the right compared to Case 3. This is because in Case 3, the individual users are unable to coordinate their queries such that each query is useful. That is, Case 3 encounters the issue of resource overload, where the sub-planners reject queries before even attempting to insert them into existing routes due to their query limits. In particular, the
A UAV sub-planner proved quite useful in this scenario; as such, each user wished to query the UAV sub-planner often. However, the UAV sub-planner accepts only a finite number of queries at any iteration, and can accommodate only a finite number of requests in its actual observation plans. Thus, in Case 3, many queries are wasted because the sub-planners are overloaded with queries.

Additionally, we note that the spike at the 200 priority level (and drop at the 100 level) is simply due to the fact that, for this dataset, the requests at the 200 priority level happened to have more feasible opportunities than the other levels. The dataset was created, however, with a roughly linearly decreasing number of feasible opportunities (and targets) as priority level increases.

5.6 CLARREO Orbit Analysis

In this section, we present analysis on the effect of varying the orbits of the assets involved in the CLARREO mission. Again, for this notional CLARREO scenario, we use the same sub-planners described at the beginning of Section 5.5. In particular, we focus on the inclination of the orbits and the Right Ascension of the Ascending Node (RAAN). Table 5-8 details each case we consider and the purpose for its inclusion in the experimental plan.
For this analysis, we consider a test set with 496 requests, distributed in a grid throughout the Earth. The grid of targets is constructed such that targets are nearly equally spaced apart. This is a common practice when seeking measurements throughout the Earth to obtain a dataset that covers the Earth evenly. The requests do not have time windows associated with them, to allow for many observations to be made.

Before showing and analyzing the results, we note that this analysis is largely demonstrative. That is, to actually show that certain orbits would provide more benefit than others would require an extremely large experiment with planning horizons on the order of years. To avoid the time and complexity this analysis would require, we conduct our runs on one or two day long scenarios to demonstrate another benefit of coordinated planning.

Table 5-9 presents the results from runs of the coordination planner on the 17 cases. We include summary statistics on dual collections, as well as the total number of requests serviced in the output. Specifically, we include the number of dual collects involving CLARREO assets, because this is of interest for the CLARREO mission.
<table>
<thead>
<tr>
<th>Case</th>
<th>Mean time between dual collects (min)</th>
<th>Max time between dual collects (min)</th>
<th>Min time between dual collects (min)</th>
<th>Number of dual collects involving CLARREO assets</th>
<th>Number under 15 minutes</th>
<th>Number under 30 minutes</th>
<th>Total number of requests</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>158.764</td>
<td>432.52</td>
<td>1.43</td>
<td>57</td>
<td>32</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>157.9006</td>
<td>434.52</td>
<td>1.43</td>
<td>57</td>
<td>33</td>
<td>6</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>157.7971</td>
<td>434.52</td>
<td>1.43</td>
<td>58</td>
<td>33</td>
<td>6</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>155.7727</td>
<td>432.52</td>
<td>1.43</td>
<td>58</td>
<td>33</td>
<td>6</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>155.0118</td>
<td>432.52</td>
<td>1.43</td>
<td>58</td>
<td>33</td>
<td>6</td>
<td>12</td>
</tr>
<tr>
<td>6</td>
<td>143.1196</td>
<td>426.97</td>
<td>1.43</td>
<td>56</td>
<td>36</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>7</td>
<td>141.3279</td>
<td>426.97</td>
<td>5.88</td>
<td>58</td>
<td>31</td>
<td>5</td>
<td>13</td>
</tr>
<tr>
<td>8</td>
<td>148.1547</td>
<td>434.52</td>
<td>10.38</td>
<td>57</td>
<td>23</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>9</td>
<td>147.6109</td>
<td>432.52</td>
<td>10.38</td>
<td>55</td>
<td>36</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>10</td>
<td>154.0321</td>
<td>432.52</td>
<td>10.38</td>
<td>54</td>
<td>27</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>11</td>
<td>162.3577</td>
<td>468.55</td>
<td>10.38</td>
<td>52</td>
<td>23</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>12</td>
<td>162.5692</td>
<td>468.55</td>
<td>10.38</td>
<td>52</td>
<td>23</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>13</td>
<td>155.7136</td>
<td>468.55</td>
<td>1.4</td>
<td>54</td>
<td>27</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>14</td>
<td>151.6358</td>
<td>468.55</td>
<td>4.4</td>
<td>54</td>
<td>27</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>15</td>
<td>152.7691</td>
<td>468.55</td>
<td>10.38</td>
<td>54</td>
<td>27</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>16</td>
<td>116.5994</td>
<td>432.52</td>
<td>4.8</td>
<td>56</td>
<td>38</td>
<td>4</td>
<td>13</td>
</tr>
<tr>
<td>17</td>
<td>112.7643</td>
<td>432.52</td>
<td>10.38</td>
<td>56</td>
<td>42</td>
<td>2</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 5-9: Case-by-Case Results

We discuss these results in the subsequent sections.

5.6.1 Varying Inclinations

We first analyze the impact of changing the inclination of the orbits. This involves comparing the following cases with each other:

- 2 vs 7 vs 9
- 6 vs 8 vs 10
Case 7 (74° inclination) increases the total number of observations made, with no significant loss in number of dual collects made. Case 9 (sun-synchronous), however, increases the total number of observations made with a visible decrease in the number of dual collects made. We see similar behavior in the cases of 6, 8, and 10. However, when we add another observatory, the Case 16 (74° inclination) output indicates that we are able to increase the number of dual collects taken and the total number of observations significantly.

5.6.2 Varying RAAN

Analyzing the impact of changing the RAAN of the orbits involves comparing the following cases:

2 vs 6
7 vs 8
9 vs 10

We see that Case 6 (RAAN difference 60°) shows an improvement in CLARREO involvement in dual collects, at the cost of fewer overall targets observed. Case 8, however, sees a large drop-off in CLARREO involvement (versus Case 7) in dual collects as a result of the interaction between the inclination and the RAAN difference chosen.

5.6.3 Dual Collect Inter-Observation Times

This section focuses on the dual collects made in the scenario. It is important to consider the time elapsed between each portion of a dual collect because, for calibration purposes, two observations must be made at nearly the same time in order to ensure that the two measurements can be compared to each other. We present the following histograms for each case below:
Here, the x-axis has units of minutes, and the y-axis has units of the number of dual collects made. Thus, each bar represents the number of dual collections made with a specific time between the observations making the collection. This figure shows that, in most cases, a spike exists in the number of collections made within 15 minutes of each other. This is indicative of the fact that the planner does its best to find dual collections as soon as they are available. Other spikes exist where two satellite assets have feasible opportunities over the same area spaced at this time, with nearly equal periods. That is, spikes correspond to the fact that the second asset’s orbit precesses over a target area $t$ minutes after the first asset (NPOESS and CLARREO1 cause the spike at $t = 200$ minutes, for example).

However, because we assumed no time windows on these targets, the planner scheduled dual collects that are many minutes, sometimes hours, apart, which is not very useful in
practice. To account for this, we can generate data sets with dual collections that are required within a specified length of time of each other. We demonstrate this in the following section.

5.7 Dual Collection Scenario

This section describes the results of a scenario in which we engineer a dataset to contain all of the possible dual collections available to the system in the upcoming planning horizon. We have stated on numerous occasions that one of the primary benefits for a coordination manager would be its ability to take advantage of simultaneous observation opportunities across different sub-planners. In this section, we show how the coordination planner we have developed can be used to coordinate observations of this variety.

We use the same 6 sub-planners (1 UAV sub-planner, 5 satellite sub-planners), and consider a 10 hour planning horizon. Before the scenario begins, we generate a dataset that includes every Simultaneous Nadir Overpass (SNO) and every feasible Simultaneous Conical Overpass (SCO) possible, where we deem a feasible dual collection to consist of separate observations of the same location at most 15 minutes apart. The scenario consists of 122 possible dual collections, some of which exist at very similar locations, as these locations tend to be at high latitudes, given the inclinations of the orbits being considered. Thus, we cannot expect all 122 collections to be made, as the scenario includes taskable assets that can only view a subset of the possible locations at one time.

After running the scenario, we find that 22 complete dual collections were made, with another 57 single observations made at other locations of interest. The figure below shows the distribution of the inter-observation times for these collections:
As expected, the inter-observation times lie between 0 minutes and 15 minutes. In fact, 2 collections are overlapping; that is, portions of them are simultaneous collections.

5.8 WRAP Scenario Analysis

This section focuses on the WRAP scenario introduced in Section 2.3.2. This scenario involves using remote sensing assets to collect data on developing and ongoing forest fires in the western United States. Thus, we generate a scenario with air and space assets and forest fire targets located in the western U.S. The analysis focuses on the marginal benefit of additional UAVs, but we consider the impact of additional UAVs while varying the number of satellite-based assets involved in the scenario as well.

We create a dataset with a grid of 186 targets located in the western U.S., as shown below:
The grid-like target set is indicative of a scenario in which the users in the scenario, whether they are environmentalists, scientists, or firefighters considering the best locations to fight the fires, desire observations uniformly throughout the areas of interest. We consider this request set in each instance we run the coordination planner. Each run varies either the number of UAVs, number of satellites, or both the number of UAVs and satellites in the scenario. Figure 5-13 shows the results of each run:

![Figure 5-12: WRAP Scenario Target Locations](image)

![Figure 5-13: Marginal Benefit of Additional Assets, WRAP Scenario](image)
In this case, we see that the additional observatory significantly impacts the quality of the plan in terms of the total priority of the targets observed by the plan. On average, the additional observatory increases the quality of the plan by 17%. This number is weighed down by the data point where there are 4 UAVs, where the percentage increase from the additional observatory is only 9%.

We also notice a roughly linear increase in total priority of the plan when the number of UAVs is increased from 1 to 3, independent of the number of satellites in the scenario. However, the benefit becomes slightly nonlinear beyond that, depending on how many satellites are present in the scenario. If we consider the increase approximately linear (for \( num\text{UAVs} = 1...5 \)), then the marginal benefit is \( \frac{1.050}{uAV} \) for 2 observatories, and \( \frac{1.625}{uAV} \) for 3 satellites, in terms of total priority. Of course, while each of these increases in plan value is highly significant, the cost of additional satellites or UAVs is quite high.
6 Conclusions and Future Work

The purpose of this chapter is to summarize the contributions of this thesis, provide a way forward for future research in this area, and state the conclusions we make from this work. The first section summarizes the contributions made by this thesis. The second section describes various possible additions to our planner and suggests different perspectives from which to address the problem. The third section presents our conclusions.

6.1 Summary of Contributions

In this section, we review each of the contributions made by this thesis, as stated in Chapter 1. The goal of our work was to formulate an algorithm that enabled a user of the coordination planner to use sub-planners to create valuable observation plans. In that line, we review each contribution made in the development of our approach in the following paragraphs:

- A complex value function that is used as an intermediate construct to build observation plans that align with user end objectives. Our value function accounts for many of the constraints of the problem and can be tuned to emphasize any of four user Measures of Performance (MOPs). The Remaining Feasible Opportunities (RFO) component makes the function forward-looking and encourages the coordinator to grant higher value to those requests with fewer opportunities remaining. Moreover, the function is built such that additional parameters of
interest, such as resolution, sun-angle, or sub-planner traffic can be smoothly incorporated.

- **An optimization problem that solves the problem of intelligently querying sub-planners at any instance in time and can be solved as a Linear Programming (LP) problem.** We use the values generated by the value function in the objective function of an optimization problem that we solve repeatedly over time. The optimization problem can be thought of as a modified assignment problem that we solve at each iteration of communication with the sub-planners to gain the most pertinent information from the sub-planners at the current time. We show that these problems can be solved as LP problems as a result of the structure of the constraint matrix.

- **Integration of the planning algorithm with two types of sub-planners, an Unmanned Aerial Vehicle (UAV) sub-planner and a satellite sub-planner.** The algorithm is embedded in software controlled in MATLAB. The coordination planner software interacts with a UAV sub-planner (coded in Java) and a satellite sub-planner (coded in MATLAB), each of which generate observation plans with their own planning algorithms. The coordination planner interprets the plans and updates the value function according to the feedback it receives.

- **Empirical and statistical testing and analysis of the value function.** To test the validity of our claims that each component of the value function is included for a specific purpose, we perform empirical and statistical analysis of the value function on various problem instances to identify relationships between each value function component and the four MOPs. We find that each component is sensibly correlated with corresponding MOPs, and that using our statistical models to optimally tune the objective function input weights is effective.

- **Development and testing of operational scenarios to demonstrate the effectiveness of the algorithm we develop and to demonstrate the benefit of coordinated planning, in general.** We create plans for a notional CLARREO mission and a notional western United States wildfire scenario. We also quantify the benefit of communication with the sub-planners, showing that feedback can improve plans
significantly, but that there are diminishing marginal returns for increasing communication beyond a point. The output is usually shown in tabular format.

- **Recommendations for future work on the coordinated planning problem.** We identify areas for improving our planner, including ways to incorporate additional observation quality metrics, weather forecasts, and sub-planner traffic data. We also discuss other approaches to this multi-faceted problem, including auction algorithms and call-center routing approaches.

6.2 **Future Work**

In this section, we review a number of areas pertaining to the Coordination Planner Problem (CPP) that remain to be addressed. They include incorporating humans into the planning process, adding fidelity to the model, and addressing the problem with other Operations Research (OR) techniques.

6.2.1 **Opportunities for Human Interaction**

A human user would presumably control the coordination planner and monitor its performance. A valuable addition to our algorithm, then, would be the ability to allow a human to enter the planning loop and provide input. Human interaction with a planning process can provide an additional check of the validity of plans, allow users to choose request-asset pairs directly, and can allow users to introduce bias easily.

A specific example of human input in the context of this thesis could consist of simply allowing users to ensure certain targets are queried on certain sub-planners. Other input could include allowing a human to evaluate the new plans that are generated by sub-planners manually to decide if the coordination planner should accept the new allocation. Yet another example of human interaction could include relaxing constraints on requests, such as time windows. Our framework assumes hard time windows, but observing some targets slightly beyond their desired time windows could enable assets to observe additional targets with value higher than that lost by failing to observe other requests entirely, thus increasing the overall value of final observation plans.
6.2.2 Adding Fidelity to the Coordination Planner

Currently, the value function takes only seven parameters as input. It is evident that there are many opportunities to include additional data in the value function to improve the coordination planner’s performance. This section describes possible additions that could be made, both to the value function and to the coordination planner algorithm as a whole. We hypothesize that these editions could be incorporated into our model without an overhaul of our problem structure and approach.

6.2.2.1 Quality of Observations

This subsection discusses additions that could be made to improve the fidelity at which we measure the quality of the observations the sub-planners generate. In that line, the quality of potential observations is largely a function of the cloud cover in the area being observed. Thus, a very useful addition to the value function would be a component that incorporates cloud cover forecasts into its formulation. The component could be in the form of a probability of cloud cover at the associated location and time, in which case the coordination planner would grant more value to requests that have a low probability of having clouds present at the associated location. Alternatively, the algorithm could screen for observations that have a forecasted chance of cloud cover below some threshold and only grant value to those observations. The inclusion of this parameter should improve the average quality of the observations made by the individual assets.

Another potential addition to the value function could be the sun-angle of each feasible opportunity. The angle at which the sun is incident to a target location at the time the target is observed can greatly affect the quality of the observation. Excessive glare, for example, can ruin an image’s worth to a user.

It can also be important to consider the time of the day at which a potential observation will be made. The quality of an observation is a function of the time of the day at which the observation is taken for a number of reasons. For instance, an electro-optical image has much more value if taken in daylight when the target is illuminated, rather than at nighttime, when visibility is low. Similarly, an infrared sensor performs best when the location surrounding its target is cool, because too much heat surrounding a target can mask the target’s infrared signature. Thus, infrared sensors perform best at night. The sun-angle and time of day at which
a possible observation will take place could be incorporated into our algorithm through inclusion in the $\overline{TSV}$ component, or in separate components.

Also, it is very important to consider the resolution of potential observations. In current operations, it is often the case that users are allowed to input minimum resolution requirements for their requests. Thus, this is an important piece to include in the coordination planner. Again, this component could be included in the $\overline{TSV}$ component, or as a separate component altogether.

### 6.2.2.2 Sub-Planner Traffic

A final addition to the value function is a measurement of each sub-planner’s ability to include additional requests into its plans. Currently, the coordination planner performs this task implicitly. That is, it does so through the measure of feasibility component, $\overline{d}$, and through communication with the sub-planners. A useful measurement of a sub-planner’s ability to include additional requests into its plans would likely require more than simply the number of targets expected in the sub-planners’ target set. It would also require more extensive measures of asset capabilities, including data storage/upload/download capabilities, and possibly the location of the targets each sub-planner is considering. In fact, this information is best incorporated through statistical analysis of individual sub-planners, as will be discussed in Section 6.2.3.

### 6.2.2.3 Planning Cycles

Another addition that would be of benefit to the coordination planner would be expanding the coordinator’s ability to include more detailed planning cycles in its planning. Currently the software handles only planning and sending/uploading phases, with the start and end times of execution phases calculated based on these two inputs. For any single sub-planner system, our current system handles only repeating planning cycles in which each planning phase is of identical length, and each sending/uploading phase is of identical length. More realistically, the software should be able to account for varying lengths of time for each of these phases, and for additional phases such as maintenance phases.
6.2.2.4  *Inter-Objective Dependencies*

In this section, we discuss complex target relationships that should be incorporated in the future. We refer to these complex dependencies among targets as *Inter-Objective Dependencies*. This term refers to the many complex relationships that may exist among different requests that are sent to the coordinating manager. One example of a relationship between requests is a dual collection, which we model in this thesis. However, many other relationships exist and it is important to consider how these relationships may be accounted for. The remainder of this section lists these inter-objective dependencies, identifies those which are accounted for in the latest version of our planner, and describes how those unaccounted for could be incorporated in the future.

**Capable of being handled:**

1. Baseline case – observe A, if possible, with any of the available assets
2. Observe A and B at the same time with different assets
3. Observe A and B desired effect
   - To distinguish from (2), time windows need not be identical

**Cannot be handled:**
4. Observe A or B, but not both
   - To incorporate (4):
     - Upon receiving feedback that A can be observed, the value function must set $v_{BP} = 0, \forall p$ (and vice-versa)
     - Initial values need not be computed differently, and assignments need not be constrained
5. Observe B at least $x$ seconds after completing the observation of A
   - To incorporate (5):
     - Suboptimal method: $ETW_B = LTW_A + x$
     - "More" optimal method: Wait until you know A is planned for, then make $ETW_B = LTW_A + x$
6. Observe B immediately after A
   - To incorporate (6):
     - Suboptimal method: $ETW_B = LTW_A$
     - "More" optimal method: Wait until you know A is planned for, then make $ETW_B = LTW_A$
     - Other option: Compute feasible opportunities before generating time windows. Find end times of possible observations for A and make the $ETW_B$ equal to this value
7. Observe A with Asset 1, 2 or 3 only
8. Observe A and B, or C and D, but not both sets
(9) Observe A, B, and C, or else none of them
(10) If A is planned for observation, plan for B
(11) Observe B during the observation of A
   • To incorporate (11):
     • Set $ET_W_A \leq ET_W_B \leq LT_W_B \leq LT_W_A$ (this can be done dynamically)
(12) Combinations of inter-objective dependencies

We note that each inter-objective dependency would require additional XML inputs alerting the coordination planner to their presence. We also conclude that our formulation, coupled with the ability to dynamically update values, allows for a construct in which many of these dependencies can be modeled by breaking a single request into multiple targets, by tweaking data fields (i.e., time windows) and/or by using the value function. However, it may be easier to design a more subtle approach to the coordination problem in order to smoothly account for all of these complex request relationships.

6.2.3 Statistical Studies of Real-World Planners

Currently, we assume that much of the planning methodologies and tendencies of sub-planners are invisible to the coordination planner. Thus, our algorithm does not take data such as the request traffic on each sub-planner as input. While this is certainly a simplifying assumption, it is also a realistic one, at least in some cases. This is due to the fact that, in practice, some sub-planners would be unwilling to provide this information to the coordinating agent. For these sub-planners (and even those with increased visibility into their planning tendencies), we propose that the coordination planner could perform statistical data analysis to improve its situational awareness.

One approach would be to create a statistical model, called a classifier, that predicts whether or not a particular target will be incorporated into a particular sub-planner’s plans at a certain time. Another approach, however, would be to create a predictor that predicts the number of targets that will be included in a sub-planner’s plans, given a set of targets as input. We propose that a predictor of this type would require the following inputs:

- Total number of requests
- Average Euclidean distance to nearest neighbor, across all requests
- Variance of Euclidean distance to nearest neighbor
- Average value of requests
- Major/Minor-axes of the ellipses that envelop the requests
- Maximum/Minimum distance of a request from UAV starting locations (UAV planner)
- UAV attributes – average speed, climb/sink rates, floor/ceiling, etc.
- Satellite attributes – average slew rate, maximum slew
- Algorithm-specific attributes
- Time of the year
  - Season, month, day, hour, etc.

The predictor could take these attributes as input, and predict which targets will be incorporated into the plans by each sub-planner. The coordination planner could incorporate these statistical models through a Robust Optimization (RO) formulation that chooses which tasks to query on which sub-planners to maximize the expected value of the targets that will be observed by the sub-planners. Given the types of inputs we suggested above, the optimization problem would likely be non-linear and possibly non-convex and poses a difficult but potentially very useful problem for future work.

6.2.4 Time-Delay Analysis

Realistically, most users who approach their preferred assets with a request and are rejected either wait until the request is accepted by their preferred asset, or approach another system to obtain their data collection. In either case, the user experiences a time delay. We hypothesize that, in addition to the four MOPs analyzed in this thesis, coordination should provide a reduction in the average time delay that users experience in trying to have their request serviced.

However, measuring this decrease in time delay is very difficult to capture in a mathematical model. That is, it is difficult to model human user behavior upon receiving feedback concerning the rejection of a past request. This would require a very high-fidelity model with much additional software.

6.2.5 Auction Algorithms

One part of real-world operations left out of our models includes a capability to allow users to pay some fee to increase their chances of being inserted into a certain asset’s observation plan. ASTER, for example, is a taskable asset that charges fees for requests for new data collections. Thus, it would be useful to incorporate cost into the planning framework. More specifically, we could build planner that conducts an auction between individual users and sub-planners, allowing each user to bid for the available resources. To model this, auction algorithms are often used.
An auction algorithm for solving the assignment problem is discussed in [52], in which the algorithm uses the dual problem to iteratively move toward the optimal solution. In [58], the authors use an auction method where, at each iteration, the central planner sends a task to each rover, the rovers plan to include the new task, and the rovers send a bid back to the coordinating agent in the amount the rover would need to travel in order to include the task. The coordinator then allocates the task to the rover that bid the lowest amount. However, in this context, we could think of the problem slightly differently.

To incorporate an auction algorithm in our problem's context, we would require additional assumptions concerning the type of feedback each sub-planner is capable of giving. Instead of simply a yes/no response, if the coordinator could receive some measure of the sub-planner's willingness to include the target in its next plan, then the coordinator could use this feedback to more intelligently update the value function. This feedback could be in the form of the distance its assets need to travel to view the target, given its current plan, or it could be in the form of the dual variables from the sub-planner's planning algorithms.

6.2.6 Call-Center Routing

The coordination planner has the mission of choosing which requests to allocate to which assets such that the benefit is maximized. One way to think of this problem is as an assignment problem (requests to sub-planners), where there is some benefit for assigning each person to a job. In a real-world problem, the coordinator might have insight into how busy the assets' schedules are at any time in addition to the benefit gained by actually servicing the request with a certain asset. Another way of looking at the problem, then, could be to view the coordination planner as a call-center and the sub-planners as agents. We could then address the problem as a call-center routing problem. This type of problem involves calls arriving to the center according to some random process, and the router is tasked with allocating the calls to the servers while considering how busy each server is to minimize the total wait time for each call. Particularly, we could view the problem by considering a system of queuing systems. Thus, the problem is well-framed by both queuing and optimization theory.

6.3 Conclusions

This thesis has addressed the problem of coordinated planning of air- and space-based sensor observations in an asynchronous and distributed environment. We have detailed a
complex value function that can be tuned to align with user end objectives, while accounting for many of the physical constraints of the problem. We presented a simple optimization problem that can be solved as an LP to intelligently query sub-planners. Our analysis shows that coordinated planning results in great benefits to users, in terms of the number of targets observed, the quality of those observations, the number of dual collections made, and the priority of the requests serviced.

We conclude that our algorithm is a simple yet effective and practical method to coordinate observations among multiple, heterogeneous assets in air and in space. The methodologies employed are quite flexible and provide a construct that allows for much fidelity to be added to the model. Moreover, through our experiments, we have found that coordinated planning could provide a great benefit to the Earth Science and intelligence communities. While this work is a coarse look at the benefit of coordinated planning, it is our hope that this work has provided one of the first approaches to a complicated planning framework with applications to many important fields.
Appendix A – List of Acronyms

AFB – Air Force Base
AIRS – Atmospheric Infrared Sounder
AIST - Advanced Information Systems Technology
AOR – Area of Responsibility
ASPEN - Automated Scheduling and Planning Environment
ASTER - Advanced Spaceborne Thermal Emission and Reflection Radiometer
BDA - Battle Damage Assessment
CALIPSO - Cloud-Aerosol Lidar and Infrared Pathfinder Satellite Observation
CALIOP – Cloud-Aerosol Lidar with Orthogonal Polarization
CCDR – Combatant Commander
CERES - Clouds and the Earth’s Radiant Energy System
CIA – Central Intelligence Agency
CLARREO - Climate Absolute Radiance and Refractivity Observatory
CNES - Centre National d’Études Spatiales
COPA – Composite Operations Planning Algorithm
CP – Constraint Programming
CPP – Coordination Planner Problem
CrIS - Crosstrack Infrared Sounder
DAR - Data Acquisition Request
DIA – Defense Intelligence Agency
DTO – Data Take Opportunity
ECI - Earth-Centered Inertial
EEI - Essential Elements of Intelligence
EO – Electro-Optical
EO-1 - Earth Observing-1
EOS – Earth-Observing System
EPOS - Earth Phenomena Observation System
ERB – Earth’s Radiation Budget
ESA – European Space Agency
ESTO – Earth Science Technology Office
ETW – Early Time Window
FOM - Figure of Merit
FOV – Field-of-View
GA - Genetic Algorithm
GEOINT - Geospatial Intelligence
GLOPAC - Global Hawk Pacific
GOES – Geostationary Operational Environmental Satellites
HASOC - Heuristic Air and Space Operations Coordinator
HUMINT - Human-Source Intelligence
IC – Intelligence Community
IMINT – Imagery Intelligence
IP – Integer Programming
IPCC – Intergovernmental Panel on Climate Change
IR - Infrared
ISR – Intelligence, Surveillance, and Reconnaissance
JAXA – Japan Aerospace Exploration Agency
JFC - Joint Force Commander
LEO – Low Earth Orbit
LOS – Line-of-Sight
LP – Linear Programming
LTW – Late Time Window
MASINT - Measurement and Signatures Intelligence
MDP – Markov Decision Process
MIP – Mixed-Integer Programming
MODIS – Moderate Resolution Imaging Spectroradiometer
MOP - Measure of Performance
MOPITT - Measurements of Pollution in the Atmosphere
NASA - National Aeronautics and Space Administration
NGA - National Geospatial-Intelligence Agency
NIIRS - National Imagery Interpretability Rating Scale
NOAA – National Oceanic and Atmospheric Administration
NPOESS - National Polar Orbiting Operational Environmental Satellite System
NPP – NPOESS Preparatory Project
NRC – National Research Council
NSA – National Security Agency
OCO – Orbiting Carbon Observatory
OGC - Open Geospatial Consortium
OLS – Ordinary Least Squares
OPS – One Pass Scheduler
OR - Operations Research
OSINT - Open-Source Intelligence
PARASOL - Polarization & Anisotropy of Reflectances for Atmospheric Sciences coupled with Observations from a Lidar
POLDER - Polarization and Directionality of the Earth’s Reflectances
POMDP – Partially Observable Markov Decision Process
RAAN - Right Ascension of the Ascending Node
RFO - Remaining Feasible Opportunity
RO – Robust Optimization
RSS – Residual Sum of Squares
SAR – Synthetic Aperture Radar
Seq – Sequential Scheduler
SCO – Simultaneous Conical Overpass
SIGINT – Signals Intelligence
SNO - Simultaneous Nadir Overpass
SP – Stochastic Programming
SPOT – Satellite Pour l’Observation de la Terre
SPPRC - Shortest Path Problem with Resource Constraints
SPS - Sensor Planning Service
TLE – Two-Line Element
TRMM – Tropical Rainfall Measuring Mission
TST – Time Sensitive Target
TU – Totally Unimodular
UAV – Unmanned Aerial Vehicle
USFS – United States Forest Service
USV – Unmanned Surface Vessel
UV – Ultraviolet
VIIRS - Visible/Infrared Imager Radiometer Suite
VRPTW – Vehicle Routing Problem with Time Windows
VVCSP - Valued Variable Constraint Satisfaction Problem
WRAP - Wildfire Research and Applications Partnership
## Appendix B – Target-to-Sensor Value Lookup Table

<table>
<thead>
<tr>
<th>Target Types</th>
<th>Sensor Types</th>
<th>EO</th>
<th>IR</th>
<th>SAR</th>
<th>CERES</th>
<th>ASTER</th>
<th>CLARREO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forest Fire</td>
<td></td>
<td>30</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>60</td>
<td>70</td>
</tr>
<tr>
<td>Volcano</td>
<td></td>
<td>50</td>
<td>30</td>
<td>30</td>
<td>60</td>
<td>65</td>
<td>65</td>
</tr>
<tr>
<td>Hurricane</td>
<td></td>
<td>40</td>
<td>50</td>
<td>70</td>
<td>70</td>
<td>40</td>
<td>75</td>
</tr>
<tr>
<td>Intelligence Target</td>
<td></td>
<td>60</td>
<td>70</td>
<td>65</td>
<td>35</td>
<td>40</td>
<td>40</td>
</tr>
</tbody>
</table>
### Appendix C – Planning Cycles Calculations

<table>
<thead>
<tr>
<th>Calculated Data</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E_p = \left\lceil \frac{H}{PL_p + UL_p} \right\rceil$</td>
<td>The number of planning cycles over the planning horizon $H$ for planner $p$.</td>
</tr>
<tr>
<td>$</td>
<td>A</td>
</tr>
<tr>
<td>$e^a_p$</td>
<td>Execution phase of sub-planner $p$ in planning period $a, E^a \subseteq P$.</td>
</tr>
<tr>
<td>$EL_p = PL_p$</td>
<td>The length of execution phases for planner $p$.</td>
</tr>
<tr>
<td>$PS_{pe} = 0$</td>
<td>The start time of planning phase $e$ of planner $p$, $e = 1, \forall p \in P$.</td>
</tr>
<tr>
<td>$PS_{pe} = (e - 1)(PL_p + UL_p)$</td>
<td>The start time of planning phase $e$ of planner $p$, $e = 2..E_p, \forall p \in P$.</td>
</tr>
<tr>
<td>$PE_{pe} = PS_{pe} + PL_p$</td>
<td>The end time of planning phase $e$ of planner $p$, $e = 1..E_p - 1, \forall p \in P$.</td>
</tr>
<tr>
<td>$PE_{pe} = \min {(ePL_p + (e - 1)UL_p), H}$</td>
<td>The end time of planning phase $e$ of planner $p$, $e = E_p, \forall p \in P$.</td>
</tr>
<tr>
<td>$US_{pe} = PE_{pe}$</td>
<td>The start time of upload phase $e$ of planner $p$, $e = 1..E_p, \forall p \in P$.</td>
</tr>
<tr>
<td>$UE_{pe} = e(PL_p + UL_p)$</td>
<td>The end time of upload phase $e$ of planner $p$, $e = 1..E_p - 1, \forall p \in P$.</td>
</tr>
<tr>
<td>$UE_{pe} = \min {e(PL_p + UL_p), H}$</td>
<td>The end time of upload phase $e$ of planner $p$, $e = E_p, \forall p \in P$.</td>
</tr>
<tr>
<td>$ES_{pe} = UE_{pe}$</td>
<td>The start time of execution phase $e$ of planner $p$, $e = 1..E_p, \forall p \in P$.</td>
</tr>
<tr>
<td>$EE_{pe} = US_p(e+1)$</td>
<td>The end time of execution phase $e$ of planner $p$, $e = 1..E_p - 1, \forall p \in P$.</td>
</tr>
<tr>
<td>$EE_{pe} = \min {H, US_p(e+1)}$</td>
<td>The end time of execution phase $e$ of planner $p$, $e = E_p, \forall p \in P$.</td>
</tr>
</tbody>
</table>
THIS PAGE INTENTIONALLY LEFT BLANK
Appendix D – Total Unimodularity Proof

A totally unimodular (TU) matrix is, by definition, an integer matrix for which every square submatrix formed from it has a determinant of -1, 0, or +1. Our polytope (from LAF below), is of the form \( R(A) = \{ x : Ax \leq b, x \geq 0 \} \). If \( A \) is TU, then all the vertices of \( R(A) \) are integer for any integer vector \( b \) [61].

Below we list properties of TU matrices that we will use for the proof:

1. If \( A \) is TU then \( [A] \) is TU, where \( I \) is the identity matrix [68]
2. Row and Column swaps in \( A \) do not effect TU [68]
3. Sufficient condition for TU: For an \( m \times n \) matrix, for every \( J \subseteq N = \{1, \ldots, n\} \) there exists a partition, \( J_1, J_2 \) of \( J \) such that

\[
\left| \sum_{j \in J_1} a_{ij} - \sum_{j \in J_2} a_{ij} \right| \leq 1, \forall i = 1..m, [69]
\]

Integer Formulation (Binary Integer Assignment Formulation (BIAF)):

\[
\text{BIAF: maximize } \sum_{r=1}^{[R^{ka}]} \sum_{p=1}^{[P]} \hat{u}_{rp} x_{rp}^{ka} \quad (D-1)
\]

subject to

\[
\sum_{r=1}^{[R^{ka}]} x_{rp}^{ka} \leq \theta_p, \quad \forall p \in P \quad (D-2)
\]

\[
(1 - S_a^{\sigma(r_1, r_2)p}) (x_{r_1p}^{ka} + x_{r_2p}^{ka}) \leq 1, \quad \forall (r_1, r_2) \in D, p \in P \quad (D-3)
\]

\[
x_{rp}^{ka} \in \{0,1\}, \quad \forall r \in R^{ka}, p \in P \quad (D-4)
\]

If we relax integrality (Linear Assignment Formulation (LAF)):

\[
\text{LAF: maximize } \sum_{r=1}^{[R^{ka}]} \sum_{p=1}^{[P]} \hat{u}_{rp} x_{rp}^{ka} \quad (D-5)
\]

subject to

\[
\sum_{r=1}^{[R^{ka}]} x_{rp}^{ka} \leq \theta_p, \quad \forall p \in P \quad (D-6)
\]

\[
(1 - S_a^{\sigma(r_1, r_2)p}) (x_{r_1p}^{ka} + x_{r_2p}^{ka}) \leq 1, \quad \forall (r_1, r_2) \in D, p \in P \quad (D-7)
\]

\[
0 \leq x_{rp}^{ka} \leq 1, \quad \forall r \in R^{ka}, p \in P \quad (D-8)
\]

For the proof, we will refer to the constraint matrix for LAF as \( A \).

To prove that the constraint matrix of LAF (modified assignment problem with side constraints) is TU, we proceed as follows:

We define the matrices \( A', A'', \) and \( A''' \) such that they correspond to each set of constraints in LAF:

\[
A = \begin{bmatrix} A' \\ A'' \\ A''' \end{bmatrix} \rightarrow \begin{bmatrix} (D - 6), \text{ summation over requests, for each sub-planner} \\
(D - 7), \text{ for each request, for each sub-planner} \\
(D - 8), \text{ flow must be between 0 and 1} \end{bmatrix}
\]

181
1. We first note that the constraints restricting each decision variable to take value between 0 and 1 (D-8) can be written as the identity matrix, I. By (1), we now only need to show that

\[ \tilde{A} = \begin{bmatrix} A' \\ A'' \end{bmatrix} \]

is TU to show that A is TU. So, we proceed by showing that we can partition the set of columns in \( \tilde{A} \) into two sets such that the sum of the entries in each set differ by no more than 1, for each row (property (3)).

2. Constraints (D-6) in LAF are the same as one set of constraints from an assignment problem. The form of these constraints is a matrix \( A' \) with a single entry of 1 in each column, and \( |R^{ka}| \) entries of 1 in each row. Thus, for any row, the sum of the entries in each of two sets of columns can always be made to differ by no more than 1. Specifically, there is zero difference if \( |R^{ka}| \) is even, and a difference of 1 if \( |R^{ka}| \) is odd.

3. All entries in \( A'' \) are 0 (corresponding to requests that do not require dual collection, or requests requiring dual collection and having an opportunity for that dual collection) or 1 (requests requiring dual collection, lacking an opportunity for that dual collection).

4. The columns \( a''_1 \) and \( a''_2 \) corresponding to related requests (same location, observation required by different assets) are always identical.

4a. The columns \( \tilde{a}_1 \) and \( \tilde{a}_2 \) corresponding to related requests (same location, observation required by different assets) are always identical, because they consist of the entries in \( a''_1 \) and \( a''_2 \) plus an additional entry of 1.

5. Because we only consider dual (e.g., simultaneous collection requiring two assets) collects in our model, then each column of \( A'' \) either contains all zeros (single request, or dual collect with a simultaneous opportunity) or it contains a positive number of ones AND there exists an identical column somewhere else in the matrix \( A'' \) (corresponding to the decision variable of the other portion of the dual collect).
6. From 4a. and 5., as long as we place the columns of related requests in different sets, we can guarantee that for each row of $\bar{A}$, the sum of the two sets will differ by 0 if $|R^{ka}|$ is odd, and 1 if $|R^{ka}|$ is even.

7. By (1), $A$ is TU.
## Appendix E – Regression Output

<table>
<thead>
<tr>
<th>MOP:</th>
<th>Average Priority</th>
<th>Average Quality</th>
<th>Observations</th>
<th>Dual Collections</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression Type</td>
<td>Best OLS</td>
<td>Best Ridge (λ = 0.5)</td>
<td>Best OLS</td>
<td>Best Ridge (λ = 11.5)</td>
</tr>
<tr>
<td>Intercept</td>
<td>0</td>
<td>556.05</td>
<td>0</td>
<td>38.38</td>
</tr>
<tr>
<td>Wp</td>
<td>647.27</td>
<td>87.78</td>
<td>37.69</td>
<td>-0.61</td>
</tr>
<tr>
<td>WTSV</td>
<td>544.31</td>
<td>-14.28</td>
<td>39.76</td>
<td>1.39</td>
</tr>
<tr>
<td>wo</td>
<td>566.56</td>
<td>8.03</td>
<td>37.37</td>
<td>-0.89</td>
</tr>
<tr>
<td>Wd</td>
<td>566.14</td>
<td>7.18</td>
<td>38.526</td>
<td>0.17</td>
</tr>
<tr>
<td>WFO</td>
<td>562.67</td>
<td>3.94</td>
<td>37.14</td>
<td>-1.04</td>
</tr>
<tr>
<td>ws</td>
<td>547.4</td>
<td>-11.54</td>
<td>39.84</td>
<td>1.46</td>
</tr>
<tr>
<td>WTW</td>
<td>558.75</td>
<td>0.04</td>
<td>37.58</td>
<td>-0.64</td>
</tr>
<tr>
<td>WTW * WFO</td>
<td>-2.2497</td>
<td>-2.22</td>
<td>0.05</td>
<td>0.12</td>
</tr>
<tr>
<td>Number of Requests in System</td>
<td>-2.2497</td>
<td>-2.22</td>
<td>0.05</td>
<td>0.12</td>
</tr>
<tr>
<td>Horizon</td>
<td>27.249</td>
<td>27.63</td>
<td>0.06</td>
<td>12.84</td>
</tr>
</tbody>
</table>
Appendix F – Paired t-Tests

This appendix shows three paired t-tests conducted to demonstrate the statistical significance of improvements made by adding the random value increase. We use a Paired t-test because we run the coordination planner with and without the randomization module on each data set. So, we have pairs of data points (with/without randomization).

Null Hypothesis: \( \mu_0 = \mu_1 \)
- The average MOP values are equal with and without randomization

Alternate Hypothesis: \( \mu_0 < \mu_1 \)
- The average MOP values are less without randomization (greater with randomization)

Test (data from Table 5-7):

\[ n = 45 \]

\( N_i = \) data point \( i \) with no randomization

\( R_i = \) data point \( i \) with randomization

\( \bar{N}_i = N_i - \bar{N} \)

\( \bar{R}_i = R_i - \bar{R} \)

Test statistic:

\[
t = (\bar{N} - \bar{R}) \frac{\sqrt{n(n-1)}}{\sum_{i=1}^{n}(\bar{N}_i - \bar{R}_i)^2}
\]

Reject null hypothesis if \( t > 1.68, n = 45, 95\% \) confidence

Results:

SIGNIFICANT for \( \text{avgPrior, numRequests, numDualCollects} \)

INSIGNIFICANT for \( \text{avgTSV} \)
References


http://geology.com/nasa/monitoring-volcanoes.shtml


http://clarreo.larc.nasa.gov/docs/CLARREO_Data_from_Decadal_Survey.pdf


http://groups.google.com/group/eol-usgs-tasker-help-page?hl=en


http://fas.org/irp/imint/niirs.htm


http://asterweb.jpl.nasa.gov/


191


