Operational Scheduling of Deep Space Radars for Resident Space Object Surveillance

by

Lindsey Blanks

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

May 2022

Author .....
Sloan School of Management
May 06, 2022

Certified by .....
Hamsa Balakrishnan
Professor, Aeronautics and Astronautics
Thesis Supervisor

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

Operational Scheduling of Deep Space Radars for Resident Space Object Surveillance

by

Lindsey Blanks

Submitted to the Sloan School of Management
on May 06, 2022, in partial fulfillment of the
requirements for the degree of
Master of Science in Operations Research

Abstract

As space becomes increasingly congested and contested, new capabilities of rivals to threaten vital assets and exploit the area for military advantages make it more important than ever for the United States to proficiently track and monitor space traffic and debris. However, currently the system of radars used by the Department of Defense to track objects in deep space operates in a way that is labor intensive, uncoordinated, and inefficient. In this thesis, we address these issues by automating and coordinating the radar scheduling process. We consider several complex radar systems that operate in an asynchronous, distributed environment and target space objects with varying priority levels, time windows, arrival frequencies, and task mission requirements.

We develop a mixed integer program capable of intelligently distributing task requests and building radar slew plans in a way that aligns with user objectives and system characteristics. We solve the optimization problem repeatedly over time, all while receiving and incorporating updated information, new task requests, and available feedback throughout the planning process. We test our methodologies on various tactical military scenarios and show that an optimization-based approach allows us to maintain custody of more space objects, better prioritize high value objects, and reduce operating costs when compared to a baseline greedy algorithm. We conclude that an automatic, centralized way of scheduling is viable and beneficial for use in the Space Situational Awareness (SSA) mission.

Thesis Supervisor: Hamsa Balakrishnan
Title: Professor, Aeronautics and Astronautics

Acknowledgments

Far too many people than I have room to name are responsible for making the last two years such an incredible experience. I feel so blessed that I have been able to learn and work alongside some of the brightest minds in the world, all while having so much fun in the process.

I would first like to thank my amazing MIT advisor, Professor Hamsa Balakrishnan, for her endless support and guidance over the past two years. I am extremely grateful that I have had the opportunity to develop and learn from someone with such incredible commitment, passion, and intellect.

Next, I would like to thank my MIT Lincoln Laboratory advisor, Robert Morrison. Thank you for your incredible patience and help as I came up to speed on all things sensors and space surveillance! Not only did your expert knowledge and feedback make this possible, but your mentorship has been invaluable during my time here.

I would also like to express how immensely appreciative I am to both the United States Air Force and MIT Lincoln Laboratory for allowing me this incredible opportunity. My hope is to take all that I have learned, both academically and personally, and use it to make a positive impact throughout my service.

I cannot express how thankful I am to all of the faculty, staff, and members of the ORC for helping me during these past two years. I am in awe of the talent, work ethic, character, and compassion that I have found in everyone in this department. Starting school during a pandemic definitely came with its challenges, but somehow you all still made me feel like I was part of a big family.

Next, I want to thank my family. Mom, Dad, Lauren, Sammy, and Alicia – I am so blessed to have you all. Thank you so much for your encouragement, advice, and willingness to always pick up the phone when I call (no matter what time zone you're in). I would not be where I am today without the unconditional love and support from all of you.

Most Importantly, I give all thanks to God for all of the blessings and opportunities He has given me so far in my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Congested, Contested, and Competitive - characterization of space power within the United States almost always falls back to these three words. Justifiably so, for space has seen a dramatic transformation in recent decades with technological advancements that have to cheaper, easier ways for actors of all kinds to enter the space arena. The movement towards a more accessible space has resulted in new opportunities as well as new challenges in ensuring secure space operations.

Currently tens of thousands of objects are being tracked in various orbits, with an unknown number of smaller objects and debris dispersed yet too small to be picked up by many modern-day sensors. In the past two years alone, the amount of objects in space has doubled, and by 2030 it is estimated that 150,000 active satellites could be in space [40]. Competition is also on the rise as private companies and government actors located around the globe look to space for prestige as well as a chance for profit. While the government and their regulatory regimes struggle to keep pace with the commercial space sector, the United States national security space enterprise is also facing many challenging questions. Nations and private actors around the world have increasing space capability, and this trend is only expected to continue as launches and providing space services becomes more routine. Although there are international agreements establishing space be kept as a peaceful domain, the ambiguity and lack of enforcement mechanisms when it comes to actually governing behavior makes this extremely difficult. The new capabilities of rivals to threaten space assets and exploit

space for military advantages make it more important than ever for the United States to proficiently track and monitor space traffic and debris.

The purpose of this thesis is to develop an algorithm to plan radar operations for obtaining observations of satellites in the deep space regime - an area we define as being 37,000 km away in the geosynchronous orbit. The algorithm, namely, the Deep Space Radar Scheduler (DSRS), will fully automate radar scheduling in a way that is flexible, centralized, and coordinated among all radar sites in the system. The development of this algorithm addresses three main challenges in deep space radar scheduling. The first challenge is generating plans that include individual heterogeneous radars, with independent planning cycles, mission sets, and users. The second challenge is planning operations which are entirely dependent upon location for several assets in a distributed environment. The third challenge is planning operations in a dynamic environment that must incorporate new information and input from the user, oftentimes on the timescale of seconds.

This chapter will introduce the topics discussed throughout this work. The first section provides an overview of the chapters included in the thesis. The second section summarizes the contributions of this thesis. Finally, the third section states the background and motivation for this work.

1.1 Thesis Overview

This thesis discusses the deep space radar scheduling problem in six chapters. An overview of each of the remaining chapters is given in the following paragraphs:

Chapter 2 explores previous literature related to the deep space radar scheduling problem. Related works involving space assets and sensor systems are reviewed to assist in the construction of the algorithm and the modeling of specific features of the problem. We then survey various scheduling algorithms more generally, with various characteristics and algorithmic approaches. We conclude by exploring the Vehicle Routing Problem in more depth, and show how a variant of this problem's network formulation can be used to model the DSRS.

Chapter 3 introduces the model used for DSRS. We first define the scope of the problem and key assumptions made during the development process. We then go through important considerations and features that must be included in the program and detail how we choose to model them in the final scheduler.

Chapter 4 details the algorithmic approach used in the DSRS. We give details on the inputs, constraints, variables, and objectives of the model. Each of the heuristics and important functions is described in detail, and the final formulation for the Mixed Integer Program (MIP) is presented. The chapter concludes with a discussion of the file inputs and outputs in the scheduler, as well as the software architecture that is used for solving.

Chapter 5 presents the results from the various schedules produced in each of the test scenarios. Specifically, we compare three different scheduling approaches in their achieved run time and Measures of Performance. We then focus on the objective function of the optimization program and show how different user objectives may be emphasized during scheduling. Finally, we use the scheduler on various operationally tactical scenarios to test its performance.

Chapter 6 summarizes the contributions of this thesis. It also provides possible extensions for the deep space radar scheduling problem as well as improvements to the current framework. The chapter then closes with the conclusions found in our work.

1.2 Thesis Motivation

The purpose of this thesis is to develop an algorithm capable of creating schedules for the viewing of target satellites by various deep space radars. The individual radar observation plans should be coordinated across an asynchronous and distributed environment, while satisfying the physical and temporal constraints of the problem. The plans must also be flexible, with the ability to incorporate new information and emphasize various, sometimes changing user objectives throughout the planning cycle. The goal of our work is to develop an algorithm that generates operation schedules

for various deep space radars that allows for the collection of valuable data for the Department of Defense Space Situational Awareness (SSA) mission.

We apply the algorithm to notional operational scenarios including a foreign launch scenario, high number of simultaneous task requests scenario, a radar outage scenario, and a large number of high-priority targets scenario. While our algorithm focuses on scheduling Department of Defense (DoD) radars for deep space viewing, the centralized planning framework could provide benefits for scheduling other assets and observing different orbits within the SSA mission.

1.3 Contributions

In developing an algorithm to address the deep space radar scheduling problem in an automated and centralized manner, this thesis hopes to make the following contributions:

- **An objective function that is capable of building schedules that align with a user's end objectives and system characteristics.** The objective function captures specific user objectives by allowing for weights to be tuned to emphasize any of the three measures of performance (total radar slew time, total task lateness/gaps, total operational cost).

- **An optimization problem that schedules various radars across time and space at any instance in time and can be solved using Mixed Integer Programming.** We create an algorithm that solves a series of optimization problems over time, all while receiving and incorporating new information and available feedback from various radar sites and the user.

- **Empirical testing and analysis of the objective cost function with two scheduling approaches.** We perform empirical analysis of the objective function components by varying the input user weights in the testing of many realistic scheduling scenarios. We also test the model on problem instances of various sizes and make-ups (different number of task requests and the percentage

of task requests at each priority). We use these problem instances to demonstrate how the scheduler would operate on a day-to-day scheduling basis and display how users can emphasize different objectives using the input weights. We also conduct experiments using a baseline greedy scheduling algorithm and an alternative, less centralized version of the DSRS and compare the results.

- **Development and testing of "tactical" operational scenarios to demonstrate the effectiveness of the algorithm in scheduling and rescheduling all deep space radar operations on short notice.** We develop and demonstrate the benefit of an automatic, centralized planner with the ability to receive and incorporate new information with tests on four relevant operational scenarios. Specifically, we use the algorithm to schedule radar operations during a foreign launch scenario, overload of observation requests scenario, a radar outage scenario, and many high-priority task requests scenario. We analyze how these scenarios affect our Measures of Performance when the DSRS is restricted in solve time since these operationally tactical scenarios oftentimes require a fast response.

1.4 Background

This section introduces the operational concept for the DSRS. Lincoln Laboratory has been working alongside the United States military to accomplish various national security objectives in the space sector, including the SSA capabilities in deep space. After describing the importance of this national space policy goal, we describe the current operations and the inefficiencies they present. Next, we discuss how we may use the DSRS within larger contexts in the future, with more sensor assets and different orbital regimes. The third section outlines real-world scenarios which we can directly apply the results of this thesis. Finally, the fourth section presents our goals for this thesis in utilizing the DSRS to address the current scheduling hurdles and inefficiencies.

1.4.1 SSA

Recent decades have seen development of more advanced space capabilities by numerous new actors. As activity in space continues to grow and advance, so too do the number of objects in orbit. As an added challenge, the actual objects entering space are smaller, faster, and more maneuverable than ever before, shrinking the timeline to respond to potentially harmful actions. Due to the incredible energy level that objects sustain in orbit, any impact or collision between these space objects could cause catastrophic damage. Managing the risks posed by this increasingly congested space environment is critical to ensuring the safety and sustainability of space operations [11]. Space services are integrated into everyday life, so a disturbance or outage to a U.S. space asset could have devastating impacts on national security, the economy, and the daily activities of the public. As space activity continues to expand and develop, we can only adequately address these possible threats if our identification and tracking techniques expand and develop in parallel. Early and accurate observation of resident space objects (RSOs) is crucial as it allows capable objects to maneuver away from danger and for the attribution of attacks or negligence to particular actors in order to appropriately respond. To protect space assets and services, the U.S. DoD works to maintain guaranteed custody of certain space objects. The guaranteed custody mission is one that is often resource intensive, because it requires having the most up-to-date information on a space object's kinematic state.

The United States maintains these SSA capabilities with the use of a variety of assets. Included in these important assets is the ground-based radar, one of the sensors responsible for monitoring space beyond 37,000 km from Earth's surface. In this thesis we will focus on how to improve the benefits offered by the deep space radars by improving utilization and coordination among various radar sites. We focus specifically on the DoD's SSA capabilities and their efforts to minimize the risk from space debris to maintain operational readiness. An efficient radar scheduler for observation of the space regime is valuable due to the highly tactical, dynamic nature of the environment, coupled with the vital satellite activities in orbit.

1.4.1.1 Various Deep Space Sensors

A system of ground-based and space-based sensors in the Space Surveillance Network are tasked in a way that allows them to work together to identify and track resident deep space objects. Each of these sensors offers different advantages and disadvantages in the SSA mission.

- *Ground-Based Optical Sensors* - This type of sensor is great at large, synoptic searches because they are able to "sweep" their field of view (FOV) across the sky and gather large numbers of observations at once. However, these sensors are limited in the sensitivity of their observations by optical aperture size - that is, the observation quality is heavily dependent on their physical size. Additionally, these sensors lack the ability to collect accurate range and range-rate information of space objects, often translating to longer observation dwell times for high quality data collection. Finally, optical sensors located on the ground are unable to operate during the daylight hours or when there is significant cloud cover overhead, further limiting their ability to accomplish the guaranteed custody mission alone.

- *Space-Based Optical Sensors* - Another type of optical sensor, the space-based optical sensor, is subject to the same advantages in large searches and disadvantages in sensitivity and range information as the ground-based version of the sensor. Space-based optical sensors do have the advantage of being able to search deep space more frequently as they are not subject to weather or nightly outages. However, these sensors are usually in low-Earth-orbit (LEO) or medium-Earth-orbit (MEO) and are thus still constrained in where and when they may conduct their searches by their orbital periods. In areas such as the solar exclusion zone, an area where the Sun comes extremely close to an optical sensor's line of sight, certain space-based sensors are unable to monitor deep space for a couple of hours, so there are gaps in coverage.

- *Ground-based Radars* - While optical sensors may be able to maintain custody of objects so long as they have little movement and maneuverability, deep space

radars are needed to fill in the gaps and maintain effective custody of RSOs. Deep space radars are able to provide more timely operations, are not limited by weather or time of day, and have an advantage over optical sensors when it comes to fixing and tracking objects in deep space. Ground-based radars are also able to provide much more accurate and efficient range and range-rate information when compared to optical sensors. Radar capacity is one major hurdle in deep space radar operations in the Space Surveillance Network due to the limited supply of this valuable space asset.

Currently, the DoD has three radars capable of searching deep space, and there are potential plans to add a fourth to the system. Efficient scheduling and utilization is needed to ensure the best use of this scarce and critical resource. Only by properly coordinating a network of deep space radars alongside the optical and space-based sensors will we have the ability to collect various forms of valuable data and will continued space awareness and dominance be possible.

1.4.1.2 Space Situational Awareness Functional Capabilities

As was previously stated, the major motivation for the DSRS comes from the desire for complete SSA. SSA in general refers to knowledge of the space environment through tracking and maintaining custody of all space objects. The United States military views SSA as an essential part of conducting missions in outer space. According to Joint Publication 3-14 [41], a document prepared to govern activities and performance of the Armed Forces of the United States, "SSA is a key component for space control because it is an enabler, or foundation, for accomplishing all other space control tasks."

Effective SSA is the only option to truly understanding and characterizing the space domain - it allows for the effective command and control that makes the desired leadership, sustainability, and dominance of space possible [42]. When it comes to maintaining this awareness, SSA is typically divided into the following four functional capabilities: Detect/Track/Identify (D/T/ID), Threat and Warning Assessment (TW&A), Characterization, and Data Integration and Exploitation (DI&E).

Executing these mission sets is essential to be able to accurately characterize the space domain and ensure the safety of U.S. and allied space assets.

- *Detect/Track/Identify (D/T/ID)* - For SSA, the D/T/ID functional capability has the primary goal of ensuring safe flight of U.S. space assets by acting in both offensive and defensive space control missions. This capability is mainly tied to the ability to search, track, maintain custody, and distinguish space objects from one another. Important data about the inventory of space objects and characterizations of space events is given and used directly by decision makers.

- *Threat and Warning Assessment (TW&A)* - TW&A is the ability to predict and characterize both potential and actual attacks. This capability is also responsible for reporting updates on the space weather environment as well as space anomalies that could be potentially harmful to U.S. assets. Because this capability relies entirely on accurate and fast information, it is heavily reliant and tied to D/T/ID. TW&A is responsible for providing advanced warning on potential threats and their impacts to space and non-space infrastructure and capabilities.

- *Characterization* - SSA would not be possible without the ability to characterize activities, intent, tactics, and impacts of activities in space. This provides the military and other decision makers with the knowledge necessary to make informed, confident decisions about allied and adversarial space activities. An important part of the deep space radar mission set is updating orbital characteristics of deep space objects for cataloging.

- *Data Integration and Exploitation (DI&E)* - DI&E can be thought of as the crucial step in SSA where all information from the other functional capabilities is fused together into a single source of relevant information. Because all of the functional capabilities are inevitably connected and dependent on one another, this is the compilation of all of the data and information from these capabilities into one streamlined source. DI&E facilitates decision making and the development of the best possible courses of action for the government.

1.4.1.3 Deep Space Missions

In the effort to provide the best possible SSA capabilities to the United States, a few mission areas are applicable to the deep space radar scheduling problem in particular. Deep space radar sensors are mostly involved in the maintaining custody of RSOs, in this context "custody" is defined to mean continuous knowledge of an object's kinematic state. Observation characteristics such as the revisit rate, dwell time, and priority level are determined by the mission area of the RSO. The following are custody mission areas that have emerged as especially important in the ever-changing deep space environment:

- *High Interest Object (HIO) Monitoring* - This mission is oftentimes referred to as "headcount." It involves the more normal, day-to-day custody mission that a deep space radar is tasked with. The radar will regularly revisit HIOs at a rate determined by the object's priority level and user objectives to maintain custody. If a HIO maneuvers or is not found during the regular headcount, an alert is sent to command and control, and detecting and tracking the object could move to a higher priority. Examples of other objects that could also be considered higher priority even during normal operations are Super High Interest Objects (SHIOs) and Separable Objects (SOs). Both of these objects are typically more maneuverable, smaller in size, and probably contain a larger radar search space. For these reasons, keeping custody of this subset of higher priority HIOs is very resource intensive and drives much of the planning for the radar's observation schedule. It is likely that the future of space will increasingly involve RSOs that are considered higher priority HIOs, so learning to efficiently track and observe these objects in a way that is not a drain on resources is of great importance [5].

- *High Value Asset (HVA) Neighborhood Clearing* - Neighborhood clearing involves searching the space around an important space asset. The deep space radar conducts this search to detect and track any potential debris or adversarial objects that may threaten the space asset. The overall goal of this mission is to

provide timely warning to operators so that an appropriate and swift response may be taken.

- *Foreign Launch Surveillance* - New foreign launches into deep space are an ever increasing high priority event which takes immediate precedence to regular daily tasking. Usually, intelligence and information about a launch arrives from other sensors who then handover the task of searching and tracking the launched object, as well as the task of providing early characterization. We will explore this event in one of the tactical operational scenarios, described in more detail in Section 1.4.1.5.

- *Discovery of Uncorrelated Targets (UCTs)* - The DoD maintains an up-to-date catalog of space objects and their orbital characteristics. When an object is detected that does not correlate with any currently known objects, it is classified as a UCT. While many sensor types may aid in the initial characterization of such objects, certain UCTs require rapid initial orbit determination which is best achieved by systems of radars.

1.4.1.4 Current Operations Framework

The structure of current operations makes coordination among various, dispersed radar sites extremely difficult, if not impossible. The following section describes in detail some of the specific characteristics that make a centralized, coordinated way of scheduling challenging and the inefficiencies that result.

Lack of System-Wide Coordination – Individual radar sites receive a list of satellites to track throughout the day in what is known as the "consolidated tasking list." Currently however, each radar site receives their tasking list for the day from a different agency with no coordination and communication between sites about what objects will be observed elsewhere. This leads to the an inefficient system in which radar sites are functioning independently - even though there is the potential to share data or functionality, this is not the case. Individual missions and observation plans

are managed completely isolated from other sensor sites, leading to a situation where some objects may be over-tracked or over-observed, while others are under-tracked or missed entirely. One reason these systems apply little coordination is the fact that they were constructed at different times, with differing capabilities, and with differing mission sets.

Asynchronous Systems – Another complication resulting from the complete independence of various radar sites is the fact that each sensor has its own planning cycles and operational methods. Individual planning algorithms, or lack thereof, common times for updating observation data, scheduled maintenance or other outage periods, all differ at different radar sites. Additionally, each radar site is responsible for accomplishing various missions in addition to that of SSA. Each radar site may also have their own specific objectives depending on their radar operators and carry out their various missions on their own schedules rather than having one common, unified goal for sensor scheduling.

Lack of Rescheduling Capability - Tasking for the deep space radar systems is currently done completely manually. After receiving the consolidated tasking list for the day, it is up to one of the radar operators to determine the observation path a radar will take to view each satellite. Not only does this take a large amount of time to initially schedule, but it also leaves little to no room for adjusting or completely rescheduling a radar with any new information throughout the planning cycle. New task requests, changes in sensor availability, updated user objectives, or emergencies all may require a change in a sensor's schedule. Incorporating this information without completely deserting an old schedule could be difficult at times, leading to little dynamic capability.

User Communities – Another real-world issue to consider in understanding the deep space radar scheduling problem is the existence of disparate user communities. Each radar site is run and operated by different personnel, so planning may be done in different ways and with different overall objectives in mind. A unified goal and

vision is key for any successful mission, and that is certainly the case in maintaining adequate SSA. Because no one radar site is able to view the entire geosynchronous belt due to FOV and capacity restrictions, maintaining custody of important space assets fails if one radar site is not accomplishing the SSA mission goals. The users and operators at the various radar sites must communicate and coordinate in some way to ensure all mission objectives are indeed accomplished.

1.4.1.5 Future Operations Framework

Currently, we are not realizing the full potential of our deep space radars. In order to keep up with the growing number of objects in space, smaller and more maneuverable than ever, we need to be smarter about how we designate tasks and schedule observations for this vital space asset. As deep space continues to evolve, there is evidence that the role of deep space radars will change as well. Radar sensors already provide valuable data and information to contribute to SSA functional capabilities, but in the future they may be called upon to fulfill even more mission sets and be tasked with more requests than what is seen today.

The ability to schedule tasks on all deep space radar assets in a synchronous manner, where all are subject to the same planning cycle, has benefits for users and may be crucial for the continued success of SSA and future space mission sets. The goal of this thesis is to create an automatic, coordinated way of scheduling deep space radars, even in a distributed and heterogeneous environment. The scheduler would offer several advantages over the current system of scheduling deep space radars.

A fully automated scheduler would shrink the observation planning timeline from its current length of hours down to seconds. This would save time for the radar operator as well as allow for faster response time and flexibility in case of schedule adjustment or rescheduling. If new information or time-sensitive task requests come in during the planning horizon, an automatic scheduler would have more dynamic capability and be able to react swiftly. Additionally, a coordinated, centralized scheduler would have better "hand-off" capability in case of scheduled maintenance or outages at a particular radar site. Coordination and data integration from across all deep

space radar sites would also allow for smarter scheduling by reducing the number of objects that go untracked or over-tracked. Deep space sensors are a powerful but scarce resource, so they must be operated in a way that achieves the best possible results.

Looking Ahead

A centralized, coordinated way of scheduling radar operations opens the door for new applications such as sensor cross-targeting. Simultaneously viewing an object or area in space with sensors in different locations may have a number of benefits. While sensors sometimes offer different capabilities based on their design characteristics (such as frequency), utilizing multiple radars to view the same satellite also allows for additional position information. Most often, radars are very beneficial at determining range information about an object in space but with the ability to make observations from multiple lines of sight, even more may be characterized about an object such as size, shape, object movement, etc. The onset of smaller, more maneuverable targets may bring the need for coordination amongst radar sites to get accurate predictions and observations quickly.

Although we do not address sensor cross-targeting in this thesis, a coordinated way of planning all of the radars in the system is the only way this would be possible. One automatic, centralized scheduler could enable near real-time observation planning for multiple sensor systems at the same time. Overall, deep space sensors could be utilized more efficiently and in a more useful way if there were more coordination between sensor sites.

Eventually, it could also be advantageous to coordinate and schedule all space sensor types and orbit regimes in one central scheduler, and the DSRS program itself may be easily generalized for this broader use. Integrating new data is done quickly due to the fact that most data processing is done in a initialization step beforehand. The optimization algorithm is also very flexible, with constraints that may be easily modified. With that being said, this work focuses on how ground-based radar systems monitor the deep space regime specifically. We also choose to design our DSRS for

use by the DoD in accomplishing SSA for national security objectives in space. The following paragraphs introduce four real-world scenarios that demonstrate how the DSRS could provide increased efficiency as well as play a larger role in the national security mission.

Foreign Launch - Since the famous "Space Race" in the 1950's, space capabilities have grown far beyond what many thought would have been possible. Today, the arena formerly dominated by the United States and Soviet Union has seen a number of countries enter with the capability to launch and function independently. Many more countries will likely enter space in the near future, each one with the hope of being able to compete in the international market and advance national security strategies that are enabled by access to space – "the ultimate high ground." Inevitably there is the concern that other countries could use their space launch capabilities to test ballistic missile technologies or other weapons in various orbits. Aside from the threat of physical danger, potential adversaries are developing weapons that could jeopardize U.S. civil and military space services such as global navigation and communications satellites [2].

One of the best ways to ensure the continued safety and sustainability of U.S. systems and operations is the ability to provide early detection and observation data on foreign space objects. Technological development has enabled a situation in which early launch detection and characterization may need to occur on a tactical timescale so that proper warnings and courses of action can be decided upon as soon as possible.

Because the ground-based radar is less restricted by time of day, weather, or physical position than other sensor types, it makes it a vital component of early launch detection and characterization. A radar operator must be able to respond immediately when intelligence arrives about a possible foreign launch. Currently however, it is hard to fit this high-priority mission directly into the sensor's more traditional day-to-day operations without completely foregoing the initially tasked observation schedule. The DSRS would allow the radar operator to input new information and incorporate the foreign launch characterization mission into a schedule along with the previous

consolidated tasking list in a matter of seconds.

Overload of Observation Requests - The reduced costs, increased access, and proliferation of space systems and technologies have driven many countries to integrate space into military, as well as commercial activities. More and more satellites entering orbit bring greater risk for collisions and the creation of debris. When looking specifically at deep space, the Geosynchronous Earth Orbit (GEO) is essential but scarce space real estate. This is due to the fact that an object in this orbit has continuous, nearly hemispheric coverage of an area, making it extremely advantageous for communication, surveillance, reconnaissance, and weather collection missions. Today the International Telecommunications Union (ITU) is responsible for assigning orbital slots in the GEO orbit on a first-come-first-served basis, but there is still debate over the specific assignment process and what may be done to create additional room for more space objects in the future.

With so many objects entering deeper and deeper into space, the way sensor scheduling occurs will need to be able to meet this higher demand. Maintaining custody of space objects requires continuous knowledge of their state, so objects must be revisited and re-characterized often. While the current system of scheduling may have been able to maintain custody of all important space objects at one point in time, future success of the SSA mission will rely on the coordination of all deep space sensors. Therefore, there is great benefit in the development and use of a centralized, coordinated scheduler like the DSRS.

Many High Priority Task Requests - As space becomes more crowded, there will also inevitably be increased risk. Governments and private actors around the world may approach space as an arena with far less rules than on Earth, making it the perfect place to engage in malicious activity. Additionally, there is also the increasing risk of collisions as more and more objects enter space.

For example, Russia carried out an anti-satellite missile test on November 15, 2021, generating thousands of pieces of space debris that continued to generate more

debris as it crashed into more objects within the orbit. Some of the material came extremely close to the International Space Station, forcing astronauts on board into a state of emergency [2]. Space debris and the danger of serious collisions is high because objects achieve extremely high velocity in space, giving them high energy with the capacity for immense collisions and the ability to stay in orbit for years.

In situations such as this one, it is an important matter of national security that the United States is able to track foreign satellites and any space debris very quickly. Swift and accurate information about the location of space objects and debris is important for providing warnings to protect our space systems, personnel, and operations. This scenario involves observing and maintaining custody of many high-priority targets at a time. The DSRS allows for better scheduling in this case by prioritizing time-sensitive, high-priority task requests while also aiming to minimize the overall slew time of the radar to visit more objects in the same amount of time.

Radar Outage - The SSA mission relies on a large system of sensors continuously working to collect and update information. A centralized, coordinated approach to scheduling allows for the sensors to operate in a more efficient manner, ensuring the distribution and path plans of jobs make the most sense from a more global perspective of surveillance.

An important scenario to consider is the effect of a radar outage in the system of deep space radars. Individual radars have their own sets of satellites they can feasibly view based on their FOV in the sky, so there will inevitably be gaps in coverage in some of these scenarios. There may also be impacts on the operational price at the rest of the radar sites in the system as they take on more jobs in attempt to mitigate the gaps in surveillance.

We conduct the radar outage scenario by removing one radar from the system for one full day of scheduling. For each of the individual radars we assess the impact on the overall operational costs at the global system level, and we examine the impact the outage has on the custody mission by looking at the number of tasks dropped.

1.4.1.6 The Role of the DSRS

There is interest and research being conducted on the impact a fully automatic, coordinated scheduling approach would have in the SSA mission. This is a look into how scheduling radars for deep space custody would work under this proposed framework.

Our goals for the project include formulating and addressing the problem of automatically scheduling sensors in a way that preserves the realisms of actual DoD operations while also remaining tractable and reasonable. We aim to demonstrate the ability to balance different objectives in planning many heterogeneous targets and radar systems in an asynchronous, distributed environment. We choose to quantify the benefit of coordinated, automatic scheduling by conducting testing on several real-world scenarios and reporting measures of performance against a baseline greedy scheduler and an alternative version of schedule optimization that is less centralized in the task request distribution among radar sites.

Chapter 2

Literature Survey

This section reviews the literature that addresses problems related to the deep space radar scheduling problem. We first review ways that radar and satellite observation planning has been studied and conducted in the past. We then discuss scheduling problems in a more general sense, focusing on works that have features similar to those found in the deep space radar scheduling problem. Finally we explore the class of problems known as the *Vehicle Routing Problem*, a well-known optimization problem whose different variants have been studied extensively. We review several proposed ways to solve this particular problem as well as review the different variants that have been studied in literature. Finally, we present and explain the choice of the model we choose to use in the DSRS.

2.1 Scheduling Algorithms

We begin by discussing various scheduling problems and the proposed algorithms and methods for solving them. We focus first on how space surveillance assets have been studied and scheduled in the past and how certain problem features relate to the DSRS. Problems involving the scheduling of sensors and satellites include characteristics that are present in our problem because they are subject to many of the same feasibility and signals processing constraints. Next, we look at algorithmic approaches to scheduling more generally, focusing on those examples with problem

features present in the deep space radar scheduling problem.

Radar Scheduling

An important data point in any radar scheduling problem is the amount of time a radar spends observing or collecting data on a particular target. It is critical that planning includes a dwell time long enough for the radar to successfully complete a task, but not an amount of time so conservative that it slows down the productivity of the radar. In [34], Li, Xu, Zhang, and Kong create a scheduling model that focuses on the relationship between task execution performance and the length of time a radar spends dwelling on a certain target in a the network. First, time periods of tasks and radar availability are determined and then tasks are assigned to radars with open time slots using a greedy algorithm and heuristic algorithm.

Reinsos-Rondinel, Yu, and Torres create a model based on a time-balanced (TB) scheduler and test on a single radar system simulation used in adaptive weather sensing in [45]. The concept of TB scheduling was first developed and applied to radar scheduling by Stafford in [49], and this methodology was later applied to scheduling radars for multiple point targets in a military application by Butler in [10]. In the algorithm, the TB values indicate the "degree of urgency" of scheduling a specific task, with positive values indicating that a sensor is already late for execution. These values are constantly changing within the system based on priority, due time of the task, and other external factors. The scheduler then simply chooses the tasks with the highest TB value to be processed next.

Kintz, Lee, and Rejto approach the problem of scheduling a single radar's observations with two separate algorithms - the first algorithm is a greedy heuristic that schedules targets based on the next soonest task that is due to be executed [5]. The second method however, their Dynamic Sensor Scheduler algorithm, models the problem as a variant of the Traveling Salesman Problem, with the objective being to minimize total distance that the radar's antenna must slew across the sky to different target locations. Even while task priority and the stochastic nature of the mission were not included in the study, the Dynamic Sensor Scheduler significantly outper-

formed the greedy solution, allowing custody of more deep space objects while also conserving valuable radar resources [5]. Our work is similar to this previous project but in our application we work to extend the problem to a system of sensors and include various new problem features rather than focus on the scheduling capacity of one radar.

Satellite Scheduling

Although our problem considers the scheduling of radars to make observations of satellites, there are many spatial and temporal constraints that are similar to those problems that involve planning satellites to make observations.

Cho, Kim, Choi, and Ahn propose a two-step binary linear program for tasking and scheduling a constellation of satellites in LEO[13]. The overall goal of scheduling is to distribute tasks to satellites and schedule their start times in a way that maximizes the predetermined performance parameters. First, the program determines the set of feasible candidate communication time intervals for each satellite-ground-station pair and then tasks are allocated [13]. Tan and P. Wang approach the problem of Earth Observing Satellite imagery with a heuristic approach[52]. The problem relates to the radar scheduling problem in that it includes the element of competing task requests and the limited agility or availability of the observing resource, in this case the satellites. The heuristic algorithm selects and schedules satellites while also automatically choosing image parameters that maximize coverage and quality of the requested images [52].

X. Wang, Gu, Wu, and Woodward formulate a robust version of a satellite scheduling algorithm to address the uncertainty in image quality that arises due to cloud cover[53]. The authors robustify the formulation with a budget uncertainty set with a bounded value determined by the satellite operational costs and how conservative the model plans to be in protecting against uncertainty. The final formulation is solved with column generation due to the large number of variables introduced in the robust linear scheduling problem [53].

2.1.1 Scheduling with Temporal Constraints

An important aspect of the radar scheduling problem is the temporal constraints associated with each task request. There has been much research in how to properly leverage operations research techniques to schedule single agents with various release times, due times, and priority levels.

One approach to scheduling with temporal constraints has been a dynamic programming approach known as the successive sublimation dynamic programming (SSDP) method. As is explained in [24], SSDP involves constructing a carefully defined relaxation of the combinatorial optimization problem and computing the optimal policy of the relaxation. If the relaxed optimal policy is also optimal in the original problem, that is the solution; if not, the relaxed version of the problem is modified to carry more detailed information about the original problem. This process continues iteratively until finding the optimal solution.

Tanaka and Fujikuma propose an exact algorithm for scheduling a single machine where idle time is permitted by using the SSDP method [50]. The algorithm is the applied to four different realistic scheduling scenarios with the goal being to minimize total weighted tardiness when release times are given .

Akturk and Ozdemir also focus on minimizing total weighted tardiness or lateness of executed tasks of a single machine in both [3] and [4]. The authors use branch and bound algorithm but introduce a new dominance rule that is able to find a local optimal solution. The authors are able to show that the computational results of their derived algorithm outperform a number of other heuristics on simulated test sets. Davari, Demeulemeester, Leus, and Nobibon employ the branch and bound method to solve a generalized single agent scheduling problem to schedule jobs that have similar features to the task requests present in our radar scheduling problem[15]. Each job has a specific processing time, release date, due date, deadline, and penalty weight. The goal then becomes to schedule these jobs in a way that minimizes tardiness of the jobs' completion while also respecting hard time window and set precedence constraints [15].

2.1.2 Dynamic Scheduling

Scheduling problems may be divided into two classes – "predictive scheduling" which addresses problems with deterministic process times, machine availability, and task requests and "reactive scheduling" which must create schedules for a dynamic environment. While predictive scheduling may create a schedule valid for the entire time span, scheduling in a dynamic environment may need to be much more flexible and have the ability to adapt to new information quickly throughout the planning horizon. One approach to dynamic programming, SSDP, has already been introduced, but in this section we review a few more approaches to scheduling in a dynamic, uncertain environment.

Several approaches to a heuristic priority rule-based scheduling are studied by Haupt in [22]. Haupt conducts a survey on various scheduling rules, ranging in complexity, in various problem instances. He finds a dynamic version of the rule-based scheduling algorithm in which only the first job is selected in a job sequence for certain, before the algorithm runs again, rescheduling or revising the remaining jobs in the sequence.

Fabrycky and Shamblin approach the dynamic environment differently - using a search-based attempt that first reduces the search space of possible solutions[17]. Then, they use probability-based sequencing algorithms to come up with completion dates that are ultimately used to schedule the tasks. Gittens uses sequencing based on an assigned allocation index[21]. The allocation index for each job is developed based on specific Markov decision processes and forward induction policies. Next, the jobs are scheduled in decreasing order of this specified index. As jobs arrive and are processed, the indices are continuously updated and arrival and service times are learned over time.

2.1.3 Coordinated Planning

This section addresses the problem of coordination in planning multiple agents for some overall objective. In most cases the sub-planners are entirely subservient to the

coordinating agent, with little ability to provide feedback or schedule operational "outages." Additionally, less research has been done with distributed and heterogeneous agents operating simultaneously.

Pedrasa, Spooner, and MacGill address the coordinated scheduling problem and how it pertains to providing energy services to homeowners [43]. Residential consumers first assign values to desired energy services and then these resources are scheduled to maximize net benefits to the system. The problem coordinates and optimizes which energy resources will be used to service certain users by using swarm optimization, a technique that optimizes a problem by iteratively improving a candidate solution based on the net benefits.

Freuder and Wallace address a coordinated satellite scheduling problem with constraints and variables similar to those used in the DSRS in [19]. The authors employ constraint programming methods and model complex task requests by creating variables for each request with associated support variables specifying the time windows, resources required, durations, and other problem features. Because of the large number of variables created, the authors employ a heuristic search and constraint propagation method to create feasible solutions in a timely manner.

In [23], Herold coordinates both air and space assets in an asynchronous and distributed environment to collect Earth observations based on requests that have specified time windows and priority levels. Herold constructs a value function as the objective in an optimization problem that can be used to solve a linear program repeatedly over time.

2.2 Vehicle Routing Problem

The main objective of the radar scheduling problem is to create an observation plan for each radar that incurs the least cost. In this case, "cost" may be viewed as a function of slew distance, the dwell time, the penalties for soft constraints in the problem, and the price associated with operating the radar. For this reason, one possible way to formulate this type of scheduling problem is as a Vehicle Routing

Problem. The Vehicle Routing Problem is an important application of optimization that first appeared in a paper by George Dantzig and John Ramser in 1959 [14]. This problem is a generalization of the famous Traveling Salesman Problem but instead of one "salesman" that must complete a route, there is a fleet of vehicles that must visit and service a set of locations. The Vehicle Routing Problem becomes difficult to solve because the number of possible routes grows extremely fast with each added customer location. For this reason, Vehicle Routing Problems are classified as NP-complete or NP-hard problems [7]. Typically, problems may be classified as either "hard" or "easy" depending on whether or not there exists an algorithm that can find a solution in polynomial time. An algorithm is said to be solvable in polynomial time if the number of steps required for algorithm is on the order of n^k where n is the complexity of the problem and k is a non-negative integer. Currently, there are no algorithms that are able to solve the Vehicle Routing Problem in polynomial time – in some cases solution methods may have to search the entire solution space of the problem to find the optimal solution. Therefore, there have been a number of methods proposed that attempt to reduce the solve time for the model by intelligently searching the solution space [39].

2.2.1 Vehicle Network Flow Formulation

The most popular model proposed to formulate the Vehicle Routing Problem uses integer programming to represent the problem as a network. Network problems represent the problem in a graph $G = (N, A)$, with a set of nodes, N, and arcs, A.

The formulation of the Traveling Salesman Problem introduced by Dantzig et al. was extended to create the two-index vehicle flow formulations for the Vehicle Routing Problem [14]. In this network structure of the problem, N nodes represent customer locations, with A arcs between nodes having an associated cost of travel, c. In the classical Vehicle Routing Problem, the set of K identical vehicles each travel exactly one route that must begin and end at the central depot, node 0. The main goal is to minimize overall cost. Therefore, the classical network flow approach to the Vehicle Routing Problem is modeled as:

$$\min \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij} \tag{2.1}$$

$$\sum_{i \in N} x_{ij} = 1 \quad \forall j \in N \backslash \{0\} \tag{2.2}$$

$$\sum_{j \in N} x_{ij} = 1 \quad \forall i \in N \backslash \{0\} \tag{2.3}$$

$$\sum_{i \in N} x_{i0} = K \tag{2.4}$$

$$\sum_{j \in N} x_{0j} = K \tag{2.5}$$

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - r(S), \ \ \forall S \subseteq N \backslash \{0\}, S \neq \emptyset \tag{2.6}$$

$$x_{ij} \in \{0, 1\} \quad \forall i, j \in N \tag{2.7}$$

In this formulation K is the number of available vehicles, S is a subset of N, and $r(S)$ is a function corresponding to the minimum number of vehicles needed to serve the set S. Constraints (2.2) and (2.3) ensure that exactly one arc x_{ij} enters and leaves each vertex associated with a customer. Constraints (2.4) and (2.5) ensure the number of vehicles leaving the depot, node 0, is the same as the number entering. Constraints (2.6) are the generalized sub-tour elimination constraints. They impose that at least $r(S)$ arcs leave set S. Finally, constraints (2.7) are the integrality constraints due to the fact that x_{ij} is a binary variable [51].

The size and complexity of the problem increases dramatically with the addition of the sub-tour elimination constraints - without them the problem would resemble a more classic assignment problem with a much faster solve time. These constraints increase problem size so much because they must prohibit any possible sub-tour that may occur in the problem, so any additional node produces a large number of new sub-tour elimination constraints. Solving the complete problem would include every single one of the 2^N sub-tour elimination constraints and the single-tour elimination constraints and be very computationally expensive. This makes the formulation im-

practical for large instances common in real-world scenarios where solve time must be low and the number of nodes N is typically large.

2.2.2 Set Partitioning Formulation

Another proposed formulation for the Vehicle Routing Problem is the set partitioning formulation. The set partitioning formulation was first introduced by Balinski and Quandt in [6] and offers a stronger linear relaxation than many other model choices. The number of constraints remains relatively low compared to other formulations such as the network flow version, but the number of variables is extremely large – one variable for every possible feasible route in the system. Therefore, in the set partitioning formulation, each vehicle route is represented by a binary n-vector a_j, representing one feasible route. Elements $a_{i,j}$ make up the a_j vectors and are equal to 1 if stop i is visited on route a_j and 0 otherwise. There is also a predetermined set cost c_j for each of the a_j vectors. Therefore, the classic set partitioning formulation is:

$$\min \sum_{j \in N} c_j x_j \tag{2.8}$$

$$\sum_{j \in N} a_j x_j = e \tag{2.9}$$

$$x_j \in 0, 1 \quad \forall j \in N \tag{2.10}$$

Where e is a vector of ones and all routes a_j are predetermined to be feasible.

The major drawback to the Set Partitioning formulation is defining the vectors a_j. The number of possible, feasible columns a_j is computationally impractical to generate, store, and finally solve. Cutting planes, branch and bound, and implicit enumeration have all been explored as possible methods to solving the problem. The cutting plane methods start by solving the linear program relaxation of the set partitioning problem and then adding additional constraints gradually to "cut away" non-integer solutions. Branch-and-bound methods similarly start by solving the linear program relaxation of the integer program to get a lower bound on the solution.

The algorithm continues to solve sub-problems in an attempt to find an integer solution [8]. These basic methods are often used in heuristics to find solutions to "hard" routing problems.

Agarwal, Mather, and Salkin approach the problem of solving the set partitioned Vehicle Routing Problem in [1] in three basic steps. First, they solve the linear programming relaxation using column regeneration. Second, using the columns from the first step's linear relaxation optimal basis, the set-covering problem is solved to obtain a heuristic solution. In the third step, a relatively small set of columns is generated and added to the problem using set partitioning theory. The optimal solution is then obtained by solving the set partitioning problem over this smaller set of columns. Desrosiers, Soumis, and Desrochers followed a very similar method to solve the Vehicle Routing Problem but with the added element of time windows for customer service and no constraints on the capacity of each vehicle [16]. Foster and Ryan also describe an integer program based on set partitioning using a heuristic algorithm with the Revised Simplex Method, a method that is the same as the standard Simplex Method aside from its actual implementation [18].

Due to the difficulty of solving large-scale routing problems to optimality, metaheuristics such as those listed above have been extremely successful in obtaining solutions. These algorithms often utilize and combine a number of the basic methods presented. Although an exact, optimal solution is not always found, these algorithms are often very efficient and can solve Vehicle Routing Problems within 0.5 or 1 percent of the optimal solution. They are also easily adaptable to fit many different variants of this class of problems [54].

2.2.3 Commodity Flow Formulations

A final formulation that has been popular in Vehicle Routing Problems is the commodity flow formulation. This formulation is similar to the network representation of the vehicle routing problem in that the problem is seen is a graph $G = (N, A)$. Additional flow variables are added to this network structure, however, representing the flow of commodities along arcs in the graph. Letchford and Salazar-González

compared several different variations of integer programming formulations, including those with a number of commodity flow variables, on a number of variants of the Vehicle Routing Problem in [32]. In 2015, Letchford and Salazar-González updated their work in [33] by presenting two new multi-commodity flow formulations that dominate previous methods with the addition of only a polynomial number of variables. The authors show that for some variants of the Vehicle Routing Problem these new methods offer significantly lower bounds than former models and may be solved in faster time.

2.2.4 Vehicle Routing Problem Variants

In the classical Vehicle Routing Problem, optimal routes are found for K vehicles to visit N customer location nodes. However, there are many ways this problem may be expanded to better fit more complicated real-world scenarios. Some possible variants to the classical Vehicle Routing Problem are: Vehicle Routing Problem with Time Windows, which constrains service to certain periods of time; Vehicle Routing Problem with with Pickup and Delivery, where precedence constraints are added to ensure pickup locations are always visited before delivery locations; Capacitated Vehicle Routing Problem, where vehicles have a limited carrying capacity on board or limited distance they are able to travel; and Dynamic Vehicle Routing Problem, where service requests are not completely known at the start of service and instead arrive during the distribution process. This section describes Vehicle Routing Problem variants that are similar to the radar scheduler problem.

The Vehicle Routing Problem with Time Windows

The Vehicle Routing Problem with Time Windows is defined as a Vehicle Routing Problem where there are specific time windows that each customer must be serviced within or service must begin. Sometimes these constraints are modeled as "hard" constraints, meaning they must be obeyed or the customer goes without service and the solution is infeasible. They may also be modeled as "soft" constraints, where

there is a penalty added to the overall "cost" if a customer receives service outside their specific time window. The objective is still to minimize total cost for all of the vehicles to visit a set of customers. The difference in this version of the problem is that there are additional constraints and variables added to ensure that the time windows restrictions are obeyed.

Kallehauge, Larsen, Madsen, and Solomon formulated the Vehicle Routing Problem with Time Windows as a multi-commodity work flow problem in [28]. Much like in the traditional Vehicle Routing Problem, there is a fleet of vehicles, V, a set of customers, C, and a directed graph, G. In their formulation, Kallehauge et al. considered the graph to contain $|C|+2$ vertices, N, with the additional two vertices being for the source and sink nodes that vehicles are required to start and end their routes. The graph also contains arcs (i, j) with an associated cost $c_{i,j}$ and time $t_{i,j}$, which in this case includes the service time at customer i. Additionally, each vehicle has a limited capacity q and each customer an associated demand d_i. Every customer also has a predetermined time window $[a_i, b_i]$ that customer i must be serviced in. Time windows for the two depots, node 0 and node $n+1$, are identical and represent the scheduling horizon. The decision variables in this problem are the binary variables $x_{i,j,k}$ which takes value 1 if vehicle k drives directly from vertex i to vertex j and 0 otherwise. The second decision variable, $s_{i,k}$ denotes the time vehicle k starts to service customer i. The final formulation is:

$$\min \sum_{k \in V} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ijk} \tag{2.11}$$

$$\sum_{k \in V} \sum_{j \in N} x_{ijk} = 1 \quad \forall i \in C, \tag{2.12}$$

$$\sum_{i \in C} d_i \sum_{j \in N} x_{ijk} \leq q \quad \forall k \in V, \tag{2.13}$$

$$\sum_{j \in N} x_{0jk} = 1 \quad \forall k \in V, \tag{2.14}$$

$$\sum_{i \in N} x_{ihk} - \sum_{j \in N} x_{hjk} = 0 \quad \forall h \in C, \forall k \in V, \tag{2.15}$$

$$\sum_{i \in N} x_{i,n+1,k} = 1 \quad \forall k \in V, \tag{2.16}$$

$$s_{ik} + t_{ij} - M_{ij}(1 - x_{ijk}) \leq s_{jk} \quad \forall i, j \in N, \forall k \in V, \tag{2.17}$$

$$a_i \leq s_{ik} \leq b_i \quad \forall i \in N, \forall k \in V, \tag{2.18}$$

$$x_{i,j,k} \in \{0, 1\} \quad \forall i, j \in N, \forall k \in V. \tag{2.19}$$

Constraints (2.13) are the capacity constraints, (2.14), (2.15), and (2.16) are the path flow constraints, and (2.17) and (2.18) are the time window constraints.

Solomon and Desrosiers conducted a survey and analysis of different methods that have been proposed for different variants of the Vehicle Routing Problem with time windows [48]. After conducting their research, they proposed that algorithm development that utilizes smart integer programming techniques like Lagrangian relaxation and column generation could be an efficient algorithm for a solution. Kohl and Madsen were the first to apply a Lagrangian relaxation on the Vehicle Routing Problem with Time Windows with a free number of vehicles [29]. Jörnsten, Madsen, and Sorensen presented a method for solving the problem with a variable splitting technique, also known as Lagrangian decomposition, to find an exact solution in finite time [27]. Kallehauge et al. show that incorporation of branching and cutting methods on solutions obtained through the Dantzig-Wolfe decomposition, where the problem is broken down into a sub-problem and a master problem, are some of the best performing algorithms to date [28].

Dynamic Vehicle Routing Problem

In the static version of the Vehicle Routing Problem, all information relevant to the planning of the routes is known before the service and routing process begins, and this relevant information does not change after the routes have been constructed. On the other hand, not all information relevant to the planning of the routes is known to the planner at the beginning of a Dynamic Vehicle Routing Problem, and this information is subject to change after the initial routes have been constructed.

Stochasticity could come from a number of different places within the Vehicle

Routing Problem. For example, there could be stochastic customers, stochastic travel times, or stochastic customer demands.

One approach to dealing with a dynamic version of the Vehicle Routing Problem is through an a-priori optimization based method. In this method, an "a-priori solution" is determined based on probabilistic nature of future events. When this approach is taken, the likelihood of certain outcomes is processed before scheduling begins so there is no need to re-optimize the problem every time new information is received. Jaillet first introduced a Probabilistic Traveling Salesman Problem in [25] in which each node is present in the network with a probability p_i, so it handles the problem of stochastic customers. Laporte, Louveaux, and Mercure formulate this Probabilistic Traveling Salesman Problem as an integer linear program and then use a branch-and-cut approach to solve it [30]. Segiun in [47] and Genreau, Laporte, and Séguin in [20] propose an exact algorithm for the probabilistic Vehicle Routing Problem based on the integer L-shaped method, an extension of the stochastic, integer case of Bender's decomposition, for both a Vehicle Routing Problem with stochastic demands and a Vehicle Routing Problem with stochastic customers. One significant drawback to the a-priori optimization approach is the fact that it relies on detailed information about the probability of customer requests and locations which is often not available in a real-world application.

When the stochastic nature of the problem is not folded into the problem prior to solving, there are a few ways for the system to respond. Gendreau, Guertin, Potvin, and Taillard model first the static and then the dynamic versions of the Vehicle Routing Problem with uncertainty in the customers [37]. They solve the problem using a tabu search heuristic, a common meta-heuristic for optimization. Malandraki in [37] and Malandraki and Daskin in [38] introduce several heuristic algorithms based on nearest-neighbor method to solve the time-dependent Traveling Salesman Problem where there are variations in travel speed along arcs in the graph depending on time of day and other unknown factors in the system.

Methods that use more local procedures may find a new solution more quickly by searching for the best insertion of the new customer into the current planned vehicle

routes, although this may not ensure global optimality. On the other hand, a global search procedure would completely resolve the problem every time an input revision has occurred. Lund, Madsen, and Rygaard implement a heuristic procedure that attempts to minimize the weighted sum of detour and delay with each new insertion [36]. Larson also focuses on local search by presenting an algorithm which is based on the fast insertion of new immediate requests received in the system, an improvement heuristic, and the insertion of dummy customers [31].

Choosing a Model

After having reviewed the applicable literature, we now present the mathematical model for solving the deep space radar scheduling problem. We choose the solution approach based on applicability of past techniques as well as the similarity of the problem constraints and overall objective.

We choose to model the DSRS as a Dynamic Vehicle Routing Problem with Time Windows. In this problem we let the various satellites associated with incoming task requests represent the customer nodes and each individual radar is represented as one vehicle. In our case, the radar sensors and the satellites are heterogeneous and operate in a dispersed environment.

We recognize that this problem is NP-hard, leading to potentially long, unpractical solve times. For this reason, we choose to first implement an initial program that cleans the input data and conducts initial feasibility checks for each satellite-sensor pair. This reduces the search space that serves as input to the Vehicle Routing Problem portion of the problem where the optimization occurs. We use the Gurobi Optimizer for solving this portion of the problem, a mathematical solver that employs the branch-and-cut algorithm to solve mixed integer programs. The branch-and-cut algorithm works by solving a linear relaxation of the mixed integer program. Then, if the acquired optimal solution to the relaxation meets all integrality constraints, it is the optimal solution. If not, integrality constraints are added to the relaxed problem iteratively, each time checking to see if all constraints are met. This process continues until an optimal solution is obtained that respects all constraints from the original

problem.

The objective in the optimization problem will be to minimize the overall "cost" to the system. The cost is a function of the slew time, the dwell time, the lateness of the tasks, and the prices associated with operating each one of the radars. This cost value is minimized while several other hard constraints must be met. Namely, a satellite must be above the radar's lowest allowable pointing elevation while it is executing a task request associated to that satellite. The radar must dwell on a satellite for the required duration of time that is dependent on the specific radar-satellite pair. The time window constraints for each of the task requests and the execution of task requests are both soft constraints in the problem that are incorporated into the lateness value in the objective function.

Chapter 3

Model Context and Development

This chapter explains the development of our model for the DSRS. The first section defines the problem scope and terminology necessary to fully understand the problem features and to define the model. The second section describes the key assumptions that are made when developing the algorithm and coming up with an appropriate solution approach. The third section provides an overview of the inputs, decisions, objectives, and constraints that we use to model the final radar scheduler.

3.1 Problem Scope

As has been previously discussed, the deep space radar scheduling problem studied in this thesis has numerous approaches that may be used for scheduling, and there are many different opportunities for the application for Operations Research (OR) techniques to reduce the inefficiencies of current operations. The problem's dynamic nature, efficient resource distribution elements, signal processing constraints, and queuing problem characteristics are all features that are well-handled by many OR approaches and researched topics.

In all scheduling scenarios we choose to focus on in this work, there are multiple resources available to make observations and take measurements of multiple targets. In each case, the user might invoke the planner to focus on certain types of targets or prioritize different objectives depending on the mission request. For example,

if an object is thought to be "red" or potentially hostile, a user may require more frequent or even constant surveillance of the target. If an object is unknown or has not been previously cataloged, the user may require many more observations as soon as possible to gather information about characteristics such as size, maneuverability, priority classification, etc. In any case, the proposed radar scheduler must be able to make decisions on which requests should be sent to which radar sites, what specific slew path each radar should take upon receiving a list of tasks, how long the radar should spend dwelling on a satellite, and all the while be continuously gathering and storing information about target satellites that have already been viewed. With the ability to directly control paths taken by each radar, a good sequence of scheduling, planning, executing, and rescheduling must be taken to ensure objects are truly being visited successfully and sensors are not being scheduled for blocks of time when they are unavailable. Additionally, the fact that radar scheduling is being done for military surveillance may require observation at specific times and may happen with very little warning. For all of these reasons, it is crucial to define the problem that this thesis addresses carefully and clearly.

3.1.1 Key Terminology

We begin by introducing the key terminology used to define this problem for clarity in how the real world scenarios translate to the proposed formulation. It is also important in detailing how certain problem features are defined and later incorporated into the model. Some terms are considered standard terminology, others are specific to military operations and procedures, and others are specific to this thesis and may not be used until potential future operations.

3.1.1.1 Task Requests and Targets

When it comes to SSA, a *request* refers to the indication that there is the need for a certain object in space to be viewed a single time. In some cases, objects must be revisited multiple times but we use separate task requests for each individual visit

to a particular object. For any sensor asset or target, a task could entail different requirements or dwell times depending on the characteristics of the target itself or the task request parameters. Each task request is made up of a certain number of *observations* depending on the priority level of the task request. Each observation is made with a preset interval of time in between (usually 10 seconds) to allow the radar to capture any movement or change that could be occurring over time. A *target* refers to a RSO, in this case a deep space RSO, that requires action be taken on it. RSO and satellites are used interchangeably in this thesis because the cases we study focuse on satellite viewing, but in reality the DoD sensors are used to keep surveillance of satellites, space debris, satellite daughter objects, space weather, etc.

For the DSRS, a target RSO with a corresponding location make up the task request – these are the inputs into the system that require action. The United States Air Force 18th Space Command is the party responsible for sending a consolidated tasking list of task requests to be viewed by a radar each day. It is important to note that requests are accepted, serviced, or executed when the radar scheduler has actually added the task request to a specific radar schedule. A task is accomplished or successful when all of the observations for the request are completed for the required duration of time. The scheduler must be able to accommodate heterogeneous request and target types, with different priority levels and varying requirements for a "successful" observation of a target.

A *user* is the human agent that may alter the preset objective function weights. Adjusting the weights of the cost function and corresponding objective function in the optimization problem is simple, allowing it to be accessible to a wide range of user types. Allowing for changes to the objective function also allows the user to have more control in the scheduling process if necessary. For example, some users may restrict the program to only allow observations by one sensor if it is cheaper or if there is some piece of observational data needed immediately at a specific radar site. Users may also prioritize certain types of targets or missions such as in the foreign launch scenario where observations have inherently high value because early detection and characterization is so critical in these cases. While the radar sites receive task

requests from the U.S. Air Force, the employees at each of the radar sites are currently responsible for executing these tasks.

We attempt to create a model that captures as many of the real-world features and details of the problem as possible to ensure it remains executable and realistic, while also balancing that with achieving a practical solve time. To model the important task request characteristics, we choose to associate the following attributes with each request:

Attribute	**Description**
Name	An object number that uniquely identifies each resident space object
Priority	The associated value with the request. Can take one of 3 discrete levels with higher number representing more important requests (high priority = high value)
Label	The associated alpha-numeric term given to each targets indicating how often it should be revisited
Location	The targets' and radars' associated latitude, longitude, and altitude
Earliest Time	A single number indicating the earliest time (starting from 0 at beginning of planning period) when the target must begin being observed
Due Time	A single number indicating the latest time (starting from 0 at the beginning of the planning period) when the target must begin being observed
Dwell Time	A number indicating the amount of time (in seconds) that a target must be observed by a radar for it to be considered successfully serviced

Table 3.1: Task Request Attributes

3.1.1.2 Coordinate System

As Table 3.1 indicates, we model all target locations as point locations defined by either latitude, longitude, and altitude or input directly into the program as a Two Line Element (TLE). There are many different ways to define a coordinate frame when it comes to point objects on and around Earth, and it is important to remain consistent throughout the problem for accurate calculations and programming efficiency. We

convert the various location inputs to define all sensor and target locations in the same coordinate system. The specifics on conversion calculations may be found in Section 4.2.2.2.

3.1.1.3 Independent Radar Sites

Currently, the U.S. deep space custody mission is conducted by the various individual *radar sites* across the globe independently. Rather than this dispersed, isolated system of scheduling, the DSRS would have master control over task request assignment and the slewing radar observation paths. We refer to the *radars* or *sensors* as the devices that actually make observations and collect data of the RSOs in deep space. There may be and are multiple sensors at the same relative location, but they all operate with their own specific observation path plan. Although for this thesis we only consider one type of space surveillance asset, the deep space radar, our methods are extendable to include both ground and space-based optical sensors as well. The main approach to the deep space radar scheduling problem may also be extended to include observations of objects in other areas of space, such as LEO and MEO. However, this would require additional astrodynamic constraints to match point locations to precise time windows.

While the DSRS maintains centralized control over scheduling, users at independent radar sites also many be able to exercise some level of control depending on the circumstance. Users at independent radar sites operate dependent upon the produced master schedule when it comes to observing deep space RSOs, but they may also be able to schedule or add in additional jobs or tasks outside of the planned task assignments. Users may also over-rule a new schedule and deny certain task assignments at one of the radar locations if necessary. In this case, the task would simply be updated as "unsuccessful" during the feedback stage before the next stage of scheduling begins. This goes along with the idea that each of the individual radar sites may be viewed as *available* or *unavailable* during the scheduling process. Unavailability may be due to planned or unplanned radar outages and may last for any length of time the user specifies.

3.1.1.4 Planning Horizon

The *planning horizon* in the problem is the length of time that the DSRS actually plans observations for. A planning horizon of 24 hours means that the scheduler will attempt to schedule all of the received task requests to satisfy user objectives to the available radars during the 24-hour time window. This makes the assumption that all task requests received occur during the planning horizon.

We model the planning horizon for the various radar sites in the same planning horizon period. Although radar sites are geographically dispersed, because we are scheduling for a full 24-hour day and the radar scheduler runs automatically starting with a global time of 0, this will take care of the need to differentiate time zones in the assignment process.

3.1.1.5 Planning Stage

The first stage of scheduling is the *Planning Stage*, which refers to the time at which the DSRS is collecting task request information and user information and then subsequently creating the plans for the various radar sites. Each Planning Stage begins by processing the consolidated target list with the most up-to-date target satellite due times. Because the formulation of the problem makes it NP-hard, we usually choose to schedule in smaller batches based on the due times that occur within the next planned iteration. This allows for a faster solving time and more optimal schedule. It is important to note that we associate each Planning Stage with a single Execution Stage iteration. Because in some planning iterations there may be little to no change in the schedule, we also run a few experiments in which the radar scheduler plans all task requests in the planning horizon at once rather than in small batches for comparison.

3.1.1.6 Approval Stage

During the *Approval Stage*, we allow the user to over-ride any of the newly created radar schedules in favor of an old schedule before the schedules are actually executed.

If this happens to be the case, we return to the Feedback and Update Stage to input this information into the system and incorporate it into a new schedule before the Execution Stage. It is important to note that this step is completely optional. Because the DSRS runs automatically, it will continue to execute the planned schedules unless the user chooses to deny a proposed plan.

3.1.1.7 Execution Stage

The period of time when a radar at a specific site is carrying out its assigned schedule is called the *Execution Stage*. We assume that observation information or measurements are being sent to operators simultaneous to the satellite being viewed, so radars may continue to execute tasks while this occurs within the same stage. It is important to note that oftentimes a schedule may be updated while radar assets are executing some task, so the scheduler may need to create schedules that are then joined into currently scheduled plans already being executed.

3.1.1.8 Feedback and Update Stage

At the end of each Execution Stage comes the *Feedback and Update Stage*. It is extremely important to keep track of which tasks were successfully completed because information updates and future scheduling is based on this information being accurate. This stage accepts new information from the various radar sites and updates the consolidated tasking list accordingly.

If there is any new feedback from the user or any system-wide events that also must be incorporated into scheduling, that is also processed during this stage of planning. Although the scheduler is accepting updates during the entire scheduling horizon, this stage is when this information is actually processed for future use.

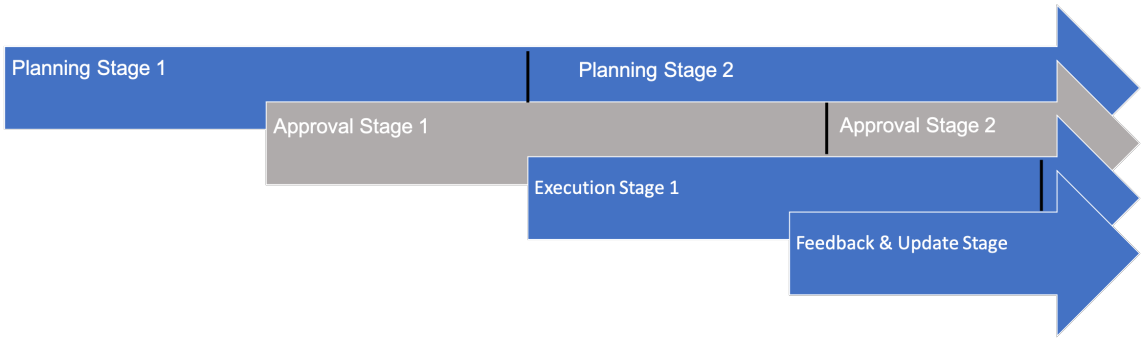


Figure 3-1: Stages of DSR Scheduler

3.1.2 Terminology and Assumptions in the United States Space Force Context

We design the DSRS for use by the DoD laboratories. We present the specific terminology used for this case in this section to ensure clarity in the translation from a military context to how it fits into the described radar scheduler context.

The DSRS receives the task requests for specific satellites each day from the 18th Space Command Squadron. 18th Space Command Squadron is the optical portion of the United States Space Surveillance Network. The command maintains a RSO catalog with all known satellites in orbit and determines a subset to be viewed by deep space radars each day – this subset is sent to the scheduler in the form of a consolidated tasking list. The radar scheduler then takes these requests as input and creates coordinated observation schedules for each of the DoD radar sites.

Included in the consolidated tasking list are alpha-numeric labels on each task request. The structure and meaning of these labels is explained in Figure 3-2. We use this system of labels to assign the task requests to the priority system and revisit structure we use in the DSRS. In our experiments, we choose to assign task requests

to one of the three discrete priorities seen in Figure 3-3.

Category:	1 Higher Priority ← 2 3 4 → 5 Lower Priority	
Suffix:		Description
	B	One track per revolution, track length of 16 observations
	C	One track per revolution, track length of 8 observations
	D	One track per revolution, track length of 4 observations
	J	Two tracks per day, separated by 6 hours, with 4 observations per track
	N	Four tracks per day, separated by at least 10° longitude, 4 observations per track

Figure 3-2: U.S. Military Task Request Labels

Figure 3-3: DSR Scheduler Task Request Labels

In this envisioned scenario for DoD space surveillance, we assume that the DSRS has the ability to assign all radar observations across all sites. All user adjustments, such as radar outages or other planned events, must be indicated to the overall scheduling level rather than just carried out at the specific radar sites themselves. In other words, the DSRS must have knowledge of any and all pertinent information during scheduling.

3.2 Modeling Assumptions

In this section we lay out the key assumptions that allow us to create a tractable and yet still realistic model. We describe our assumptions about the unknown, probabilistic nature of the problem, the amount of knowledge we have about the various radar planning cycles, the astrodynamics of the problem, our model of both time and space, and the sensor-to-target satellite interactions.

3.2.1 Stochastic Nature of the Problem

The radar scheduling problem is inherently stochastic because the arrival time and rate of task requests to the system each day is unknown. There is uncertainty surrounding the frequency and requirements of task requests that may not be included in the initial RSO consolidated tasking list at the beginning of each planning cycle. While many day-to-day operations would include viewing the targets only contained in this predetermined list, there is always the probability of additional targets coming in, of varying priorities, that need to be scheduled during the planning process. As launches into space become more routine, this is something that is likely to occur at a much higher frequency in the future.

We model the stochastic nature of the problem by using a program that randomly draws certain task requests from the U.S. military's list of relevant RSOs that could potentially require viewing in the geosynchronous belt. We sample from this list with a predetermined probability throughout the process of scheduling. Additionally, each time a task request is sent to a radar as part of an observation schedule there is the possibility that the observation will be unsuccessful during the execution phase, although this is very uncommon. There are a number of reasons this could possibly occur – the search for the target could take longer than is planned and thus limit the amount of time left in the schedule for the actual observation collection, the radar could point and dwell on a target from an angle that is not advantageous for the best measurement quality, or the radar scheduler could even override a previously scheduled observation in the event of timely, high-priority target observation requests

entering the system.

For this thesis, we assume that feedback received from the individual radar sites is accurate, and we build in additional slew and dwell times to be conservative in case of longer target search or detection, longer slew time, or other factors that may slow operations. By building in additional time as a "factor of safety" for the stochastic nature of the problem, we will not only reduce the amount of time resources spend on observations that end up being unsuccessful, but we are also able to relax some of the more complex stochastic features of the problem when it comes to modeling task completion.

3.2.2 Simulation of Real-World Problem

For each planning horizon of 24 hours, we allow the user to specify the number of planning iterations that must occur. At each iteration, the DSRS runs through each individual stage of scheduling – the scheduler will update the schedule based on new information if necessary, send the newly updated schedules to the various radar sites, allow for denial of schedules at the radar sites if applicable, and finally begin the next phase of execution. We assume that each radar site is able to provide observation information and measurement data to the user during the Execution Stage of the schedule. The radar scheduler is also continuously receiving feedback from radar sites and accepting new task requests through all phases of scheduling, so the next iteration may begin the planning process with the most up-to-date information and be able to incorporate it appropriately.

3.2.3 Complete Knowledge Operational Cycles and Data

For our radar scheduling problem, we also make the assumption that the DSRS has full knowledge of the operational cycles and important characteristics at the independent radar sites. Radar characteristics such as operational capabilities, slew rate and slew acceleration, field of view restrictions, and similar data points are all well-studied and documented by experts in the field. Because the orbital characteristics

of satellites are continuously updated, the approximate locations are also assumed to be correct as listed in the RSO catalog maintained by the U.S. military. We also assume the data collected on the satellite characteristics such as size, radar cross section (RCS), maneuverability, etc. is also accurate. Data sets for "normal" target size and corresponding RCS values include target RCS values over many aspect angles to ensure more accurate, robust estimates.

The only time when adjustments may need to be made to the reported slew and dwell numbers is when there is an UCT spotted in deep space by one of the various sensor assets. In this case observations need to be taken in an effort to quickly and accurately characterize the object, but because of the unknown nature of the object it may take longer to search, detect, and adequately observe than well-known RSOs. Examples of UCTs that may need more attention and resources include space debris, daughter objects that broke or separated from another RSO, or objects that were newly launched into deep space.

Another important consideration is the fact that some of the deep space radars are tasked with other mission sets or are responsible for space surveillance in different orbital regimes as well. At all of the radar sites, the DSRS may have difficulty determining the times reserved for maintenance or slotted for other mission tasks. For this reason, we make the assumption that the radar scheduler has full knowledge of the planned execution cycles at the radar sites. In in the actual scheduling process, we choose to combine all scheduled radar "outages" that may make them unavailable into one category of "scheduled outage" that may be input by the user prior to the next scheduling iteration.

3.2.4 Radar Signals Modeling

In this section we describe how we incorporate radar signal processing features into our model of the problem. Specifically, we discuss how we determine satellite dwell durations, how we model radar and satellite locations, and how we model radar slew times. More detailed calculations are included in Chapter 4.

3.2.4.1 Satellite Dwell Time Calculations

Various radar and RSO features must all be considered and incorporated in order to determine how long a radar must dwell on a specific target. The dwell time, or the length of time a radar spends "servicing" or observing a particular RSO, is entirely dependent upon the amount of time required for that specific radar to achieve a high enough signal-to-noise ratio (SNR) at the specific target. Various radar and RSO features can affect the time for a sufficient SNR and they are discussed below.

RCS and Target Size

The size of a target satellite and its associated RCS value are critical parameters when determining a radar's capacity. RCS is the electromagnetic signature of an object and a measure of how detectable the object is by a radar sensor measured in decibels (dB). The RCS values determine how long a radar will need to dwell on a specific target satellite in order for the observations to be "successfully" executed. There is a very strong correlation between the physical size of a target object and its RCS value.

Based on the relationship found and verified in previous work [5], we choose to model the relationship as the following:

$$RCS_{50} = (TargetSize)^2 * 0.20$$

Here we define "TargetSize" as the average of the RSO's height, width, and length in meters (m).

RCS Fluctuations

Another important consideration when it comes to RCS is the fluctuations in this value due to the target object's shape and angle to the sensor. RCS is a key characteristic in driving the radar's capacity and resource allocation, so it is important to select the right value while still accounting for these fluctuations. Aside from satellites that are intentionally designed to calibrate sensors by returning a constant RCS

value, all objects in space will exhibit some amount of RCS fluctuation.

When it comes to maintaining custody in the SSA mission, it is an important matter of national security that radars are able to detect a target satellite with an extremely high probability. This means that radars must be able to detect a target even when the RCS value has fluctuated very low for the objects size. For this reason, we choose a more conservative estimate for the RCS value used in the model, to capture instances where low fluctuations occur. Sometimes low RCS fluctuations can lead to dwell times that are magnitudes longer than the median RCS value, so this adjustment ensures that we are allotting enough time in the schedule to truly dwell on objects for enough time to collect successful and useful observations. On average, a 95% probability of detection equates to an adjustment of -11.7 dB from the median RCS value [5], so we choose to make this adjustment when using RCS values to calculate dwell time.

RSO Maneuverability

Another factor that comes into play when determining the observation number requirements as well as the total dwell time for a sensor-target pair is the maneuverability of the object itself. The maximum acceleration and total delta velocity that a satellite is able to achieve drive most of the maneuverability of a given object. Some objects that are smaller in size may be harder to detect due to a smaller RCS, but they are also limited by their capacity to store fuel to perform maneuvers in orbit. In contrast, larger objects typically have more room to store fuel and thus typically have a higher total available change in velocity.

Objects that are able to maneuver may take longer to track and detect than those that are somewhat restricted to one position in orbit, therefore, keeping custody of larger, more maneuverable objects may be very resource intensive and costly in scheduling.

Signal-to-Noise Ratio

Because of the importance of deep space custody in maintaining SSA for national security, an extremely high probability of detection and low probability of false alarm for target verification is needed. For this thesis, we follow the same numerical values as was used by Kintz et al. in [5], a probability of detection set at 99.9% and a probability of false alarm set at 10^{-5}. Using these values we can follow the Neyman-Pearson threshold to determine the required SNR for a dwell [46].

After the SNR value is determined, we use the following equation to solve for the radar dwell for a single observation:

$$SNR_{j,k} = \frac{P_T G_T}{4\pi R_T^2} \frac{\sigma}{4\pi R_R^2} \frac{G_R \lambda^2}{4\pi} \frac{B\tau N_p}{L} \frac{1}{kT_R B}$$

Where we define:

Variable	**Definitions**
P_T	Transmit power of the radar
G_T	Transmit gain of the radar
R_T	Range from radar transmitter to target
R_R	Range from target object to receiver
G_T	Receive gain
λ	Wavelength
B	Bandwidth
τ	Pulse length
N_p	Number of pulses
L	Losses
k	Boltzmann Constant, $1.38064852 * 10^{-23}$ m^2 kg s^{-2} K^{-1}
T_R	Receiver temperature
N_R	Number of receivers
N_T	Number of transmitters

Table 3.2: SNR Variable Definitions

The dwell time for a single observation for a sensor-target satellite pair is found by solving the equation for the integration time required to achieve the specified SNR.

Dwell Time

Each satellite has a specified number of required observations determined by its alpha-numeric label as well as its size and maneuverability. The previously calculated dwell time is the time for one observation, and we assume there should be a few seconds between observations taken by the radar so they are not completely identical. The specifics of these calculations are discussed in Chapter 4.

3.2.4.2 Radar and Satellite Point Locations

In the DSRS we choose to model all locations for satellites and radars as point locations in the Range-Azimuth-Elevation (RAE) coordinate system. Although the locations of sensors and space objects are originally defined and input to the program as Latitude-Longitude-Height (LLH) coordinates, the RAE coordinate system makes the most sense when doing later calculations for slew time. This is due to the fact that slew time for a sensor is a function of the angle pointing from the sensor to the target object in space. For the starting point or "depot" positions of the radars' antennas, we assume that it begins the first Execution Stage with the antenna pointing straight towards the geosynchronous orbit. Because the RAE coordinate system is position defined by angles from some reference point, the point location of each satellite will be different for each of the independent radar sites.

Coordinate conversions for both the target RSOs and the radar sensors occur in a initialization program that prepares all of the data before the mixed integer optimization program. There are k (number of radars) different lists of satellite locations calculated in this heuristic. Specific explanation of calculations may be seen in Chapter 4.

3.2.4.3 Radar Slew Times

As was previously mentioned, the time a radar spends slewing across the sky is dependent upon its location and other system characteristics. The slew distance is calculated as a function of the azimuth and elevation angles for each of the satellites

from the perspective of the radar's location on Earth and each radar is characterized by a specific slew rate and acceleration in the azimuth and elevation directions. These inputs allow for the slew time computation.

Because the slew time is calculated based on the angular location from the perspective of one radar site, these calculations must be done k (number of radars) times. The data initialization program that occurs before the optimization portion outputs k NxN matrices of slew times, where N is the number of satellites with corresponding task requests. The slew time is calculated based on the azimuth and elevation angles for two target satellites from the perspective of one radar location.

3.2.4.4 Modeling Time and Space

When it comes to modeling temporal and spatial features in a problem, there are generally two approaches – one involves discretizing space and time and the other involves modeling the two as continuous. Continuous modeling of time and space tends to be more complex and translating into longer solve times, so we choose to look at discrete time and space in this thesis. Usually, this is done by transforming the real-world problem into a graphical structure that is easier to visualize and analyze.

Because this thesis focuses on target objects specifically in the geosynchronous belt, we are able to make the assumption that the actual location of the nodes in the graphical structure will not change much from time period to time period. This is due to the fact that objects in the geosynchronous orbit have an orbital period that is the same as that of the rotation of the Earth. The synchronization of objects in this orbit creates ground tracks that appear as points on the Earth's surface, thus allowing us to model these objects as point locations.

Given the astrodynamics features still present in the problem, we allow the radar antenna to slew to any location at any time. That is, we do not discretize three-dimensional space in regions or sections, but rather allow the target satellite locations to take on any value from a continuum. We also allow sensor pointing angles to take on arbitrary values and target locations, time windows, and required durations are arbitrary in that there is no finite set from which we choose values. We allow a

continuum of latitude, longitude, and time values quantized to four decimal places.

3.2.4.5 Quality of Observations

One metric of interest for potential users of the DSRS is the quality of the measurements that it is able to plan for and achieve. The quality of an observation is dependent on many different controllable and uncontrollable factors, including the following: target satellite type, sensor type, and length of observation.

Although there are far more factors that may play into the quality of an observation, we model the quality using these three values. We do this by determining the threshold for an observation with a quality that is still considered adequate and deemed "successful" by the user for each target object-sensor pair. We then determine the minimum dwell time needed for this level. The needed quality of the observation plays a role in the required SNR and number of observations collected.

3.2.4.6 Real-World Sensors

As has been previously stated, many aspects of this problem rely on the specific characteristics of the deep space radars included in the model. Characteristics such as sensor slew rate, slew acceleration rate, system temperature, system loss, etc. may change from sensor to sensor. There is also a big difference between sensors that operate at different frequencies when it comes to mission sets and priorities. The various wave and frequency ranges used by radars can be seen in Figure 3-4.

We address these differences in modeling in the initialization step prior to the optimization program. We take these characteristics as input and prepare data and sets appropriately based on these values so the program is flexible and generalizable.

Because this thesis focuses on the scheduling of deep space radars within the DoD, we choose to model the actual radars used for military deep space custody in this problem. Although there are three radars that are currently operational, we model a fourth radar named "Radar 4" in this study as it will enter the space surveillance system in the near future.

- *HUSIR-X* - This deep space radar is located in Westford, MA. It is an X-band radar that is used for imaging and tracking.

- *Millstone Hill Radar* - This deep space radar is located in Westford, MA. It is an L-band radar that is used exclusively for tracking objects.

- *Altair* - This deep space radar is located at Reagan Test Site on the Marshall Islands. It is a VHF-band radar that is used for tracking space objects, specifically missiles for ballistic missile defense (BMD).

- *Radar 4* - We choose to model a radar at White Sands Beach, AZ. This is a proposed radar not yet under operational but has plans to enter in the future. It is assumed to be an L-band radar used for tracking space objects.

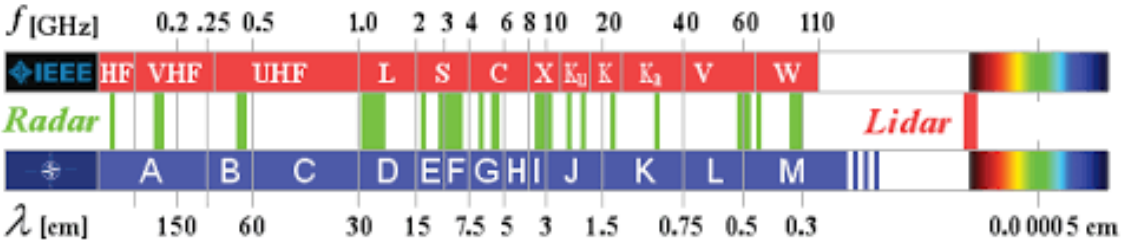


Figure 3-4: Waves and Frequency Ranges used by Radar [44]

3.2.4.7 Real-World Satellites

Much of the modeling and scheduling in this problem relies on the characteristics of the space objects that serve as targets for the radars. Characteristics such as size, RCS, maneuverability, and priority change the requirements and standards for a successful task completion in the model.

For this reason, it is extremely important that we model the target satellites with the best actual data and approximations to ensure the problem remains realistic and useful. In this thesis we use the military's continuously updated catalog of space objects in the geosynchronous belt for the satellite location, size, and launch data. Because the information about specific RCS values, revisit times, and observation

collection requirements may be sensitive, we come up with a way to quickly generate realistic values for these specific characteristics drawn from a set constructed after consulting with engineers in the field. This step occurs in the data initialization portion of the program, so it is easily generalized to include a wider range of objects.

Chapter 4

Algorithmic Approach

In this chapter we give more details on the mathematical formulation and calculations used in the DSRS. The first section introduces the overall flow of the model, giving details on the functions and calculations used in various parts of the initial heuristic and optimization program. Next, we discuss how we specifically implement the DSRS in practice.

4.1 Mathematical Formulation

In order to solve the deep space radar scheduling problem, we develop an optimization problem with an objective function designed to maintain custody of all deep space objects in the consolidated tasking list by minimizing the overall lateness and jobs dropped, slew time, and total operating cost. Although the objective function of the optimization program has default weights on each of the objective function's terms, we also allow the coefficients of the function to be adjusted by the user in order to better capture user objectives with the user weights. The final formulation of the problem is a modified version of the Vehicle Routing Problem that is able to quickly and intelligently schedule the deep space radar sensors.

The objective function contains various elements that capture specific satellite and radar characteristics that may change the value of the penalty or benefit to the overall system during the optimization. We are also able to alter radar schedules

dynamically based on feedback from the user as well as new outside information as it is introduced into the system. We accomplish this by using a program that incorporates feedback data into the list of requested tasks and use a rolling horizon-type approach to accommodate interaction with the various radar sensor sites over time. We also use the initialization portion of the program to execute feasibility checks in order to reduce the search space of the optimization program and allow for faster solve time. In this initial heuristic we also calculate various components of the radar scheduling objective function that are used later as inputs into the optimization portion of the scheduler.

To get better insight into the execution of the functions, interactions, and flow of information in the DSRS, we present a diagram of the program in Figure 4-1.

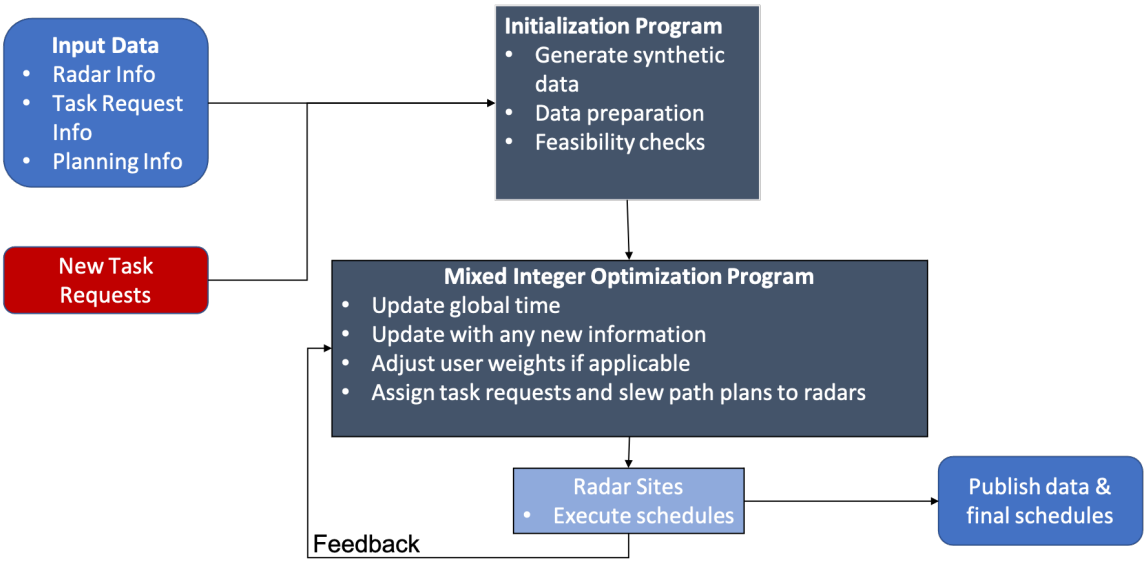


Figure 4-1: Flow of Scheduling

4.2 Notation and Definitions

We choose to solve the deep space radar scheduling problem by using a modified version of the Vehicle Routing Problem using a mixed integer program – a program that utilizes both continuous and binary variables to optimize some objective. We

will now define the constructed sets of data, inputs, and decision variables that are used in the mathematical formulation.

4.2.1 Input Sets

In order to schedule radars for satellite viewing for the entire planning period, the user must specify or input several sets of information. The following are the sets that are used in the DSRS.

Set	**Description**
A	Set of planning periods scheduled for the planning horizon, number of iterations of scheduling
T	Set of all task requests and corresponding satellite information
K	Set of available radars and corresponding radar information

Table 4.1: Sets used to define DSRS

Although these are the only sets needed to initially define the model at the beginning of planning, there are various pieces of information included or calculated from these sets that are used later as input into the optimization portion of the scheduling program. Users may also indicate additional information about the planning period such as scheduled outages at one or more of the radar sites or adjustments to the default user weights for optimization.

Table 4.2 displays the inputs associated with the satellite task request data. Some of this information is input directly by the user via the previous sets while other values are calculated in the beginning of the scheduler during the data preparation and initial heuristic phase. This information is initialized at the beginning of the scheduling process, but then continuously updated throughout the full planning period. This information for each task request is stored in a running log so users may access it at any time during scheduling.

Task Request Inputs	Descriptions
Obj_Name_i	A unique numerical identifier corresponding to the individual RSO i.
$Location_i$	The location of the RSO associated with task request i as defined by latitude (degrees), longitude (degrees), and altitude (meters)
$Size_i$	the size of the RSO associated with task request i. This is later used to calculate the RCS of i
RCS_i	The RCS value associated with task request i (dBsm)
$Priority_i$	The RSO's priority associated with task request i. Note that an increased value corresponds to increased priority. We allow the priority to take on 4 discrete levels from 1 to 4.
$Revisit\_Time_i$	The amount of time in between radar visits to the object corresponding to task request i.
$Rem\_Revisits_i$	The number of remaining times a radar must visit the RSO corresponding to task request i before the end of the planning period.
$Early_i$	The time corresponding to the beginning of the time window for task request i to begin to receive service (in minutes from the start of the planning period)
Due_Time_i	The time corresponding to the end of the time window for task request i to begin to receive service (in minutes from the start of the planning period).
Num_Obs_i	The number of observations required during for task request i.

Table 4.2: Task Request Data Inputs

Table 4.3 includes the inputs associated with the individual radars that are available for planning. Some of this information is input directly by the user via the initialization sets, while other values are uploaded directly from lab sites to get the most up-to-date radar information.

Finally, there are several system-wide characteristics that serve as inputs into the DSRS, seen in Table 4.4. All of these values have default settings if the schedule is operating under normal, daily scheduling conditions, but the user is also able to alter these inputs prior to scheduling.

Radar Inputs	Descriptions
$Depot_k$	The location of the radar k as defined by its latitude (degrees), longitude (degrees), and altitude (meters)
P_k	The transmit power of radar k (watts)
$Duty\_Cycle_k$	The duty cycle of radar k (ask units)
$freq_k$	The center frequency of radar k (Hz)
$Diam_k$	The diameter of radar sensor k (meters)
App_Eff_k	The aperture efficiency of radar k (percentage)
$Temp_k$	The receive temperature of radar k (K)
CPI_k	The coherent processing interval (CPI) of radar k (Hz)
$Rate\_AZ_k$	The slew rate in the azimuth angle direction of radar k (m/s)
$Rate\_EL_k$	The slew rate in the elevation angle direction of radar k (m/s)
Acc_AZ_k	The acceleration in the azimuth angle direction of radar k (m/s^2)
Acc_EL_k	The acceleration in the elevation angle direction of radar k (m/s^2)
$Loss\_dB_k$	The system loss of radar k (dB)
$Outage\_time_{s,k}$	Time of scheduled outage s for radar k, if one is scheduled (in minutes from the start of the planning period).
$Outage\_length_{s,k}$	The length of time for the scheduled outage s for radar k if one is scheduled (in minutes from the start of the planning period).

Table 4.3: Radar Data Inputs

System Inputs	Descriptions
$Time\_Horizon$	The length of the planning horizon (minutes)
$Iterations$	The number of iterations of updating and rescheduling for the duration of the planning horizon
$Start\_Time$	The start time of the planning horizon (minutes)
$Scenario\_Class$	This numerical value indicates which of the five possible scenarios for scheduling ranging from day-to-day scheduling (1) to one of the four operational scenarios (2-5).

Table 4.4: System-Level Data Inputs

4.2.2 Initialization Program

After reviewing results from previous works and studying the success in using heuristics in speeding up solve time for scheduling problems, we create a simple heuristic that is implemented before the optimization program is executed. The sets A, T, K

are run through an initial heuristic to process and prepare data for optimization as well as perform feasibility checks on satellite-sensor pairs. Specifically, this includes generating synthetic values for which the actual values may be sensitive or confidential, calculating more computationally expensive data values prior optimization, and running FOV and capacity feasibility checks.

4.2.2.1 Generate Synthetic Data

Because of the classified nature of much of the data available from the DoD on satellites in deep space, an important part of the data pre-processing step is the generation realistic synthetic portions of the data. Namely, the $Revisit\_Time$, Num_Obs, Due_time, and RCS values are calculated using various methods for each of the satellites in the consolidated tasking list. For the $Revisit\_Time$, Num_Obs, and Due_Time values, we randomly sample from a predefined uniform distribution of data values based on the satellite's assigned priority level. These distributions are created after consulting with engineers in the field.

In order to calculate the RCS values for a satellite, we first sample random values from a list of common satellite dimensions to ensure we have a realistic satellite size. We then use these dimensions to calculate the $Linear\_Size$ of satellite i as the average of the satellite's height, width, and length. Next, we calculate the median RCS value based on the relationship derived in Chapter 3.

$$RCS_i = (TargetSize)^2 * 0.20$$

From there, we adjust the RCS values to ensure protection against low fluctuations in RCS by performing the adjustment of -11.7 dB to ensure a 95% probability of detection.

4.2.2.2 Converting Coordinates

When it comes to optimizing scheduling for deep space radars, an important consideration is how long the radar will spend slewing across the sky to different locations.

All slew acceleration and velocity information is reported in the azimuth and elevation directions. Because these values are based on angular location, it is dependent upon where the radar is physically located on Earth and where the radar's antenna is pointed at the beginning of the planning period. This section describes how to convert from the input LLH coordinate values to the more useful RAE coordinate values that will be used in the rest of the problem.

We assume that the antenna for each of the radars begins pointed at 0 degrees in the azimuth direction and directly towards the geostationary orbit near the equator. We then calculate the $depot_k$ value for each radar's position.

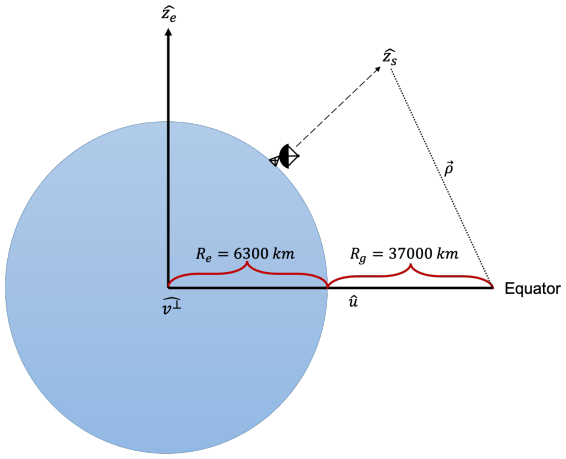


Figure 4-2: Terms for Coordinate Calculation

We define the sensor position as P_s and use the World Geodetic System (WGS84) definitions of $a = 6378137$ and $e = 8.1819190842622e^{-2}$ in the following calculations [35].

- we convert sensor position, $\overrightarrow{P_s^{LLH}}$ to $\overrightarrow{P_s^{ECR}}$

$$N = a/\sqrt{1 - e^2 * \sin(lat)^2}$$

$$x = (N + alt) * \cos(lat) * \cos(lon)$$

$$y = (N + alt) * \cos(lat) * \sin(lon)$$

$$z = ((1 - e^2) * N + alt) * \sin(lat)$$

$$P_s^{ECR} = (x, y, z)$$

- define the sensor zenith angle, $\hat{z_s} = \frac{\overrightarrow{P_s^{ECR}}}{||\overrightarrow{P_s^{ECR}}||}$

- we define earth zenith angle, $\hat{z_e} = [0, 0, 1]$ and calculate $\hat{v^{\perp}}$, to get orthogonal angle to sensor and earth zenith angles with magnitude one.

$$\hat{v^{\perp}} = \frac{\hat{z_s} * \hat{z_e}}{||\hat{z_s} * \hat{z_e}||}$$

- we define $\hat{u}$, the vector pointing to the geosynchronous belt from the sensor as $\hat{u} = \hat{z_e} * \hat{v^{\perp}}$

- Now, having defined $R_e = 6300$ km as the Earth's radius and $R_g = 37000$ km as the range to the geosynchronous belt from Earth, we calculate the sensor depot position vector with correct magnitude as $\overrightarrow{P_{Geo}} = (R_e + R_g)\hat{u}$

- This is the sensor depot position vector in earth-centered, earth-fixed coordinate system so we now convert $\overrightarrow{P_{Geo}^{ECR}}$ to $\overrightarrow{P_{Geo}^{RAE}}$. We use several new terms seen in Figure 4-3 defined in the sensor frame in this calculation, where ϕ is elevation angle and θ is azimuth angle.

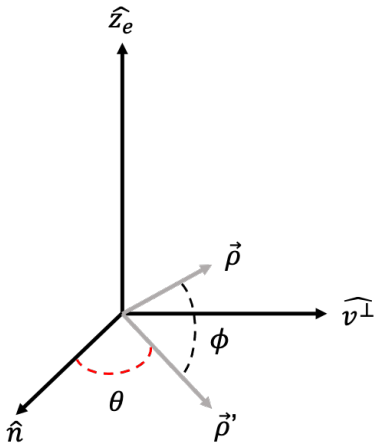


Figure 4-3: Terms for Azimuth Angle Calculation

We begin by calculating range:

$$\rho = ||\overrightarrow{P_s^{ECR}} - \overrightarrow{P_Geo^{ECR}}||_2$$

$$range = \sqrt{(x_s - x_{Geo})^2 + (y_s - y_{Geo})^2 + (z_s - z_{Geo})^2}$$

Next, we calculate the elevation angle:

$$\sin\phi = \hat{z_s} * \frac{\overrightarrow{P_s^{ECR}} - \overrightarrow{P_s^{ECR}}}{||\overrightarrow{P_s^{ECR}} - \overrightarrow{P_s^{ECR}}||}$$

$$elevation = \sin^{-1}(\hat{z_s} * \frac{\overrightarrow{P_s^{ECR}} - \overrightarrow{P_s^{ECR}}}{||\overrightarrow{P_s^{ECR}} - \overrightarrow{P_s^{ECR}}||})$$

Finally, we calculate the azimuth angle:

We define $\overrightarrow{\rho_{SF}}'$ as ρ in the sensor's coordinate frame (SF):

$$\overrightarrow{\rho_{SF}}' = \begin{bmatrix} \overrightarrow{\rho} * \hat{n} \\ \overrightarrow{\rho} * \hat{v^{\perp}} \\ 0 \end{bmatrix}$$

Next, we convert to the ECR frame:

$$\overrightarrow{\rho^{ECR}}' = \begin{bmatrix} \hat{n} & \hat{v^{\perp}} & \hat{z_s} \end{bmatrix} \begin{bmatrix} \overrightarrow{\rho} * \hat{n} \\ \overrightarrow{\rho} * \hat{v^{\perp}} \\ 0 \end{bmatrix} \overrightarrow{\rho^{ECR}}$$

We are able to perform this operation because we are projecting $\overrightarrow{\rho^{ECR}}$ onto the ground plane $\begin{bmatrix} \hat{n} & \hat{v^{\perp}} & 0 \end{bmatrix}$.

$$Azimuth = \cos^{-1}\hat{n} * \hat{\rho^{ECR}}'$$

$$\text{where } \hat{\rho}' = \frac{\overrightarrow{\rho^{ECR}}'}{||\overrightarrow{\rho^{ECR}}'||}$$

Similarly, we calculate the locations of all satellites from the perspective of each of the k radars present in the problem. The last step is the only one that changes in the calculations, with the satellites position S^{LLH} replacing the term P_{Geo}^{LLH}. The RAE coordinate system defines locations based on the angles from a specific point on Earth, so we must calculate the satellite's point locations k times, once for each of the radar locations as the reference point.

4.2.2.3 Feasibility Check

This section describes how we incorporate physical constraints in the problem into the optimization problem. Because we are viewing satellites from locations on Earth, we must account for the fact that not every satellite in orbit is visible from a given location on Earth at all times. We make the assumption that radar sensors are able to slew anywhere above 10 degrees above the level of the ground. Therefore, after converting the radar depot locations and satellite locations to the RAE coordinate system, we check the polar angle. The polar angle is simply $90 - elevation$ angle, measured in degrees. As long as the polar angle from the radar location to the satellite location in space is above 10 degrees, it is feasible that the radar can view the satellite.

Another consideration is the best sensor-task request pairings based on the radar's defining characteristics such as frequency. The choice of frequency for a radar depends on the radar's application and usually involves trade-offs among several factors [12]. Because the deep space radars in the space surveillance system are involved in different missions and consequently have different frequencies, some may be capable of executing certain types of task requests while others may not be. We perform checks to determine which radars are capable of performing which task requests in the given consolidated tasking list beforehand.

For each of the k radars we create a set of feasible satellites and satellite paths in the sky and store them to be used as input in the optimization portion of the program. We choose to check feasible satellite and path requirements prior to the mixed integer program to reduce the search space in the optimization, speeding up solve time.

4.2.3 Mixed Integer Optimization Problem

After the initial heuristic is executed, scheduling begins during the optimization portion of the program. The next section describes the various features and calculations that occur in this step.

4.2.3.1 MIP Objective Function

This section describes how the objective function in the mixed integer program is defined, calculated, and constructed to address the problem of radar scheduling. We discuss the various elements in the objective function, giving details as to why they were selected and how they are appropriately scaled and computed. Because we use a convex combination of elements as the objective function in the optimization problem, it is important to scale the data so it becomes more generalized and the program will not be biased towards some feature that is simply higher in magnitude.

As was previously mentioned, the objective function of the DSRS is a linear, convex combination of several components. A subset A of a vector space is said to be convex if any point $x \in A$ and all scalers θ are of the form $x = \theta_1 x_1 + \theta_2 x_2 + ... + \theta_k x_k$ with $\theta_1 + ... + \theta_k = 1, \theta_i \geq 0$. This is said to be a convex combination of the vectors X, and all convex combinations of vectors in A constitute the convex hull of A. In a linear integer program, the optimal solution is also optimal over the convex hull of all feasible solutions [9].

In our case, we introduce user weights, u, in place of the θ and vectors specific to the radar scheduling problem take the place of the x values in the example. The objective function is then constructed as follows: for any task request i and j and radar k, the scaled value of the objective function is a convex combination of the vectors and the user weights, where $\sum u = 1$. Therefore, each value in the objective function lies in the range [0,1]. Algebraically, we calculate the value for any request i, j, served by radar k, as follows:

$$\text{Cost}^t_{i,j,k} = f(\text{late}_j, \text{price}_k, \text{dwell}_{j,k}, \text{slew}_{i,j,k})$$

$$\text{Cost}^t_{i,j,k} = u_{late}\text{late}_j + u_{price}\text{price}_{i,j,k} + u_{slew}\text{slew}_{i,j,k}$$
$$\text{where } u_{late} + u_{price} + u_{slew} = 1$$

We include the superscript t that corresponds to a given planning iteration number to allow the value of the objective function to change throughout the planning period based on the feedback received from the various radar sites, planned or unplanned events such as outages, and the user's input. This objective function provides a construct to compute a relative cost to the system of a certain radar and sensor-to-sensor pair based on some parameters. The subsequent sections will go into more detail describing the notation used in the function and how each component is calculated.

4.2.3.2 Objective Function Components

In this section we describe each component of the objective function that we use for the DSRS. It is important to note that each of these values are specific to the scheduler on a specific planning cycle - they are dependent on the sensors available as well as the requested tasks for the planning period. These values are also not seen by the user or reported in the final schedules at the various radar sites; they simply serve as a metric for controlling what is being optimized in the MIP. At each iteration, the scheduler chooses the best task distribution and slew path plans for each of the radars based on the decided upon Measures of Performance. Thus, the values inherent to each request and those values inherent to the sensors are not the scheduler's only concern in the planning process. Each value computed by the objective function is for a radar and satellite-satellite pair based on user weights and other problem specific features. The objective function components themselves are selected because they are directly tied to one or more of the Measures of Performance in the problem, making them important features for the user to be able to modify if needed.

Lateness

This component of the objective function is a measurement of the lateness of the time of the start of service, t_i, to task request i. The lateness portion is included to enforce

the time constraints in the problem. While there is the option to strictly enforce the time constraints of the tasks by making them tight constraints in the problem, this version allows the optimization to program to drop task requests in lieu of some other objective if it decreases overall cost to the system. The user is able to adjust the weights and tightness of the time window constraints before planning begins.

Because the problem is a minimization problem and $lateness_i$ is a variable calculated and input directly to the objective function, our goal is to minimize the $lateness$, with the ideal value equal to 0. The $lateness_i$ is computed for each task request i based on the $priority_i$ that is associated with that task request. Note that the higher the assigned priority, the more important or urgent that task request. We use the input task request $priority_i$ to calculate the penalty incurred for being late or not executing task request i. The penalty value is one of five discrete values from 0 to 100. To scale the penalty value in this portion of the objective function, we divide each penalty value by the maximum penalty of 100.

$$\bar{p_i} = \frac{penalty_i}{100}$$

In order to model and enforce the time constraints as soft constraints, we model them using variables $Early_i$ and Due_Time_i as well as a_i and b_i. As was previously stated, $Early_i$ and Due_Time_i specify the earliest and latest time for start of service of i, respectively. The variables a_i and b_i are positive, nonzero numbers only when the task is serviced outside of a specified time window.

We model this feature of the problem with the following constraints:

$$t_i + a_i \geq Early_i \quad \forall i \in T$$
$$t_i - bi \leq Due\_Time_i \quad \forall i \in T$$

If the user wishes to enforce tight time window constraints, there is an option to set constraints specifying that all a_i and b_i variables should be strictly equal to zero.

Additionally, we keep track of any unexecuted task request i with constraints containing variables $x_{i,k}$ and g_i. The variable $x_{i,k}$ is a binary variable equal to one when radar k executes task request i. However, g_i is a binary variable equal to one when the task request i is not completed successfully. In this way, the g variables keep track of the "gaps" in the system as it schedules for the day.

We model this feature of the problem with the following constraint:

$$\sum_{k} x_{i_k} + g_i = 1 \quad \forall i \in T$$

This finally leads to our calculation of the *lateness* variable that is input directly into the objective function of the DSRS.

$$lateness_i = \sum_{i \in T} 0.1(\bar{p_i})(a_i + b_i) + \sum_{i \in T} (\bar{p_i}) g_i$$

We weight the portion of the *lateness* that penalizes being outside of the time window for a certain task request by 0.1 times the penalty of the task request, versus weighting the portion of *lateness* that penalizes dropping a task request by the full value of the penalty of the task request. This is to reflect the idea that making some observation of a given satellite, even if it is earlier or later than is ideal, is still much better for maintaining custody than never observing the satellite at all.

Slew Time

As was previously mentioned, each radar in the system has specific characteristics that differ from site to site. The slew rate and slew acceleration rate, in both the azimuth and elevation directions, are important radar characteristics affecting the slew time from location to location in space.

We use the angular locations in RAE as input in this step of the program. The starting location in azimuth and elevation angles and the new location in azimuth and elevation are used as input in the slew time calculation.

We begin by calculating the distance and time it takes a radar to get up to its

max velocity in the following calculations:

$$acc\_time_{az} = \frac{rate_{az}}{acc_{az}}$$
$$acc\_time_{el} = \frac{rate_{el}}{acc_{el}}$$
$$acc\_dist_{az} = 2 * (1/2 * (acc_{az}) * acc\_time_{az}^2)$$
$$acc\_dist_{el} = 2 * (1/2 * (acc_{el}) * acc\_time_{el}^2)$$

Next we check to see if the distance from one location to another is less than acc_dist in the azimuth and elevation directions. If the distance is less, we assume the radar antenna accelerates for half of the distance and then decelerates the other half of the distance to stop on the target.

$$\text{if } |az_i - az_j| \ < acc\_dist_{az} \text{ then:}$$
$$dist = |az_i - az_j|/2$$
$$slew\_time_{az} = 2(2 * \frac{dist}{acc_{az}})^2$$
$$\text{if } |az_i - az_j| \ \geq acc\_dist_{az} \text{ then:}$$
$$slew\_time_{az} = 2 * acc\_time_{az} + \frac{|az_i - az_j| - acc\_dist_{az}}{rate_{az}}$$

Next, we calculate the elevation direction with the same steps:

$$\text{if } |el_i - el_j| \ < acc\_dist_{el} \text{ then:}$$
$$dist = |el_i - el_j|/2$$
$$slew\_time_{el} = 2(2 * \frac{dist}{acc_{el}})^2$$
$$\text{if } |el_i - el_j| \ \geq acc\_dist_{el} \text{ then:}$$
$$slew\_time_{el} = 2 * acc\_time_{el} + \frac{|el_i - el_j| - acc\_dist_{el}}{rate_{el}}$$
$$final\_slew = \max(slew\_time_{az}, slew\_time_{el})$$

Because slew time is calculated based on the angular location of the satellite from the radar's location on Earth, all of the possible slew times for one radar k from

any location i to any location j must be considered. This means that the slew time calculations must be done k (number of radars) times for each of the TxT matrices. These slew times are all compared in the optimization problem in order to minimize the slew time in the generated schedules. We scale the slew times before they are input into the optimization problem by dividing by the maximum distance a radar in the system may slew to reach a task request.

Operational Price

Another important factor that comes into play when scheduling is the individual price associated with operating each of the radars, $price_k$. If the goal is to minimize overall total cost to the system, it may be cheaper to schedule more task requests to certain sensors than others. Operational price differences may result from a variety of factors including age of the sensor, location of the sensor, limited operational capacity due to maintenance, and mission set differences. For example, the radar Altair has a primary mission of missile defense, so the price of devoting radar time and resources to the deep space custody mission may be higher than other radar sites where this may not be the case. We consult experts in the field to come up with rough price estimates for each radar in the model. We scale these numbers to be in the range (0,1] by dividing each price by the maximum operational price in the system.

We define "price" to be the price of operating one specific radar k for 1 min, and we use the radar's total slew time plus its total dwell time on all targets in the schedule as "operational time." So, in addition to calculating the slew time for each radar as was detailed in the previous section, we also need to calculate the dwell time for each radar.

Dwell Time The total dwell time is the amount of time a radar must stay at a given RSO's location in order for the task request to be considered successful. As was previously explained in Chapter 3, we use the SNR with a built in factor of safety for added robustness to determine this time.

Using given radar and task request parameters, we use the Radar Equation solved

for SNR to calculate the SNR for one second as was described in Chapter 3. This value, $SNR\_1sec_{i,k}$, is specific to the radar-task pair (i,k) and is reported in dB.

$$SNR\_1sec_{i,k} = \frac{P_T G_T}{4\pi R_T^2} \frac{\sigma}{4\pi R_R^2} \frac{G_R \lambda^2}{4\pi} \frac{B\tau N_p}{L} \frac{1}{kT_R B}$$

We then use this value to calculate the dwell time for a single observation based on the calculated RCS of the space object and the task request requirements.

$$obs_{i,k}^d = \frac{10^{SNR\_req_k/10}}{SNR\_1sec_{i,k}}$$

If all observations in a task request have the same requirements, then all obs^d variables for a specific radar-task request pair will have the same values. However, depending on the location of the satellite and the specifics of the task request, this may not always be the case.

It is also important for observations to not be taken immediately after one another during a task request. Because one of the main goals of satellite surveillance is gathering and updating orbital and satellite features continuously, observations must be executed in a way that allows for the best possible data points. By allowing even a few seconds to pass in between observations, characteristics such as current positional information, satellite movements, and maneuverability become available. Therefore, we add the requirement that 10 seconds must pass before the start of the next observation on one satellite. Then we use the input number of observations for each task request to calculate the total dwell time. For example, if task request i has 3 required observations, the total dwell time in seconds would be

$$final\_dwell_{i,k} = obs_{i,k}^1 + 10 + obs_{i,k}^2 + 10 + obs_{i,k}^3 + ...$$

We scale the $final\_slew_{i,j,k}$ components before they are input into the objective function by dividing by the maximum slew time from one node in the system to another node in the system. Likewise, we scale the and $final\_dwell_{i,k}$ components by dividing by the maximum dwell time of all of the radar-satellite pairs in the system.

We then use the $final\_slew_{i,j,k}$, $final\_dwell_{i,k}$, and the $price_k$ to come up with the total operational price for executing a schedule with the following variable:

$$op\_price = \sum_{i,j,k} final\_slew_{i,j,k} price_k + \sum_{j,k} final\_dwell_{j,k} price_k$$

Number of Active Radars

Another component we include is the objective function is the number of radars that are actively being used in each iteration. It could be beneficial for the user to try to find a feasible schedule where the minimum number of radars are operating at one time, perhaps to set aside the unused radars to work on some other mission set or task.

We use the binary variable r_k to keep track of whether or not radar k is used in a given planning iteration. Therefore, we model the number of active radars component as:

$$active\_radar = \sum_{k} r_k$$

Total Operational Time

An additional factor that may be important for scheduling purposes is the amount of time the radar spends slewing to and then dwelling on a particular satellite. This component is very similar to the *Operational Price* component, but this time we do not factor in radar price in the parameters; the goal is to simply reduce the amount of time the radar is operating rather than the overall price associated with the operation. We once use $final\_slew$ and $final\_dwell$ parameters that have been scaled in the same way described as in the Operational Price Subsection. Therefore, the component is modeled as:

$$tot\_time = \sum_{i,j,k} final\_slew_{i,j,k} + \sum_{j,k} final\_dwell_{j,k}$$

4.3 Vehicle Routing Problem Construction

In this section we present the actual mixed integer programming formulation used to schedule various radars within the DoD system. The formulation closely resembles a Vehicle Routing Problem variant, with the "vehicles" being represented by the radars and the "customer locations" being the satellites in deep space. We use the objective function established and defined in the previous section to solve the routing problem in a way that meets hard feasibility constraints, while also minimizing the lateness, slew time, and overall operational cost.

4.3.1 Deep Space Radar Scheduling Static Formulation

Although a static formulation for the classic Vehicle Routing Problem was given in the Literature Review in Chapter 2, in this section we present the modified formulation of a Vehicle Routing Problem that is used in the DSRS. This is the model that is solved at the beginning of each iteration during the Planning Stage.

The objective function is constructed as previously defined, with the final objective value being entirely dependent on the radar-satellite-satellite pairings chosen. Although we do have a recommended default vector of user weights u, we also allow the user to input these weights directly, which impacts the schedule and final objective value achieved during scheduling.

Some of the constraints are similar to those found in the classic Vehicle Routing Problem model, with slight modifications to account for the additional feasibility and details of this specific problem.

$$\min \sum_{i \in T} \sum_{j \in T} \sum_{k \in K} u_l(lateness) + u_s(slew\_time) + u_p(op\_price) + u_r(active\_radar) + u_t(tot\_time) \tag{4.1}$$

$$\sum_{k \in K} x_{i,k} + g_i = 1 \quad \forall i \in F_k \tag{4.2}$$

$$\sum_{k \in K} x_{i,k} \leq 1 \quad \forall i \in T \tag{4.3}$$

$$\sum_{j \in T} y_{i,j,k} = x_{i,k} \quad \forall k \in K, i \in T \tag{4.4}$$

$$\sum_{i \in T} y_{i,j,k} = x_{j,k} \quad \forall k \in K, j \in T \tag{4.5}$$

$$t_j \geq t_i + final\_dwell_{i,k} + final\_slew_{i,j,k} - M(1 - \sum_{k \in K} y_{i,j,k}) \ \forall k \in K, i \in T \tag{4.6}$$

$$t_i \geq Early_i - a_i \quad \forall i \in T \tag{4.7}$$

$$t_i \leq Due\_Time_i + b_i \quad \forall i \in T \tag{4.8}$$

$$\sum_{i \in S_k} \sum_{j \in S_k} y_{i,j,k} \leq |S_k| - 1 \quad S_k \subseteq T \tag{4.9}$$

$$x_{i,k}, y_{i,j,k}, g_i \in \{0, 1\} \quad \forall i \in T \tag{4.10}$$

$$t_i, a_i, b_i \geq 0 \quad \forall i \in T \tag{4.11}$$

In the final static formulation, x, y, and t are the decision variables. $x_{i,k}$ is a binary variable that is equal to 1 when radar k is assigned to task request i sometime in the planning iteration. The $y_{i,j,k}$ variable is a binary variable that is equal to one when radar k takes a path from task request i directly to task request j. Finally, t_i is a continuous variable that chooses what the start time of service is for task request i.

Constraint (4.1) is the objective function that minimizes some or all of the components depending upon the user weights, u. (4.2) is the constraint that enforces that each satellite must be visited one time or a "gap" is declared. Because the feasibility check determines which radars are able to view and successfully observe which satellites in the initialization program, the MIP only considers the radar-satellite pairs that are feasible. This information is included and stored in the feasible set of satellites for each radar F_k. (4.3) ensures that at most one radar is assigned to a specific task request. (4.4) and (4.5) are both constraints that ensure the radars form a tour; in other words, if the variable $y_{i,j,k}$ equals 1, then both the corresponding $x_{i,k}$ and $x_{j,k}$ variables must also be 1. (4.6) is a constraint that ensures the start of service at a new location does not begin before the start of service at the previous location plus the dwell time at the previous location plus the slew time from the previous location to the new location. (4.7) and (4.8) are both time window constraints. (4.9) is a

constraint that eliminates all sub-tours within the various vehicle routes to ensure that all are connected and there are no infinite or closed loops. Finally, (4.10) and (4.11) constrain the various problem variables to being binary and positive numbers, respectively.

After each Execution Stage of the scheduler, we receive feedback and any updated user inputs that are then incorporated back into the objective function and relevant constraints before resolving the optimization problem.

4.3.2 Dynamically Updating and Scheduling

In order to make the DSRS useful for realistic space surveillance it must be able to adapt to changes in the environment that may affect future or currently planned scheduling. It is important for the scheduling program to keep track of which task requests were completed successfully and which requests went uncompleted. The radar scheduler is also constantly receiving new information about the system, user objectives, and the arrival of new task requests throughout the planning horizon. This information must be incorporated into future scheduling, and in this section we describe how we update the information that serves as input in the MIP after each iteration to include past feedback and requests as well as new information.

4.3.2.1 Types of Feedback

Updating with new information and potentially altering the current schedule happens regularly at the beginning of each new iteration of scheduling following the Execution Stage *or* if the execution of the current schedule is stopped directly by the user. After either of these events occur, the DSRS will receive feedback about which tasks were completed and which were not. The scheduler will also keep track of radars that may still be in the process of executing a task request at the time of the feedback process and may not be immediately available to accept new tasks.

Feedback may come in a few different forms. One form of feedback is direct feedback from the user - this could be in the form of updated user weights that are

used to emphasize certain values in the objective function of the MIP, it could be an update as to which radars are unavailable for scheduling for a certain amount of time, or it could be more specific requirements about which radars must fulfill certain task requests.

Additionally, feedback information is regularly received automatically from the individual radar sites about the state of task requests in the system. Tasks may be *accepted but not completed*, *rejected*, or *successfully completed*. If a task is rejected by the system we assume that it will not be accepted in the future and it is removed from the list of task requests. This is, unless this decision is overridden by the user. Most often, this occurs when all available radars in the system have no way to feasibly execute the task request. If a task is accepted but not yet completed, the task request stays in the current consolidated tasking list for the system and may or may not be scheduled in the future. This depends largely on whether or not a newly updated schedule that incorporates the request is seen as an "improvement" by the system. If a task request is successfully completed in a previous execution phase, the request is either updated to reflect its next due time, or if the request requires no further observations for the day, it is documented as completed and removed from the consolidated tasking list altogether.

4.3.2.2 Choosing a Schedule

As was discussed above, feedback from previous planning iterations can affect the system-wide schedules and allocations of task requests. In some cases, a specific radar may accept a task request that makes it impossible to execute a previously accepted task request - the new task request essentially forces out a previously accepted one. In order to deal with the fact that rescheduling could possibly result in some requests being forced from the system if they are replaced by new ones, the scheduler must check for conflicts of current plans with existing plans. At each planning iteration t, we compute the objective value of the schedule based on the specified user weights. The incorporation of new information gained during the feedback phase could change the value of the objective function and have significant impact on the schedule, so it

is important to compare both the previous value with the value of a new schedule, taking into account any penalties associated with missing new tasks. It is important to note that we choose to evaluate the system cost when comparing different schedules on a global level rather than at a local level at each individual radar site. The only time the new, updated schedule is immediately chosen to replace the previous without any such comparison is if it is specified by the user. This capability comes into place most of the time in more operationally tactical or urgent scenarios where a RSO may need to be tracked and characterized immediately, regardless of the overall impact on the objective function value.

In the case of no user override, the DSRS will automatically compare the schedule based on the objective value of a previous schedule with adjustments for missing new tasks against the objective value in the newly created schedule. If $Obj\_Val^t \leq Obj\_Val^{t-1}$ then we forgo the previous schedule in favor of the new one and update problem parameters and radar path plans to reflect the decision.

4.4 Alternative Version of the DSRS

The DSRS is meant to employ optimization at every step of the planning process - in both the task distribution steps, where task requests are assigned to specific radars, but also the individual slew path plans of the radars. While this type of scheduling may be more ideal because of the ability to take a more global approach by optimizing for the entire surveillance system at once, it does come with computational expense. The DSRS is formulated in a graphical structure that includes nodes for every radar and satellite in the system and arcs for every possible combination of radar and satellite-to-satellite pairings. For a problem size that is typical of current scheduling, the scheduler is able to quickly solve to optimality and create schedules in a matter of seconds. With that being said, solve time could become impractical if problem sizes increase to much larger levels in the future - a possibility if space activities and services continue to rapidly expand in the coming years.

For this reason, it may be important to introduce various heuristics prior to opti-

mization that could help speed up the run time of the scheduling program. Although heuristics are not able to ensure global optimality, they are often able to significantly decrease solve time and still produce very useful and effective results. For this reason, we create an alternative version of the DSRS which we refer to as Alt-Scheduler.

In this scheduling approach, we use a heuristic to distribute the tasks to the radars before any optimization occurs in the problem. This heuristic is implemented immediately following the initialization step because it uses the resulting feasible sets of satellites for each radar. Satellites that are only feasibly observed by one radar are automatically assigned to those radars. After that, the radars alternate in accepting feasible task requests so that the number of task requests assigned to each radar is relatively even. This process continues until all of the task requests have been assigned from the planning period.

After this heuristic, the optimization program is equivalent to that of the DSRS for each individual radar. The problem set up and formulation is the same except that there is one radar and its predetermined list of task requests being optimized at a given time. This decreases the number of variables, constraints, and potential path plans, allowing for a faster solve time and output solution.

The goal of this thesis is to develop a completely centralized, coordinated scheduler. The DSRS has been developed to do this with optimization at every step, but it is not the only option. With the Alt-Scheduler we explore another scheduling approach and compare the results with that of the DSRS in Chapter 5.

4.5 Implementation of DSRS

This section describes our implementation of the above algorithms in software. We begin by describing the software architecture and detail the algorithmic flow of the DSRS. Then we discuss the format of the input and output data with examples, and finally we discuss the development of the experimental tests.

4.5.1 Software Architecture

To test our algorithm's ability to generate efficient schedules, we implement the algorithm in software that is available and widely used by one of the radar operators, Lincoln Laboratory. The radar characteristics for all radar sites are reviewed and maintained by Lincoln Laboratory in a macro-enabled excel file that we upload into a Python Jupyter notebook. The satellite data may be accessed by a csv file containing all satellite input characteristics or this information may be uploaded directly in the form of a TLE from space-track.org. We upload, generate, and clean the data for the problem and control the software using Python. To solve the scheduling optimization problems, we call Gurobi Optimizer, an optimization modeling system that utilizes a branch-and-cut algorithm.

4.5.2 Input Data

The data for requests being considered by the radar scheduler are input in one of the two ways previously mentioned - information may be read in from a csv file as seen in Figure 4-4 or this information may be uploading directly into Python from www.space-track.org. Both methods require data cleaning and formatting accomplished with the initialization program.

The radar data is all contained in one macro-enabled excel file that is maintained and updated routinely by the DoD laboratories. This file is also read into python for data cleaning and initialization.

The last pieces of information for solving the DSRS are the user inputs. The vector of user weights, the number of planning iterations and planning horizon, and the time of day are the required parameters. These values have default settings but may be changed at any time in the scheduling process.

4.5.3 Test Set Development - Operational Scenarios

Each run of the DSRS requires a significant amount of data input, so we automate the process of pulling satellites to be included in the consolidated tasking list and

Tasking_List

Name	Obj_Num	Intl Code	Launch Date	Period	Longitude	Label	rand
METEOSAT 3	19215	1988-051A	15-Jun-88	1485.3	154.3° W	4D	0.112659999
ANIK B1 (TELESAT-4)	11153	1978-116A	16-Dec-78	1442.7	105.9° E	3D	0.309818795
SKYNET 4F	26695	2001-005B	7-Feb-01	1452.3	26.2° W	3N	0.057335003
OPS 3151	9803	1977-007A	6-Feb-77	1476.5	178.6° W	3J	0.267188379
ANIK A1 (TELESAT 1)	6278	1972-090A	10-Nov-72	1457.1	31.6° W	4C	0.457792654
NSS 806 (INTELSAT 806)	25239	1998-014A	28-Feb-98	1453.4	6.2° E	3D	0.96159121
GOES 13	29155	2006-018A	24-May-06	1436.1	61.3° E	4N	0.883485332
AMC-21	33275	2008-038B	14-Aug-08	1436.1	124.9° W	1N	0.916431791
GSAT 19	42747	2017-031A	5-Jun-17	1436.1	47.9° E	4D	0.324161142
RADUGA 1M-1	32373	2007-058A	9-Dec-07	1448.8	80.9° E	2B	0.545240456
ITALSAT 1	21055	1991-003A	15-Jan-91	1440.6	113.9° W	2D	0.599615503
TDRS 13	42915	2017-047A	18-Aug-17	1436.1	11.6° W	1D	0.699675609
KOREASAT 5	29349	2006-034A	22-Aug-06	1436.1	113° E	2D	0.542720465
GARUDA 1	26089	2000-011A	12-Feb-00	1441.3	106° E	4C	0.41128797
GAOFEN 4	41194	2015-083A	28-Dec-15	1436.1	105.7° E	3D	0.811811553
ESA/GOES	9931	1977-029A	20-Apr-77	734	151.3° W	1B	0.322619813
RADUGA-1 8	34264	2009-010A	28-Feb-09	1442.7	56.5° E	4B	0.170734585

Figure 4-4: Example Input Data

generating task request requirements with a program in Python to randomly and quickly generate realistic request data.

Although this program is used for all testing of the scheduler, we control the volume and rate of requests received and the specific task request characteristics to give rise to more operationally tactical scenarios for testing. Some data were chosen manually based on realistic operational values after consulting with engineers in the advanced sensors field. Other values such as satellite RCS, revisit times, and observation request arrival rates were chosen from a set of realistic values that was constructed after consulting engineers in the advanced sensors field. Satellite positions for these scenarios were also extracted from www.space-track.org and [26] and thus also represent real satellite locations and information.

Random number generators and the constructed sets are used to create the request data. The request priorities and penalties are chosen from a discrete uniform distribution, while the time windows and RSO sizes are chosen from continuous uniform distributions with reasonable upper and lower bounds.

4.5.4 Output

The final output of the program provides the assignments of requests to radar sites and the planned start time of all task request servicing. The file also contains important satellite information for locating and tracking in the future and includes information about when and how often the satellite will need to be serviced again. This published schedule can be seen in Figure 4-5. We additionally output a separate file for each individual radar site containing each radar's specific slew path plan and timing throughout the planning horizon. These files are updated throughout the planning process to reflect any changes in scheduling, so that the latest version produced is the most up-to-date.

Full_Day_Test

Obj_Name	Longitude	RCS	Priority	Revisit_Time	Rem_Revisits	Due_Time	Service_Time
4510	-135.8	10	3	60	23	61.810319523706100	1.8103195237060800
11484	-27.1	6	3	60	23	61.7861108587864	1.786110858786400
29272	128.0	0	3	60	23	67.27475424528800	7.274754245287960
15993	-65.8	7	3	60	23	64.4952637537237	4.4952637537237
11558	139.4	8	3	60	23	68.40416565860100	8.40416565860099
26719	-132.9	10	3	60	23	63.48363582011080	3.4836358201108200
25331	-135.2	-2	2	360	3	361.96760163513400	1.967601635133630
23651	175.3	9	2	360	3	365.37566022288700	5.375660222887260
4510	-135.8	10	3	60	22	128.36937504820400	68.36937504820360
11484	-27.1	6	3	60	22	121.79671464706500	61.79671464706500
29272	128.0	0	3	60	22	123.70680851296200	63.70680851296230
15993	-65.8	7	3	60	22	123.16144592811200	63.16144592811240
11558	139.4	8	3	60	22	125.9442890786390	65.9442890786389
26719	-132.9	10	3	60	22	129.4339506272470	69.4339506272471
41903	110.1	-5	2	360	3	433.29642317799400	73.29642317799430
39616	-74.0	10	2	360	3	424.3268632210000	64.3268632209995

Figure 4-5: Example Published Schedule Output

We also report on the primary Measures of Performance after the program is complete. Intermediate output updates on these Measures of Performance and feedback on the status of task requests are possible but must be specified by the user.

Chapter 5

Results and Analysis

This chapter presents and evaluates the results of the DSRS. In the first section we compare the results of the scheduler against two other scheduling approaches – a greedy scheduler, and the modified version of the DSRS, the Alt-Scheduler. We compare the solution approaches by comparing the run time results as well as their achieved Measures of Performance. In the first section we also present the results of a more specific cost analysis that was conducted between the three scheduling methods to better examine how price may affect the overall created schedules.

The chapter then focuses on the objective function of the optimization portion of the DSRS. We perform various empirical tests to ensure the function responds as is intended when the user weights are adjusted.

The third section utilizes the introduced operational scenarios to analyze the performance of the scheduler. We analyze the achieved Measures of Performance, program run time, and any gaps in coverage or custody to evaluate the performance of the scheduler in these more operationally tactical scenarios that are likely to be more common in the future.

5.1 Comparison of Scheduling Approaches

In this section we provide an initial evaluation of the performance of three different scheduling approaches, each varying in the level of coordination and optimization

present in task distribution and assignment. To provide an initial evaluation on which method achieves the best results in the deep space radar scheduling problem, we analyze two characteristics: the run time and the overall radar slew times achieved in each method.

The three algorithms that are discussed in this section are the Greedy Scheduler, an alternative version of the DSRS, the Alt-Scheduler, and the DSRS.

The Greedy Scheduler plans operations in a manner that automates the way current operational scheduling in manually done. Each radar site is given a randomly assigned predetermined tasking list at the beginning of the planning period. After receiving the list, the scheduling process is very straightforward – the radar simply orders and executes tasks in the order of the next soonest due time. It is important to note that while the Greedy Scheduler does function as a baseline comparison in the following examples, it is still a faster, more efficient scheduling approach to the current way radars in the DoD are scheduled due to the fact it produces schedules automatically versus employees manually creating schedules each day.

The Alt-Scheduler schedules in a way that preserves the feature of current operations wherein the task requests are not optimally assigned to radar sites, but instead uses a heuristic based on feasibility to distribute the requests. The scheduler does however optimize the actual path the radar takes to slew to these locations. After receiving the predetermined tasking list for a specific radar, the scheduler then works to meet the time constraints of the problem while minimizing the time the radar spends slewing across the sky.

Finally, the DSRS optimizes the entire process of task distribution, assignment, and path planning. Rather than receiving a subset of the tasking list at each site, the full profile of task requests for a planning period serve as input into the centralized DSRS. From there, the tasks are distributed to each radar site with an accompanying path plan that attempts to minimize slew time. Because there are a number of possible radar-satellite paths that are possible in this scenario, we attempt to speed up solve time by reducing the solution space before optimization occurs with feasibility checks that eliminate any infeasible solutions based on radar capacity and field of view

restrictions.

We expect that the DSRS will have the longest solve time due to the larger set of variables and constraints in the mixed integer optimization portion of the problem. We also expect that this solution will achieve the best resulting operations plan due to its more global optimization approach rather than the more localized approach with Alt-Scheduler and the lack of optimization in the Greedy Scheduler.

5.1.1 Testing Method

To analyze the various scheduling approaches, thirty pseudo-random tests were generated by randomly sampling in the manner introduced in Chapter 4. This method allows us to manipulate the proportion of objects at certain priority levels in the tasking list. For these experiments, we chose to include various problem sizes of satellites from a list of High Interest Objects (HIOs). These HIOs are typically assigned a level two or three priority and only have one required revisit during the planning horizon. We choose to test with HIOs because it makes the final observation plans simpler to compare. The problem instances include satellites of various sizes, highly dispersed locations, and encompassing a range of dwell requirements.

There are five test sets for each of the following problem sizes: 80, 100, 120, 140, 200, and 260. In each of these cases we also include all four radar sensors as available for scheduling. We begin by increasing the problem size by increments of 20 when we analyze the more realistic task request sizes of 100 - 140, but then we increase the increment the problem sizes by increments of 60 to analyze the limits of solve time in the larger test cases of 200 and 260 task requests. For the set of run time tests alone we do not employ the batch scheduling approach that we normally use in the DSR-Schedule and Alt-Scheduling approaches because we want to evaluate the time it takes each method to solve while working to schedule all tasks at one time.

5.1.2 Run Time Comparison

The goal of the run time comparison is to determine how fast the Greedy scheduler, the Alt-Scheduler, and the DSRS are able to create operations plans that solve the deep space radar scheduling problem. While we are concerned with which scheduling approach solves the fastest and slowest, we are mostly interested in examining which of the schedulers is able to solve large problem instances in "reasonable" amounts of time for military operational scheduling. After consulting with radar operators, we determined that a "desirable solve time" for operationally tactical or urgent scenarios need be below 60 seconds. To test this capability, the thirty test sets discussed in the previous section were solved using each method. Each test set size was solved five times to ensure trends were consistent across samples.

We expect that the Greedy Scheduler will have the fastest run time by far – this is because the scheduling approach and resulting algorithm is relatively straightforward and simple. We expect the Alt-Scheduler and the DSRS to have longer run times because we are actually optimizing the slew paths in these these algorithms, and it takes time for various path plans to be compared and formed. We expect that DSRS will have the longest run time to optimality because the solution space is much larger due to its optimization of both task allocation and the task execution path plan. Table 5.1 provides the run times of each of the testing runs.

The data presented in Table 5.1 indicate that for each test set the Greedy Scheduler indeed created a path plan for the radars in the shortest amount of time. The Alt-Scheduler solved in the second fastest amount of time and still achieved desirable solve times even for the largest test case of 260 HIO task requests, with the maximum solve time for the Alt-Scheduler being only 13.71 seconds.

By comparison, the DSRS was able to achieve consistently desirable solve times for test cases up to 120 HIO task requests. Beyond this number, the program was still able to solve the problem to optimality, but with solve times that fluctuated from below 30 seconds to upwards of 1,000 seconds of solving in one of the large test cases.

Table 5.2 shows the averages of the previous run times results for each problem

Problem Size	Greedy (s)	Alt (s)	DSR (s)
80	0.12	1.34	7.34
80	0.12	1.40	5.88
80	0.11	1.45	5.63
80	0.13	1.86	5.44
80	0.12	1.20	9.12
100	0.13	1.78	11.19
100	0.14	1.72	10.87
100	0.13	1.59	10.02
100	0.18	1.94	9.61
100	0.14	1.90	9.13
120	0.15	2.03	11.76
120	0.15	1.91	13.62
120	0.15	2.25	13.60
120	0.16	1.85	18.54
120	0.15	2.04	10.89
140	0.19	2.62	65.34
140	0.20	2.94	36.55
140	0.17	3.45	23.14
140	0.18	2.84	117.12
140	0.17	3.17	53.75
200	0.23	4.95	74.46
200	0.54	6.00	97.37
200	0.23	5.47	56.29
200	0.27	5.16	63.28
200	0.24	5.33	54.51
260	0.65	10.55	1161.11
260	0.23	7.94	281.99
260	0.47	7.71	299.26
260	0.49	13.71	283.91
260	0.48	6.81	312.74

Table 5.1: Run time results for schedulers at various problem sizes

Problem Size	Greedy (s)	Alt (s)	DSRS (s)
80	0.12	1.45	6.68
100	0.15	1.79	10.16
120	0.15	2.02	13.68
140	0.18	3.00	59.18
200	0.30	5.38	69.18
260	0.46	9.34	467.80

Table 5.2: Average run times for schedulers at various problem sizes

size tested. The average run times support that both the Greedy Scheduler and Alt-Scheduler are able to solve all problem size test cases in an amount of time that is useful and desirable in tactical military scenarios. The DSRS is, on average, able to schedule problem instances up to 140 task requests in the appropriate amount of time for a more urgent scenario. It is important to note that engineers estimate the number of incoming task requests does not exceed 100 in a day currently. So while the DSRS is fully able to handle even higher fluctuations in current scheduling, if in the more distant future this number does indeed grow, additional heuristics like the one employed in the Alt-Scheduler could be added to DSRS to reduce solve time in larger problem instances.

Another important metric to consider is the variance of the run times. Because the scheduling approach is for the use of the Department of Defense, a consistent solve time is highly preferable. If a scheduler takes much longer than expected, it could put behind critical operations and opportunities for appropriate response or action. For this reason, we report the standard deviations of the run times for the three different scheduling algorithms in Table 5.3.

Problem Size	Greedy (s)	Alt (s)	DSRS (s)
80	0.01	0.24	1.55
100	0.02	0.14	0.85
120	0.01	0.15	2.96
140	0.01	0.31	36.17
200	0.13	0.39	17.59
260	0.14	2.81	387.77

Table 5.3: Standard deviation of run times for schedulers at various problem sizes

The DSRS has a standard deviation that is magnitudes higher than the other two scheduling approaches, very dependent on the problem size and the specific random problem instance. We once again see the benefit to an initial heuristic before optimization in the Alt-Scheduler. In this case we see its effect on the variance of the run time, as it is able to achieve a much smaller magnitude standard deviation than the full optimization approach in the DSRS.

5.1.3 Slew Comparison

The overall time that radars in the surveillance system spend slewing during a planning period is an extremely important metric for comparison. Not only is the slew time a large driver in the overall cost of operations for the day, but it also determines how much time is leftover in a planning period that may be allotted to fulfill other important space surveillance missions and tasks. Two of the scheduling approaches, the Alt-Scheduler and DSRS, use optimization as an approach to scheduling. These algorithms allow the user to input specific weights on the elements in the objective functions to emphasize certain metrics while meeting hard constraints. In the following test cases, we choose to emphasize the lateness element in the objective function so that the most weight is associated with minimizing tardy or missed tasks. In later results we show the effect that increasing the weights on the slew or cost variables has on the overall slew time in the system.

To compare the total slew time in the resulting operations plans produced by each scheduling approach, we once again run the thirty tests discussed in Section 5.1.1 and used in the Run Time Comparison Section.

The results in Table 5.4 indicate that the DSRS is able to minimize the achieved total slew time in a majority of problem instances. In a few tests, the Alt-Scheduler has a lower total slew time than the DSRS. This is due to the fact that the Alt-Scheduler uses a heuristic for the initial assignment of tasks to radars whereas the DSRS uses optimization. The DSRS works to minimize the total slew time but also distributes tasks to radars in a way that minimizes operational costs. For example, HUSIR-X has a higher price associated with radar operation, so even though it may be able to slew to a satellite in a shorter amount of time than one of the other radars in the system, the DSRS may not always assign the task request to HUSIR-X if it is actually cheaper to assign that request to a different, even further radar. We provide the average slew results for each problem size test in Table 5.5.

It is once again important to consider the variance in slew times achieved by each of the scheduling methods. If radar operators are able to consistently estimate the

Problem Size	Greedy (min)	Alt (min)	DSRS (min)
80	189.92	94.58	92.05
80	172.94	91.55	87.07
80	164.13	102.58	96.19
80	193.92	91.59	91.45
80	157.03	88.72	87.83
100	222.84	124.95	112.18
100	196.78	121.27	112.38
100	225.67	120.39	120.04
100	182.46	119.52	111.09
100	233.06	120.98	105.36
120	259.24	122.35	131.12
120	231.46	135.23	124.26
120	244.41	139.26	131.93
120	265.79	128.58	124.68
120	257.81	134.24	114.31
140	281.28	169.90	142.25
140	368.91	158.24	153.71
140	315.23	156.17	152.44
140	306.48	178.50	168.07
140	305.09	167.04	139.49
200	236.27	36.28	34.53
200	271.23	39.47	38.14
200	264.24	40.64	34.86
200	267.78	36.20	34.15
200	261.80	38.86	34.71
260	333.53	50.09	42.83
260	288.41	53.11	48.66
260	376.41	52.22	44.79
260	246.31	52.64	43.29
260	309.02	46.68	41.63

Table 5.4: Slew time results for schedulers at various problem sizes

Problem Size	Greedy (min)	Alt (min)	DSRS (min)
80	29.26	15.63	15.48
100	35.36	20.23	18.72
120	41.95	21.98	20.87
140	52.56	27.66	25.19
200	260.26	38.29	35.27
260	310.74	50.95	44.23

Table 5.5: Average slew times for schedulers at various problems sizes

amount of time a radar will spend slewing, they are better able to determine how much time is available for other mission tasks or outages. We present the standard deviations of the slew times for the three different scheduling algorithms in Table 5.6.

Problem Size	Greedy (min)	Alt (min)	DSRS (min)
80	2.66	0.88	1.04
100	3.58	0.34	0.87
120	2.29	1.09	1.17
140	5.41	1.51	1.88
200	13.87	1.97	1.62
260	48.68	2.64	2.72

Table 5.6: Standard Deviation of slew times for schedulers at various problem sizes

The Greedy Scheduler has a standard deviation that is magnitudes higher than the other two scheduling approaches - this value increases as problem size increases and is very dependent on the randomly sampled task requests. We see that both the Alt-Scheduler and DSRS have much lower standard deviations, indicating that the average amount of time the radars spend slewing in a planning period is more predictable and is less likely to have high fluctuations.

5.1.4 Cost Analysis Comparison

In this section we run a more substantial comparison on the cost achieved by each of the models for pseudo-random samples of HIO satellites. We once again include satellites with one required revisit during planning for ease of comparison, but for these tests we keep the sample size at a constant 100 task requests per planning period.

As was previously mentioned, each radar has a particular price associated with its operation. This price serves as a sort of "price weight" because it is scaled before optimization. In this section, we vary the base cost associated with radar operation, but leave the weights at a constant ratio because these values do not change. For example, if the base cost is \$1 and the price weight for a radar is 0.5, then the operational price for the radar is (\$1 x 0.5) = \$0.5. One can imagine that cost of operating radars for space surveillance could change for a multitude of operational

reasons or gradually over time with the rise and fall of prices of other goods and services. For this reason, we test the operational cost at values from the set [.1, 15.0] dollars per operational minute.

The average results of these experiments are reported in the Appendix, with the Average Total Cost values reported in hundreds of dollars. The results show that the DSRS is consistently able to distribute tasks and create slew path plans for the radars that achieve a lower cost than the other two methods. This is due to the fact that the DSRS optimizes both the radar-task assignment process and the radar slew path plans simultaneously, so it is able to assign task requests to the radars and with start times in the cheapest way possible, even as price increases. The results also indicate that the Alt-Scheduler is still able to achieve lower operational costs than the Greedy Scheduler in all instances due to its ability to optimize the slew path plan of the radars during scheduling.

Another important observation is that the Greedy Scheduler's cost of operations experiences many different fluctuations whereas both the Alt-Scheduler and the DSRS are able to remain much more steady with changes in price. High fluctuations in cost are very undesirable because of the difficulties it may cause in budgeting for the radar sites. Having the ability to accurately predict price of operations allows for better resource allocation and builds confidence in the scheduling approach. We see that as operational cost increases, these patterns and average differences in results become all the more pronounced.

5.2 Objective Function Component Analysis

We use the objective function introduced and detailed in Chapter 4 as the way for users to adjust optimization outcomes to reflect different mission goals. In this section, we conduct empirical testing to observe how modifying the user weights in the objective function can be used to lead to different scheduling outcomes.

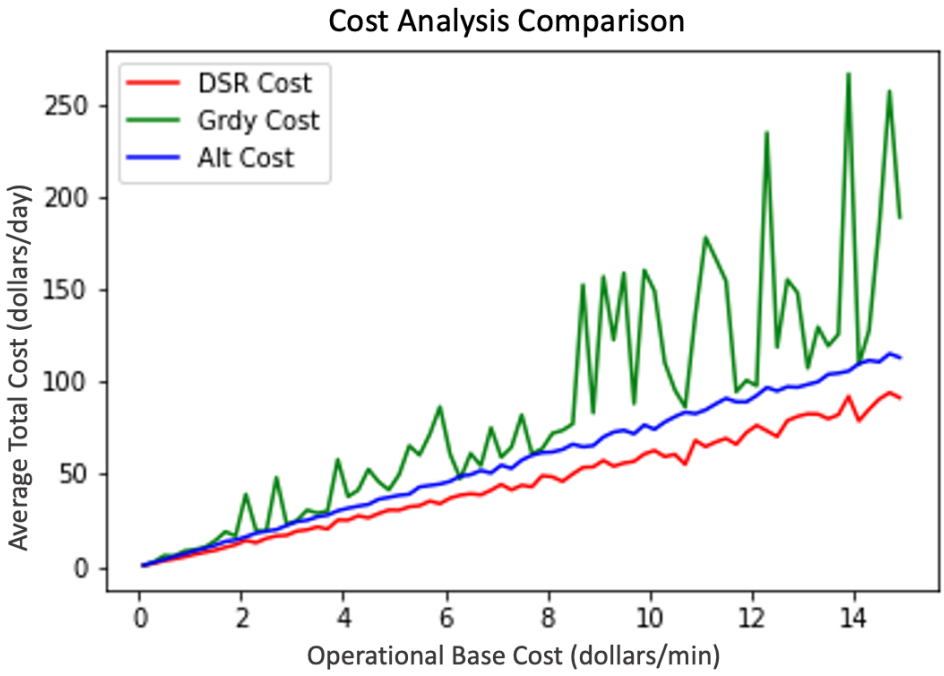


Figure 5-1: Cost Analysis and Comparison

5.2.1 Full Day Experiments with DSRS

For this section we conduct experiments where we vary the input user weight u for the various objective function components: the *lateness* component, the $final\_slew$ component, and the op_price components. Although there are more components introduced in Chapter 4 that may be minimized in the objective function, we choose to focus on these three elements in the analysis because they are the most common features that radar operators are currently likely to seek to control during scheduling.

For each of the sections to follow, tests are performed using psuedo-random samples of 100 task requests. Each test contains the same proportion of task requests from each priority level - 70% low priority (priority level one), 20% medium priority (priority level two), and 10% high priority satellites (priority level three). In this test set, high priority satellites must be revisited every hour, medium priority satellites must be revisited every six hours, and low priority satellites only require one visit per day. This gives a total of 390 total satellite task request visits to schedule in the planning period. Additionally, to make the scenario more realistic and dynamic we also add in one additional randomly drawn high priority task every two hours. We

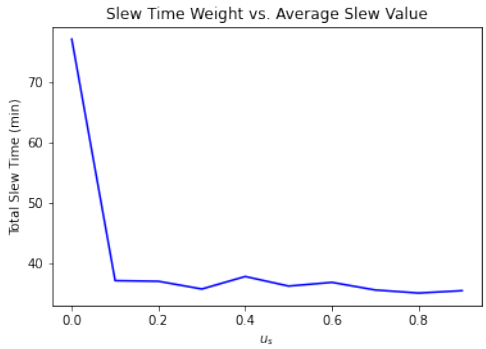
allow all four radar sensors to be available for scheduling in the entire planning period in the scenarios. For these experiments we employ a rolling horizon type scheduling approach where we schedule the task requests in batches based on the next soonest due times. We choose to solve the problem with a 24 hour planning horizon and 24 iterations, or one per hour. We report all cost results in hundreds of dollars.

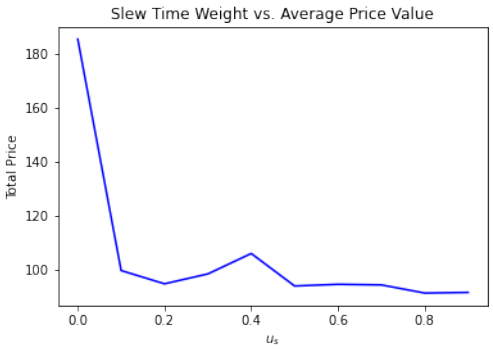
5.2.1.2 Total Slew Component

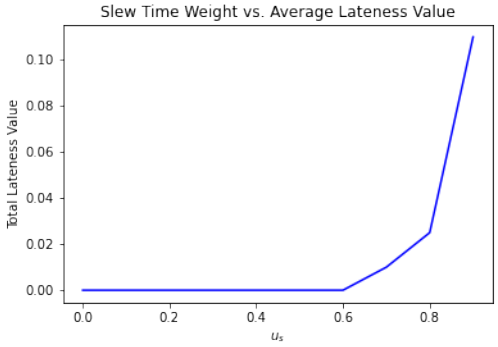
In this section we show the effect of increasing the overall slew time component's user weight, u_s, while keeping all other input weights at 0 except for the the overall lateness parameter, which we set equal to $1 - u_s$. The lateness component enforces the time constraints and gaps in coverage by penalizing tardiness or missed tasks. We average the output values for our Measures of Performance over five runs for each of the 11 values of u_s in the range [0.0, 1.0]. The average results of these tests are reported in Table 5.7.

u_s Weight	Avg Slew (min)	Avg Total Cost (USD)	Avg % Dropped
0	77.26	185.43	0
0.1	37.03	99.91	0
0.2	36.92	95.06	0
0.3	35.64	98.65	0
0.4	37.72	106.21	0
0.5	36.13	94.21	0
0.6	36.74	94.81	0
0.7	35.47	94.59	1
0.8	34.96	91.58	2.5
0.9	35.36	91.82	11
1.0	0	0	100

Table 5.7: DSRS - Change in u_s Results







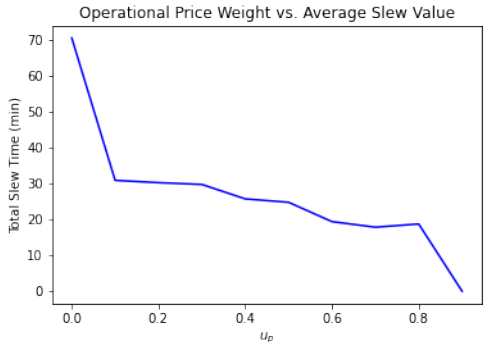
By design, when the weight u_s associated with the $final\_slew$ is emphasized in the objective function during scheduling, the slew time decreases on average. We see that the slew time is minimized and tends to flatten out as there are no better options for reducing overall slew time in the system, although there are some irregularities that are likely data-driven because task requests are randomly sampled and added to the program during scheduling. We see that as $final\_slew$ is weighted at around 0.7 or higher in the model, the scheduler starts opting to drop certain task requests in an effort minimize slew time because it is so highly emphasized in the model.

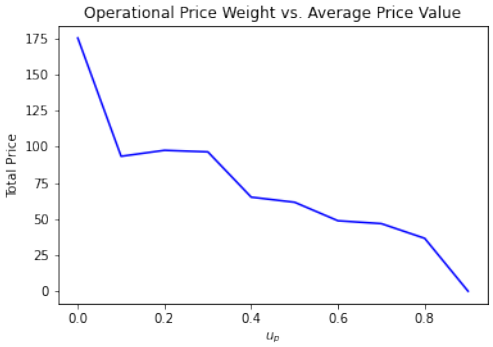
5.2.1.3 Total Price Component

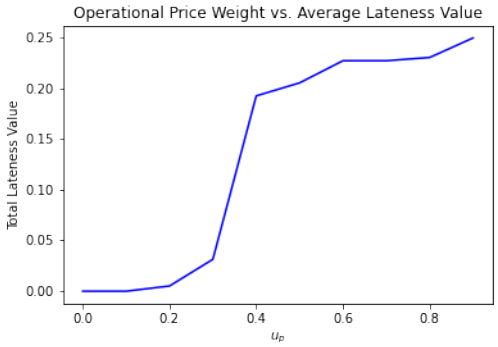
In this section we show the effect of increasing the overall operational price component's user weight, u_p, while keeping all other input weights at 0 except for the the overall lateness parameter, which we set equal to $1 - u_p$. We average the output values for our Measures of Performance over three runs for each of the 11 values of u_p in the range [0.0, 1.0].

u_p Weight	Avg Slew (min)	Avg Total Cost (USD)	Avg % Dropped
0	70.50	175.31	0
0.1	30.86	93.39	0
0.2	30.21	97.53	2
0.3	29.72	96.44	12.25
0.4	25.70	65.10	75
0.5	24.75	61.57	80
0.6	19.36	48.76	88.5
0.7	17.82	46.78	88.5
0.8	18.71	36.60	89.75
0.9	0	0	97.25
1.0	0	0	100

Table 5.8: DSR - Change in u_p Results







In this case we see that the user weight u_p associated with the *op_price* component has a roughly linear relationship with average slew time and average price values. This is expected as the average price value is so closely tied to the amount of time the radar spends slewing across the sky. We also see a roughly positive linear relationship with the price weight and the average lateness value. This is due to the fact that as radar

operations become more expensive, the radar is more likely to drop tasks or execute tasks outside of their specified time windows if it decreases the operating costs.

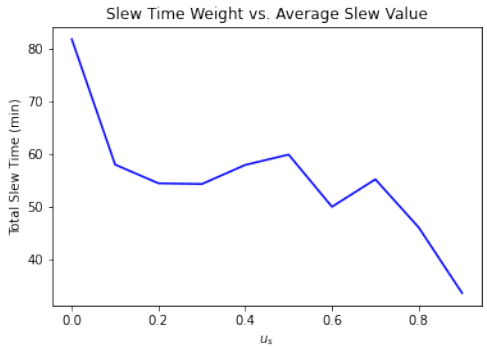
5.2.2 Full Day Experiments with Alt-Scheduler

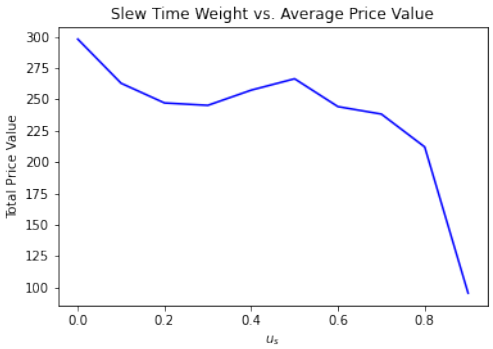
For this section we conduct experiments where we vary the input user weights in the same way as the Section 5.2.1 with the DSRS, but this time with the Alt-Scheduling approach. We once again vary the user weights for the selected objective function parameters: the *lateness* component, the $final\_slew$ component, and the op_price components.

The tests on the Alt-Scheduler are identical to those run on the DSRS. Each test set 0f 100 task requests contains 70% low priority, 20% medium priority, and 10% high priority satellites. We once again introduce one randomly drawn additional high priority task every two hours. We allow all four radar sensors to be available for scheduling in the entire planning period in the scenarios.

5.2.2.2 Total Slew Component

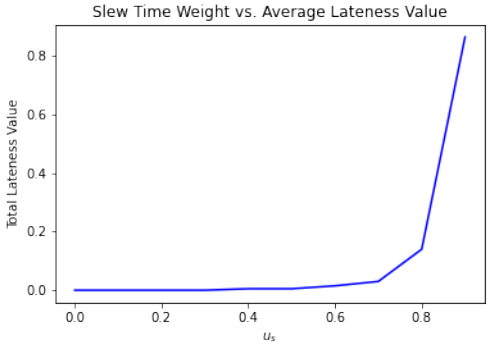
Here we show the effect of increasing the overall slew time component's user weight, u_s, while keeping all other input weights at 0 except for the the overall lateness parameter, which we set equal to $1 - u_s$. We average the output values for our Measures of Performance over three runs for each of the 11 values of u_s in the range [0.0, 1.0]. The average results of these tests are reported in Table 5.9 below.





u_s Weight	Avg Slew (min)	Avg Total Cost (USD)	Avg % Dropped
0	81.77	298.18	0
0.1	58.01	262.95	0
0.2	54.48	247.26	0
0.3	54.34	245.35	0
0.4	57.95	257.48	0.5
0.5	59.91	266.54	0.5
0.6	50.03	244.32	1.5
0.7	55.23	238.39	3
0.8	46.12	212.18	14
0.9	33.71	95.66	86.5
1.0	0	0	100

Table 5.9: Alt - Change in u_s Results



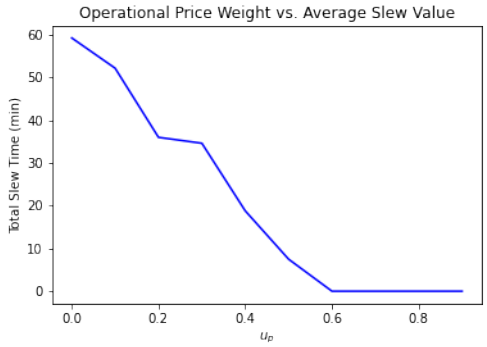
We can see in the results that an increase in the u_s variable indeed decrease the average slew time and average cost, but at a much slower rate than in the DSRS. This is most likely due to the fact that the Alt-Scheduler is only able to minimize slew time through the individual radar slew path plans, rather than also by optimizing task distribution for this goal. In these experiments we see the slew time reach more of a steady state in the range of 55 – 60 minutes. The slew time only dramatically decreases when the slew component weight is significant enough that the Alt-Scheduler begins dropping many more tasks. This is also why we see a large jump in the lateness value around a weight of 0.6 – 0.7.

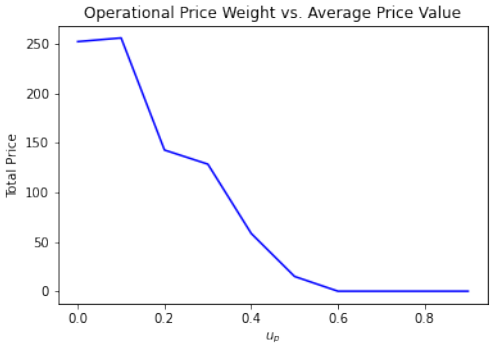
5.2.2.3 Total Price Component

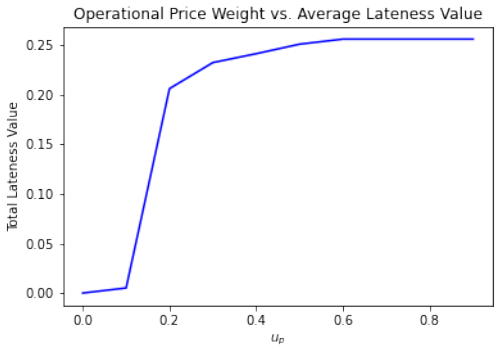
Here we show the effect of increasing the overall operational price component's user weight, u_p, while keeping all other input weights at 0 except for the the overall lateness parameter, which we set equal to $1 - u_p$. We average the output values for our Measures of Performance over three runs for each of the 11 values of u_p in the range [0.0, 1.0]. These are reported in Table 5.10.

u_p Weight	Avg Slew (min)	Avg Total Cost (USD)	Avg % Dropped
0	79.18	292.26	0
0.1	52.11	255.97	2
0.2	35.95	142.68	80.5
0.3	34.57	128.41	90.75
0.4	18.78	58.37	94.25
0.5	7.48	14.90	98
0.6	0	0	100
0.7	0	0	100
0.8	0	0	100
0.9	0	0	100
1.0	0	0	100

Table 5.10: Alt - Change in u_p Results







We once again see a negative relationship in the weight associated with the operational price, u_p, and the average slew and average price values. Because the Alt-Scheduler is unable to redistribute tasks to radar sites in order to decrease operations, we see that tasks are dropped earlier to decrease prices instead. Many of the early dropped tasks come from those assigned to the more expensive radar sites to operate, such as HUSIR-X or Altair.

5.3 Operational Scenarios

Because the deep space radar scheduling problem in this thesis is for the purpose of SSA as it pertains to national defense, it is important to consider this context when testing the algorithm. The DSRS is designed for use by the U.S. DoD, with hopes of being useful in the future even as space continues to grow and progress. We now turn to the more tactical operational scenarios introduced in Chapter 1, some which may occur in current operations and others which are likely to become part of the deep space radar mission in the future.

For each of the following scenarios, we choose to test the full version of the DSRS and use preset user weights that place a majority of the emphasis on the timeliness of task completion with the rest of the emphasis on minimizing the operational price for a day of surveillance. We choose to calculate the base price at \$1/min of radar operation. Because all of these scenarios are considered tactical or urgent, we allow the scheduler to run for a maximum of 60 seconds before cutting off and accepting the operations plan; however, in all of the scenarios the scheduler was able to schedule to

optimality in the alloted time due to the rolling horizon batch scheduling approach.

5.3.1 Foreign launch

In this scenario, we assume that a foreign launch occurs and requires urgent characterization and tracking. For this reason, we take all deep space radar sensors in the system away from the custody mission to put towards the foreign launch detection mission for a time period of two hours. While in some cases this scenario may not require all radars be "unavailable" for scheduling in the DSRS, we model the effects of taking one radar out of the system at a time in a later subsection.

We test this scenario by modeling with 100 task requests as in previous sections. Each test contains the same proportion of task requests from each priority level - 70% low priority (priority level one), 20% medium priority (priority level two), and 10% high priority satellites (priority level three). In this test set, high priority satellites must be revisited every hour, medium priority satellites must be revisited every six hours, and low priority satellites only require one visit per day. We once again add in one randomly drawn high priority task request every two hours to make the scenario more realistic. For each test, we choose a random $global\_time$ to indicate a foreign launch in need of characterization has occurred. Planning for the deep space custody mission halts for a period of two hours before resuming for the rest of the day. The scenario is conducted 10 times, and the average results are reported below in Table 5.11, once with the results from no launch and then again on the same problem instance with the indication that a foreign launch has occurred.

Scenario	Avg Slew (min)	Avg Total Cost (USD)	Avg % Dropped
No Launch	41.97	127.59	0
Launch	39.34	128.17	12.0

Table 5.11: Foreign Launch Results

Interestingly, we see that while the average slew time in the event of a launch is slightly lower, the average cost is slightly higher than when no launch occurs. This is due to the fact that as the DSRS gets back online after the launch, it attempts to

"catch up" on previous tasks even if they may be served late. As the system recovers from the launch, tasks may be sent to radars that are available, albeit more expensive, leading to the increased cost. Because the scheduler was offline, 12% of objects still do not have guaranteed custody for the full length of the planning period due to the outage.

These results show the benefit of having a way of scheduling that is centralized and allows for coordination between sensor sites. The DSRS is able to regain custody of objects very quickly following the foreign launch because it is able to quickly distribute tasks in a way that allows for swift recovery on a system-wide level. The DSRS is able to do reassign and reschedule in a way that maximizes custody for the remainder of the planning period even after the unscheduled outage event.

5.3.2 Overload of Observation Requests

In this scenario, we assume that for a given planning period of scheduling a large number of task requests are sent as input to the DSRS compared to what is considered regular and routine. After consulting with radar operators, currently a scheduling day will consist of a maximum of 100 task requests. For this scenario, we choose to test with double this amount and schedule a day with 200 task requests.

We draw the 200 task requests and keep the same proportion of priority levels as in previous tests - 70% low priority (priority level one), 20% medium priority (priority level two), and 10% high priority satellites (priority level three). We also continue to add in one randomly drawn high priority task every two hours. The scenario is conducted ten times, and the average results are reported below in Table 5.12.

Avg Slew (min)	Avg Total Cost (USD)	Avg % Dropped
56.69	224.49	0.4

Table 5.12: Task Request Overload Results

Because we choose to emphasize guaranteed custody of all objects as well as minimization of the operational price, while the average slew time and average values are indeed higher with more task requests, the scheduler was still able to produce

operations plans with less than one, only .4%, of objects that did not have guaranteed custody, on average.

These results once again show the value in having an automatic, central scheduler when it comes to planning radar operations. With double the number of task requests from what is commonly seen, the current, isolated system of scheduling would further waste valuable money and resources. Additionally, without a way to intelligently plan task distribution and path assignment, there is also the increasing chance that we lose custody of many more RSOs than what is possible by employing optimization. Finally, further human time and energy is wasted manually creating the schedule for this unprecedented number of task requests while the automatic scheduler is able to adjust and plan accordingly in a matter of seconds.

5.3.3 Many High Priority Targets

In this scenario, we imagine a situation in which there is a very large proportion of high priority satellites (priority level three) that need to be observed every hour of the planning period. This scenario is likely in the future if objects become more maneuverable or if more critical national security services are provided in space that need almost continuous coverage to guarantee custody.

To model the scenario, we again draw 100 task requests from the master list of satellites kept and maintained by the U.S. military. This time, however, we alter the proportion of high priority satellites from 10% to 30%, and then 20% of medium priority satellites and 50% of low priority satellites. This brings the total task requests up to 850, once again with the additional high priority task requests being added in randomly every two hours. The scenario is conducted 10 times and the average results are reported below in Table 5.13.

Avg Slew (min)	Avg Total Cost (USD)	Avg % Dropped
60.97	236.19	0.2

Table 5.13: Many High Priority Task Requests Results

We once again see an increase in the average slew time and average cost values

due to the increase in the number of task requests in this scenario. We do see that in this particular case however, the number of average satellites that the scheduler is unable to keep custody of is only 0.2%. This reflects the fact that, although there are many objects the radar surveillance system is charged with keeping custody of, many of these additional task requests are high priority and therefore incur a much larger penalty if they are dropped from the schedule than the lower priority tasks. Indeed, after the lower priority tasks have been visited for their required visits, most of the radars' operations plans include it slewing to high priority objects for continued coverage.

For many of the same reasons as is the case in the Overload of Observation Requests Scenario, the DSRS's automatic, centralized approach to planning radar operations is extremely beneficial. While slew time is inevitably increased with so many more high priority task requests, we see that the average items we are able to keep in custody is also very high. This assignment and scheduling is done quickly and intelligently so that the user does not have to manually adjust.

5.3.4 Radar Outage

The final scenario for scheduling is the Radar Outage Scenario. In these experiments, we assume that one of the deep space radars in the space surveillance system is unavailable for the entirety of the planning period. This could occur if a radar requires maintenance, if the radar is needed for another mission, or any other number of other various user-specified reasons. In this section we explore how losing a radar in the system affects the system gaps in coverage as well as the overall cost of operation.

Testing for this scenario is conducted with a consolidated tasking list that contains 100 task requests with the same 70/20/10 split in low/medium/high priority level satellites. The tests were conducted with each radar removed from the system ten times. The results for for each radar being removed from the surveillance system are reported below in Table 5.14.

In these results we can see that the most dramatic increase in cost occurs when Radar_4, located in White Sands, Arizona, is removed from the radar system. When

Removed Radar	Avg Slew (min)	Avg Total Cost (USD)	Avg % Dropped
HUSIR-X	47.24	137.86	0
MHR	43.23	145.27	0
Altair	33.74	75.59	34.4
Radar_4	42.72	160.92	0

Table 5.14: Outage Results

this occurs, the other radars are able to adjust to take over the task requests and maintain custody of all objects, but it does require more expensive radars and slew path plans on average.

While the average cost and average slew time values are the lowest when Altair is removed from the surveillance system, this is largely due to the fact that this is the only radar that leads to a large gap in coverage when removed. Indeed, about 34% of objects do not have guaranteed custody in this case. This is due to the physical location of Altair - in the Marshall Islands. The other radars in the system are geographically located such that they are unable to fill in the gaps in custody because they are unable to adequately make observations. Overall, these scenarios do display the benefit to a centralized and coordinated way of scheduling the radar surveillance system. When one radar is removed, the other radars in the system are largely able to take over and keep custody of objects, which is not the case when scheduling is done independently. With current, "stovepiped" system of scheduling, a radar outage would simply mean that the radar's assigned tasks would go completely unfulfilled for the length of the outage. Alternatively, with the DSRS's system of planning there is limited loss of custody from a system perspective. The only time when items go completely unobserved when using the default user objective weights is when the remaining radars are not feasibly capable of executing the task requests.

The results of these scenarios provide some examples of cases where an automatic, centralized, and coordinated way of conducting scheduling operations is extremely beneficial. The DSRS is more resilient when faced with both planned and unplanned outages, with the ability to recover quickly and reschedule if necessary. It has the ability to hand off tasks to other radars or make up for missed task requests to quickly

regain custody. It also handles large volumes of task requests with varying priorities in a way that keeps operating costs and slew times much lower than other approaches to scheduling. Finally, the DSRS saves the user valuable time; manually scheduling, rescheduling, and keeping track of object information when faced with large volumes of task requests or unplanned events and outages would be extremely difficult if not impossible to do in an appropriate amount of time.

Chapter 6

Conclusions and Future Work

The purpose of this chapter is to summarize the work presented in this thesis. The first section presents the contributions made throughout this thesis. The second section proposes possible improvements to our scheduling approach and suggests possible uses and future extensions for the deep space radar scheduling problem. The third and final section is a summary of our conclusions.

6.1 Summary of Contributions

In this section we review the contributions of this thesis that were first introduced in Chapter 1. The goal of this thesis was to create a scheduling algorithm to address the inefficiencies present in current deep space radar scheduling operations within the DoD. We summarize the contributions of this work in the following paragraphs.

- **Formulation of an objective function that is capable of building schedules that align with a user's end objectives and system characteristics.** We create an objective function for the optimization portion of scheduling that is able to change and prioritize different user weights. These user weights may be tuned to emphasize the lateness value of the task, the overall slew time, the total operating time, the total operating price, or the number of active radars. While we have default weights that prioritize timeliness of tasks while minimizing operational costs, the user is able to adjust these metrics at any point in

the planning period.

- **Implementation of an optimization problem that schedules various radars across time and space at any instance and can be solved using Mixed Integer Programming.** Our algorithm is able to schedule large problem instances in a dynamic environment by solving a series of optimization problems over time. By focusing on scheduling the objects in blocks of time, the solve time is kept at a reasonable length and the schedules better reflect changes to the system or new information.

- **Empirical testing and analysis of objective cost function in two different scheduling approaches.** We perform testing on the objective function components by varying the user weights in two different schedulers - namely, the Alt-Scheduler and the DSRS. We also test the various scheduling approaches on problem instances of various sizes and make-ups and compare the achieved Measures of Performance as well as average run time.

- **Development and testing of operationally "tactical" scenarios to demonstrate the benefits and effectiveness of the algorithm in scheduling and rescheduling all deep space radar operations on short notice.** Through the testing of these scenarios, we are able to demonstrate the resilience and recover capability of the scheduler in outages and other unforeseen events. We also show the advantages of a centralized, coordinated way of planning when it comes to filling in gaps in coverage or custody during scheduling.

6.2 Possible Extensions and Future Applications

While the DSRS is able to address some of the inefficiencies with current scheduling practices in the space surveillance mission, it still has many areas that can be improved and extended. Some of our suggestions are listed in the following paragraphs.

6.2.1 Expand Surveillance Capabilities to Other Orbits

This work focuses exclusively on keeping custody of objects in the geosynchronous belt, but the space surveillance mission encompasses much more than this. Critical U.S. objects and systems are located throughout LEO and MEO, which also require up-to-date characterization and guaranteed custody. Additionally, in the future it may also be important to keep custody of objects farther into deep space than the geosynchronous orbit. Because the number of sensors that are capable of executing the space surveillance mission are limited, a centralized and coordinated way of scheduling these assets could be extremely beneficial. Working to allow the DSRS to plan missions in more orbital regimes is a promising extension that could accomplish this goal. The extension would require additional constraints to control for the newly introduced physical and temporal limitations that come with planning in new orbits, but the basic formulation could remain similar to our work.

6.2.2 Inclusion of Additional Sensor Types

This thesis focuses on constructing the slew path plans for deep space radars located across Earth, but there are many other sensor assets available to aid in the space surveillance mission. As was previously introduced in Chapter 1, ground-based optical sensors, space-based optical sensors, and smaller radars are additional assets that could be incorporated into the scheduling approach and algorithm for the custody mission.

There are a few options in how coordinated and centralized planning could occur in the future. One approach is accomplished with one centralized scheduler that is responsible for distributing tasks to each sensor in the system *and* constructing the observation path plans for each of these sensors. This is similar to how the DSRS currently operates, with master control over all the deep space radars in the system. Another option is to create one master scheduler that is responsible for assigning tasks to a specific sensor asset type, and then utilizing "sub-planners" to distribute tasks amongst that specific sensor and create their path plans. This two-step approach

would allow for more detailed and specific planning at the sensor level, but loses some of the global perspective in scheduling. If this were the case, the DSRS would serve as the deep space radar "sub-planner."

6.2.3 Improvements to the Current Framework

While there are exciting ways this work may be extended, there are also areas for improvement within our current framework that may allow for its continued utilization even as space becomes more congested, contested, and competitive in the future. One improvement is the creation of fast, intelligent heuristics to continue to decrease the achieved solve time during scheduling. Currently, there is one simple heuristic tested in this work that assigned tasks to radars based on feasibility sets, but no other methods are used to attempt to create smarter radar-task request matches. Methods like assigning "clusters" of task requests that are located close together to a specific radar may help minimize the slew time without employing more costly optimization techniques.

Another possible way to improve this work is through additional ways of evaluating and altering the objective function of the optimization portion of the program. We focus our efforts mainly on minimizing overall slew time and operational costs while meeting time constraints, but new formulations may be able to better capture more advanced mission objectives or user preferences. This could be through the addition of more objective function terms or by modeling more complex interactions and the use of nonlinear terms.

Another possible supplement to our current framework is altering our optimization program to make it more robust to uncertainty in the system. Although we make adjustments to some of the estimates in data to protect against uncertainty, robust optimization techniques allow us to formalize this process with well-studied methodologies. Because scheduling in the space environment carries many unknowns, the use of robust optimization allows for protection against factors and features that are inherently unpredictable.

6.3 Conclusions

In summary, this thesis has addressed the problem of scheduling deep space radars in a centralized and coordinated way for space surveillance. We have created a mixed integer program that is capable of scheduling these task requests in a way that captures the physical and temporal constraints in the problem, as well as incorporates unique satellite and radar characteristics. We have also created an objective function that is capable of capturing specific objectives through adjustable user weights.

Our analysis shows that the scheduling approach used in the DSRS is beneficial to the DoD's scheduling for space surveillance because of its ability to better coordinate plans, respond to changes or new system information, and quickly, automatically generate schedules. The ability of DSRS to create efficient, optimized schedules saves the user's time, the radars' operating time, and costs of operations. We have also provided an alternate scheduling approach through the Alt-Scheduler that is capable of significantly decreasing the solve time associated with creating feasible schedules, while still greatly reducing the slew time and operational costs when compared to traditional scheduling.

We conclude that our algorithm is a simple approach to coordinate and schedule task requests among multiple, heterogeneous deep space radars. The methods employed in the DSRS are a viable option for scheduling by the DoD because of their adherence to important constraints, prioritization of high value objects, allowance of human operator input in the creation of plans, and overall flexibility and resilience in scheduling. While this work is one step in examining the benefits of a centralized, coordinated scheduler for this type of national security mission, it is our hope that the framework and scheduling approach may be useful in observation planning in the future.

Appendix A

List of Acronyms

BMD - Ballistic Missile Defense

D/T/ID - Detect/Track/Identify

DI&E - Data Integration and Exploitation

DoD - Department of Defense

FOV - Field of View

GEO - Geosynchronous Orbit

HIO - High Interest Object

HVA - High Value Asset

ITU - International Telecommunication Union

LEO - Low-Earth-Orbit

LLH - Latitude-Longitude-Height

MEO - Medium-Earth-Orbit

MIP - Mixed Integer Program

OR - Operations Research

RAE - Range-Azimuth-Elevation

RCS - Radar Cross Section

RSO - Resident Space Object

SHIO - Super High Interest Object

SNR - Signal-to-Noise-Ratio

SO - Separable Object

SSA - Space Situational Awareness

SSDP - Successive Sublimation Dynamic Programming

TLE - Two-Line Element

TW&A - Threat Warning and Assessment

UCT - Uncorrelated Target

Appendix B

Tables

Weight	DSR Cost	DSR Slew	Grdy Cost	Grdy Slew	Alt Cost	Alt Slew
0.1	0.63	6.34	0.77	18.42	0.77	17.27
0.3	1.81	7.31	2.50	15.76	2.35	17.57
0.5	3.23	8.33	6.014	16.73	3.81	18.12
0.7	4.17	6.81	5.79	16.60	5.54	18.57
0.9	5.28	7.77	8.49	19.96	6.95	18.51
1.1	6.64	7.84	9.21	24.12	8.62	19.50
1.3	7.78	8.04	10.52	21.92	9.95	17.78
1.5	8.84	6.90	14.03	19.86	11.56	18.94
1.7	10.43	6.65	18.91	18.34	13.63	20.29
1.9	11.87	7.01	16.56	18.67	14.43	14.91
2.1	14.21	7.25	39.14	15.55	16.13	18.12
2.3	12.97	6.83	19.65	19.82	18.16	16.87
2.5	15.27	7.43	19.10	16.14	19.19	16.86
2.7	16.64	7.37	48.17	22.96	20.04	15.71
2.9	16.98	7.10	22.88	15.62	22.29	16.66
3.1	19.23	8.53	24.92	20.83	24.35	19.14
3.3	19.95	6.74	30.58	14.30	25.08	17.16
3.5	21.51	7.34	28.95	12.47	26.95	17.68

3.7	20.34	6.33	30.08	15.44	27.87	16.76
3.9	25.20	7.40	57.95	18.03	30.23	18.07
4.1	25.15	8.12	37.86	23.15	31.62	20.05
4.3	27.48	7.73	41.43	18.74	32.71	17.55
4.5	26.42	6.59	52.68	20.09	33.65	15.04
4.7	28.71	7.72	45.89	18.06	36.45	16.51
4.9	30.63	7.50	41.51	17.27	37.47	17.87
5.1	30.53	8.01	49.71	17.83	38.56	16.19
5.3	32.38	7.14	65.36	19.30	39.17	17.57
5.5	32.96	7.58	60.30	22.72	43.10	18.91
5.7	35.45	6.68	71.49	21.65	43.89	17.77
5.9	34.05	6.48	86.40	19.50	44.66	16.57
6.1	37.08	7.32	60.74	19.58	46.13	16.01
6.3	38.65	6.12	47.46	16.70	48.96	19.11
6.5	39.40	8.29	61.23	19.49	49.71	19.43
6.7	38.81	7.59	54.53	15.98	51.93	17.28
6.9	41.35	7.82	75.04	22.12	50.63	15.46
7.1	44.45	8.59	59.07	21.99	54.87	17.78
7.3	41.48	7.70	64.62	22.78	53.09	15.47
7.5	44.01	6.51	82.03	20.50	57.60	18.56
7.7	43.13	6.72	61.16	17.73	60.22	18.23
7.9	49.13	7.28	63.71	19.14	61.66	17.22
8.1	48.56	7.38	72.03	22.68	61.96	19.93
8.3	46.09	6.78	73.68	19.86	63.35	15.13
8.5	50.11	7.44	77.17	21.80	66.17	17.11
8.7	53.63	7.35	152.42	22.62	64.75	14.80
8.9	53.93	6.92	83.09	21.37	65.42	15.41
9.1	57.41	6.89	156.72	16.62	70.07	15.41

9.3	54.29	6.27	122.59	19.81	72.75	19.25
9.5	55.97	7.24	158.69	20.24	73.72	18.02
9.7	56.93	7.32	88.12	21.44	71.74	15.96
9.9	60.87	6.97	160.27	15.86	76.59	17.17
10.1	62.81	7.11	148.94	15.17	74.21	13.75
10.3	59.49	6.89	110.35	19.18	78.16	15.71
10.5	60.70	6.52	95.41	16.38	81.09	17.49
10.7	55.30	5.77	86.25	16.26	83.47	20.98
10.9	68.26	6.54	136.03	17.26	82.70	16.47
11.1	64.87	7.71	177.91	15.54	84.73	16.32
11.3	67.39	7.95	166.34	15.83	87.99	18.76
11.5	69.32	7.47	154.35	19.41	90.98	19.18
11.7	66.13	6.68	94.51	20.37	89.07	17.46
11.9	72.52	8.31	100.71	17.98	89.06	16.83
12.1	76.37	8.48	97.77	21.46	92.51	18.10
12.3	73.44	6.52	234.60	20.68	96.89	20.23
12.5	70.32	7.12	118.61	22.92	94.95	16.17
12.7	78.91	7.73	155.12	22.90	97.26	16.63
12.9	81.21	8.11	147.98	16.43	96.96	15.81
13.1	82.57	7.43	107.37	16.54	98.48	16.55
13.3	82.45	7.67	129.51	20.54	99.93	16.13
13.5	79.93	7.63	119.26	20.85	103.96	16.78
13.7	82.08	7.11	125.64	21.96	104.58	15.99
13.9	92.08	7.72	266.41	19.29	105.72	18.17
14.1	78.74	7.74	109.17	18.72	109.69	19.02
14.3	85.05	7.04	127.81	20.61	111.50	18.20
14.5	90.61	7.23	184.42	17.91	110.66	17.18
14.7	94.03	7.52	256.98	19.03	115.10	20.46

14.9	91.33	6.75	188.97	20.59	113.05	17.82

Table B.1: Cost Analysis Results

Bibliography

[1] Yogesh Agarwal, Kamlesh Mathur, and Harvey M Salkin. A set-partitioning-based exact algorithm for the vehicle routing problem. *Networks*, 19(7):731–749, 1989.

[2] National Air and Space Intelligence Center. Competing in space. 2018.

[3] M Selim Akturk and Deniz Ozdemir. An exact approach to minimizing total weighted tardiness with release dates. *IIE transactions*, 32(11):1091–1101, 2000.

[4] M Selim Akturk and Deniz Ozdemir. A new dominance rule to minimize total weighted tardiness with unequal release dates. *European Journal of Operational Research*, 135(2):394–412, 2001.

[5] J.K. Rejto A.L Kintz, R.K. Lee. Dynamic sensor scheduler: Capacity simulations for deep space custody radars. 2019.

[6] Michel L Balinski and Richard E Quandt. On an integer program for a delivery problem. *Operations research*, 12(2):300–304, 1964.

[7] Dimitris Bertsimas and John N Tsitsiklis. *Introduction to linear optimization*, volume 6. Athena Scientific Belmont, MA, 1997.

[8] Norman Biggs. The traveling salesman problem a guided tour of combinatorial optimization, 1986.

[9] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[10] Joseph MacKay Butler. *Tracking and control in multi-function radar*. University of London, University College London (United Kingdom), 1998.

[11] John Paul Byrne, Robin Dickey, and Michael P. Gleason. A space policy primer: Key concepts, issues, and actors.

[12] Joseph P Camacho. Federal radar spectrum requirements. *2000.*, 2000.

[13] Doo-Hyun Cho, Jun-Hong Kim, Han-Lim Choi, and Jaemyung Ahn. Optimization-based scheduling method for agile earth-observing satellite constellation. *Journal of Aerospace Information Systems*, 15(11):611–626, 2018.

[14] George B Dantzig and John H Ramser. The truck dispatching problem. *Management science*, 6(1):80–91, 1959.

[15] Morteza Davari, Erik Demeulemeester, Roel Leus, and Fabrice Talla Nobibon. Exact algorithms for single-machine scheduling with time windows and precedence constraints. *Journal of Scheduling*, 19(3):309–334, 2016.

[16] Jacques Desrosiers, François Soumis, and Martin Desrochers. Routing with time windows by column generation. *Networks*, 14(4):545–565, 1984.

[17] WJ Fabrycky and JE Shamblin. A probability based sequencing algorithm. *Journal of Industrial Engineering*, 17(6):308, 1966.

[18] Brian A Foster and David M Ryan. An integer programming approach to the vehicle scheduling problem. *Journal of the Operational Research Society*, 27(2):367–384, 1976.

[19] Eugene C Freuder and Richard John Wallace. *Constraint Programming and Large Scale Discrete Optimization: DIMACS Workshop Constraint Programming and Large Scale Discrete Optimization, September 14-17, 1998, DIMACS Center*, volume 57. American Mathematical Soc., 2001.

[20] Michel Gendreau, Gilbert Laporte, and René Séguin. An exact algorithm for the vehicle routing problem with stochastic demands and customers. *Transportation science*, 29(2):143–155, 1995.

[21] John C Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society: Series B (Methodological)*, 41(2):148–164, 1979.

[22] Reinhard Haupt. A survey of priority rule-based scheduling. *Operations-Research-Spektrum*, 11(1):3–16, 1989.

[23] Thomas Herold, Mark Abramson, Hamsa Balakrishnan, Alexander Kahn, and Stephan Kolitz. Asynchronous, distributed optimization for the coordinated planning of air and space assets. In *AIAA Infotech@ Aerospace 2010*, page 3426. 2010.

[24] T. Ibaraki. Successive sublimination methods for dynamic programming computation. *Annals of Operations Research 11*, 1987.

[25] Patrick Jaillet. *Probabilistic traveling salesman problems*. PhD thesis, Massachusetts Institute of Technology, 1985.

[26] Eric Johnston. List of satellites in geostationary orbit, 2022.

[27] Kurt O Jörnsten, Oli BG Madsen, and Bo Sørensen. *Exact solution of the vehicle routing and scheduling problem with time windows by variable splitting*. 1986.

[28] Brian Kallehauge, Jesper Larsen, Oli BG Madsen, and Marius M Solomon. Vehicle routing problem with time windows. In *Column generation*, pages 67–98. Springer, 2005.

[29] Niklas Kohl and Oli BG Madsen. An optimization algorithm for the vehicle routing problem with time windows based on lagrangian relaxation. *Operations research*, 45(3):395–406, 1997.

[30] Gilbert Laporte, Francois V Louveaux, and Hélene Mercure. A priori optimization of the probabilistic traveling salesman problem. *Operations research*, 42(3):543–549, 1994.

[31] Allan Larsen and OB Madsen. *The dynamic vehicle routing problem*. PhD thesis, Institute of Mathematical Modelling, Technical University of Denmark, 2000.

[32] Adam N Letchford and Juan-José Salazar-González. Projection results for vehicle routing. *Mathematical Programming*, 105(2):251–274, 2006.

[33] Adam N Letchford and Juan-José Salazar-González. Stronger multi-commodity flow formulations of the capacitated vehicle routing problem. *European Journal of Operational Research*, 244(3):730–738, 2015.

[34] Xueting Li, Longxiao Xu, Tianxian Zhang, and Lingjiang Kong. A scheduling method of generalized tasks for multifunctional radar network. In *2019 International Conference on Control, Automation and Information Sciences (ICCAIS)*, pages 1–6, 2019.

[35] Franz Josef Lohmar. World geodetic system 1984—geodetic reference system of gps orbits. In *GPS-Techniques Applied to Geodesy and Surveying*, pages 476–486. Springer, 1988.

[36] Karsten Lund, Oli BG Madsen, and Jens Moberg Rygaard. Vehicle routing with varying degree of dynamism. 1996.

[37] Chryssi Malandraki. *Time-dependent vehicle routing problems: Formulations, solution algorithms and computational experiments*. PhD thesis, Northwestern University, 1989.

[38] Chryssi Malandraki and Mark S Daskin. Time dependent vehicle routing problems: Formulations, properties and heuristic algorithms. *Transportation science*, 26(3):185–200, 1992.

[39] Blair Ellen Leake Negron. Operational planning for multiple heterogeneous unmanned aerial vehicles in three dimensions. 2010.

[40] John O'Callaghan. Space debris: Special report. *Aerospace America*, 2022.

[41] Chairman Joint Chiefs of Staff. Joint publication 3-14: Space operations. 2020.

[42] Air Force Doctrine Publication 3-14 Counterspace Operations. Space situational awareness. 2021.

[43] Michael Angelo A. Pedrasa, Ted D. Spooner, and Iain F. MacGill. Coordinated scheduling of residential distributed energy resources to optimize smart home energy services. *IEEE Transactions on Smart Grid*, 1(2):134–143, 2010.

[44] Pulse Radar. Classification of radar systems (1).

[45] Ricardo Reinoso-Rondinel, Tian-You Yu, and Sebastián Torres. Multifunction phased-array radar: Time balance scheduler for adaptive weather sensing. *Journal of Atmospheric and Oceanic Technology*, 27(11):1854–1867, 2010.

[46] Mark A Richards. *Fundamentals of radar signal processing*. McGraw-Hill Education, 2014.

[47] Réne Séguin. Problemes stochastiques de tournees de vehicules. 1996.

[48] Marius M Solomon and Jacques Desrosiers. Survey paper—time window constrained routing and scheduling problems. *Transportation science*, 22(1):1–13, 1988.

[49] WK Stafford. Real time control of a multifunction electronically scanned adaptive radar (mesar). In *IEE Colloquium on Real-Time Management of Adaptive Radar Systems*, pages 7–1. IET, 1990.

[50] Shunji Tanaka and Shuji Fujikuma. A dynamic-programming-based exact algorithm for general single-machine scheduling with machine idle time. *Journal of Scheduling*, 15(3):347–361, 2012.

[51] Paolo Toth and Daniele Vigo. *The vehicle routing problem*. SIAM, 2002.

[52] Pei Wang and Yuejin Tan. A heuristic method for selecting and scheduling observations of satellites with limited agility. In *2008 7th World Congress on Intelligent Control and Automation*, pages 5292–5297, 2008.

[53] Xinwei Wang, Yi Gu, Guohua Wu, and John R Woodward. Robust scheduling for multiple agile earth observation satellites under cloud coverage uncertainty. *Computers & Industrial Engineering*, 156:107292, 2021.

[54] N Wassan. Reactive tabu adaptive memory programming search for the vehicle routing problem with backhauls. *Journal of the Operational Research Society*, 58(12):1630–1641, 2007.