Planning and Scheduling Proximity Operations for Autonomous Orbital Rendezvous

by

Christopher J. Guerra
B.S. Electrical and Computer Engineering,
Carnegie Mellon University, 2001

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of
Master of Science in Aeronautics and Astronautics
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
June 2003

Author ....
Department of Aeronautics and Astronautics
May 23, 2003

Certified by ....
Dr. Lance A. Page
Charles Stark Draper Laboratory, Inc.
Thesis Supervisor

Certified by ....
John J. Deyst, Jr.
Professor, Department of Aeronautics and Astronautics
Thesis Supervisor

Accepted by ....
Edward M. Greitzer
H.N. Slater Professor of Aeronautics and Astronautics
Chair, Committee on Graduate Students

[Except for this sentence, this page intentionally left blank.]

Planning and Scheduling Proximity Operations for Autonomous Orbital Rendezvous

by

Christopher J. Guerra

Submitted to the Department of Aeronautics and Astronautics
on May 23, 2003, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

This thesis develops a mixed integer programming formulation to solve the proximity operations scheduling problem for autonomous orbital rendezvous. The algorithm of this thesis allows the operator to specify planned modes which encode the chase satellite's operations. The scheduler optimally places these modes in the midst of the environmental conditions that fall out of the chase satellite's orbit parameters. The algorithm manages resources, i. e. battery state of charge, and observes temporal constraints.

Experiments show that the scheduler responds to changes in a variety of situations. It accommodates changes to the constraints in the modes. Relaxing or tightening the restrictions on the resources illuminates the algorithm's responsiveness to practical resource demands. Changes to the definition of optimality via a cost function indicate that the scheduler reacts to a diverse set of parameters.

Thesis Supervisor: Dr. Lance A. Page
Title: Charles Stark Draper Laboratory, Inc.

Thesis Supervisor: John J. Deyst, Jr.
Title: Professor, Department of Aeronautics and Astronautics

[Except for this sentence, this page intentionally left blank.]

Acknowledgments

This thesis represents the hard work, prayers, and dreams, not just of myself, but of all the people who have invested in me. Many of them are nameless, but I will attempt to mention a few of the more obvious ones.

My parents have been a consistent source of support and inspiration throughout my life, and I wouldn't be writing this without them. From teaching me that "reading is the key to knowledge," to preventing me from being second banana, and to those little pep talks on the phone when my motivation falters, I cannot adequately express my gratitude. Thank you for guiding me to this waypoint on my life journey.

To my brother, who continues to amaze me, I am proud of you.

To all of my extended family especially those unselfish grandparents who really knew how to raise families, my hat goes off to you.

I send a special thanks to Lance for his courage in selecting me as his first student at Draper Lab. Your availability in all of the big and small questions has been unwavering. I am also grateful for your witty personality which comes through at all of the important times.

More thanks go to Professor Deyst who has advised me both academically and technically in this thesis. Thanks for taking me on as your student in spite of your many commitments.

To Draper Laboratory and MIT, thanks for providing this opportunity to further my education. It has been an amazing ride that I hope will continue.

To all of the teachers and administrators at Our Lady of Victory School and Saint Joseph High School in Victoria, Texas, thanks for your dedication and sacrifice in educating me. Your presence in my life has made a huge difference.

To all of the hardworking people at Carnegie Mellon, especially those professors who really care about their students, thanks for the wonderful technical background you gave me.

I extend personal thanks to all of my past employers who have hired me over the years. Thanks for taking a chance on me.

A huge thanks go out to all of my friends. You have been the source of much support and supplied a lot of the fun over the years. How else would I have passed the high school Shakespeare sections and my first year of physics and ECE at CMU.

To the men and women of the space shuttles, Columbia and Challenger, and other space agency members, who have perished, thank you for your ultimate sacrifice. You are a source of inspiration.

One last thank you goes out to all of the nameless and faceless people who have done great things to help me over the last quarter century.

Si quieres paz, defiende la vida.

Ad Majorem Dei Gloriam,

Chris

ACKNOWLEDGMENT

May 23, 2003

This thesis was prepared at The Charles Stark Draper Laboratory, Inc., under Contract RF1-438068.

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

Christopher J. Guerra .... [illegible]

[Except for this sentence, this page intentionally left blank.]

Contents

[Except for this sentence, this page intentionally left blank.]

List of Figures

List of Tables

[Except for this sentence, this page intentionally left blank.]

Chapter 1

Introduction

This thesis considers scheduling a low earth orbiting satellite's proximity operations in the context of orbital rendezvous. In particular, the thesis looks at proximity operations related to an observation mission.

World space agencies have a significant desire to apply autonomous rendezvous technology in their space programs; some agencies already have working systems, and others could stand to upgrade their capabilities. Autonomous rendezvous software would benefit a wide variety of systems. An inspection satellite for the International Space Station or the Space Shuttle could provide a capability that NASA or its counterparts currently do not have. On orbit maintenance or assembly is another application of autonomous rendezvous technology. Future plans for human space travel to Mars would require some autonomous rendezvous technology to assemble the crafts. In general on orbit assembly is difficult and extremely expensive because humans must do the work. Developing this technology eliminates an obstacle to building more complex systems on orbit. The work of this thesis represents one of the tools necessary to fly a completely autonomous rendezvous system.

First, this chapter presents the problem in general terms. Then, it offers an explanation of some core semantics. Finally, it guides the reader through the essential sections of the thesis.

1.1 Overview

The scheduler's primary function is to optimally schedule the high-level goals of the proximity operations while satisfying the mission's temporal demands and resource needs. The user defines the mission as a set of planned requirements. The scheduler uses a set of environmental conditions that encodes features, which are a function of the chase satellite's orbit and specific, mission controlled operations including communication windows. It is the scheduler's job to make sure the planned requirements are consistent with the environmental conditions. This means the scheduler should optimally schedule the planned requirements in the midst of the environmental events while both considering and managing the temporal constraints and the resources. The optimality of a resultant schedule will be a function of the plan's execution time and the resource values where the goal is to simultaneously minimize the execution time and maximize the resource values. Chapter 3 will clarify any ambiguities regarding the problem in this thesis.

1.2 Key Terms

Proximity operations will refer to the set of high-level goals that the chase satellite should perform when it is within a specific range of its target, and when it has acquired its target with the acquisition sensor. These proximity operations occur during a specific range of the rendezvous process. The range is defined as the region in space before the actual capture and when the chase craft has acquired the target with a rendezvous sensor; reliable acquisition with the sensor can only occur when the chase craft has established a line of sight with the target craft. In a typical hierarchical planner (cf. Section 2.3.1), this range and acquisition data would arrive at this scheduler as a parameter from a higher level in the autonomous system.

The two other significant terms are the chaser and target craft. These refer to the *chaser*, which actively seeks out the *target*, which passively maintains its orbit. The problem posed in this thesis assumes that the target is both passive and cooperative.

Here passive means that the target does not actively communicate with the chaser to facilitate rendezvous, and cooperative means that it does not intentionally avoid the chaser.

1.3 Roadmap

This thesis presents the problem in a standard scientific format. Chapter 2 looks at the significant body of literature with regard to this planning and scheduling problem. This includes a look at past attempts to create systems for the autonomous rendezvous and capture problem. It gives further insight into some of the theory behind planning and scheduling.

This theory provides a foundation for the problem description of Chapter 3. The problem described in Chapter 3 is well-suited to the application of a linear program. Unfortunately, the complete problem requires a higher degree of search capability to solve the arrangement of the planned requirements among the environmental events. Chapter 4 delineates a mixed integer programming formulation to the problem. Mixed integer programming inherently encapsulates the necessary search capability that affords a complete methodology for finding the optimal schedule.

The experimental section of Chapter 5 presents the results of several experiments. The chapter defines a class of observation missions and shows the algorithms response to changes in certain mission parameters. One of the changes looks at specific constraints placed on one of the planned requirements. Another type of experiment in the chapter involves changes to the definition of optimality, and the thesis presents the results of those cases. Finally, Chapter 6 offers some conclusions along with suggestions for future research.

[Except for this sentence, this page intentionally left blank.]

Chapter 2

Review of Literature

This chapter explores the body of literature related to autonomous rendezvous and planning and scheduling. First, this chapter looks at autonomous rendezvous and capture with a special look at international approaches to the problem. The literature review turns more theoretical in its review of scheduling and then planning. Section 2.3 discusses deliberative planners, planning architectures. It mentions a method to evaluate plan quality, which is useful for replanning and for iteratively repairing a plan.

2.1 Autonomous Rendezvous and Capture

Autonomous rendezvous is a technology that the United States space programs expect to develop more thoroughly. The Russians have a system which they developed during the Soviet era [15]. Their system which performs capture in addition to rendezvous relies on direct radio communication between the two satellites. Current, U. S. rendezvous technology uses scripted autonomy where events are laid out on a timeline and the mission unfolds predictably. The Japanese and Europeans are in the process of developing such technology. The U. S. expects to use this knowledge in the Mars Sample Return mission and in resupplying the International Space Station (ISS). Current U. S. rendezvous technology depends on a costly per mission design. The capability to rendezvous under autonomous control would increase the flexibility and reduce the cost of the current approach.

2.1.1 A Survey of Rendezvous and Capture

Michael Polites of NASA submits a history of *Automated* Rendezvous and Capture [15]. In particular he discusses Soviet-cum-Russian rendezvous technology and a history of the approach and the future tack of the United States.

The Russian Approach

Polites points out that current Russian and American rendezvous systems share similar hardware including a guidance computer, inertial rate and attitude sensors, radar, and video cameras. The Russian system, while automatic, is not autonomous; it does not *decide* how to perform its rendezvous. A fully autonomous system would intelligently choose how to perform its rendezvous given a specific goal, i. e. a target.

The Russian system, which is currently used by uncrewed Progress ships docking with the ISS, depends heavily on two-way radio communication. The target satellite transmits a beacon signal which the chaser can acquire from a range of 200 km. Upon acquisition of this beacon, the chase vehicle acknowledges by activating a transponder. This transponder serves as a reflector which the chase vehicle can illuminate to derive range data. At 200 m the chase vehicle initiates a fly around and the docking port radios on the target begin communicating to provide range and attitude information. Finally, at 20 m the chaser's inertial system is the sole source of navigation data, and the chaser performs a *high-impact* docking.

Dated-technology comprises the Russian rendezvous and docking system. It lacks many of the desiderata of current space engineers' wish lists, but the system works—with a substantial amount of robustness.

American Rendezvous and Capture: Present and Future

Perhaps, it is easier to begin with the future of American rendezvous technology because the present is not as exciting. Polites indicates a definite need for automated rendezvous technology. First, he mentions the ISS, for which the space shuttle docks using per-flight rendezvous software; as previously mentioned, the Russians use their automated system. The Mars Sample Return Mission and any crewed Mars missions will require autonomous rendezvous technology, part of which this thesis may provide.

Previous American efforts form the building blocks of the space shuttle's rendezvous technology. This includes experiments in the Gemini era, and the lunar excursion module link-up with the command module of Apollo. The Space Shuttle Orbiter performs similarly. After some ground tracking and a launch into a precisely designed orbit, Mission Control computes the on orbit burns to put the orbiter within 74 km of its target. From within this range the orbiter initiates voice communication with the crew of the target vehicle. The rendezvous radar and laser ranging device provide tracking, range, and range rate functionality. Additionally, a video camera on the centerline of the docking port gives another set of information for the docking phase of the approach. The crew and Mission Control can determine the degree of manual versus automatic control to provide, but a human is always in the loop.

2.1.2 Japanese Experiment on Autonomous Rendezvous and Capture

The National Space and Development Agency of Japan (NASDA) performed two rendezvous experiments in 1997 and 1998 [10]. The first experiment tested the equipment in the docking/approach phase which is in the 0–2 m range. The second experiment featured a whole mission which begins with the chase satellite as far as 12 km from the target satellite. It concluded with the chase satellite making a physical connection with the target.

The rendezvous mission plan consisted of the three phases listed in Table 2.1. In the Relative Approach Phase, it used the Global Position System (GPS) to maintain relative navigation. The Final Approach Phase leveraged line of sight (LOS) control with the aide of a Rendezvous Radar (RVR), which is a laser powered radar for acquiring the target satellite. The Docking Approach Phase used a camera functioning

Table 2.1: NASDA Rendezvous and Capture Mission Phases

Phase	Description
1	Relative Approach Phase
2	Final Approach Phase
3	Docking Approach Phase

as a proximity sensor (PXS). In this phase the satellite flight computer commanded motion with six degrees of freedom.

The mission configuration depended critically on two-way communication between the chase and target crafts. Chapter 3 will delineate the advantages and disadvantages of this approach. Briefly, the two-way communication gives a low-cost solution to the rendezvous and capture problem, but the target satellite must be equipped with communication hardware. A rendezvous system without this feature poses a greater challenge in that the chase craft must have the capability to determine its target's relative position without the target's help.

2.1.3 Preventing Plume Impingement and Avoiding Obstacles

Another important technology to perform autonomous rendezvous is the capability to avoid obstacles and engine exhaust plumes. Arthur Richards et al. describe a technique in [17]. Their method uses mixed-integer linear programming (MILP) to navigate the immediate region around a satellite. Logical constraints represent the relative attitudes of the two craft, with a cost function that models fuel usage. The goal is to find an optimal trajectory that satisfies the logical constraints and minimizes fuel use. The most relevant example application they cite is the use of a microsatellite to inspect the International Space Station. This study has influenced the work described in this thesis insofar as Richards applies MILP, also called MIP, to the rendezvous problem [4]. While Richard's work utilizes MIP to effect rendezvous, this thesis considers the intricacies of scheduling rather than navigation.

2.2 Scheduling

Planning and scheduling often come as a pair because their functions are closely linked. While this thesis focuses primarily on scheduling, it is important to recognize and understand how scheduling fits into a plan [11].

2.2.1 Scheduling Within a Plan

Chien et al. posit a set of requirements which a planning and scheduling system should fulfill [5]. The requirements are

- An expressive constraint modeling language to allow the user to define naturally the application domain
- A constraint management system for representing and maintaining spacecraft operability and resource constraints, as well as activity requirements
- A set of search strategies for plan generation and repair to satisfy hard constraints
- A language for representing plan preferences and optimizing these preferences
- A soft, real-time replanning capability
- A temporal reasoning system for expressing and maintaining temporal constraints
- A graphical interface for visualizing plans/schedules (for use in mixed-initiative systems in which the problem solving process is interactive).

The primary difference between NASA's ASPEN (Automated Scheduling and Planning ENvironment) and the scheduling algorithm in this thesis is the scope of its relevance. While ASPEN is generalizable to a broad class of problems, its primary thrust is for scheduling events in a densely populated timeline.

Briefly, their work includes an "early commitment, local, heuristic, iterative search approach to planning, scheduling, and optimization." In particular, the iterative repair allows for incremental changes. If only a small constraint changes, then the plan can be repaired rather than recomputing it. One drawback is that the heuristic may avoid searching some valid plans or search a plan multiple times. This issue becomes less severe because the local aspect of the search allows for increased computational and memory efficiency.

While the preceding list represents the requirements of a planning and scheduling system, it does not explicitly state the algorithm. Chien defines planning and scheduling as taking high-level goals and converting them to low-level activities. Further, the system should satisfy any constraints and optimize the plan quality. There are obvious reasons to satisfy constraints, but seeking optimality is more subtle. The generated plan needs to be efficient and execute in a reasonable amount of time. Specific goals might have many feasible solutions, but if they overuse or abuse resources they are less useful than an optimal plan.

2.2.2 Scheduling Satellite Observations

Birgit Sauer describes an integer programming method to scheduling satellite operations [19]. This approach is a source of significant inspiration for this thesis. Sauer's algorithm is not intended for in flight use, but rather by an operator of a satellite or classes of satellites. Sauer's thesis examines three cases: spin stabilized, 3-axis stabilized, and constellations of 3-axis stabilized satellites.

The scheduler maximizes the mission's science value across the time horizon. This approach uses a linear programming model of the mission. The integer variables describe the use of the instruments which generate the mission's scientific value. The linear programming model in conjunction with the instrument constraints form the operational framework for the scheduler. Other important considerations result from the aforementioned classes of satellites, but they are beyond the scope of this thesis.

2.2.3 Scheduling with LP and MIP

Section 2.1.3 describes using a MIP to compute proximity operations maneuvers under the considerations of obstacle avoidance and plume impingement. The use of a MIP for rendezvous inspires the work in this thesis. To apply this tool, a planner and scheduler must model specific variables not the least of which is time. This section looks at the approach of Dechter et al.

Temporal Constraint Networks

Dechter, Meiri, and Pearl consider temporal constraint networks and present techniques for representing time and satisfying constraints [7]. Their approach represents time as a set of continuous variables: $X_1, \dots, X_n$. Here, each variable represents a point in time.

The next step in modeling the network is to specify the constraints. Constraints are represented as sets of closed intervals. The closed property becomes important when applying the formulation to a Linear Program (LP) or a MIP.

$$\{I_1, \dots, I_n\} = \{[a_1, b_1], \dots, [a_n, b_n]\} \tag{2.1}$$

For unary constraints T_i, there is the disjunction

$$(a_1 \leq X_i \leq b_1) \vee \cdots \vee (a_n \leq X_i \leq b_n). \tag{2.2}$$

To develop some intuition, consider these T_i as the constraints, which bound the event, X_i's execution time. The following disjunction suggests that the event, X_i, a communication activity for example, must occur between mission time intervals $[12, 20]$, minutes or $[59, 67]$, minutes.

$$(12 \leq X_i \leq 20) \vee (59 \leq X_i \leq 67)\,. \tag{2.3}$$

Then there is the binary constraint, T_{ij}, which defines the allowable distance between $X_j - X_i$. It is written as

$$(a_1 \leq X_j - X_i \leq b_1) \vee \cdots \vee (a_n \leq X_j - X_i \leq b_n). \tag{2.4}$$

Equations 2.1 – 2.4 describe constraints on the network parameters, but they do not describe the network's structure. The simplest way to characterize the network structure uses a state transition matrix. For a simple temporal network, a single interval, I_j, encodes either a unary or a binary constraint. Dechter et al. show that the Floyd-Warshall algorithm (i. e., the All-pairs-shortest-paths algorithm) can solve the simple temporal problem in polynomial time.

Chapter 3 will show how Equations 2.2 and 2.4 can be applied to represent temporal constraints for a scheduling problem.

2.3 Planning

This section will consider several of the more common planning methods which are currently in practice. These deliberative approaches to planning live on finite state machines, and they typically create plans that satisfy the pre- and post- conditions of each planned activity. Often, they account for temporal optimality and resource management.

This section will examine several key elements of planning. First, it looks at an approach that decomposes the planning process. Decomposition makes the problem more tractable and provides a simpler replanning capability. This section will also delineate several planning architectures and a scheme to manage resources. Another feature of this section is the treatment of a method to evaluate plan quality and a replanning notion known as iterative repair.

2.3.1 Hierarchical Planning

Mark Abramson et al. present a hierarchical approach to planning and scheduling [1]. They focus on a top down methodology to plan earth observations that maximize science value. In particular, the *Earth Phenomena Observing System* (EPOS) models a fleet or constellation of satellites to provide data about the Earth. The hierarchical approach decomposes the problem into three distinct tiers. In descending order these are the system, collaborative, and satellite tiers. Each tier focuses on a particular goal. The system tier wants to know which targets to observe and which satellite platforms should perform the observations. The collaborative tier examines exactly which satellite to pair with a target and at what time. The lowest tier manages the individual fuel usage of a satellite, attitude control, and data management. One significant advantage to the tiered approach is that a lower level can replan itself without affecting its parents.

This decomposition makes for a smooth interface with various planning and scheduling tools. They apply integer programming, network optimization, and astrodynamics "to calculate optimized observation and sensor tasking plans." Specifically, EPOS uses astrodynamics to decide whether a satellite can observe a target at time, t, whether the target will be in sunlight or darkness during observation, and to properly orient the sensor with respect to the target. This information is converted into binary data which indicates whether a sensor *can* or *cannot* observe a target; they call these data *dynamic inputs*. To perform the satellite-to-target assignment they use two classes of integer decision variables, which maximize a function of the dynamic inputs. These solutions pass to a lower tier of the hierarchy that computes specific pointing commands for each active satellite. This tier bases its decisions on variables which represent the sensor gimbal angle and the sensor's ability to observe its target, both of which depend on t.

2.3.2 Heuristic Planning

Tackling a mission in its entirety poses an intractable problem because of the amount of uncertainty and the number of decisions. Dungan et al. describe a method which uses a greedy search in conjunction with a heuristic to schedule fleets of satellites with higher fidelity models [8]. In particular, they model data storage and manage communications unlike many previous efforts.

To schedule the earth observation support activities, they implemented the Constraint Based Interval (CBI) planning framework from a proprietary planning system called EUROPA. This system uses state variables which are timelines that represent data management activities. Even at this level of scheduling, they encountered lengthy running times. In light of this, and the fact that the assignment of satellites to targets has a very large search space, they implemented a "planning algorithm that combines heuristics, stochastic search, and constraint propagation." The algorithm randomly selects an observation with the heuristic; it randomly selects a time slot; and it propagates the constraints until it is impossible to propagate further or until the plan is inconsistent. The algorithm refines this process a specific number of times while checking for consistency. On the heuristic, Dungan et al. posit a form of contention which depends on the sum of available data storage space and available time slots.

2.3.3 Architecture for Temporal Reasoning

Deep Space One is a NASA space probe that tested several new technologies. Among these was a planning and scheduling system called the New Millennium Remote Agent (NMRA). The main premise in the agent is that the Planner/Scheduler operates the *deliberative* layer and the executive (cf. Section 2.3.4) controls the *reactive* layer [11]. Because of this arrangement, the executive issues the low-level commands to the spacecraft's subsystems, and the planner performs the high-level computation. This leaves the reactive layer with a simple, scripted ability to respond to a dynamic environment. When the environment changes drastically, the planner/scheduler must perform a replan. Muscettola et al. describe this ability and focus on addressing the temporal constraints.

Their approach is based on the simple temporal networks that Dechter et al. described [7]. Essentially, the planner incrementally adds events to a partial plan. The algorithm propagates the events frequently and always propagates when adding an equality constraint. Because of their system's structure, there is no disparity between actions and states. They describe, instead, parallel threads which model each state variable. These threads are linked at time points which activate the affected threads. The advantage here is that moving a time point only directly affects one state variable. They apply an "incremental version of the Bellman-Ford algorithm [7]."

2.3.4 Executive Architecture

While the goal of this thesis is to treat the scheduling problem within a planner, an executive's architecture offers valuable design principles. Pell et al. describe their design for the New Millennium Remote Agent's (NMRA) executive or sometimes called EXEC [13, 14]. Typically, in autonomous systems, this component decomposes a plan into the activities which the various subsystems must execute. Their hybrid design incorporates aspects of both a procedural and deductive system. Here the procedural executive performs the higher-level functions that include managing locks, synchronization, hierarchical task decomposition, and others. The deductive executive serves in a more computation intensive capacity performing state inference and optimal failure recovery.

The procedural component of their executive manages properties and locks associated with a particular activity. Each activity runs in its own thread, and it can issue a variety of signals which indicate a change in the status of the activity or one of its properties.

The deductive executive "can be viewed as a discrete model-based controller that attempts to keep the spacecraft state on a trajectory that achieves a set of high-level input properties." This component views the spacecraft or the autonomous system as a set of synchronous finite state machines. The deductive executive has a monitor function which Pell et al. call mode identification (MI). MI infers the spacecraft's state using a conflict-directed best-first search algorithm.

In conjunction with the recovery function, MI becomes MIR. This component is critical to the interface between the procedural and deductive executives. The recovery feature provides functionality to the system in the event of a fault.

In general, Pell et al. use a very modular design in EXEC. Their goal is to create an executive that software developers can think of from a high-level. In combination with a modular design this high-level approach gives developers the option to port the software between a variety of autonomous systems. As do Chien et al. in Section 2.2.1, Pell et al. offer a set of *capabilities* which they strive to include in their system.

These are flexible plan execution, configuration management, resource management, an action-definition language, and system-level fault-protection support.

2.3.5 Evaluating Plan Quality

While searching for an optimal plan, the ability to evaluate the quality of a plan is useful. Gregg Rabideau et al. offer preferences which serve as "quality metrics for variables in complete plans [16]." The five types of variables which they use to consider plan quality are

- local activity variable
- activity/goal count
- resource/state variable
- resource/state change count
- state duration

More specifically, the preference assigns a value to these variables in the range, $[0, 1]$. Based on this mapping *improvement experts* can increase the quality of the plan. Usually, this requires considering how to change the variable and observing whether it yields an increase or decrease in the observed value.

2.3.6 Iterative Repair

Iterative repair is a scheduling method that takes a complete plan and refines it to improve the performance. The previously mentioned planning and scheduling methods are of a deliberative nature. They differ from iterative repair in that they take partial plans and incrementally improve them or lengthen them. Monte Zweben et al. present a system, GERRY, that implements *constraint-based iterative repair* to schedule space shuttle ground maintenance operations [20].

The constraint-based method assigns a penalty to any constraint violations where the goal is to minimize the sum of the violations. This provides the ability to evaluate a current schedule and compare it to its repaired counterpart keeping the better of the two. GERRY examines both resource and state constraints which each have their own heuristics for repairing the schedule.

Table 2.2: Heuristics for repairing resource constraint violations

Fitness	Move the task whose resource requirement most closely matches the amount of overallocation.
Temporal Dependents	Move the task with the fewest number of temporal dependents.
Distance of Move	Move the task that does not need to be shifted significantly from its current time.

Table 2.2 enumerates the heuristics for resolving resource constraint violations, and the following list describes the heuristics for repairing state constraint violations.

1. Insert a new task that sets the state correctly from the start-time to the end-time of the violated task.

2. Move the violated task forward to a time where the constraint is satisfied.

3. Move the violated task forward to a time where the state can be changed (by a new task) without causing additional state violations. Then insert the new task, thus changing the state for at least the duration of the violated task.

4. Move the violated task backward to a time where the constraint is satisfied.

5. Move the violated task backward to a time where the state can be changed without causing additional state violations. Then insert the new task with an effect that will change the state for the violated task.

Based on the computed penalties of the present schedule, these tools make it possible to repair the schedule. Zweben et al. point out a disadvantage. Iterative repair is not a *complete* search method, so there are cases where the algorithm will run to the maximum allowable iterations without finding a solution; iterative repair can get "stuck" in local minima. The upside is that the algorithm can quickly accommodate changes and work easily with preexisting schedules.

Dealing with Dynamic Events

Historically, goal-based systems required teams of schedulers to create a mission plan. This type of plan could be changed only during infrequent communication windows. Onboard planners and schedulers provide autonomy to replan for certain events. Rather then using probabilistic methods to treat an uncertain future, Chien et al. approach the replan using iterative repair for *dynamic events* [6]. The idea is that if an event arises that would increase the mission's science value, then a replan would accommodate that feature. Likewise, if something detrimental occurred, such as a failure to acquire a guide star, then the planner could replan. Iterative repair is useful because onboard planning consumes significant resources. The planner for the Deep Space 1 mission requires four hours to generate a three day plan. The ability to replan can nicely accommodate incremental changes.

The iterative repair algorithm is straightforward, and it admits a hierarchical approach to planning with the understanding that long horizon plans will contain less detail than short horizon plans. Table 2.3 shows the iterative part of the algorithm. The initialization requires that plan set, P, and goal set, G, are initialized to their respective null conditions, and S must be set to the current state.

Table 2.3: Algorithm for iterative repair

Step	Operation
1	Update G to reflect new goals or goals that are no longer needed
2	Update S to the revised current state
3	Compute conflicts on (P, G, S)
4	Apply conflict resolution planning methods to P (within resource bounds)
5	Release relevant near-term activities in P to RTS [Executive] for execution
6	Goto Step 1

2.3.7 Resource Management

Erann Gat and Barney Pell describe their treatment of *abstract resource management* in the context of the NMRA [9]. The NMRA applies the three layer architecture endemic to many autonomous systems. This includes a deliberative and a reactive layer tempered by the sequencer. The problem they consider involves managing resources as discrete values while attempting to ensure that "parallel tasks do not interfere with one another." This problem is intractable because there are innumerable, unforeseen situations which may make some parallel tasks incompatible.

Instead, Gat and Pell propose the *property lock* which is a "data structure that signals a task's intention to make a property take on a particular value." When the planner wants to plan a task, it will first subscribe the property. Depending on the state of the world, the response will be one of three cases. If no other task owns it then the subscribing task becomes the owner. Second, if another task is subscribing to the lock and the outcome values are compatible, then they will share the lock, but not be owners. Finally, if another task owns the lock then the intent to subscribe will be denied.

This application of a property lock became part of the NMRA which flew on Deep Space 1. It simplifies describing the configuration management routines.

[Except for this sentence, this page intentionally left blank.]

Chapter 3

Problem Description

The preceding chapter described several approaches to planning and scheduling, and it reviewed some of the tools which will be necessary for the algorithm set forth in this thesis. As described in Section 2.2.1, Chien et al. define planning and scheduling as a decomposition process. A planner/scheduler takes high-level goals and turns them into low-level activities while satisfying constraints and optimizing the plan. The list on page 25 enumerates the requirements of the planning / scheduling system.

This thesis will examine planning and scheduling during satellite proximity operations. These activities occur when the target is within a line-of-sight and when the chase satellite has acquired the target. Radar and laser sensors are possible and appropriate ways to acquire the target satellite. During these proximity operations, scheduling the low-level activities becomes critical because of the demand on scarce resources and the constraints imposed by the use of the resources and sensors.

This chapter will define the problem and provide the mathematical language to perform the operations to create an optimized schedule.

3.1 Summary

This thesis assumes as given a discrete sequence of planned activities for the satellite to perform. Each activity is defined by a set of modes and a set of constraints that remain constant throughout the activity. The modes correspond to subsystem modes such as spacecraft attitude mode and various spacecraft payload modes. The constraints may limit the environmental conditions under which the activity can occur. For example an observation mode may be constrained to occur in either sunlight or during an eclipse.

This thesis assumes the order in which to perform the activites is fixed, but the specific transition times are not. Temporal bounds may be optionally placed on the transitions times and activity durations. The scheduling problem in this study selects the optimal transition times between the activities.

3.2 High-level Goals

The scheduling problem in this thesis requires that the planner/scheduler determine the placement of desired events among environmental events. Planned events refer to the *high-level* events that Chien et al. describe in Section 2.2.1. The following section will examine the environmental conditions with which the scheduler must contend. Subsequently, the planned requirements are discussed. Finally, this section explores temporal considerations and resource management.

3.2.1 Environmental Conditions

The environmental conditions occur beyond the control of the spacecraft system. They consist of day and night time intervals, communication windows, and other factors which are necessary to consider when scheduling the satellite activities for proximity operations. They have both direct and indirect effects on the chase craft's resource usage and significantly influence the schedule.

This thesis considers three types of environmental conditions: daylight and communication over two distinct communication bands. This formulation treats communication windows as environmental conditions because they are dictated externally. This formulation discretizes the *events* or points in time where the conditions change; rather then making daylight be a continuous function of light intensity, the formulation describes the lighting condition as either lightness or darkness. Similarly, two

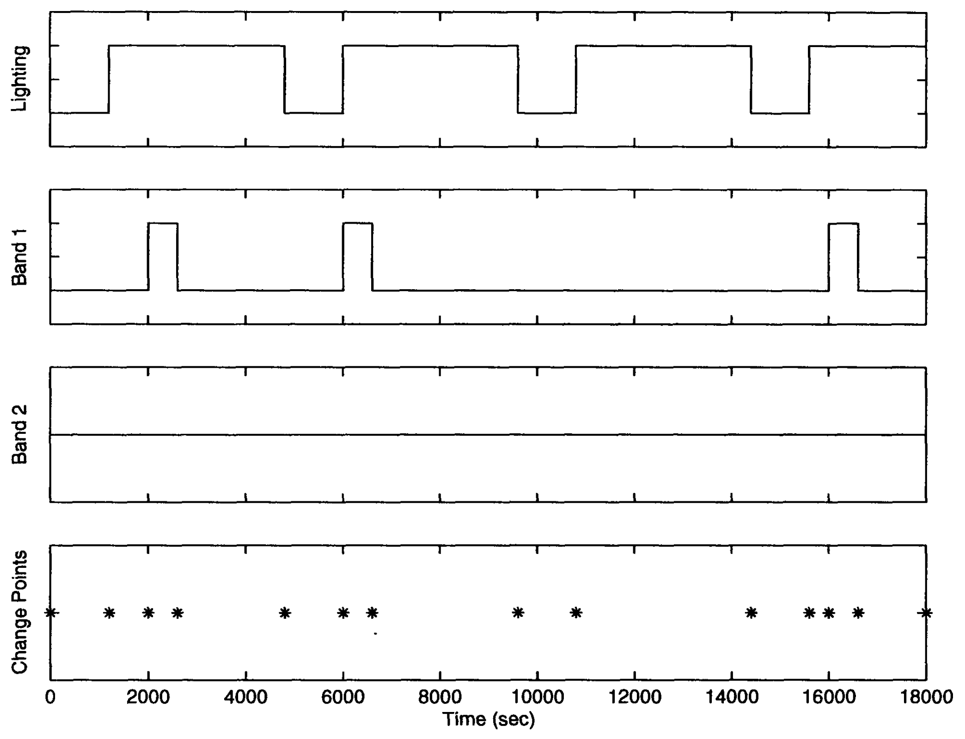


Figure 3-1: An example environmental timeline

states represent both bands of communication events—on or off. This provides simplicity in evaluating the environmental conditions in the context of the planned events. It is important to note that this formulation readily admits more complex descriptions of environmental conditions, but for the sake of simplicity the examples here are binary.

Viewing a timeline is the easiest way to understand the encoding of the environmental conditions. Figure 3-1 encodes the environmental conditions into a timeline of binary events. The horizontal axis labels time in seconds. The first row represents the cycle of lighting as the satellite moves in its orbit. Note that in this simulation the periods of daylight are longer than the periods of darkness. This is an artifact of the satellite's orbit. This particular orbit tends to expose the craft to a higher percentage of lightness than darkness. This facilitates observing the target in daylight and maintaining the battery's charge. The second row shows three communication events, and the third row indicates that no Band 2 communication events will occur.

The fourth row is a row of change points. This represents the collection of points in time where an environmental condition changes. In the event where two events change state simultaneously at 6000 s, only one change point is necessary to encode the information. This is the benefit of using a standalone data structure to store the discrete environmental values. This structure is called the matrix of environmental conditions $\mathbf{E} \in \mathbb{Z}^{\varepsilon \times N}$. Where ε is the number of environmental conditions considered in the plan, and N is the total number of environmental events. Each column represents the discrete value of all ε environmental parameters in a given interval of time. The vector is specified as

$$\vec{e}_j = [\tilde{e}_{1j} \cdots \tilde{e}_{\varepsilon j}]^T \in \mathbb{Z}^{\varepsilon} \qquad \forall j \in \{1, \ldots, N\} . \tag{3.1}$$

In general, the value $\tilde{e}_{ij}$ denotes the environmental condition of the ith parameter during the interval j. Note that $\tilde{e}$ refers to an element of $\vec{e}$. e refers to values of time which Section 3.2.3 will clarify.

3.2.2 Planned Requirements

While this is an autonomous system, an operator must define the satellite's mission. This is the first bullet in Chien's common elements of a planning and scheduling system. For this thesis, the user specifies a set, $\mathcal{P}$ of planned events that occur in monotonically increasing order.

$$\mathcal{P} = \{p_i \in \mathbb{R}^{\oplus}, \forall i \in [0, \ldots, M]\} \tag{3.2}$$

These p_i are the times which the scheduler must compute. Associated with these times are the intervals which are written

$$P_k = [p_{k-1}, p_k] \quad \forall k \in [1, \ldots, M] . \tag{3.3}$$

P_k emphasizes the discrete nature of the intervals.

Each interval constrains the plan into a specific set of constraints, denoted $\vec{m}_j$. The $\vec{m}_j$ serve as a set of discrete values that define the chase satellite's systems configuration during an interval. Associated with each interval is a starting and ending time, also known as planned events or planned times. Section 3.2.3 discusses the temporal constraints on each planned event.

At each planned event the user designates the constraints or modes as a set of discrete values in a vector, $\vec{m} \in \mathbb{Z}^{\mu}$. Here, $\mathbb{Z}$ is the set of integers and μ is the number of constraints considered in each planned interval. Concatenating all $\vec{m}_j$ yields the matrix $\mathbf{M}$ of Equation 3.4 which this thesis calls the matrix of *planned modes.*

$$\mathbf{M} = \begin{bmatrix} \vec{m}_1 & \vec{m}_2 & \cdots & \vec{m}_M \end{bmatrix} \tag{3.4}$$

Each $m_{ij} \quad \forall i \in [1, \ldots, \mu]$ refers to the ith plan parameter of planned interval P_j.

Table 3.1 enumerates the mode for each element, m_i, of $\vec{m}_j$. The lighting requirement refers to whether the event must occur in daylight, darkness, or either. The communication events are binary: either the event occurs in a communication window or it does not. The attitude mode is composed of several discrete values which have a significant affect on other systems. These values include an option to periodically observe the target craft to maintain relative position. Another option is to point at the earth or the sun to perform communication activities or illuminate the solar arrays. The sensor mode describes the state of the cameras, instrumentation, acquisition sensor, etc.

Table 3.1: Description of planned event parameters

Parameter	Description
m_1	Lighting constraints
m_2	Communication Band 1 constraints
m_3	Communication Band 2 constraints
m_4	Attitude mode
m_5	Sensor mode

Unless stated otherwise, this thesis assumes the description in Table 3.1, $\mu = 5$, and M is the number of planned events in a plan. Figure 3-2 is an example of a user-defined plan where M, the number of planned events is seven.

Examining Figure 3-2, note that the horizontal axis is marked with numbers from zero to seven. As an example interval zero to one denoted as planned event one is the user's definition of the craft's behavior during that event. Similarly, the interval labeled one to two denotes the operators specification of the parameters for the second planned event. Rows two and three capture two *boring* cases where the operator has specified that neither types of communication events should occur during this plan; this means the chase craft's communication requirements are unconstrained.

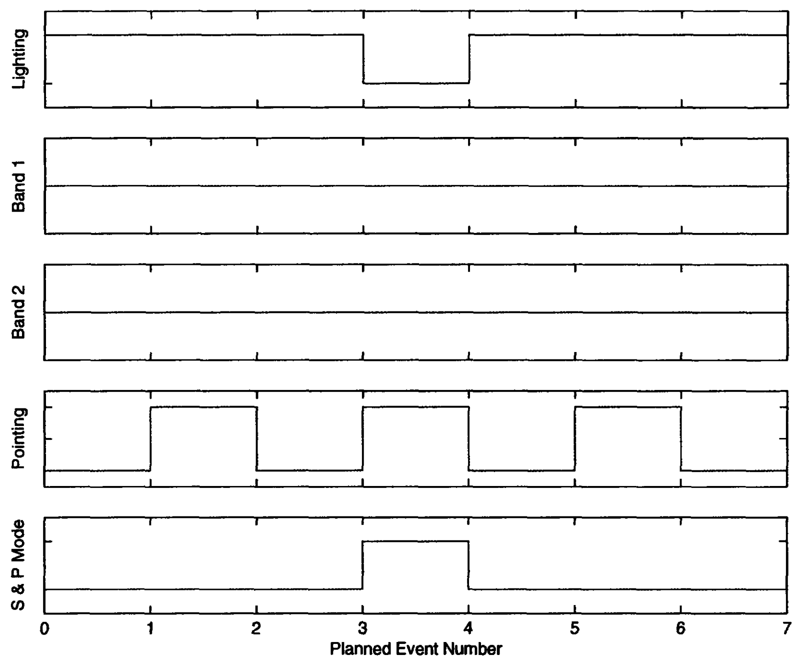


Figure 3-2: An example of the plan that a user would specify

3.2.3 Temporal Considerations

Thus far the thesis has shown how to represent environmental and planned parameters in this formulation. Section 3.2.1 briefly examined the temporal considerations, but it neglected a rigorous approach. First, this section will define the major ideas for the planned modes and environmental events. Then, it will study the interactions between the two. Finally, it will consider an approach to allow for temporal constraint descriptions using the two entities.

Intervals Defined

Section 3.2.1 indicates that each environmental event begins and ends at a specific time. A specific event has its beginning time e_i and its ending time e_{i+1} where e_i refers to a particular time and $\vec{e_j}$ refers to a vector of environmental parameters. Where the complete set of environmental times is

$$\begin{aligned} &\mathcal{E} = \left\{e_i \in \mathbb{R}^{\oplus}, \forall i \in [0, \ldots, N]\right\} \\ &\text{Must satisfy} \quad e_{i-1} \leq e_i \quad \forall i \in [1, \ldots, N]\,. \end{aligned} \tag{3.5}$$

The formulation requires that $|\mathcal{E}| = N + 1$*. Typically, $e_0 = 0$, but this formulation allows for any value in $\mathbb{R}^{\oplus}$ such that $e_i \leq e_{i+1} \quad \forall i \in [0, \ldots, N]$.

It is important to realize that each value, e_i $\forall i \in [0, \ldots, N]$, describes both the beginning time of an interval and the ending time of the previous interval. This is not true for the initial and final intervals because e_0 and e_1 define the initial interval and e_{N-1} and e_N describes the final interval.

Recall that within these intervals the orbiting craft will experience environmental conditions codified in the $\vec{e_j}$. Moving across an e_i boundary into another interval with different $\vec{e_j}$ places the craft under different environmental conditions. This idea becomes critical in selecting a feasible-cum-optimal plan.

Figure 3-3 depicts the relationship between the interval labels, E_j, and their related time boundaries, e_i.

*$|\mathcal{E}|$ means the cardinality of $\mathcal{E}$

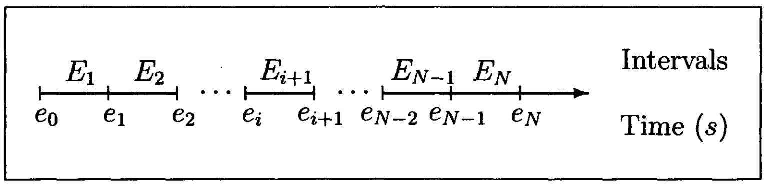


Figure 3-3: Pictorial definition of environmental events

As an example, the e_i for the environmental conditions of Figure 3-1 are

$$\begin{aligned}\mathcal{E} &= \{0, 1200, 2000, 2600, 4800, 6000, 6600, 9600, \\ &\quad 10800, 14400, 15600, 16000, 16600, 18000\}\,. \end{aligned} \tag{3.6}$$

The planned modes' intervals are defined similarly. The key difference is that these time boundaries are sought as solution values (i. e. they are unknown at the outset). Consider the set of time boundaries

$$\begin{aligned}&\mathcal{P} = \{p_i \in \mathbb{R}^{\oplus}, \forall i \in [0, \ldots, M]\} \\ &\text{Subject to} \quad (p_{i-1} \leq p_i) \wedge (p_0 = e_0) \wedge (p_M \leq e_N) \qquad \forall i \in [1, M]\end{aligned} \tag{3.7}$$

Here, two planned mode times can occur simultaneously. This is the case where a mode requires no time to execute, as when $p_i = p_{i+1}$. Later, this section will present additional constraints that a user may impose on the p_i. The restriction that $p_i \leq p_{i+1}$ implies that the user must specify the planned modes in a sequential order, and thus this order is already known. This formulation specifies that $e_0 = p_0$ because it is convenient to define the first planned interval to begin at the same time as the first environmental interval; thus eliminating the need to solve for p_0. The other restriction, $p_M \leq e_N$ comes from the fact that a desirable schedule is valid for a given range of times, i.e. $[e_0, e_N]$. This is significant because this is the only range in time with viable environmental data, and this information is essential to creating a solution.

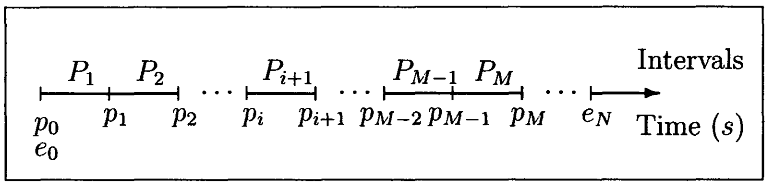


Figure 3-4: Pictorial definition of planned modes

More formally, the set of environmental intervals is

$$\mathcal{I}^E = \{E_i\} \qquad \forall i \in \{\mathbb{Z} \cap [1, N]\}, \tag{3.8}$$

where E_i is the interval $[e_{i-1}, e_i]$. E_i is the interval in which the conditions defined by $\vec{e}_i$ hold.

The planned intervals have the analogous set

$$\mathcal{I}^P = \{P_i\} \qquad \forall i \in \{\mathbb{Z} \cap [1, M]\}. \tag{3.9}$$

The interval P_i has temporal bounds p_{i-1} and p_i. In this interval the planned modes $\vec{m}_i$ are valid.

As with the e_i, each p_i represents both the *end time* for the interval P_i and the *start time* for the interval P_{i+1}, with the same exceptions of the initial and final values, p_0 and p_M, respectively.

Finally, Figure 3-4 gives insight into the planned mode timeline. The timeline is similar to the one for the environmental events. The important points to note are the definitions of the first and last times. A more interesting aspect arises in a subsequent section which examines the interactions between these intervals when they overlap.

Temporal Constraints

One of Chien's requirements for a planning and scheduling system, in the list of Section 2.2.1, (p. 25), is the implementation of "a temporal reasoning system for expressing and maintaining temporal constraints." This section describes the mathematics for encoding this temporal reasoning system. Previously, this thesis showed how to define the planned modes in terms of time. The elements of the set, $\mathcal{P}$, serve as temporal boundaries for each of the $\vec{m}_j$. Imposing certain constraints on the p_i effectively characterizes the temporal realities of the mission. The three types of temporal constraints are

1. External sources of start time and duration constraints (see following list)
2. Joint interval ordering
3. Interval compatibility

First, this thesis examines reasons to provide this functionality for a planning and scheduling system. Table 3.3 presents several approaches to encoding constraints. The rationale for providing temporal constraint functionality is

1. A certain planned mode needs to last *exactly long enough* to coincide with a communication window.
2. Guidance, navigation, and control require that a mode last *at least* a certain amount of time; or require that it last *less than* a certain amount of time.
3. The user determines that an activity should last *less than*, *more than*, or *exactly* a certain amount of time.
4. Equipment in a specific mode needs to be used for a certain amount of time.
5. Extra time is needed to renew a resource.

In this list, there are some commonalities among the reasons for describing the temporal constraints. These commonalities can be boiled down to *exactly*, *greater than*, and *less than.* An operator might see these as in Table 3.2.

When specifying a mission, the operator may think of its duration in terms of the *User Language* of Table 3.2. The planner must take the high-level user specification and turn them into values that can be related with the indicated relational operators. The inequalities of Table 3.3 implement this operation. This thesis assumes that the high-level translation of the user language into the more primitive relational operator specification has already been performed for a given mission.

Table 3.2: User language in terms of relational operators

Relational Operator	User Language
$\geq$	*minimum end time / duration* greater than at least more than no less than
$\leq$	*maximum end time / duration* less than at most no more than
$=$	*exact end time / duration* equal to exactly both more than and less than

Table 3.3: Methods for encoding temporal constraints, $t \in [e_0, e_N]$

Minimum end time	$p_i \geq t$	$\Leftrightarrow$	$-p_i \leq -t$	(3.10)
Maximum end time	$p_i \leq t$			(3.11)
Exact end time	$p_i = t$	$\Leftrightarrow$	(3.10) $\wedge$ (3.11)	(3.12)
Minimum duration	$p_i - p_{i-1} \geq t$	$\Leftrightarrow$	$p_{i-1} - p_i \leq -t$	(3.13)
Maximum duration	$p_i - p_{i-1} \leq t$			(3.14)
Exact duration	$p_i - p_{i-1} = t$	$\Leftrightarrow$	(3.13) $\wedge$ (3.14)	(3.15)

Table 3.3 presents the inequalities which are used to encode the user language of Table 3.2. For completeness, it is allowable to use conjunctions of minimum and maximum end times or conjunctions of minimum and maximum durations. When specifying these temporal constraints the user must exercise care so as not to create an infeasible situation. Of course it is entirely possible that a set of constraints makes the schedule infeasible, and the solver must indicate such a situation.

The value t refers to a constant in $[e_0, e_N]$ for Equations 3.10–3.12. Equations 3.13–3.15, must have $t \in [0, e_N - e_0]$. The reader should notice that this constant could land within any E_i complicating the problem. This is not always the case, but it is an eventuality which must be considered.

The two inequalities of 3.10 and 3.13 also reveal another form to express the same relationship using $\leq$. This becomes important when describing solution techniques in Section 3.3.2.

The second point in the first list of Section 3.2.3 (p. 46) refers to the joint interval ordering. This regards the arrangement between the p_i and the e_j. This ordering represents a difficult problem because a particular arrangement directly affects several significant factors. Section 3.2.4 reveals that the ordering affects how the resource values evolve over time. Once the ordering is specified, the problem still remains as to how best to position the p_i, as in sliding beads on a wire. Because of the difficulty of this problem, Chapter 4 presents a more comprehensive approach to solving this problem.

The final item in the first list of Section 3.2.3 (p. 46) regards interval compatibility. The process of computing the ordering requires that overlapping intervals satisfy certain requirements. For instance a planned interval might have the requirement that it occur in daylight, so it must only be paired with an environmental interval of the same nature. This implies the existence of a binary function

$$v(i,j) = \begin{cases} 1 & \vec{m}_i \text{ compatible with } \vec{e}_j \\ 0 & \text{otherwise} \end{cases} \tag{3.16}$$

$$\forall i \in [1,\ldots,M],\ j \in [1,\ldots,N].$$

The binary function compares the planned constraints encoded in $\vec{m}_i$ with the environmental conditions encoded in $\vec{e}_j$. If $v(i,j)$ evaluates to a *false* condition the respective overlapping intervals P_i and E_j are constrained so that they do not overlap in the final solution.

3.2.4 Resource Management

In a planning and scheduling system Chien mentions, in his second point, that a robust system should address resource constraints. This section presents the tools to treat linearly the resource usage for various types of resources. Table 3.4 characterizes several resources which a typical satellite might need to maintain. This list is not exhaustive, but it highlights three major classes of resources. The science is listed as a *finite* resource; Sauer describes a scheduling algorithm that maximizes science in a mission [19]. Sauer indicates that there is a finite amount of science that an earth observing satellite might be able to acquire as it orbits because the earth's state changes and the satellite will rarely repeat the same track.

Table 3.4: Types of resources and their management objectives

Type	Resource	Objective
Finite	Science [1, 19]	maximize science gain minimize loss of science
Semi-finite	Fuel	minimize usage maximize storage
Renewable	Temperature	minimize temperature gain maximize temperature loss
	Memory	minimize data loss maximize data retention
	Battery charge	minimize battery use maximize battery charge

Renewable resources are the most general of types because they include both a finite and infinite dimension. That is, the chase craft can exhaust them completely, or renew them without end. In particular, this thesis will take the approach of maximizing battery state of charge (SOC) or rather minimizing battery loss. The final resource type, *semi-finite*, is a special case of a renewable resource. In theory, fuel or some other commodity might be replenished, but this would occur over significant lengths of time which would exceed the temporal order of magnitude for proximity operations. For this reason, this thesis assumes that semi-finite resources fall outside of its scope.

Mathematical Formulation

The following formulation is a linear representation for modeling various resources. Some will argue that temperature, or battery charge do not behave linearly which is true, but assuming short times, a linear function can approximate their behavior. Beginning with the simple function, f which is linear in time elapsed, Δt yields the equation

$$f(\Delta t) = \alpha \Delta t \tag{3.17}$$

Here, α, represents the rate of increase or decrease of the resource over the duration, Δt.

Any resource, r will have upper and lower bounds so

$$\begin{aligned} r_k &= \min\{r_{k-1} + \alpha \Delta t, ub\} & (3.18) \\ r_k &\geq lb & (3.19) \\ \forall k &\in [1, \ldots, M + N - 1] \end{aligned}$$

This equation does not give the complete picture because it neglects that the resource values r_k are defined at each p_i $\forall i \in [1, \ldots, M]$ and e_j $\forall j \in [0, \ldots, N-1]$. This thesis assumes the value $r_0 \in [lb, ub]$ is given as an input.

Next, defining

$$\alpha = \alpha(\vec{m}_i, \vec{e}_j) = \alpha_{i,j} \tag{3.20}$$

which is the resource rate. The arguments of α represent the parameters $\vec{m}_i$ of P_i and $\vec{e}_j$ of E_j, which are needed to determine how the spacecraft uses its resource during the intersection of P_i and E_j. A skeptic might say that both vectors are not necessary in the cases of the intervals $[p_{i-1}, p_i]$ or $[e_{j-1}, e_j]$, but there is always an active planned and environmental interval with valid parameters. There is one exception to this rule. In the interval defined as $[p_M, e_N]$, the timeline has entered a joint interval where the P_i and their associated $\vec{m}_i$ are no longer defined. This explains why it is unnecessary to compute the last r_j^e and that $|\mathcal{R}| = M + N - 1$. Note also that it is only necessary to compute those $\alpha_{i,j}$ for which $v(i,j) = 1$, because if $v(i,j) = 0$, then P_i and E_j are not allowed to overlap.

The two statements of Equations 3.18 and 3.19 model the linear evolution of the resources with saturation. Equation 3.19 is a policy statement that allows this thesis to assert a minimum value for the resource. The saturation value, ub, comes from a natural modeling constraint; for a battery this would represent full capacity. Without saturation, Equations 3.18 and 3.19 could be equalities. With saturation allowed, it is sufficient to provide Equations 3.18 and 3.19 in conjunction with

$$r \leq ub. \tag{3.21}$$

In the context of a linear program the resource values will be pushed to one of the limits as needed. This occurs in Figure 3-5. The figure shows two cases of evolution

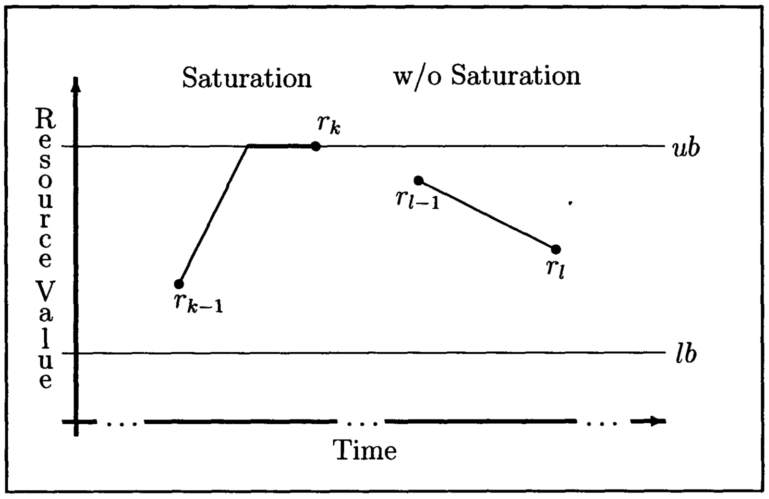


Figure 3-5: Resource evolution with and without saturation

where the left interval begins at r_{k-1} to reach saturation at which point it continues to maintain that value such that $r_k = ub$. The right part of the figure shows the resource value decreasing, but it remains in the interval, $[lb, ub]$.

The problem with Δt is that it does not readily fit into the formulation based on p_i and e_j. The following are the intervals which we will use to compute Δt and their associated resource limit equations. The resource limits are valid on the later bound, i.e. the right-handed bound.

$$\mathcal{I}_1^J = \begin{cases} [e_0, p_1] & \Rightarrow \quad -\alpha_{1,1}p_1 + r_1^p \leq r_0 - \alpha_{1,1}e_0 \\ [e_0, e_1]_{P_1}{}^{\dagger} & \Rightarrow \quad r_1^e \leq -r_0 - \alpha_{1,1}e_0 + \alpha_{1,1}e_1 \end{cases} \tag{3.22}$$

$$\mathcal{I}_k^J = \begin{cases} [p_{i-1}, p_i]_{E_j}{}^{\ddagger} & \Rightarrow \quad r_i^p - r_{i-1}^p + \alpha_{i,j}p_{i-1} - \alpha_{i,j}p_i \leq 0 \\ [p_{i-1}, e_j] & \Rightarrow \quad r_j^e - r_{i-1}^p + \alpha_{i,j}p_{i-1} \leq \alpha_{i,j}e_j \\ [e_{j-1}, p_i] & \Rightarrow \quad r_i^p - r_{j-1}^e - \alpha_{i,j}p_i \leq -\alpha_{i,j}e_{j-1} \\ [e_{j-1}, e_j]_{P_i}{}^{\dagger} & \Rightarrow \quad r_j^e - r_{j-1}^e \leq -\alpha_{i,j}e_{j-1} + \alpha_{i,j}e_j \end{cases} \tag{3.23}$$
$$\forall k \in [2, \ldots, M+N-1], \quad j \in [1, \ldots, N-1], \quad i \in [1, \ldots, M]$$

For Equations 3.22 and 3.23, several key points must be clarified. The $f(\Delta t)$ itself expands linearly into the r_i^e and r_j^p. These are the resource values at the end of the intervals, $\mathcal{I}_k^J$, that end at environmental time, e_j, and planned time, p_i, respectively.

In Equation 3.23, there are four possible combinations of joint intervals. In particular these combinations influence the proper parameters to select for the $\alpha_{i,j}$. For example, the interval, $[e_{j-1}, p_i]$, requires the use of the $\alpha_{j,i}$. It takes the current environmental conditions and the current planned mode parameters. In this interval, that corresponds to the parameters $\vec{e}_j$ and $\vec{m}_i$. However, there is the case which must consider the intervals that are bounded by purely environmental or planned events: $[p_{i-1}, p_i]$ or $[p_{j-1}, p_j]$. In these intervals, it becomes necessary to examine the previous unrepresented event to determine the correct parameters to use.

Formally,

$$\mathcal{R} = \left\{ r_0, r_j^e, r_i^p \in \mathbb{R} \mid lb \leq r_0, r_j^e, r_i^p \leq ub \right\} \tag{3.24}$$
$$\forall i \in [1, \ldots, M], \quad j \in [1, \ldots, N-1].$$

Each value of r is bounded from above and below. These bounds typically come from both a policy and a modeling requirement. In the examples in this thesis for SOC the bound from below is a policy constraint that prohibits the battery from dropping

† P_i added to emphasize the presence of a planned interval.
‡ E_j added to emphasize the presence of an environmental interval.

below a specific level. The upper bound is a constraint that models the battery's saturation level. Another key point regards the initial state of the resource, r_0. This value typically fits into the specified bounds, but the user may be willing to relax the policy constraint for certain eventualities.

3.3 Problem Statement

The preceding sections have shown the mathematical framework necessary to fully describe the problem. The remainder of this chapter will use these tools to present a formal, comprehensive view of the problem and define what it means to have a solution.

In the list of Section 2.2.1 (p. 25), Chien enumerates the requirements for a planning and scheduling system. With special emphasis on constraint and resource management this chapter describes a system that optimally schedules planned events in the midst of environmental events. This scheduling occurs in the context of proximity operations which are considered to be intermediate range with respect to a rendezvous. The thesis assumes that this schedule is useful as part of a larger plan for autonomous rendezvous. Here a *larger plan* refers to a plan which accounts for both the long range and capture phases of rendezvous.

3.3.1 Problem

The first thing to consider is that the scheduler must exert control over the events' execution times, p_i. These values have a direct effect on how the chase satellite consumes its resources, r_i^e and r_j^p, our other control variables. In operations research parlance, these are called *decision variables.* This thesis may refer to these as

$$\vec{x} = \left[p_1\, p_2 \cdots p_M \quad r_1^p\, r_2^p \cdots r_M^p \quad r_1^e\, r_2^e \cdots r_{N-1}^e\right]^T. \tag{3.25}$$

The goal here is to minimize the execution time and maximize the resources. This is

$$\min_{p \in \mathcal{P}, r \in \mathcal{R}} \left[C\left(p_i\right) - \sum_{j=1}^{N-1} r_j^e - \sum_{k=1}^{M} r_k^p \right]. \tag{3.26}$$

The function $C(p_i)$ allows for several types of cost functions. The first one tries to minimize the amount of time it takes to execute all of the planned events. This is written as

$$C\left(p_i\right) = \sum_{i=1}^{M} p_i. \tag{3.27}$$

If the primary concern is to minimize the start to finish time of the whole schedule then the function should be

$$C\left(p_i\right) = p_M. \tag{3.28}$$

Using a special definition of $p_M = e_N$ then the previous equation becomes

$$C\left(p_i\right) = p_{M-1}. \tag{3.29}$$

This allows the scheduler to minimize the execution time of the planned events with the final time in a fixed position.

Next, encoding all of the constraints is complicated by the placement of the p_i within the e_j. This placement problem is challenging enough to require additional mathematical formulation so at this point the thesis assumes that the placement has been solved. The following is an example of a placement with three planned modes and seven environmental events. The placement was earlier referred to as the joint intervals.

$$e_0 = p_0 \leq e_1 \leq e_2 \leq e_3 \leq p_1 \leq e_4 \leq e_5 \leq e_6 \leq p_2 \leq p_3 \leq e_7 \tag{3.30}$$

This example has several features which are significant. The first two elements are always the same; that is $e_0 = p_0$. The thesis asserts the definition $e_0 = p_0$ to achieve a consistent format in the ordering. The thesis also refers to this same statement

(Equation 3.30) omitting the inequality symbols with the implication that they still exist.

Another powerful way to represent this ordering is to omit the inequality symbols, assign the values

$$p_i = 1 \quad e_j = 0 \qquad \forall p \in \mathcal{P}, e \in \mathcal{E}, \tag{3.31}$$

and write it as a sequence. The example of Equation 3.30 becomes

$$(0\,1)\ 0\,0\,0\,1\,0\,0\,0\,1\,1\ (0).$$

Here, the parentheses denote fixed placement of the binary values. This thesis defines the relationship of $e_0 = p_0$ as $e_0 \leq p_0$, but only for this interval ordering. Similarly, it makes the last event environmental because of the requirement, $p_M \leq e_N$.

At this point it becomes clear how to write the constraints. The goal is to fit them into a linear optimization framework as in

$$\mathbf{A}\vec{x} \leq \vec{b}. \tag{3.32}$$

Here, $\vec{x}$ represents the decision variables of Equation 3.25. The following is the matrix-vector pair

$$\mathbf{A} = \left[\begin{array}{c|c} \mathbf{U} & \\ \mathbf{V} & 0 \\ \mathbf{W} & \\ \hline 0 & \mathbf{Y} \\ \hline \multicolumn{2}{c}{\mathbf{Z}} \end{array}\right], \quad \vec{b} = \begin{bmatrix} \vec{b}_{\mathbf{U}} \\ \vec{b}_{\mathbf{V}} \\ \vec{b}_{\mathbf{W}} \\ \vec{b}_{\mathbf{Y}} \\ \vec{b}_{\mathbf{Z}} \end{bmatrix}. \tag{3.33}$$

The first pair, $\mathbf{U}$, $\vec{b}_{\mathbf{U}}$, encodes the constraints of the definition of set $\mathcal{P}$ in Equation 3.7. The second pair, $\mathbf{V}$, $\vec{b}_{\mathbf{V}}$, represents the additional commanded constraints whose rationale was described in the second list of Section 3.2.3 (p. 46). The third pair, $\mathbf{W}$, $\vec{b}_{\mathbf{W}}$ contains the joint interval constraints. These first three pairs are linear in the p_i. The pair $\mathbf{Y}$, $\vec{b}_{\mathbf{Y}}$ is linear in the r_i^p and the r_j^e and they encode the resource policy and saturation constraints. The final pair is linear in the p_i, r_j^e, and r_k^p. It is represented in the pair, $\mathbf{Z}$, $\vec{b}_{\mathbf{Z}}$, which helps define the temporal evolution of the resource.

3.3.2 Solutions

Now that the problem is defined, what defines an optimal solution? Continuing to assume the optimal joint intervals, the optimal solution which is the optimal schedule will be the one which minimizes Equation 3.26 subject to Equation 3.33. This optimal schedule will still observe all of the constraints. The p_i will be specified as if they are beads on a timeline (Pack-Kaelbling) with range $[e_0, e_N]$. Their interval constraints will be satisfied, as will the other temporal constraints which might creep up as described in the second list of Section 3.2.3 (p. 46).

Chapter 4

A Mixed Integer Programming Approach

The critical assumption of Section 3.3.1 is that the relative ordering of p_i and e_j is given. The arrangement of these times and their respective intervals governs the evolution of the resources and suggests that the planned modes agree with the environmental conditions that overlap. Within these instances of planned and environmental overlap the algorithm must still select how to optimally move the p_i within its respective joint interval to make an optimal plan accounting for both the resources limits and execution time.

In the spirit of operations research, this chapter extends the linear formulation of Chapter 3 into a MIP formulation that considers not only the joint interval problem, but also the issue of complete temporal optimality. The software package *XPRESS-MP* will be used in conjunction with Matlab to generate examples and results.

Section 4.1 introduces some binary decision variables which are critical to selecting the optimal ordering of the p_i among the e_j. Section 4.2 amends the linear formulation to capitalize on the new binary decision variables. Section 4.3 shows how to combine all of the modified constraints and apply the solver. Chapter 5 will then discuss some examples and their solutions.

4.1 Solving the Joint Interval Problem

The joint interval problem refers to the arrangement of p_i among the e_j, or rather how do the P_i overlap with the E_j. A binary decision variable, $b_{m,n}$, can characterize this overlap. This variable can be integrated into the other constraint equations mentioned in Section 3.2. The solver can discretely select the interval arrangement, which activates a particular set of $b_{m,n}$, and based on these values it judges the optimality of the execution time and resource limits.

4.1.1 Defining the Interval Variable

The interval variable, $b_{m,n}$, is defined as

$$\begin{aligned} b_{m,n} = 1 &\Leftrightarrow P_m \text{ overlaps } E_n \\ &\Leftrightarrow P_m \cap E_n \neq \emptyset \\ &\Leftrightarrow (p_{m-1} \leq e_n) \wedge (e_{n-1} \leq p_m) . \end{aligned} \tag{4.1}$$

This translates into four different styles of overlapping intervals depicted pictorially in Figure 4-1. Each of these overlap cases satisfies the conjunction of Equation 4.1.

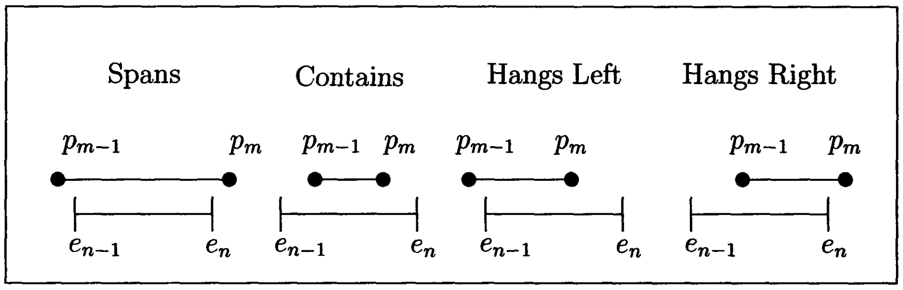


Figure 4-1: Four types of interval overlap

Section 4.1.3 will more thoroughly examine the mathematical representation of these four overlap cases and describe how the $b_{m,n}$ can identify each of the instances.

4.1.2 Defining the Interval Space

Collecting all of the $b_{m,n}$ together forms the interval space

$$\mathbf{B} = [b_{m,n}] \in \{0,1\} \qquad \forall m \in [1,\ldots,M+1], \quad n \in [1,\ldots,N]. \tag{4.2}$$

In a solution, the path will be specified as a series of ones from the top left position, $b_{1,1}$ to the bottom row, $b_{M+1,n}$. The number of ones in the search space should be at least $M+1$ and no more than $M+N$. All other entries in $\mathbf{B}$ are zero. In a valid, feasible path, the ones are adjacent and only traverse down and to the right. Figure 4-2 is an example of a possible path in the interval space based on the inequality of Equation 3.30. The solid blocks indicate that $b_{m,n} = 1$ and the unfilled blocks, that $b_{m,n} = 0$. The easiest way to extract the timeline from the interval space is to observe

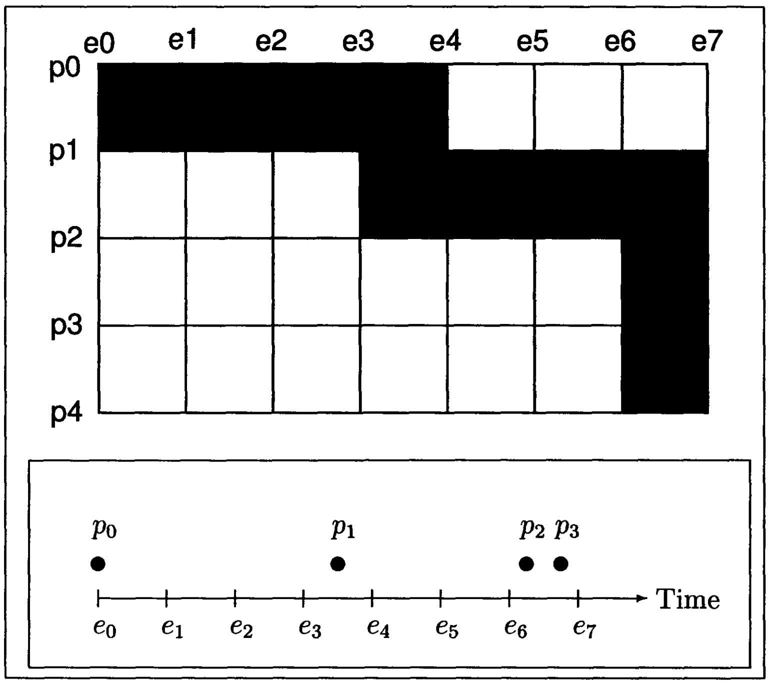


Figure 4-2: An example path in interval space with its associated timeline

the transitions of the shaded blocks. By the definition of Section 3.3.1, the reader knows that each order begins with a 0 1 which indicates that $e_0 = p_0$; this ordering always ends with a 0. Otherwise, to extract the body of the ordering follow the path.

At each transition, i.e. a move right or down, record a move to the right as a 0 and a move down as a 1.

It is useful to note that the interval space of Figure 4-2 refers to a value p_4 which is non-existent in its path description in Equation 3.30 and on the timeline of the same figure. This additional planned event is necessary to capture the placement of the final planned event, in this case p_3.

These equations are the *structural* equations of the matrix **B**. They capture the idea that a path must originate in the upper left corner and move only down and right to the bottom row. The first equation indicates that the both the first planned and first environmental intervals must always overlap.

$$b_{1,1} = 1. \tag{4.3}$$

The next equation indicates that along any diagonal running from the northeast to the southwest there is at most one instance of overlap.

$$\sum_{\{m,n\,:\,n+m=q+2\}} b_{m,n} \leq 1 \qquad \forall q \in [1,\ldots,M+N-1] \tag{4.4}$$

Explicitly, expanding Equation 4.4 for the overlap space of Figure 4-2 gives

$$\begin{aligned}
q=1 &\Rightarrow b_{1,2}+b_{2,1} \leq 1\\
q=2 &\Rightarrow b_{1,3}+b_{2,2}+b_{3,1} \leq 1\\
q=3 &\Rightarrow b_{1,4}+b_{2,3}+b_{3,2}+b_{4,1} \leq 1\\
q=4 &\Rightarrow b_{1,5}+b_{2,4}+b_{3,3}+b_{4,2} \leq 1\\
q=5 &\Rightarrow b_{1,6}+b_{2,5}+b_{3,4}+b_{4,3} \leq 1\\
q=6 &\Rightarrow b_{1,7}+b_{2,6}+b_{3,5}+b_{4,4} \leq 1\\
q=7 &\Rightarrow b_{2,7}+b_{3,6}+b_{4,5} \leq 1\\
q=8 &\Rightarrow b_{3,7}+b_{4,6} \leq 1\\
q=9 &\Rightarrow b_{4,7} \leq 1
\end{aligned}$$

The following two inequalities assert the property of *forward propagation.* That is the path moves down and to the right, but obviously can only makes one move at a time.

$$b_{m,n} - b_{m+1,n} - b_{m,n+1} \leq 0 \qquad \forall m \in [1,\ldots,M],\ n \in [1,\ldots,N-1] \qquad (4.5)$$

$$b_{m,N} - b_{m+1,N} \leq 0 \qquad \forall m \in [1,\ldots,M] \qquad (4.6)$$

Equation 4.5 suggests that from $b_{m,n}$ the path moves to the right $b_{m,n+1}$ and down $b_{m+1,n}$. Other constraints such as the diagonal property of Equation 4.4 prevent the path from traversing both to the right and down from the same $b_{m,n}$. Figure 4-3 indicates this erroneous situation.

Equation 4.6 prevents the path from trying to move down when there is not a subsequent row. In this case the path may only move right. This implies moving across environmental intervals as the overlap path has already reached the last planned event.

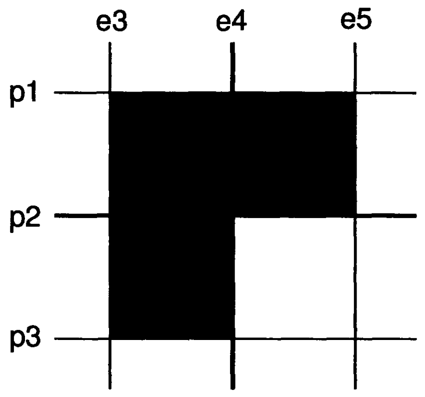


Figure 4-3: Forward propagation without applying other constraints

The next four equations encode the *backward propagation* constraints. These take a particular $b_{m,n}$ and enforce the idea that it should have come from above, $b_{m-1,n}$, or from the left (not politically speaking), $b_{m,n-1}$.

$$\begin{aligned} b_{m,n} - b_{m-1,n} - b_{m,n-1} &\leq 0 \quad \forall m \in [2,\ldots,M],\ n \in [2,\ldots,N] && (4.7) \\ b_{m,1} - b_{m-1,1} &\leq 0 \quad \forall m \in [2,\ldots,M] && (4.8) \\ b_{1,n} - b_{1,n-1} &\leq 0 \quad \forall n \in [2,\ldots,N] && (4.9) \\ b_{M+1,n} - b_{M,n} &\leq 0 \quad \forall n \in [1,\ldots,N] && (4.10) \end{aligned}$$

The last pair of structural inequalities implies that upon arriving at the final planned event, the path terminates. This revokes any reason to move right; moving right is unnecessary because the implication of an environmental condition is that it occurs regardless if a planned mode overlaps.

$$\sum_{n\in[1,\ldots,N]} b_{M+1,n} \leq 1 \qquad (4.11)$$

$$\sum_{n\in[1,\ldots,N]} b_{M+1,n} \geq 1 \qquad (4.12)$$

Note that these inequalities are opposite which in fact is another way to say equals. This is useful for some MIP solvers.

There is redundancy included in the equations. Most MIP solvers are capable of rejecting redundant constraints, so eliminating these cases is not a concern.

4.1.3 Using the Interval Variable to Determine the Previous Event

Figure 4-1 presents the four specific ways in which planned and environmental intervals can overlap, but without any method to determine which case actually exists. This section will delineate the IP equations and relate them to *true* and *false* states. This will be useful in isolating the four cases to treat the resource limits which are dependent on this knowledge.

Contains

This type of overlap features a planned interval entirely contained within an environmental interval. In the joint interval form, this is

$$e_{n-1} \leq p_{m-1} \leq p_m \leq e_n. \tag{4.13}$$

In the interval space if the algorithm considers the $b_{m,n}$ then this overlap type refers to the overlaps above, $b_{m-1,n}$, and below, $b_{m+1,n}$. This type of overlap is present for the $b_{m,n}$ and results in the following statements

$$(true) \Leftrightarrow 2 - b_{m+1,n} - b_{m-1,n} = 0 \tag{4.14}$$

$$(false) \Leftrightarrow 2 - b_{m+1,n} - b_{m-1,n} \geq 1 \tag{4.15}$$

$$\forall m \in [2, \ldots, M], \ n \in [1, \ldots, N]$$

Hangs Left

Denoting this type of overlap as

$$p_{m-1} \leq e_n \leq p_m \leq e_{n+1}. \tag{4.16}$$

In the interval space this situation refers to a $b_{m,n+1}$, its left neighbor, $b_{m,n}$, and its bottom neighbor, $b_{m+1,n+1}$. The following statements will be useful in writing the other constraint equations.

$$(true) \Leftrightarrow 3 - b_{m,n} - b_{m,n+1} - b_{m+1,n+1} = 0 \tag{4.17}$$

$$(false) \Leftrightarrow 3 - b_{m,n} - b_{m,n+1} - b_{m+1,n+1} \geq 1 \tag{4.18}$$

$$\forall m \in [1, \ldots, M], \ n \in [1, \ldots, N-1]$$

There is a special instance for this overlap situation, $m = 1$. This means $b_{2,1} = 1$

and of course $b_{1,1} = 1$. Thus, the equations become

$$(true) \;\Leftrightarrow\; 1 - b_{2,1} = 0 \tag{4.19}$$

$$(false) \;\Leftrightarrow\; 1 - b_{2,1} \geq 1. \tag{4.20}$$

Hangs Right

A third instance of overlap is the case of Hangs Right. In the joint interval language this is denoted as

$$e_{n-1} \leq p_m \leq e_n \leq p_{m+1}. \tag{4.21}$$

The central overlap variable is $b_{m+1,n}$. This case refers to its upper neighbor, $b_{m,n}$ and its right neighbor $b_{m+1,n+1}$. Writing the other constraints will be easier with the following conditional statements

$$(true) \;\Leftrightarrow\; 3 - b_{m,n} - b_{m+1,n} - b_{m+1,n+1} = 0 \tag{4.22}$$

$$(false) \;\Leftrightarrow\; 3 - b_{m,n} - b_{m+1,n} - b_{m+1,n+1} \geq 1 \tag{4.23}$$

$$\forall m \in [1, \ldots, M-1],\; n \in [1, \ldots, N-1].$$

If these bounds appear short or to be off-by-one that is because this formulation is not concerned with the extra planned event, p_{M+1}, or rather with what happens after visiting $p_M \leq e_N$.

Spans

The remaining overlap type is the Spans case where a planned interval spans over a whole environmental interval. This is

$$p_m \leq e_{n-1} \leq e_n \leq p_{m+1}. \tag{4.24}$$

Focusing on the central interval variable which for this case is $b_{m,n}$, indicates its left neighbor as $b_{m,n-1}$, and its right neighbor as, $b_{m,n+1}$. These yield the following statements

$$\begin{aligned}
(true) \quad &\Leftrightarrow \quad 2 - b_{m,n-1} - b_{m,n+1} = 0 && (4.25)\\
(false) \quad &\Leftrightarrow \quad 2 - b_{m,n-1} - b_{m,n+1} \geq 1 && (4.26)\\
&\forall n \in [2, \ldots, N-1], \; m \in [1, \ldots, M].
\end{aligned}$$

This overlap type has a special case when $n = 1$. That is

$$\begin{aligned}
(true) \quad &\Leftrightarrow \quad 1 - b_{2,1} = 0 && (4.27)\\
(false) \quad &\Leftrightarrow \quad 1 - b_{2,1} \geq 1. && (4.28)
\end{aligned}$$

4.1.4 Integrating the Interval Variables

The preceding sections laid the groundwork for the MIP. Using the interval space of Section 4.1.2 and the constraints from Section 3.3.1 the two entities are necessary to solve both the joint interval problem and the temporal/resource limit problem. The first task is to examine the temporal constraints and their relationship to the interval variables. Then, the thesis will consider the interval variables along with the resource limits.

4.1.5 Evaluating Interval Compatibility

The first list of Section 3.2.3 (p. 46) refers to the categories of temporal constraints. This includes the joint interval problem which the formulation of this chapter can fully solve. This list mentions interval compatibility which is briefly described in Section 3.2.3 using a function, $v(i, j)$. Because of the definition of the interval space, the binary function must accommodate the new size, which adds the N extra values of row $M + 1$. Redefining Equation 3.16 gives the function

$$v(m, n) = \begin{cases} 1 & m = M + 1 \\ 1 & (P_m \text{ compatible with } E_n) \wedge (m \leq M) \\ 0 & \text{otherwise} \end{cases} \tag{4.29}$$

$$\forall m \in [1, \ldots, M+1], \; n \in [1, \ldots, N].$$

It should be noted that the idea of *compatible with* refers to the overlapping intervals as defined in Equation 4.1.

Using the function of Equation 4.29 each of the $(M+1)N$ instances of variable overlapped should be evaluated. If any are determined to be prohibited instances then the algorithm must explicitly declare

$$b_{m,n} = 0. \tag{4.30}$$

This will result in a set of constraints with no solution. If at the beginning $b_{1,1} = 0$ then there is a conflict with the constraint of Equation 4.3; this schedule would be deemed infeasible. Likewise if all $b_{M+1,n} = 0 \; \forall n \in [1, \ldots, N]$, then this schedule would be infeasible. In most cases, the user specifies the requirements properly, and the solver readily finds a solution.

4.2 Adapting the Linear Model

To this point, this chapter has defined the interval variable, its related decision space, and specified a formulation (Section 3.3.1) for solving the planning and scheduling problem. The remaining step is to use the interval space to find the optimal solution at the same time as choosing the relative ordering of the p_i among the e_j. The first task is to examine the temporal constraints and their relationship to the interval variables. Then, the interval variables along with the resource limits will be considered.

4.2.1 Temporal Constraints

Section 3.2.3 defined the p_i such that they each successive p_i is greater than or equal to the previous value. The MIP must enforce these inequalities

$$p_{m-1} \leq p_m \qquad \forall m \in [2, \ldots, M] \tag{4.31}$$

to maintain the formulation of Chapter 3.

To facilitate the placement of the planned events among the environmental events, it is imperative to use the idea in Equation 4.1 and relate the p_i to the e_j. The following inequalities manage this placement.

$$\begin{aligned} e_{n-1} &\le p_m + T\left(1 - b_{m,n}\right) \quad \forall m \in [1, \ldots, M]\,,\ n \in [2, \ldots, N] && (4.32) \\ p_{m-1} &\le e_n + T\left(1 - b_{m,n}\right) \quad \forall m \in [2, \ldots, M+1]\,,\ n \in [1, \ldots, N-1] && (4.33) \\ &\text{where } T \gg e_N \text{ is a very large positive constant} \end{aligned}$$

These inequalities yield the joint intervals described in Section 3.2.3. In the case where interval P_m overlaps E_n, then $b_{m,n} = 1$ leaving the statements $p_{m-1} \le e_n$ and $e_{n-1} \le p_m$, which is Equation 4.1. When $b_{m,n} = 0$, the value T overpowers the inequality, effectively *relaxing* the constraint, so that it does not restrict the solver in finding the solution.

Finally, the algorithm must observe the rules regarding the beginning and end of the schedule. This requires the explicit statement,

$$\begin{aligned} p_0 &= e_0 && (4.34) \\ p_m &\ge e_0 \quad \forall m \in [1, \ldots, M] && (4.35) \\ p_m &\le e_N \quad \forall m \in [1, \ldots, M] && (4.36) \end{aligned}$$

4.2.2 Resource Evolution

The next step in completing the formulation is to specify the resource evolution. This necessitates the continuous real variables, r^p_m and r^e_n along with the initial resource value r_0 at time $e_0 = p_0$. First this section will discuss the resource consumption or replenishment rate $\alpha_{m,n}$. Then it will consider modifications to the inequalities of the resource evolution in Equations 3.22 and 3.23.

The function $\alpha_{m,n}$ computes the rate of consumption or replenishment of a resource in the given joint interval. It takes into account the parameters of the planned modes, $\vec{m}_m$, and the environmental conditions, $\vec{e}_n$. The function, $\alpha_{m,n}$ should evaluate to a value that realistically describes increases and decreases in the resource values within the allowed range of the upper and lower bounds, $[lb, ub]$ across the considered temporal boundaries. The temporal bounds come from the solution and they are directly linked to the times described by P_m and E_n.

It is useful to note that the final additive term of the following equations is a function of the overlap variables. They are written in a way which corresponds to the truth values of the four types of overlap specified in Section 4.1.3. In these final additive terms, the coefficient L serves a purpose similar to T. The value, L, relaxes the constraint when the quantity it multiplies evaluates to a nonzero value. Like T, L should be a large positive value.

Equation 3.22 contains the special cases for the general formulation. Even with the new interval space, there are still some special cases. The first of these occurs when the first joint interval is $[e_0, p_1]$, and so it ends with resource value r_1^p. These are

$$r_1^p \leq r_0 + \alpha_{1,1}(p_1 - e_0) + L(1 - b_{2,1}). \tag{4.37}$$

Similarly, if the first interval ends with an environmental event then,

$$r_1^e \leq r_0 + \alpha_{1,1}(e_1 - e_0) + L(1 - b_{1,2}). \tag{4.38}$$

This means that the first interval was $[e_0, e_1]$, and it ends with resource value r_1^e.

The following two equations represent a unique case because they combine elements of Equations 3.22 and 3.23. The first one is written for a static planned event, p_1 and a dynamic environmental event, e_n such that we have the interval $[e_n, p_1] \quad \forall n \in [1, \ldots, N-1]$ which leaves the resource value r_1^p.

$$r_1^p \leq r_n^e + \alpha_{1,n+1}(p_1 - e_n) + L(2 - b_{1,n+1} - b_{2,n+1}) \quad \forall n \in [1, \ldots, N-1] \tag{4.39}$$

The second of the two hybrid equations represents the resource value at r_1^e within the interval $[p_m, e_1]$ $\forall m \in [1, \ldots, M-1]$.

$$\begin{aligned} r_1^e &\leq r_m^p + \alpha_{m+1,1}(e_1 - p_m) + L(3 - b_{m,1} - b_{m+1,1} - b_{m+1,2}) \\ &\forall m \in [1, \ldots M-1] \end{aligned} \tag{4.40}$$

The next four inequalities model the resource limits for the general cases of resource evolution seen in Equation 3.23. This inequality treats the contains relationship where the interval is written $[p_{m-1}, p_m]$, and it represents the resource value, r_m^p.

$$\begin{aligned} r_m^p &\leq r_{m-1}^p + \alpha_{m,n}(p_m - p_{m-1}) + L(2 - b_{m+1,n} - b_{m-1,n}) \\ &\forall m \in [2, \ldots, M],\ n \in [1, \ldots, N] \end{aligned} \tag{4.41}$$

For the Hangs Left interval, $[e_n, p_m]$ there is the inequality for the resource value r_m^p.

$$\begin{aligned} r_m^p &\leq r_n^e + \alpha_{m,n+1}(p_m - e_n) + L(3 - b_{m,n} - b_{m,n+1} - b_{m+1,n+1}) \\ &\forall m \in [2, \ldots, M],\ n \in [1, \ldots, N-1] \end{aligned} \tag{4.42}$$

The final two inequalities respectively represent the Spans case of interval $[e_{n-1}, e_n]$ and the Hangs Right case of interval $[p_m, e_n]$. They model the resource value at r_n^e.

$$\begin{aligned} r_n^e &\leq r_{n-1}^e + \alpha_{m,n}(e_n - e_{n-1}) + L(2 - b_{m,n-1} - b_{m,n+1}) \\ &\forall n \in [2, \ldots, N-1],\ m \in [1, \ldots, M] \end{aligned} \tag{4.43}$$

$$\begin{aligned} r_n^e &\leq r_m^p + \alpha_{m+1,n}(e_n - p_m) + L(3 - b_{m,n} - b_{m+1,n} - b_{m+1,n+1}) \\ &\forall n \in [2, \ldots, N-1],\ m \in [1, \ldots, M-1] \end{aligned} \tag{4.44}$$

4.3 Combining the Amended Formulation

The preceding sections have modified the original formulation of Chapter 3. The remaining step is to make a complete statement about the method to achieve a practical solution. Table 4.1 enumerates the six steps required to implement the MIP. Only the fifth step requires the use of a solver package.

Table 4.1: Implementing a MIP

1.	Specify planned modes, **M**, and environmental conditions, **E**
2.	Check for interval compatibility
3.	Compute resource rates
4.	Collect interval, temporal, and resource constraints
5.	Evaluate MIP with a solver package
6.	Review solution

Specifying the planned modes and environmental conditions is straightforward. Using their data, the algorithm evaluates the interval compatibility with the function $v\,(m, n)$. If a given overlap case evaluates to 1 then perform Step 3 which is to compute the associated resource rate, $\alpha\,(m, n)$. Only compute the resource rate for valid overlapping intervals since the solver cannot place two intervals that are invalid. Finally, collect all of the constraints in one place. First, there is the vector of decision variables.

$$\vec{x}_{MIP} = \left[b_{1,1}\; b_{1,2} \cdots b_{1,N}\; b_{2,1}\; b_{2,2} \cdots b_{2,N} \cdots b_{M+1,N}\; \vec{x}^T\right]^T \tag{4.45}$$

Note that the new vector expands the overlap variables along with the previous vector of decision variables, $\vec{x}$, from Equation 3.25.

As in Section 3.3.1, here the associated constraints are specified. The matrix **B** encodes the structural definition of the overlap matrix. The matrix is defined to include individual constraints which restrict overlap due to a conflict; specifically this requires encoding $b_{m,n} = 0$ for those restricted overlap cases. To include the temporal constraints, it is necessary to encode Equations 4.31–4.36 along with the other commanded temporal constraints from the second list of Section 3.2.3 on page 46. Modifying $\mathbf{V}$ to include these results in $\tilde{\mathbf{V}}$. For the MIP formulation, the matrix of joint interval constraints, $\mathbf{W}$ is unnecessary. Instead, $\mathbf{B}$, will aid the solver in selecting the correct overlap conditions, i. e. joint intervals. The MIP formulation keeps the resource policy and saturation constraints of $\mathbf{Y}$ but modifies $\mathbf{Z}$ such that it is linear in terms of $b_{m,n}$, p_m, r^e_n, and r^p_m. This leaves the constraint matrix and vector as

$$\tilde{\mathbf{A}} = \left[\begin{array}{c|c|cc} \mathbf{B} & 0 & 0 & 0 \\ \multicolumn{2}{c|}{\tilde{\mathbf{V}}} & 0 & 0 \\ 0 & 0 & \multicolumn{2}{c}{\mathbf{Y}} \\ \multicolumn{4}{c}{\tilde{\mathbf{Z}}} \end{array}\right], \quad \vec{b} = \begin{bmatrix} \vec{b}_{\mathbf{B}} \\ \vec{b}_{\tilde{\mathbf{V}}} \\ \vec{b}_{\mathbf{Y}} \\ \vec{b}_{\tilde{\mathbf{Z}}} \end{bmatrix}. \tag{4.46}$$

Finally, applying the constraints of Equation 4.46 in a solver package using the cost function of Equation 3.26 yields the optimal schedules. The following chapter will examine several cases resulting from the method of Table 4.1.

For a thorough application of the algorithm in Table 4.1, see Appendix B in conjunction with the example of Section 5.2.2.

[Except for this sentence, this page intentionally left blank.]

Chapter 5

Results

In Chapter 4, the thesis defined a comprehensive approach to solving the proximity operations scheduling problem. This chapter will focus on reviewing results and experiments.

In particular, this chapter will examine several test cases which offer a realistic representation of an observation mission's proximity operations. Experiments rooted in these test cases will reveal some of the algorithm's features and highlight its capability and extendibility. Here *extendibility* refers to the solution method's robustness to modifications, which will increase the fidelity of the output.

Section 5.1 delineates the language and basic elements of the missions to be discussed. Then, Section 5.2 gives some examples to accustom the reader to the general mission format for the remaining experiments. The first one presents a very basic mission with only three planned modes. The second example schedules a mission with more realistic activities.

Sections 5.3–5.6 all consider the constrained and unconstrained cases of the mission's main observation goal. An example of a constrained case would be to constrain the observation to occur in daylight or darkness. The unconstrained case does not care whether it occurs in daylight, darkness, or both. Section 5.5 examines a minor modification to the formulation that affects resource management, and ultimately the solution. This requires the addition of a continuous decision variable, some additional constraints based on this variable, and its inclusion in the cost function. Finally, Section 5.6 explores a modification to the cost function that seeks to reduce the execution time of specific planned modes.

5.1 General Mission Framework

The remainder of this chapter will explore several example plans and their respective MIP solutions. Figure 5-1 delineates the planned modes for a basic mission. Each block represents a mode and directly corresponds to a m_j of which there are $M = 7$. Usually the first and last modes are *station keeping* modes because they serve as start and goal states and act as a sort of buffer between other events. They provide the opportunity for the scheduler to delay or hurry the subsequent and previous activities. Furthermore, in a hierarchical planner, the start and goal modes function as handles for the planner to grab and use in its algorithm [1, 8, 11].

SK	Station Keeping
T	Transitional Maneuver
Obs	Observation

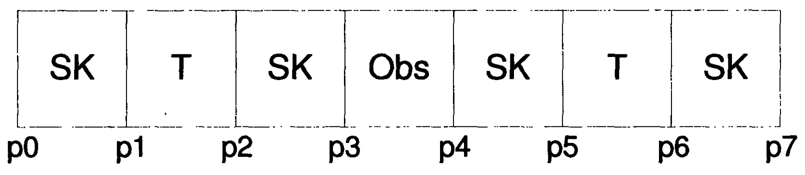


Figure 5-1: A basic proximity operations observation mission

In general the chase satellite may perform the station keeping activity along the radial, $\bar{r}$, or velocity, $\bar{v}$, vectors of the target craft. It is worth noting that in terms of fuel consumption, station keeping on the radial vector is expensive. More generally, station keeping anywhere but on the velocity vector is costly*. Section 5.6 offers an approach to minimize this fuel cost.

Additionally, station keeping is an excellent mode to perform communication contacts for uploading and downloading data. During these modes, the satellite operators

*Mathematically, any combination of $\bar{r}$, $\bar{v}$, or $\bar{r} \times \bar{v}$ (cross-track) can transmit navigation updates or commands giving the final authorization to proceed with the rendezvous. There are two communication bands. *Band 1* represents a low data rate system whereas *Band 2* allows transfers of large amounts of data. This thesis assumes that Band 2 conveys information rich material such as video and images.

In keeping with its multipurpose role, station keeping may also serve as a mode to update the relative position error between the chaser and target using the acquisition sensor. More often, it will be a mode to regenerate and/or dissipate resources.

The last point of station keeping addresses a subtle issue, and one which must be explained carefully. Recall the validating function of Equation 3.16; this determines whether an environmental and planned interval are compatible based on the $\vec{e}_j$ and $\vec{m}_i$. For example, consider the application of the validating function to the planned modes and environmental conditions resulting in the following valid, shaded overlaps of Figure 5-2. Here, the planned interval, P_2, was constrained to occur in daylight,

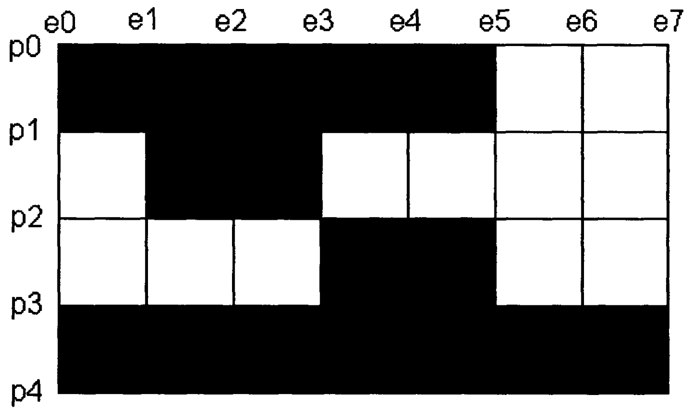


Figure 5-2: An example of overlap space depicting $v\left(i,j\right)$ with no solution

and the planned mode, P_3, should occur during a communication window. Now, unless the daylight occurs just before the beginning of the communication window, then there is no valid path and therefore no solution. The space does not allow a solution because some of the intervals are incompatible. In Figure 5-2 intervals P_2

and E_4 are unshaded and, therefore, incompatible. Inserting an extra unconstrained station keeping mode between the two offending planned modes, intervals P_2 and P_3, remedies this conflict.

The *observation* mode in Figure 5-1 gives the chase craft the opportunity to observe the target using its camera to make images and video. In certain circumstances, it might transmit video to the earth using Band 2. During this time the chaser might use passive motion to move around the target, or it may use fuel to physical fly around the target. Typically, the observation mode is temporally constrained to occur with finite duration as a result of a communication window duration, astrodynamics constraints, or both.

Finally, the *transition* mode uses the thruster to transit between station keepings. The guidance computer on the chase satellite gives the scheduler temporal constraints governing the transition maneuver. These are integrated into the solution.

5.2 Test Cases

Beyond the basic mission framework, this chapter begins the exploration of the algorithm's capabilities. Two examples follow. The first is an extremely simple case. The second one will conform to the more rigorous example laid out in the framework.

5.2.1 Example 1

This is a case of a changing from one station keeping mode to another station keeping mode via a transition maneuver. This operation will be important in Section 5.6. There the thesis examines station keeping on the $\bar{r}$ versus the $\bar{v}$.

Figure 5-3 captures the three modes which the user specifies. In this example, guidance commands the transition maneuver to last 225 *sec* which the algorithm must accommodate. All three modes are unconstrained with respect to lighting or communication requirements. The first and third modes use the *cycle* pointing mode, which periodically observes the target for position updates, and the spacecraft spends the remainder of its time orienting the solar arrays toward the sun. The second mode

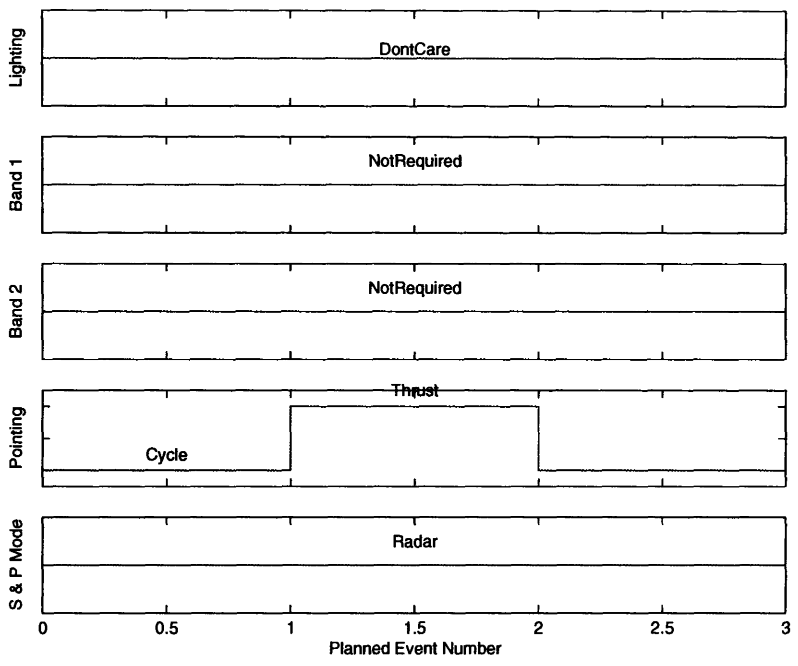


Figure 5-3: Example 1: User-specified planned modes

uses the *thrust* attitude mode which orients the chase satellite's thruster to perform the maneuver. The last row shows that the spacecraft uses its *radar* for tracking the target in each of the modes.

The current plan must contend with the environmental conditions depicted in Figure 5-4. The reader should note that there are several Band 1 communication windows, which the user decided not to use in the mission plan.

The following solution applies the MIP algorithm to the problem with the following formulation. It uses the decision variables of Equation 4.45 subject to the constraints of Equation 4.46, and it uses the cost functions specified in Equations 3.26 and 3.29, explicitly stated here.

$$\min_{p \in \mathcal{P},\, r \in \mathcal{R}} \left[p_2 - 0.1 \sum_{m=1}^{3} r_m^p - 0.1 \sum_{n=1}^{21} r_n^e \right] \tag{5.1}$$

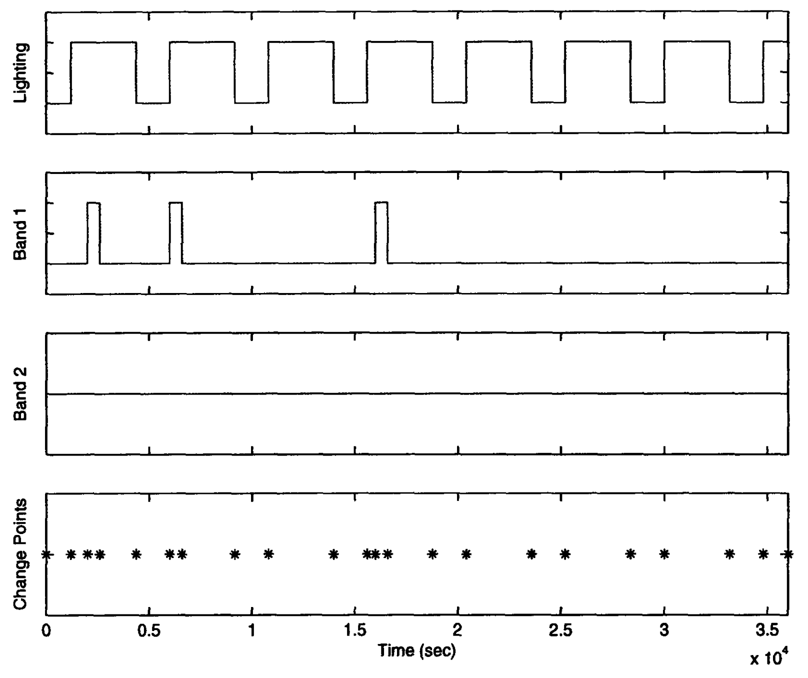


Figure 5-4: Example 1: Environmental conditions

This cost function indicates the intent to minimize the time when the second to last event occurs and to maximize resource values. The negative coefficients on the resource values ensure that they are maximized, where the resource here is the state of charge (SOC) of the chase satellite's battery.

Applying the algorithm to the problem yields the optimal path shown in Figure 5-5. Note that in Figure 5-5 the algorithm schedules p_0, p_1, and p_2 within the first environmental interval, $E_1 = [e_0, e_1] = [0, 1200]$.

Finally, Figure 5-6 presents the complete solution timeline. The first planned mode labeled 1 in the last row is barely distinguishable since it has zero duration. Likewise the transition maneuver only lasts for a handful of minutes so its label, 2, is difficult to see. In this solution, $p_0 = p_1 = 0$ *sec*. This behavior is expected because the initial state of charge is 0.75 where full capacity is 1.0. The solver realizes that it can immediately schedule the transition maneuver without consuming enough charge

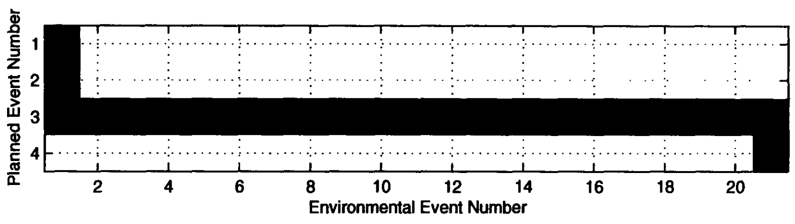


Figure 5-5: Example 1: Optimal path

to violate the lower bound of 0.6.

There are several important features in Figure 5-6. First, the reader should know that the dotted lines represent environmental conditions and the solid lines are the planned activities. In cases where the planned modes directly coincide with the environmental conditions, they will appear as solid lines.

Next, the operator never desired to use any of the communication windows, so they remain unscheduled. Notice that the corresponding resource rate is diminished during these windows. Regardless of the user's desire to use a Band 1 communication window the satellite must activate its communication equipment as a fail-safe measure. Also, consider that during periods of sunlight the resource rate tends to be positive unless the other environmental conditions conspire with the sensor and payload mode to consume more power than the power source can generate. Over the course of the mission, notice the upward trend in the charge as it approaches saturation.

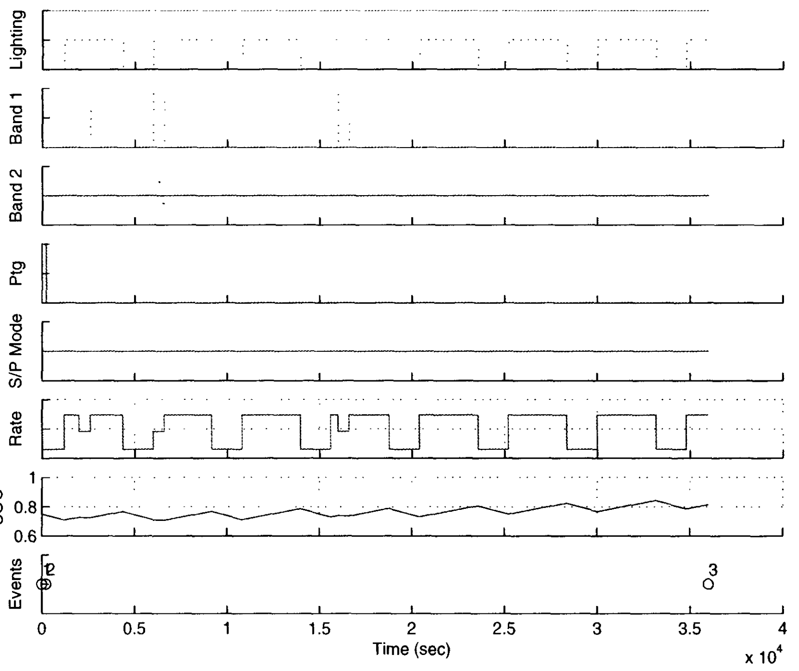


Figure 5-6: Example 1: Optimal schedule

5.2.2 Example 2

The second example encapsulates a more representative observation mission than the example of Section 5.2.1. This mission uses ten modes pictured in Figure 5-7. The

SK Start	SK Band 1	T	SK	SK Acquire	Obs Band 2	SK	SK Band 1	T	SK Goal

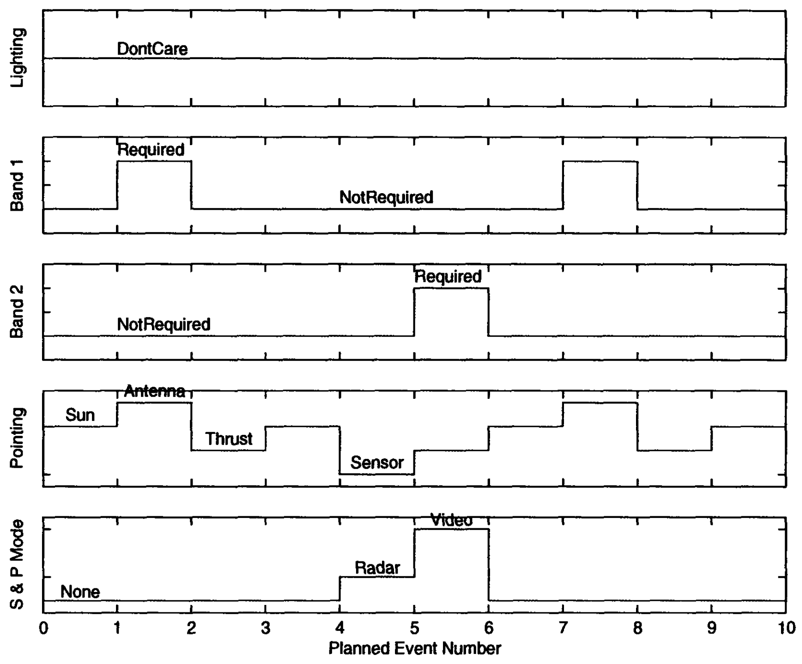


Figure 5-7: Example 2: User-specified planned modes

station keeping mode labeled with *acquire* simply denotes that the chase satellite will use the opportunity to acquire the target. Typically, acquisition occurs before proximity operations begins, but the aim here is to expose the algorithm's scope of capabilities. The user constrains this mode to endure at least ten minutes and execute for no more than forty-five minutes.

In this mission, the lighting requirements are all unconstrained as indicated in Figure 5-7. There is a requirement to use Band 1 in modes P_2 and P_8. Mode P_6 communicates using Band 2. The pointing requirements in this mission use the *antenna* and *sensor* modes. The antenna attitude mode orients the spacecraft's antenna to ensure good signal transmission and reception, and the sensor attitude mode points the spacecraft's sensor at the target to acquire it. In the sensor and payload requirements of the last row, the chase satellite uses its *radar* to detect the target, and it uses equipment to make a *video*. For more background information on this mission see Appendix B.

The environmental conditions presented in Figure 5-8 are the same as those of the example in Section 5.2.1 except for the presence of the Band 2 condition.

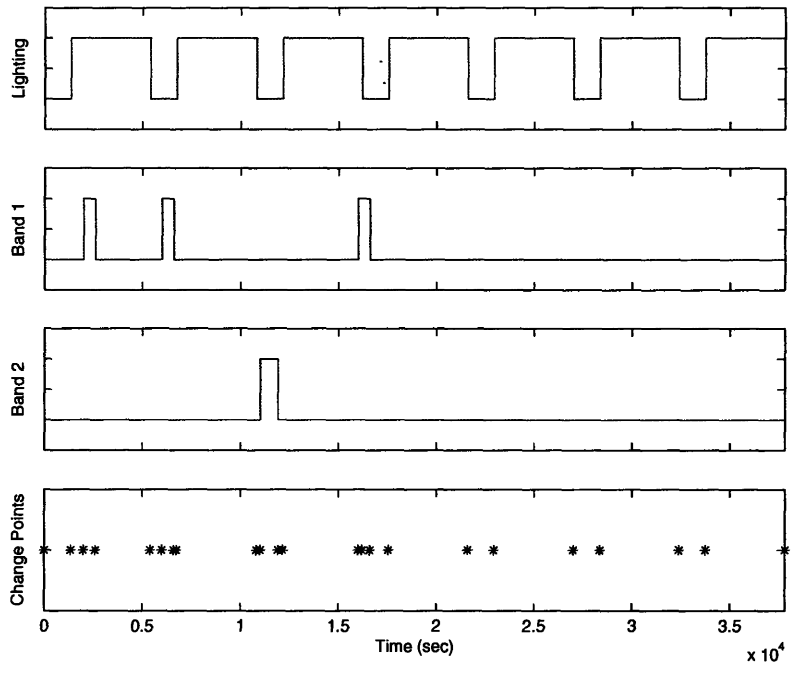
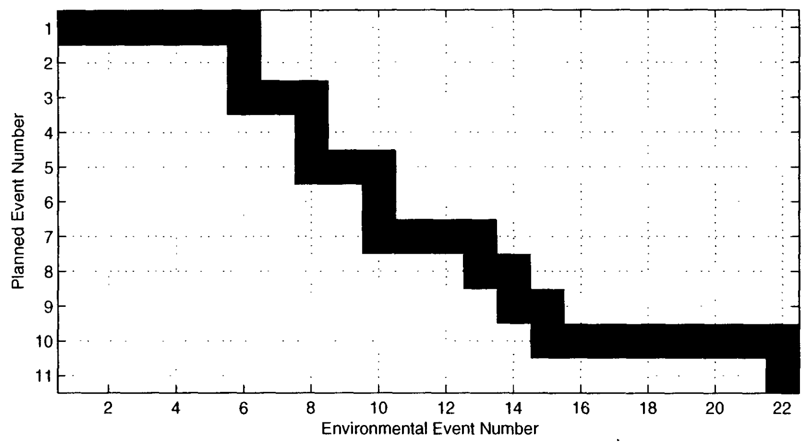


Figure 5-8: Example 2: Environmental conditions

Finally, applying the same algorithm of Section 5.2.1 under the cost function of Equation 5.2 gives the optimal path and schedule of Figure 5-9.

$$\min_{p\in\mathcal{P},\, r\in\mathcal{R}} \left[p_9 - 0.1\sum_{m=1}^{10} r_m^p - 0.1\sum_{n=1}^{22} r_n^e \right] \tag{5.2}$$

The optimal schedule indicates several significant trends. First, the row labeled *Rate* indicates that the observation mode between events five and six ranks as the greatest power consumer. Second, when presented with the choice of the three Band 1 communication conditions, the solver correctly selects two windows and admits the Band 2 condition between them. Furthermore, the algorithm selected the second Band 1 communication window versus the first one. The hypothesis here is that the solver favors placing power hungry events during periods of darkness so as to fully leverage the battery's regenerative capabilities during the times of sunlight. Appendix B offers greater depth concerning the details of this example.



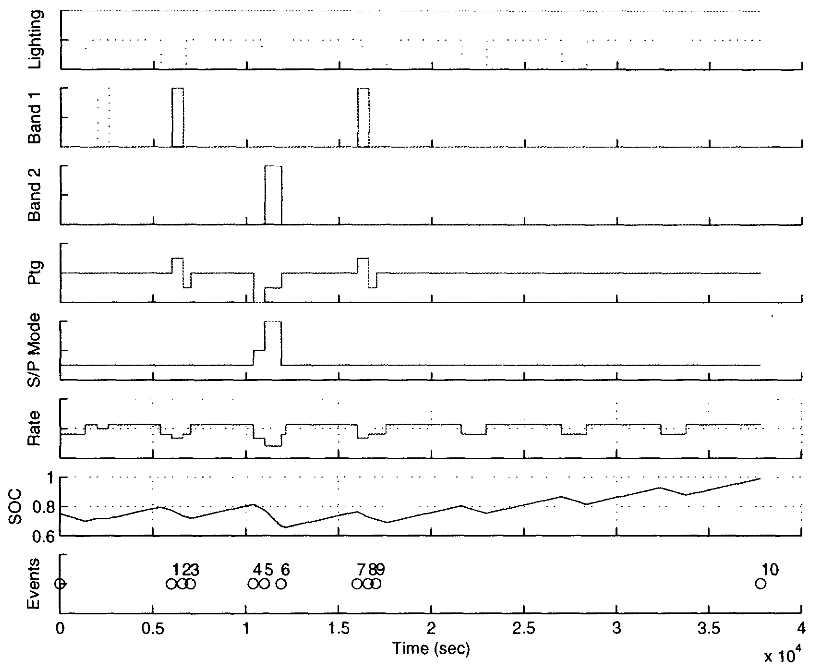


Figure 5-9: Example 2: Optimal path and schedule

5.3 Constraining the Main Mission Objective

The example of Section 5.2.2 provides a realistic view of how the algorithm might schedule a real mission. Unfortunately, this realistic mission does not shed light on the full potential of the algorithm. In particular, the communication windows tend to *force* the algorithm into a very small set of discrete decisions in the overlap space. The examples in this section serve as a baseline for subsequent experiments, omit communication windows in both the environmental conditions and as user specified requirements. Instead, the observation mode will fulfill one of the three possibilities: don't care, daylight, or darkness. All of the lighting requirements in the other modes will be unconstrained. The user will specify all of the plans to be of the format in Figure 5-1.

5.3.1 Unconstrained Observation Mode

The following example represents the wholly unconstrained case as depicted in Figure 5-10. There are some temporal exceptions. The first transition maneuver must last seventeen minutes, and the last one should take three and a half minutes. Lastly, the observation mode has between twenty and sixty-seven minutes to execute.

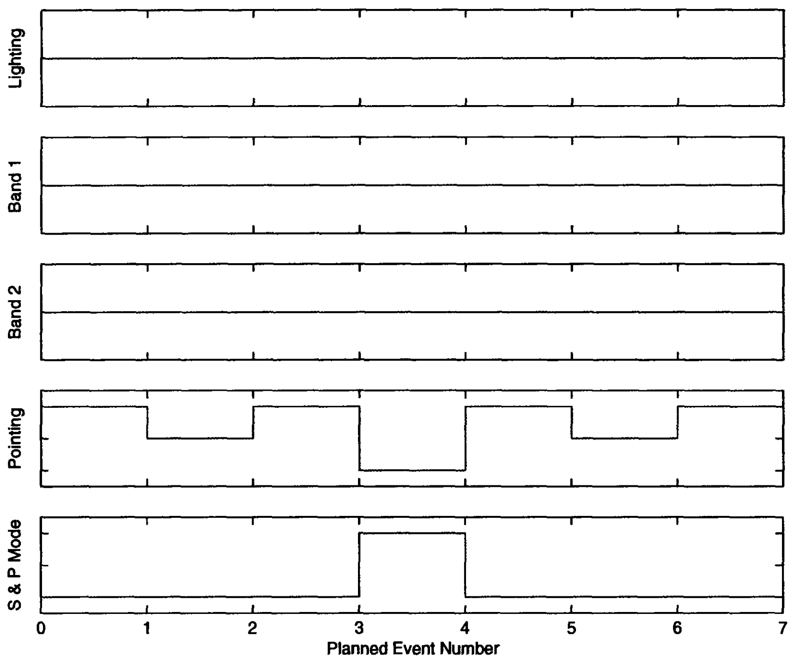


Figure 5-10: Example 3: User-specified planned modes

Figure 5-11 indicates the environmental conditions this mission will face. The remainder of this chapter will refer to these conditions as they will be applied throughout.

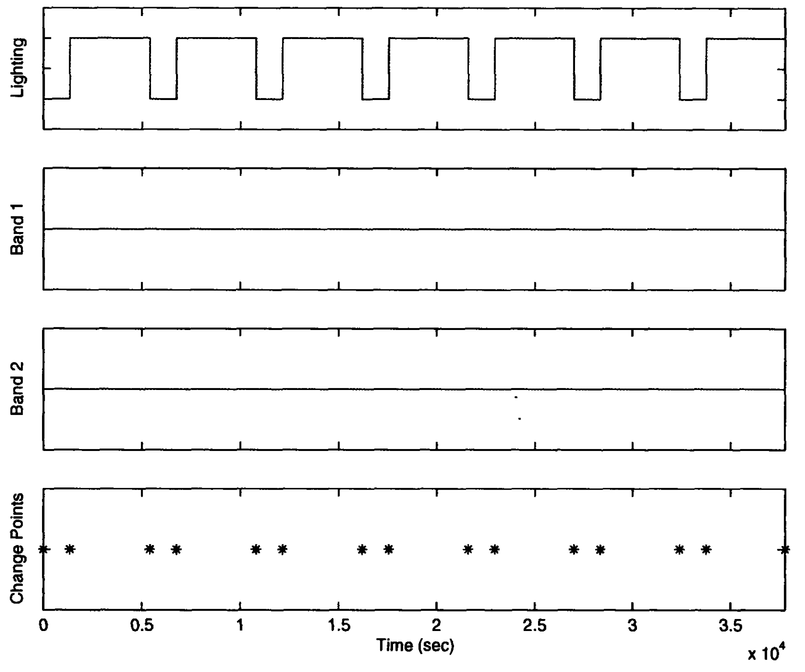
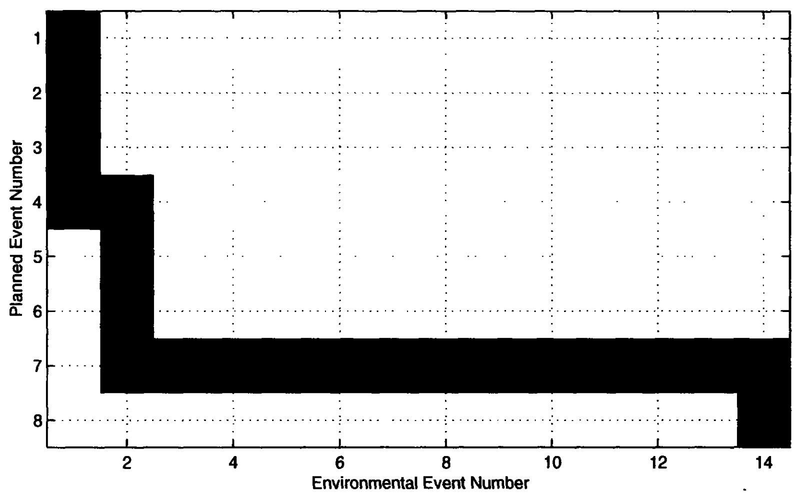


Figure 5-11: Example 3: Environmental conditions

Applying the algorithm as defined in Section 5.2.1 with the cost function of Equation 5.3 gives the solution path and schedule of Figure 5-12.

$$\min_{p \in \mathcal{P},\, r \in \mathcal{R}} \left[p_6 - 0.1 \sum_{m=1}^{7} r_m^p - 0.1 \sum_{n=1}^{14} r_n^e \right] \tag{5.3}$$

The solution schedule fulfills the expectation that it would schedule the activities as quickly as possible. The first station keeping mode has no duration. The transition maneuver follows with its finite duration. Next the observation mode is scheduled. The important point here is that just after the observation activity, the battery's state of charge approaches the minimum allowed value of 0.6. The solver intentionally allowed the resource to evolve in this manner because over the course of the mission, it can maximize the overall charge. This is an artifact of the cost function.



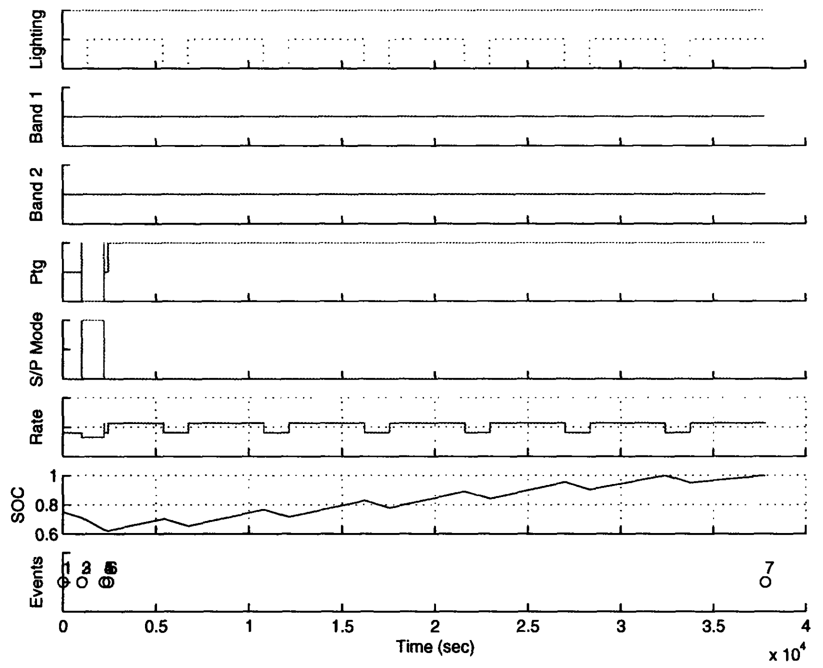


Figure 5-12: Example 3: Optimal path and schedule

5.3.2 Lighting Constrained Observation Mode

This mission is identical to the mission of Section 5.3.1 except that the observation mode is constrained to occur in daylight. Figure 5-13 indicates that difference. The environmental conditions in Figure 5-11 apply here. Again, applying the algorithm

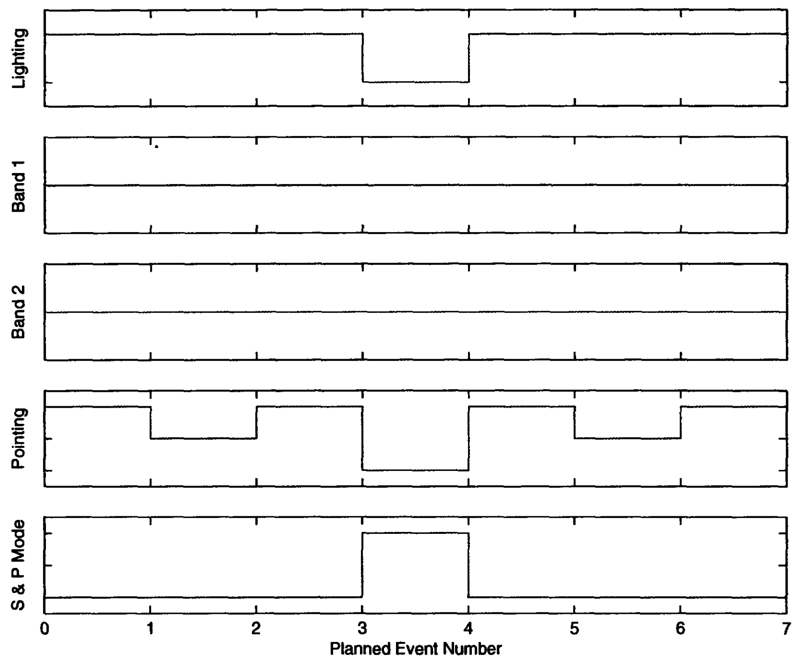
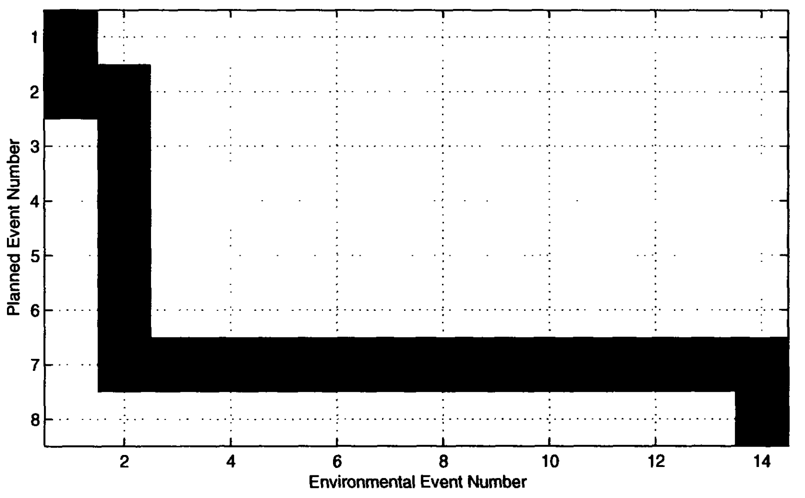


Figure 5-13: Example 4: User-specified planned modes

of Section 5.2.1 with the cost function of Equation 5.3 gives the solution path and schedule in Figure 5-14.

This solution is similar to the solution for the mission in Section 5.3.1. This is co-incidental because the finite duration of the transition maneuver forces this maneuver to end just before the satellite enters the next period of daylight. At this point, the scheduler places the observation mode (p_3 to p_4) without using much of its preceding station keeping mode (p_2 to p_3). As in the example of Section 5.3.1, the resource value drops to its minimum allowable value, and this occurs by design.



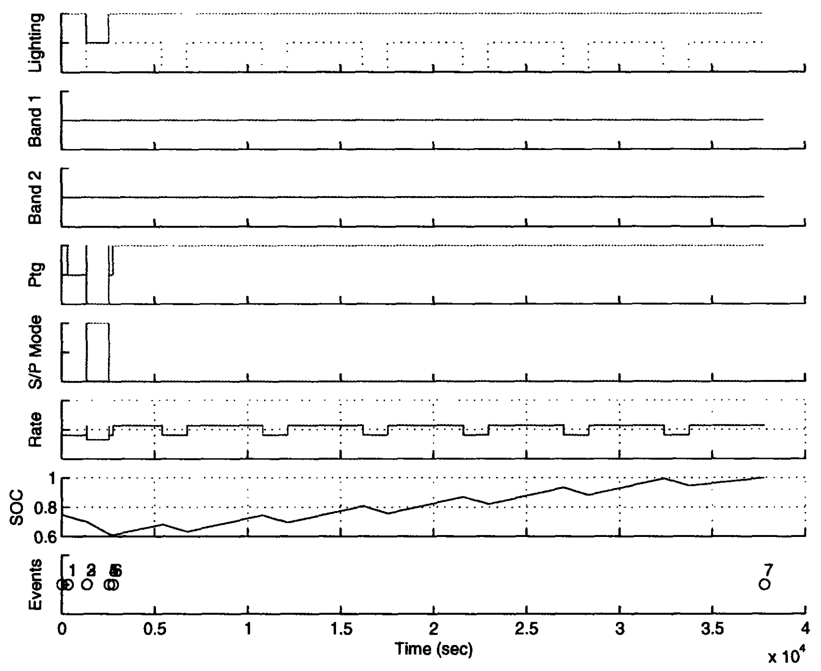


Figure 5-14: Example 4: Optimal path and schedule

5.3.3 Darkness Constrained Observation Mode

At the risk of being redundant, the following mission is identical to the previous one with the exception that the observation mode is constrained to execute in darkness. Figure 5-15 captures these planned modes. For the respective environmental conditions see Figure 5-11.

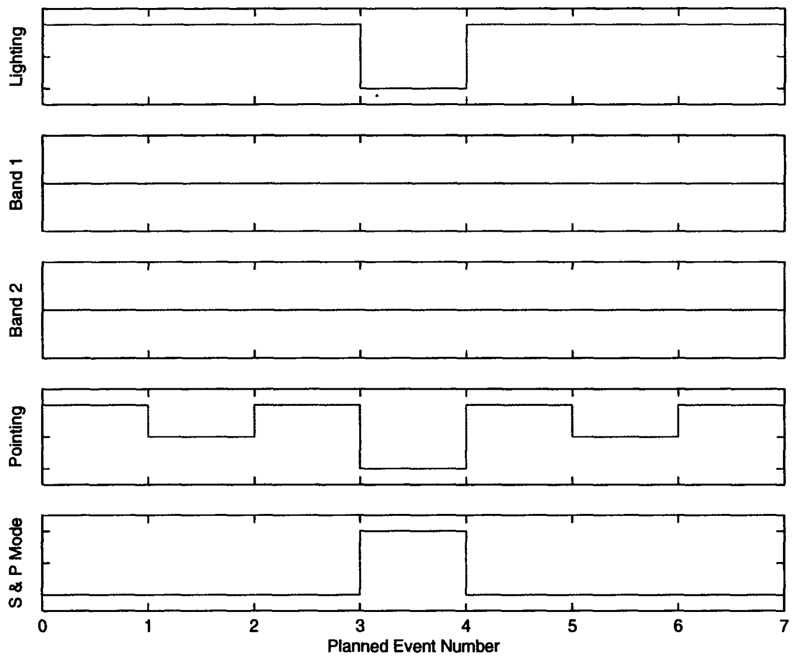
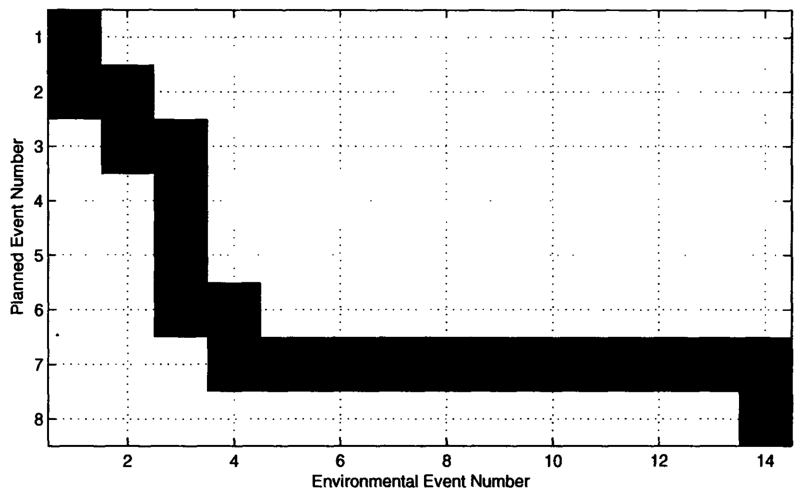


Figure 5-15: Example 5: User-specified planned modes

Finally, applying the algorithm of Section 5.2.1 with the cost function of Equation 5.3 gives the solution path and schedule in Figure 5-16.

Predictably, the algorithm chooses the second period of darkness. It does not have the option of selecting the first one because the transition maneuver's duration is long enough to exclude the observation mode from the first period of darkness. Because of the "delay" in scheduling the observation mode, the state of charge exhibits a gentler rise throughout the course of the mission.



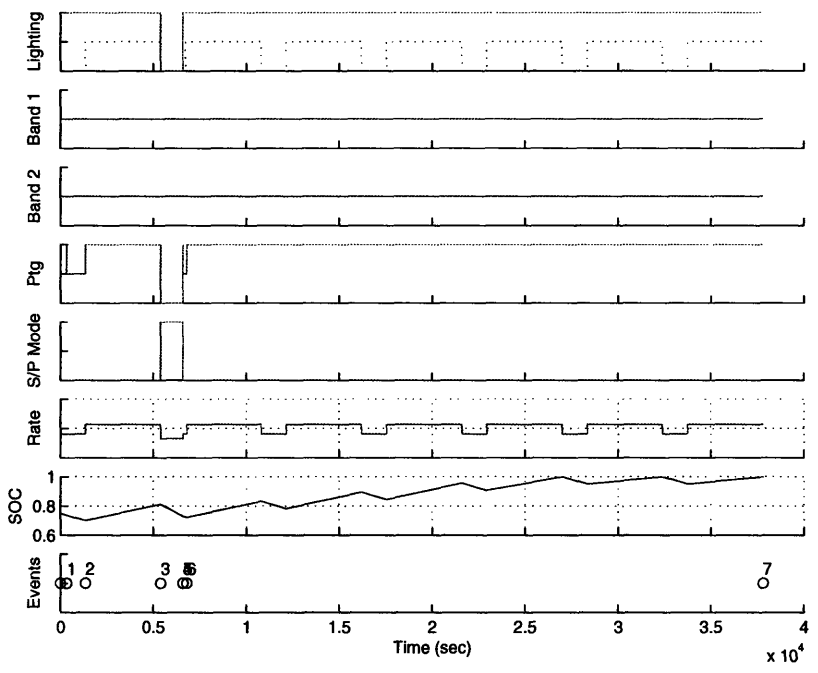


Figure 5-16: Example 5: Optimal Path and schedule

5.4 Changing the Minimum Allowable Resource Value

The examples of Section 5.3 appear to schedule the planned activities as expected. This section is an experiment to gauge the algorithm's performance for different minimum allowable resource values. Minimum state of charge (SOC) is another way to refer to this value. The following experiments in Sections 5.4.1–5.4.3 consider minimum allowable SOC values in the set, $\{0.1, \ldots, 0.9\}$. The values 0.0 and 1.0 are not included because in reality they are unreasonable.

The hypothesis is that as the level increases, the scheduler will postpone the power hungry modes until the satellite accumulates enough charge to complete the mission.

5.4.1 Unconstrained Observation Mode

The experiment here uses the same parameters as the unconstrained example of Section 5.3.1. This includes the planned modes of Figure 5-10 in addition to the environmental conditions of Figure 5-11. Figure 5-17 shows the observation event's end time, p_4, (see Figure 5-1) versus the discrete values in the set of minimum allowable state of charge variables. For the minimum SOC of 0.8 and 0.9, there was no solution, i. e. they are infeasible.

The observation mode end time and the minimum allowed SOC create a discrete space as in Figure 5-17. For example the observation event must last at least twenty minutes. With this knowledge, the scheduler can then evaluate based on the environmental interval, the SOC rate during that period. If the rate violates the minimum allowable value then the solver discards that case. Here a *case* refers to a partial path through the overlap space. Changing cases modifies the ordering of planned and environmental events within this overlap space.

For the given planned modes, environmental conditions, and minimum SOC 0.1 through 0.6, inclusive, the solver finds the same solution path and schedule in Figure 5-12. For minimum SOC, 0.7, the solution path and schedule are in Figure 5-18.

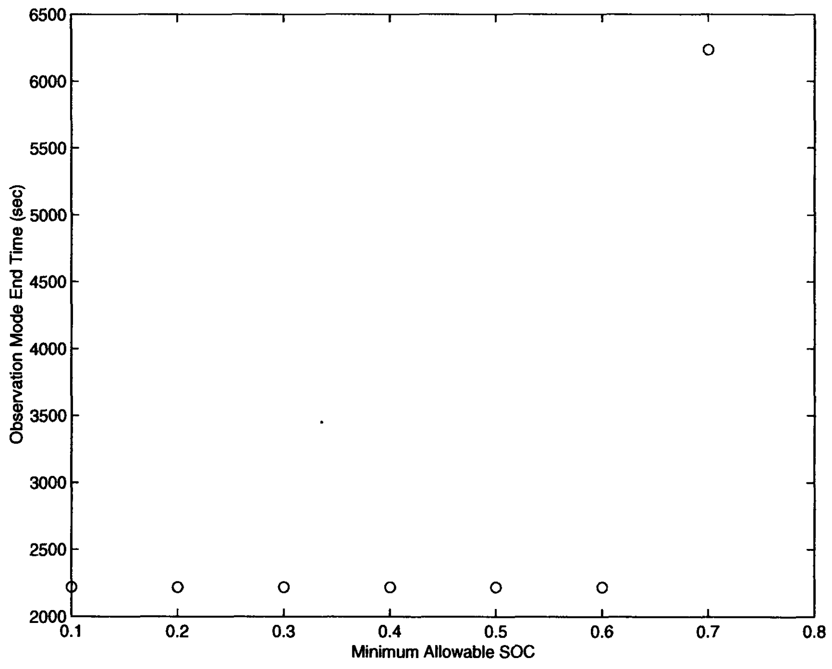
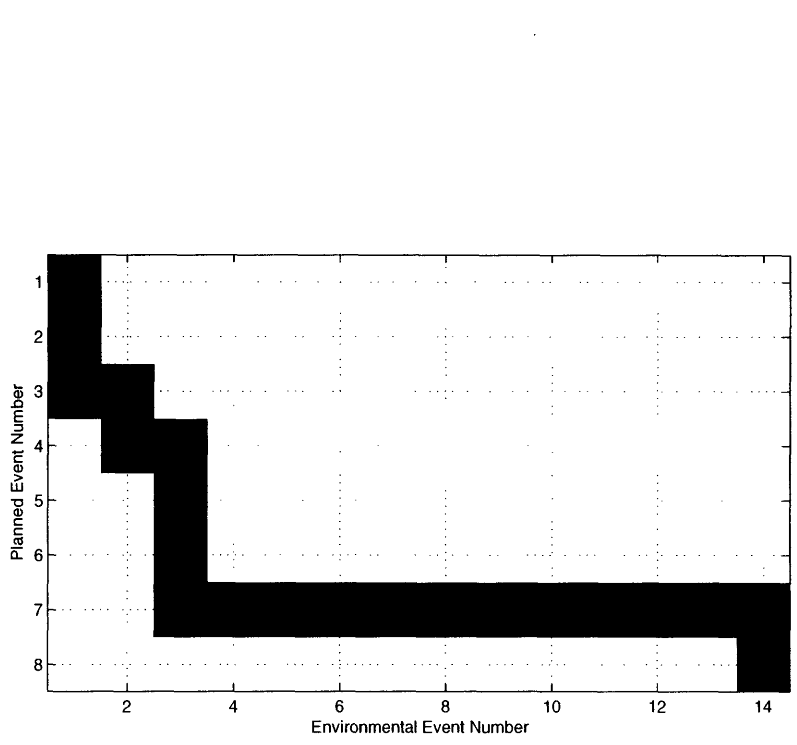


Figure 5-17: Experiment 1: Observation End Time vs. Minimum Allowable Value

The result for the experiment with a minimum SOC of 0.7 shows that the solver correctly delayed the observation event to a later time while allowing the battery more time to charge. As mentioned previously, the solver placed the event during darkness since it cannot charge its solar array driven battery during this time.



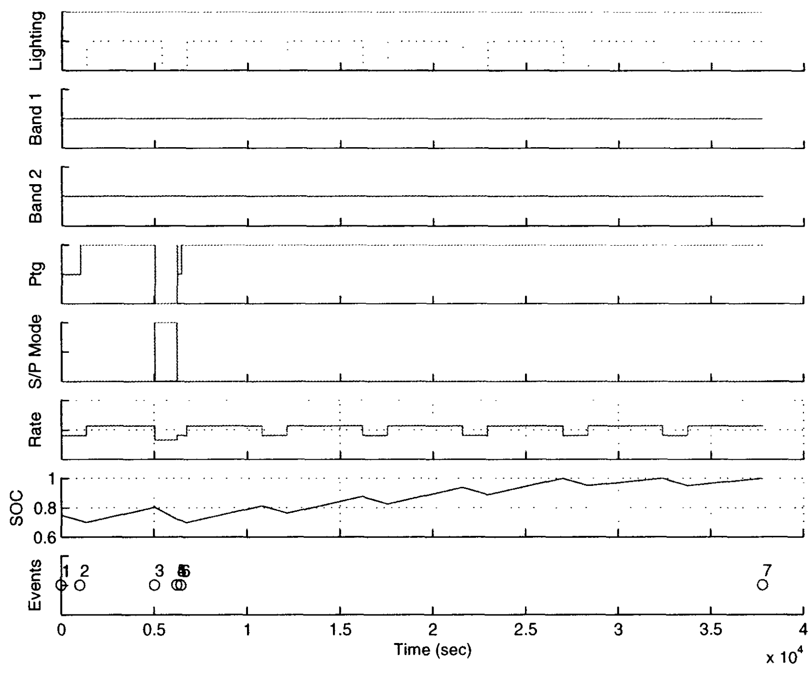


Figure 5-18: Experiment 1: Optimal Path and Schedule for *min SOC* = 0.7

5.4.2 Lighting Constrained Observation Mode

Performing the same experiment with the observation mode constrained to occur in daylight yields a similar result. The only difference between Figure 5-17 and 5-19 is that the observation end time is much later for the sun constrained case. This occurs because with a minimum SOC of 0.7, the satellite must wait through nearly two full periods of sunlight before enough charge accumulates to perform the observation activity. Contrast this with the unconstrained case which can perform the maneuver during the more advantageous nighttime period.

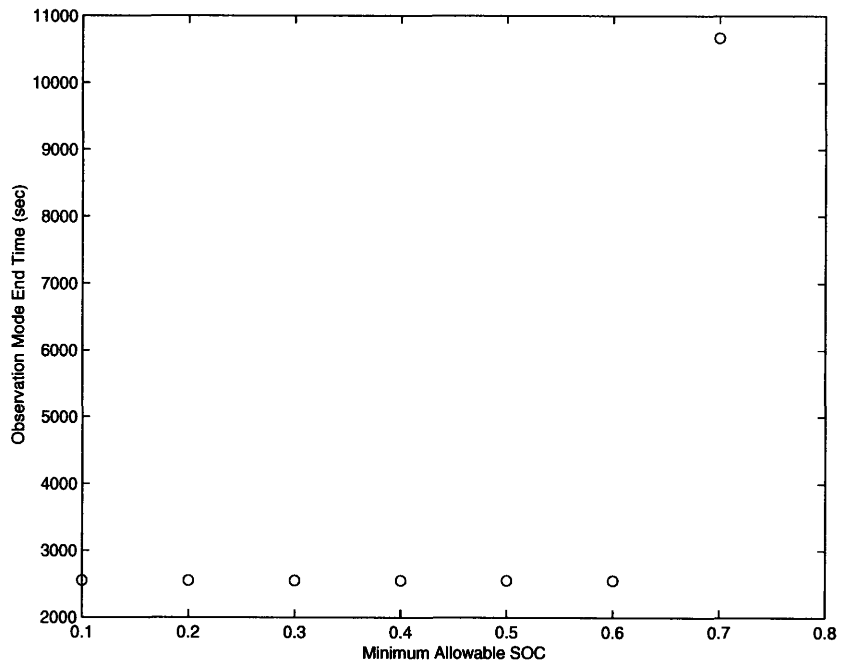
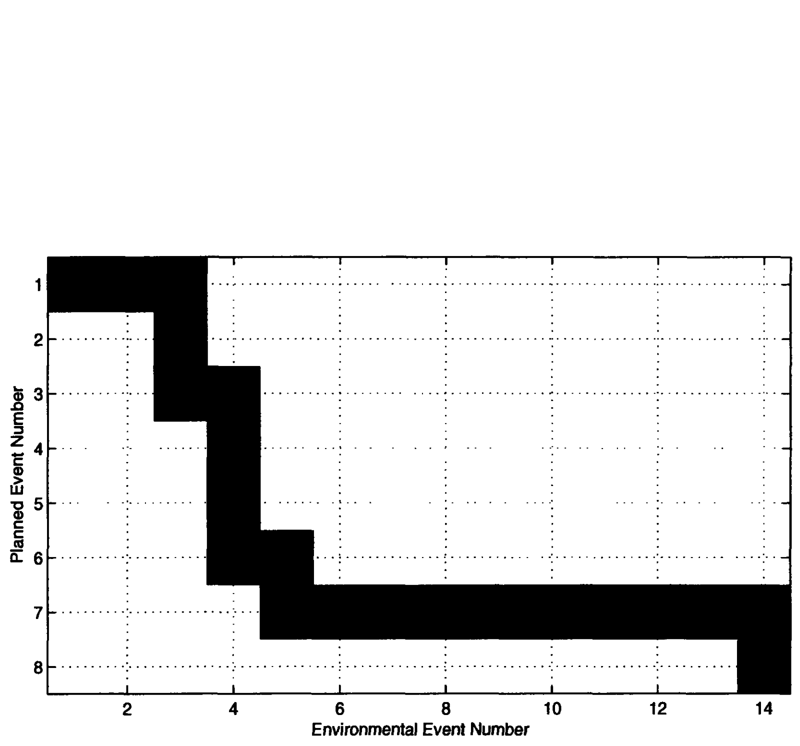


Figure 5-19: Experiment 2: Observation End Time vs. Minimum Allowable Value

Figure 5-20 gives the solution path and schedule for the 0.7 case where the solutions for the other minimum allowed SOC values (0.1–0.6) are the same as Figure 5-14.



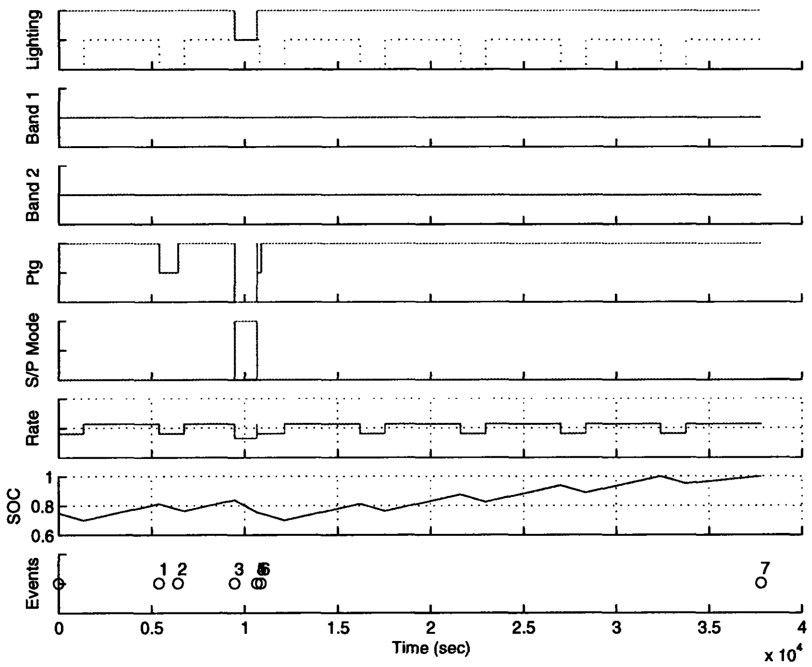


Figure 5-20: Experiment 2: Optimal Path and Schedule for *min SOC* = 0.7

5.4.3 Darkness Constrained Observation Mode

Constraining the observation mode to occur in darkness with the various values of minimum SOC produced no change. The results are the same for minimum values of SOC 0.1 through 0.7 with the remaining two, 0.8 and 0.9 being infeasible. To revisit the solution path and schedule, see Figure 5-16. This lack of difference among the solutions for the various minimum SOC values is a result of the advantage to placing the observation event in darkness; that is the chase satellite can replenish its battery.

5.5 Resource Weighting Schemes

Up to this point, the thesis maximizes the sum of every resource value at all of the planned and environmental events. This section focuses on two methods for improving the treatment of the resource values. Section 5.5.1 explores a modification to the formulation which gives extra control over the resource values; this change allows the user to maximize the lowest resource value. Section 5.5.2 incorporates this change, but it modifies the cost function so that the magnitude of the p_m does not dominate the cost. This modification allows the user to exert greater control over the resource values while at the same time controlling the solution time.

5.5.1 Naively Bounding the Resource Values

The formulation up to now maximizes the sum of all of the resources values. This section examines whether it is reasonable to maximize the lowest resource value. The addition of one new continuous decision variable, β, to the decision variables in Equation 4.45, can perform this function. It is necessary to add the constraints

$$\beta \leq r^e_n \quad \forall n \in [1, \ldots, N-1] \tag{5.4}$$

$$\beta \leq r^p_m \quad \forall m \in [1, \ldots, M] \tag{5.5}$$

to the constraints in Equation 4.46. Finally, the modified cost function builds on Equations 3.26 and 3.29.

$$\min_{p\in\mathcal{P},\, r\in\mathcal{R},\, \beta} \left[p_{M-1} - \beta - \lambda_p \sum_{m=1}^{M} r_m^p - \lambda_e \sum_{n=1}^{N-1} r_n^e \right] \tag{5.6}$$

Here, λ_p and λ_e represent small positive constants which provide the maximization effect for the resource values. Otherwise they run the risk of being neither maximized nor minimized. All of the examples to this point use values of 0.1.

The experiments include the unconstrained, lighting constrained and darkness constrained observation modes evaluated for a range of minimum allowed SOC. The experiments of this section are the same as those of Section 5.4, and they yield the same results. This fulfills the expectations because the new decision variable, β, bounds the lowest resource value. As the minimum allowed resource value increases to approach β, the solver must respond by moving events to accommodate the stricter demand on the resource.

5.5.2 Modifying the Cost Function

All of the previous cost functions in this thesis are a naive attempt to minimize the plan's execution time and maximize the resource value. This section presents a new cost function, which gives the user more control over the lowest resource value and the plan execution time. The additional benefit of the new cost function allows the user to define the emphasis between the execution time and minimum resource value. The user has the opportunity to further evaluate the relationship between these parameters defined as λ_β and $\lambda_{p_{M-1}}$. These are the controls for the minimum resource bound and plan execution time, respectively.

The new cost function is

$$\min_{p\in\mathcal{P},\, r\in\mathcal{R},\, \beta} \left[\frac{\lambda_{p_{M-1}}}{e_N - e_0} p_{M-1} - \lambda_\beta \beta - \lambda_p \sum_{m=1}^{M} r_m^p - \lambda_e \sum_{n=1}^{N-1} r_n^e \right] . \tag{5.7}$$

Note the fractional coefficient of the p_{M-1} term. This normalizes the planned event time. In the examples of this thesis this value is on the order of tens of thousands of seconds, and it tends to dominate the smaller resource values, r_n^e and r_m^p, and beta value, β. The following experiments capture the same planned modes and environmental conditions. Instead, these consider changing the value, λ_β, and evaluating its impact on the observation maneuver's end time. For these experiments, $\lambda_e = \lambda_p = 0.0001$. Equation 5.8 captures the

$$\lambda_{p_{M-1}} = 1 - \lambda_\beta \quad \forall \lambda_\beta \in \{0.0, 0.1, \ldots, 0.9, 1.0\} \tag{5.8}$$

relationship between λ_β and $\lambda_{p_{M-1}}$. Each of the following three sections presents a figure of the observation activity's end time versus the value of λ_β.

Unconstrained Observation Mode

This experiment uses the planned modes and environmental conditions of Figures 5-10 and 5-11. As β's input cost, λ_β increases, several discrete jumps occur. The first occurs between 0.5 and 0.6, and the second between 0.9 and 1.0. The understanding here is that as the emphasis on the minimum resource bound, β, increases, the scheduler increases the amount of time before it executes the observation maneuver. Rather than belabor the point with the figures of optimal paths and schedules, the reader may refer to these in Appendix A. In Figure 5-21 the observation end time and the whole schedule for the first range of values from 0.0 to 0.5 is the same as the solution in Figure 5-12. The paths and schedules for the other two cases, 0.6 to 0.9 and 1.0 appear in Figure A-1 and A-2 of Appendix A.

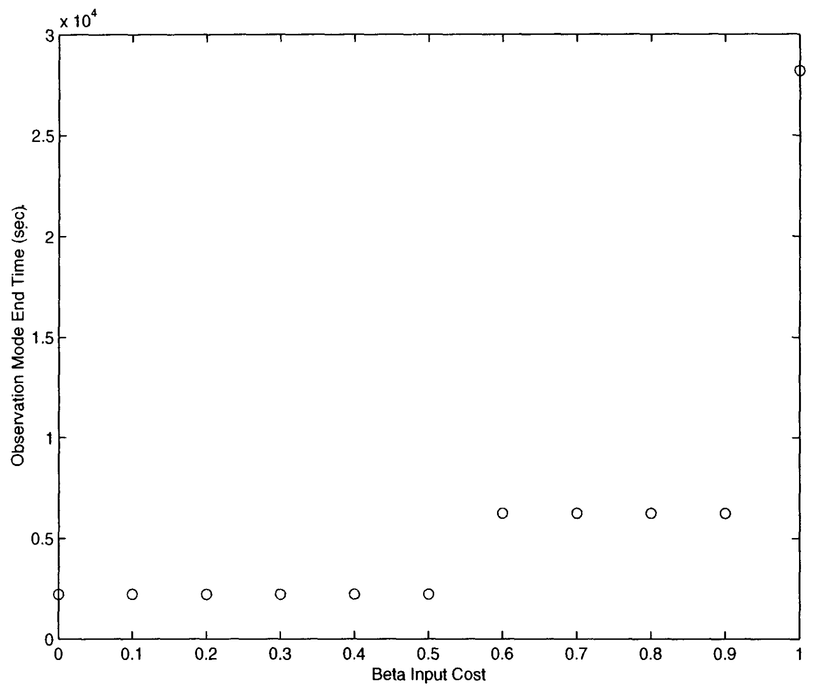


Figure 5-21: Experiment 4: Observation End Time vs. λ_β

Lighting Constrained Observation Mode

The following experiment increments the values of λ_β through the previously defined values. Recall that as $\lambda_{p_{M-1}}$ decreases, λ_β increases and vice versa. This helps explore the relationship between the two values. In Figure 5-22 there are four distinct solution schedules. The implication is that constraining the observation maneuver to occur in daylight presents the algorithm with a range of options. As emphasis on λ_β increases, the algorithm delays the activity to allow for more sunlight exposure to charge the satellite's battery. As in the unconstrained case, the first values of λ_β yield the same solutions as in the example of Section 5.3.2 and the experiment in Section 5.5.1.

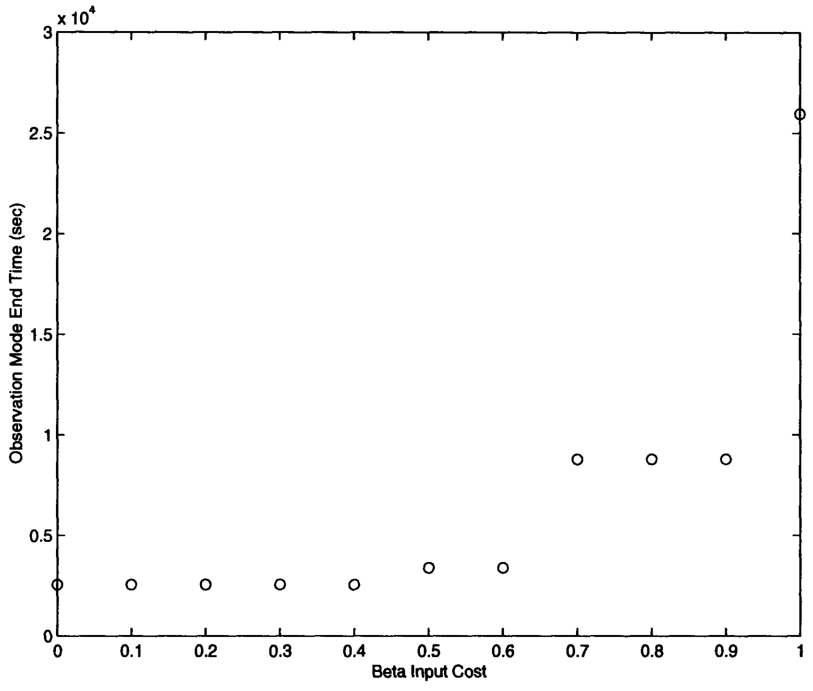


Figure 5-22: Experiment 5: Observation End Time vs. λ_β

Darkness Constrained Observation Mode

The final experiment of this section constrains the observation mode to occur during darkness. Figure 5-23 shows how the change in λ_β and $\lambda_{p_{M-1}}$ affect the solution time. Over the range, the solver favors only one solution except for in the last value, $\lambda_\beta = 1.0$. One explanation for this lies in the fact that placing the observation activity during nighttime is the optimal decision to make. Furthermore, the solution for the λ_β values in the majority of the cases is the same as the solution of Figure 5-16 in the example of Section 5.3.3, and the same as the experiment in Section 5.5.1. For the solution where $\lambda_\beta = 1.0$ see Figure A-6.

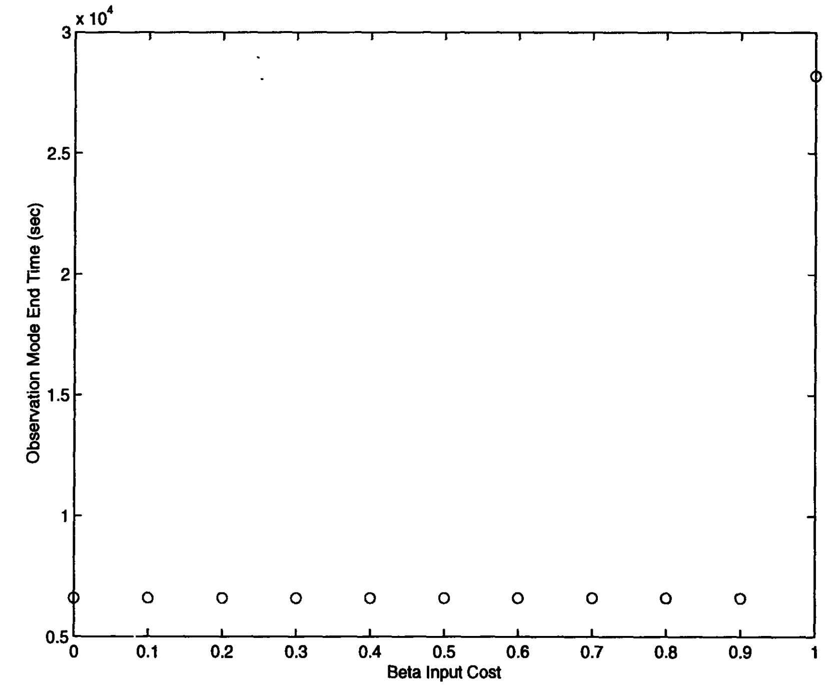


Figure 5-23: Experiment 6: Observation End Time vs. λ_β

5.6 Weighting Planned Events

Certain modes use additional time dependent resources which will remain unmodeled. In this section, the unmodeled resource is fuel. The assumption is that the chase craft consumes fuel proportional to the amount of time it spends within a specific fuel consuming mode. Thus, the goal is to minimize this resource consumption by minimizing the mode's execution time, and this is achieved by penalizing the execution time of the fuel consuming modes.

In the current missions, the station keeping modes just before and just after the observation maneuver take place along the radial vector, $\bar{r}$, of the target satellite. This consumes fuel directly proportional to the length of the mode. The goal of this section will be to show that the algorithm can reduce the amount of execution time for these modes with a modification to the cost function.

Consider the following terms for addition to the cost function of Equation 5.7. These terms add a penalty to the duration of the station keeping modes in question. Controlling the values $\lambda_{SK,1}$ and $\lambda_{SK,2}$ lets the user assign the desired importance to each of the modes.

$$\frac{\lambda_{SK,1}}{e_N - e_0}(p_3 - p_2) + \frac{\lambda_{SK,2}}{e_N - e_0}(p_5 - p_4). \tag{5.9}$$

This thesis sets

$$\lambda_{SK,1} = \lambda_{SK,2} = \lambda_{SK}, \tag{5.10}$$

and for each trial assigns a values from the set $\{0.0, 0.1, \ldots, 0.9, 1.0\}$ to λ_{SK}.

The experiments are the same ones as in the previous sections; they examine the unconstrained case, and the daytime and nighttime constrained observation maneuvers. The other parameters of the cost function are $\lambda_\beta = 0.9$ and $\lambda_{p_{M-1}} = 0.1$.

5.6.1 Unconstrained Observation Mode

Of the three cases, the algorithm has the most freedom to move the observation maneuver in the unconstrained case. Along with this freedom comes the responsibility to properly adjust the durations of the adjacent station keeping modes. The hypothesis is that as the penalty on the durations increases, the net duration between the two modes will decrease. Figure 5-24 shows each mode's duration independently. For any of the eleven trials except the first, the net duration is less than all of the previous values. In the end, both of the modes have zero execution time.

One interesting feature of this experiment occurs when λ_{SK} increases from 0.1 to 0.2. The algorithm is able to improve the overall cost by increasing the duration of the second station keeping mode while significantly decreasing the duration of the first station keeping mode.

Solution paths and schedule for each of the cases appear in Appendix A.

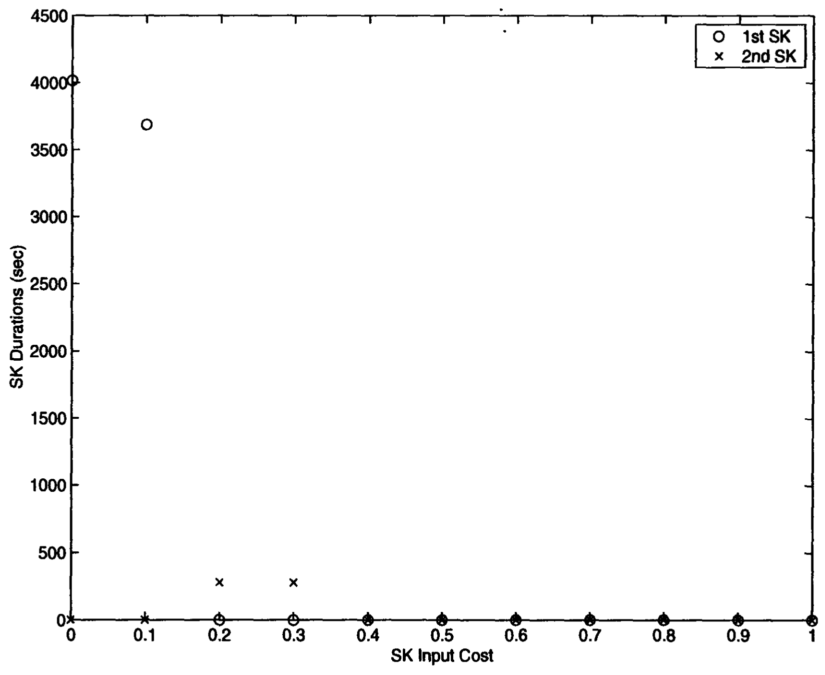


Figure 5-24: Experiment 7: SK Duration vs. SK Cost

5.6.2 Lighting Constrained Observation Mode

With the environmental conditions of these sections, constraining the observation maneuver to occur during sunlight gives the algorithm an intermediate number of options compared to the unconstrained case and the darkness constrained case. Unlike the unconstrained case, the lighting constrained case executes each trial with zero duration in the second station keeping mode. The first station keeping mode decreases as the input cost increases. Figure 5-25 presents these results. The solutions are in Appendix A.

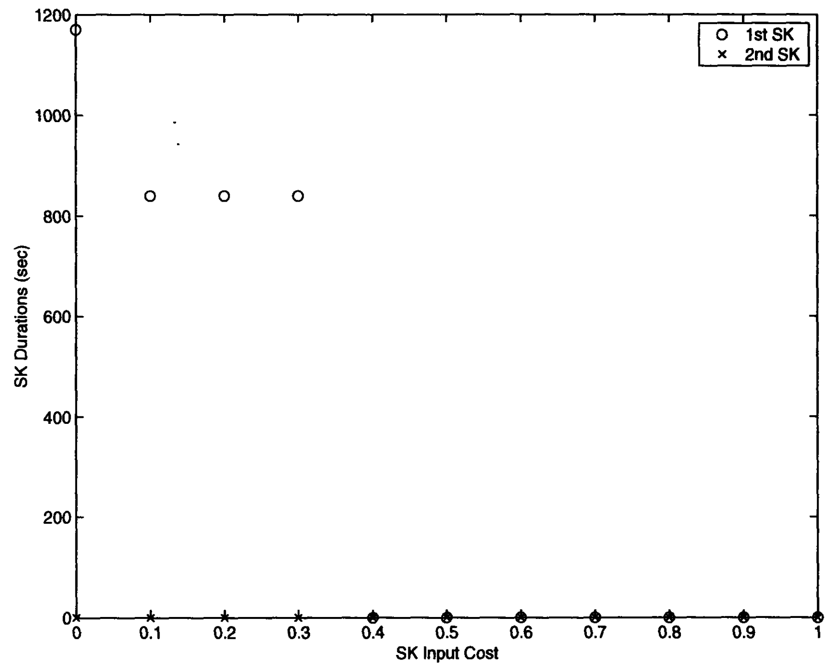


Figure 5-25: Experiment 8: SK Duration vs. SK Cost

5.6.3 Darkness Constrained Observation Mode

Constraining the observation maneuver to occur in darkness gives the solver fewer possible options. In reality there are three distinct solutions in this experiment and in the sunlight constrained experiment. As with the sunlight case, the second station keeping mode has zero duration in all eleven trials and the first one decreases as its input cost increases. Figure 5-26 gives the results. Appendix A shows the unique solution paths and schedules.

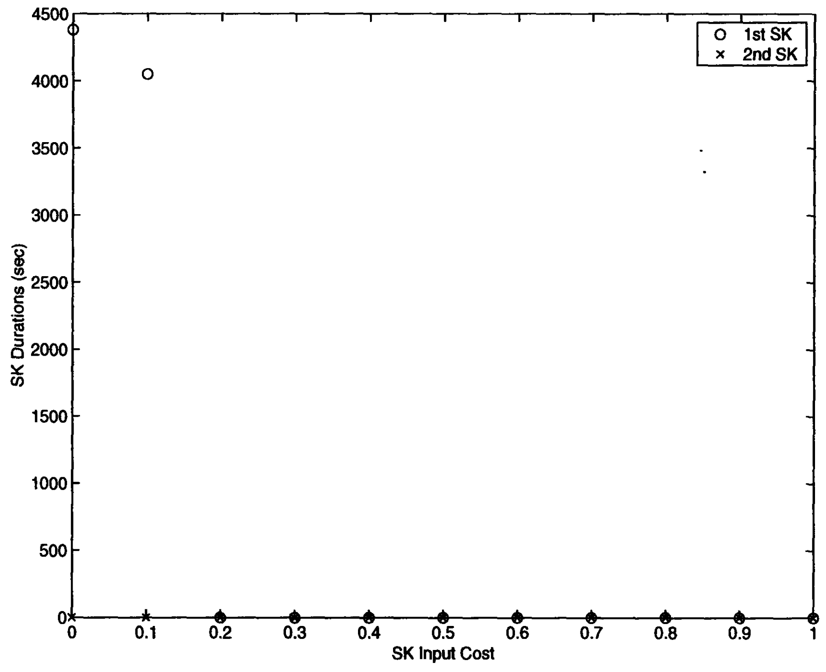


Figure 5-26: Experiment 9: SK Duration vs. SK Cost

Chapter 6

Conclusions

This thesis develops a mixed integer programming formulation to treat the orbital rendezvous proximity operations scheduling problem. The user specifies planned modes, which describe the chase satellite's state during the mission. This state considers the desired lighting constraints, communication operations, pointing mode, and sensor and payload mode. This list is not exhaustive, but in conjunction with the environmental conditions, the planned modes determine the resource rate. The environmental conditions describe the world surrounding the chase satellite; these conditions are predetermined based on the orbit, and on the availability of the communication channels.

The algorithm discretely selects the optimal relative ordering of the planned modes and environmental intervals, and it simultaneously computes the optimal, continuous values of the planned times. A user specified cost function governs the optimality of the solution.

6.1 Summary

The analysis in this thesis considers a general observation mission that focuses on an observation mode. This observation mission begins and ends with the same set of modes—a pair of station keeping modes joined by a transition maneuver (see Figure 5-1). The thesis analyzes this mission as the user specifies the observation mode to occur without constraint, in daylight, and in darkness.

The analysis of this algorithm comes from two perspectives. The first examines the solver's response to changes in the minimum allowed resource value. An associated experiment modifies the MIP formulation with an additional decision variable that bounds the minimum resource value. The other experiments posit more utilitarian cost functions, which specify the user's optimality preferences for a particular mission.

6.2 Capabilities

The experiments suggest that the algorithm adeptly solves practical cases. If an overlapping planned mode and environmental condition consume more resources throughout their duration than are available, it will naturally be an infeasible mission. The analysis here suggests that the solver recognizes the cases where overlapping planned modes and environmental conditions are infeasible. It examines all other cases to find a solution before declaring infeasibility. Similarly, the scheduling algorithm is complete as long as the underlying MIP solver is complete.

The MIP approach provides a certain amount of flexibility and robustness. The experimental results (Chapter 5) offer a modification to the MIP formulation, which requires very few changes to fully implement. Similarly, the MIP design admits other modifications both large and small. It allows for more resources, and it considers other modes and conditions with ease. This solution method gives the user the option to define the criteria that determines whether a planned mode and environmental condition can overlap. This determination can be made under a nearly limitless degree of considerations.

The cost function captures the user's concept of how to treat the resources and execution time. The final cost of a schedule provides a valuable tool for scoring sets of schedules where the lower cost schedule is more desirable. This scoring feature lends itself to integration within a hierarchical planner, which would poll this algorithm. Based on the scores of the resultant schedules, it would generate the optimal plan.

Finally, this algorithm can be used as a verification tool. There is still a fair amount of reluctance among the customers of the aerospace community to allow a spacecraft the autonomy that this thesis suggests. To allay these concerns, this solution method would be a powerful tool to evaluate the quality of schedules generated by other techniques. Alternative algorithms for this scheduling problem might bypass the built-in search capabilities of the commercial MIP solvers in favor of proprietary methods. This algorithm can verify solutions generated with the proprietary methods.

6.3 Limitations

This solution approach leaves room for refinement, especially to increase the fidelity of the formulation. The algorithm by design treats deterministic problems. This formulation makes it more difficult to consider cases where the environmental conditions may change unpredictably. Consider the difficulties associated with a communication window that is subject to an uncertain delay. This would require the algorithm to use a set of heuristics or techniques of probability to formulate the optimal solution schedule.

Associated with this problem is the solver's ability to issue a replan. Using the methods developed here, if only one or two requirements change, it is necessary to recompute the complete solution. There are more effective ways to use the complete output of the mixed integer program to simplify replanning.

This thesis neglects other problems such as dependent events. For example, the transition maneuvers ignore simple requirements such as pressurizing a fuel line before a thruster fires. This type of activity needs to occur outside of the bounds of the transition maneuver yet it is a direct result of the transition.

Finally, the approach of this thesis treats the planned modes and environmental conditions as discrete events. While this may suffice for communication windows, modeling sunlight as a discrete value might adversely affect some computations. Furthermore pointing modes in reality are continuous values as they represent attitude. Modeling attitude as a discrete value means the scheduler sacrifices some fidelity. This might be a problem if the user wanted to optimally choose the satellite's attitude for a specific requirement.

6.4 Future Work

Many of the algorithm's limitations suggest the future areas of research. The author understands that no space faring craft has ever flown software that performs mixed integer programming in support of an autonomous guidance, navigation, and control package. This idea makes customers cringe because of the large size of the non-flight validated mixed interger programming software. One way to ease them into acceptance is to remove the aspects of the problem that require a MIP solver. This requires solving the joint interval problem with a different technique. A greedy search of the overlap space applying a pure linear program could yield the same solutions. This approach should be more acceptable because it will require less work to flight validate linear programming software and the accompanying search algorithm as opposed to commercial mixed integer programming software.

6.4.1 Greedy Search

Looking at the joint interval problem without the robustness of the MIP requires careful thought. A major question is how to incorporate the cost function into the search. Clearly, the goals are to minimize solution time and maximize the resource values. Unfortunately, the cost function for a complete mission does not mean the same thing for the abbreviated mission represented in a partial path of the overlap space. Integrating a cost function for a linear program over the partial space of overlaps, as the greedy search progressed, would be a challenge. Furthermore, consider the difficult situation in which a planned event might be torn between the temporal desire to execute quickly, versus the resource's desire to endure longer to recharge the battery. This push and pull might occur across a discrete change in the environmental conditions which would complicate the situation.

6.4.2 Generating the Cost Function

Currently, the cost function is selected on a per mission basis. A fully autonomous system needs a cost function that will leave the satellite in a usable state at the end of the mission, but also one that will achieve its goals. Defining optimality for a mission can be complicated because it depends on several features including the types of modes involved, the user-defined order of the modes, etc. One of the experiments (Section 5.6) considers penalties for some of the costlier station keeping modes. There may be a case where the satellite would gain a benefit from extending a mode's execution time. In this thesis, the observation mode of the experiments is constrained to execute for a specific duration. Instead, providing some kind of benefit related to the observational value of the mode might be a more powerful way to design the cost function. This is merely one example of the myriad considerations that might be significant in generating the cost function.

[Except for this sentence, this page intentionally left blank.]

References

[1] Mark Abramson, David Carter, Stephan Kolitz, et al. The Design and Implementation of Draper's Earth Phenomena Observing System (EPOS). Technical report, American Institute of Aeronautics and Astronautics, 2001.

[2] Dash Associates. *XPRESS-MP Reference Manual.* Dash Associates Limited, 1999.

[3] Dash Associates. *XPRESS-MP User Guide.* Dash Associates Limited, 1999.

[4] Dimitris Bertsimas and John N. Tsitsiklis. *Introduction to Linear Optimization.* Athena Scientific, 1997.

[5] S. Chien, G. Rabideau, R. Knight, R. Sherwood, B. Engelhardt, D. Mutz, T. Estlin, B. Smith, F. Fisher, T. Barrett, G. Stebbins, and D. Tran. ASPEN – Automated Planning and Scheduling for Space Mission Operations. SpaceOps, June 2000.

[6] Steve Chien, Russell Knight, Andre Stechert, Rob Sherwood, and Gregg Rabideau. Using Iterative Repair to Increase the Responsiveness of Planning and Scheduling for Autonomous Spacecraft. In *IJCAI99 Workshop on Scheduling and Planning Meet Real-time Monitoring in a Dynamic and Uncertain World,* August 1999. http://jmvidal.cse.sc.edu/masreadinggroup.html.

[7] Rina Dechter, Itay Meiri, and Judea Pearl. Temporal constraint networks. In *Artificial Intelligence.* Elsevier, 1991.

[8] Jennifer Dungan, Jeremy Frank, Ari Jónsson, Robert Morris, and David E. Smith. Advances in Planning and Scheduling of Remote Sensing Instruments for Fleets of Earth Orbiting Satellites. Technical report, NASA Ames Research Center, Moffett Field, CA 94035, 2002.

[9] Erann Gat and Barney Pell. Abstract Resource Management in an Unconstrained Plan Execution. In *Proceedings of the IEEE Aerospace Conference*, Snomass, CO, 1998. IEEE.

[10] Masaaki Mokuno, Isao Kawano, and Toru Kasai. Experimental Results of Autonomous Rendezvous Docking on Japanese ETS-VII Satellite. In *22nd Annual AAS Guidance and Control Conference*, volume 99. AAS, 1999.

[11] Nicola Muscettola, Paul Morris, Barney Pell, and Ben Smith. Issues in Temporal Reasoning for Autonomous Control Systems. In *Proceedings 2nd International Conference on Autonomous Agents*, 1998.

[12] Leslie Pack-Kaelbling. Techniques of Artificial Intelligence. Lectures, Electronic materials, 2002. MIT Course 6.825.

[13] Barney Pell, Gregory A. Dorais, Christian Plaunt, and Richard Washington. The Remote Agent Executive: Capabilities to Support Integrated Robotic Agents. In *Working Notes of the AAAI Spring Symposium on Integrated Robotic Architectures*, 1998.

[14] Barney Pell, Edward B. Gamble, Erann Gat, et al. A Hybrid Procedural/Deductive Executive for Autonomous Spacecraft. In *Proceedings 2nd International Conference on Autonomous Agents*, 1998.

[15] Michael E. Polites. Technology of Automated Rendezvous and Capture in Space. *Journal of Spacecraft and Rockets*, 36(2):280–291, March–April 1998.

[16] Gregg Rabideau, Barbara Engelhardt, and Steve Chien. Using Generic Preferences to Incrementally Improve Plan Quality. In *Fifth International Conference on Artificial Intelligence Planning and Scheduling*, Breckenridge, CO, April 2000.

[17] A. Richards, T. Schouwenaars, J. P. How, and E. Feron. Spacecraft Trajectory Planning with Avoidance Constraints Using Mixed-Integer Linear Programming. *Journal of Guidance, Control, and Dynamics*, 25(4):755–764, July–August 2002.

[18] Stuart Russell and Peter Norvig. *Artificial Intelligence a Modern Approach.* Prentice-Hall, Inc., Upper Saddle River, New Jersey 07458, 1995.

[19] Birgit M. Sauer. Autonomous Mission Scheduling for Satellite Operations. Master's thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, May 1997.

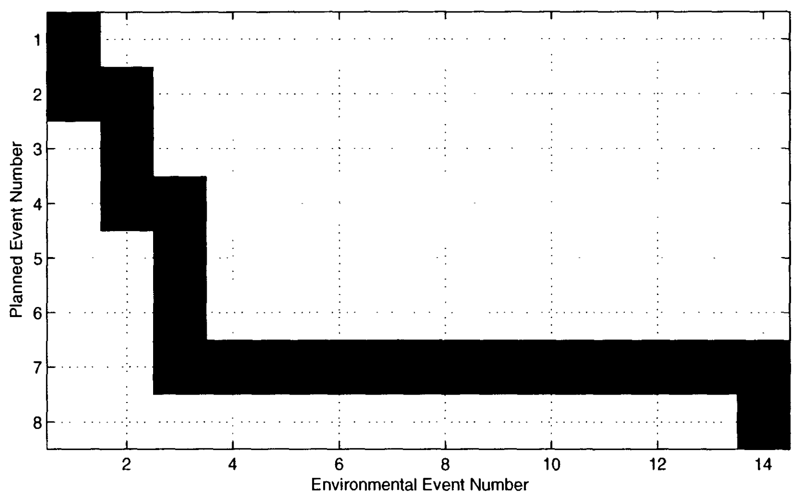
[20] Monte Zweben, Eugene Davis, Brian Daun, and Michael J. Deale. Scheduling and Rescheduling with Iterative Repair. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(6):1588–1596, November/December 1993.

[Except for this sentence, this page intentionally left blank.]

Appendix A

Additional Figures

This appendix contains additional figures of the results from the experiments described in Chapter 5.



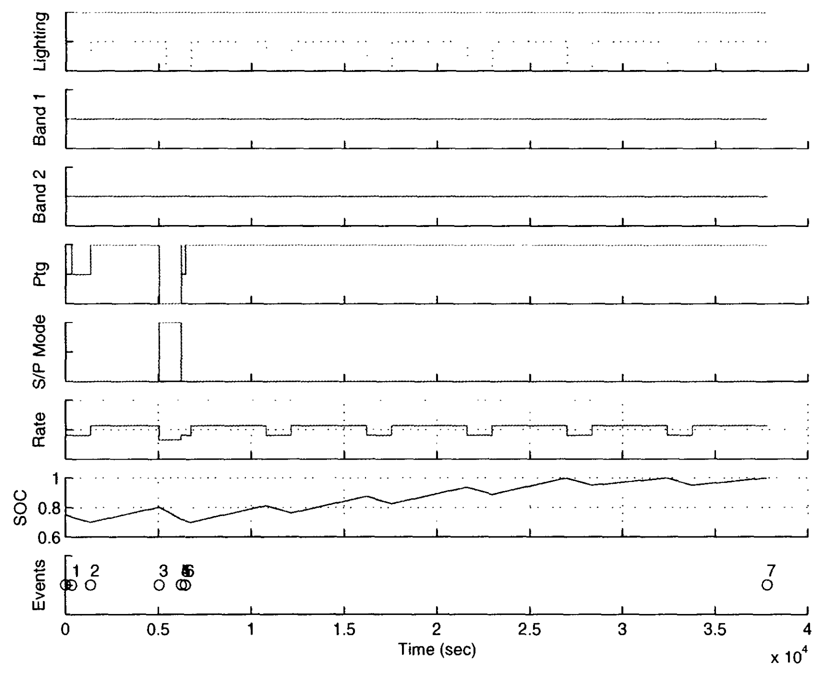
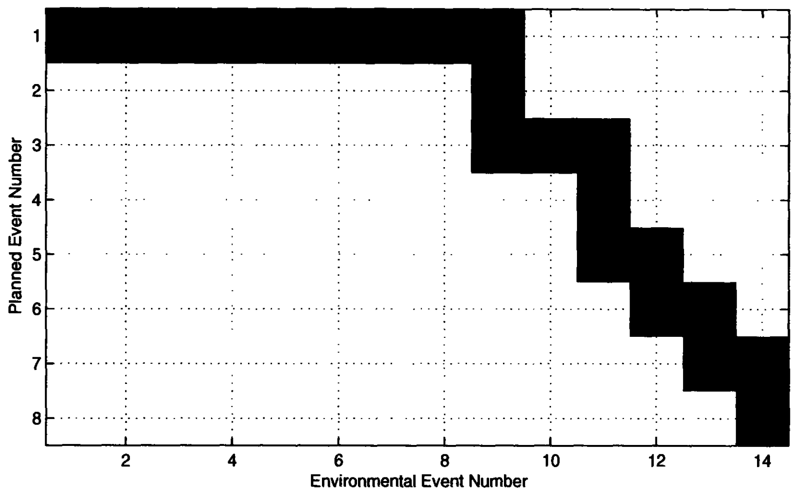


Figure A-1: Experiment 4: Optimal Path and Schedule for $\lambda_\beta = 0.6, \ldots 0.9$



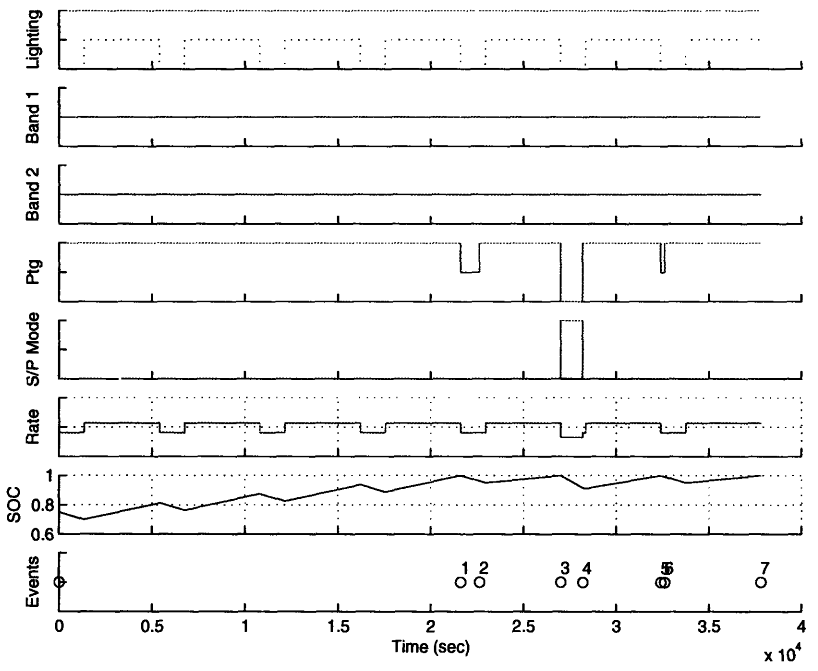
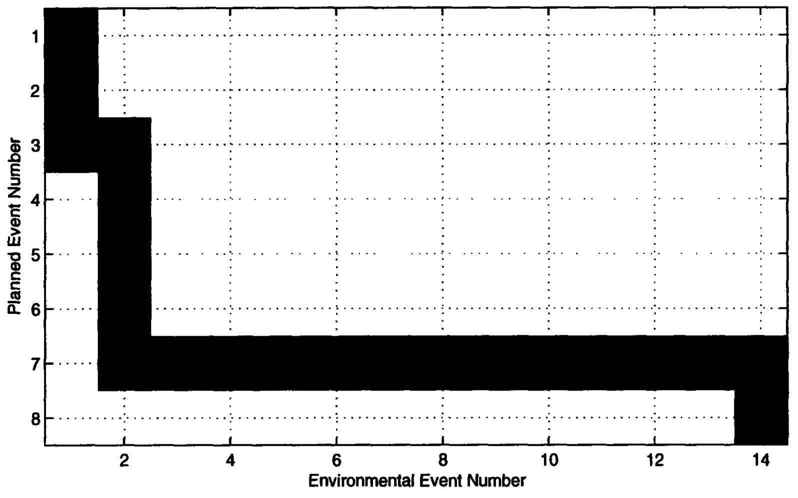


Figure A-2: Experiment 4: Optimal Path and Schedule for $\lambda_\beta = 1.0$



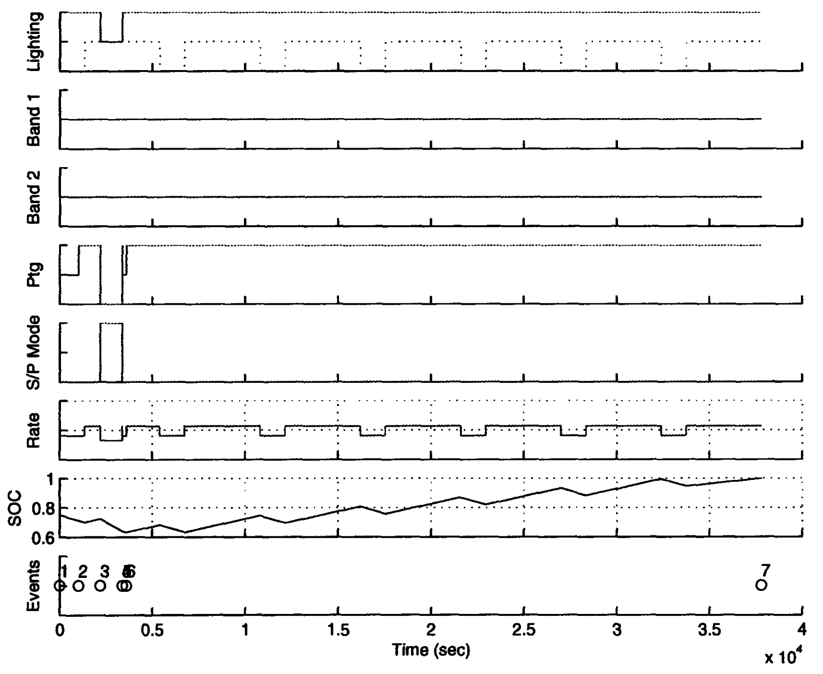
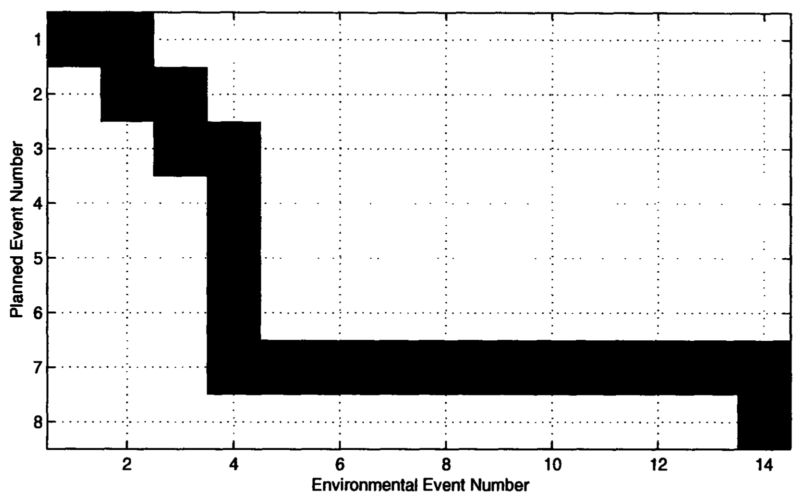


Figure A-3: Experiment 5: Optimal Path and Schedule for $\lambda_\beta = 0.5,\ 0.6$



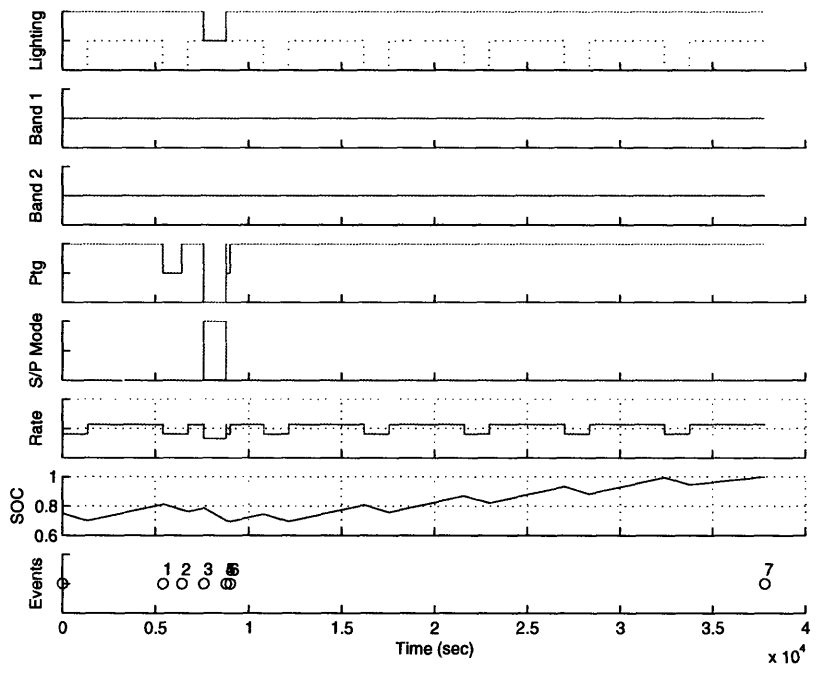
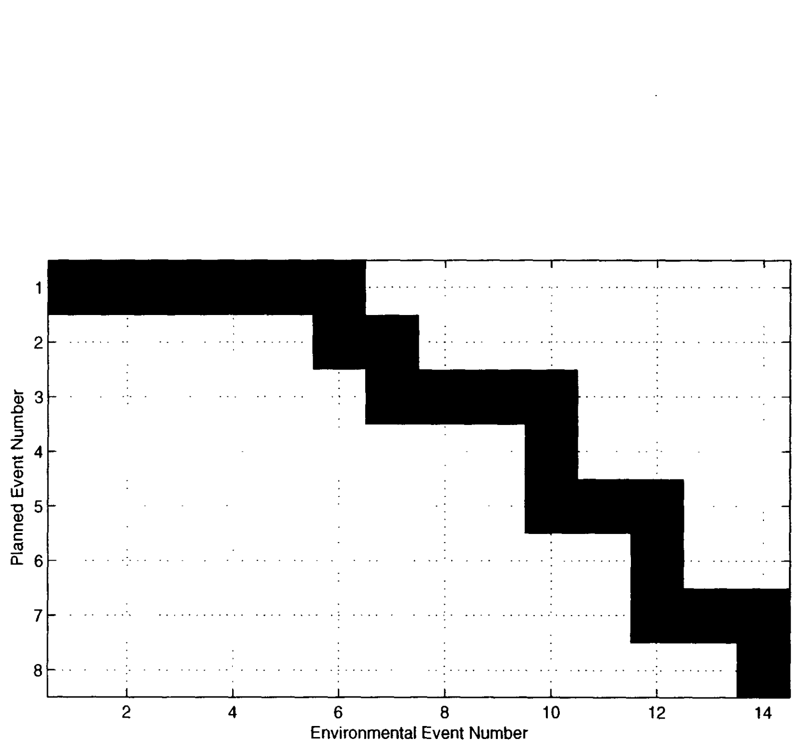


Figure A-4: Experiment 5: Optimal Path and Schedule for $\lambda_\beta = 0.7,\ 0.8,\ 0.9$



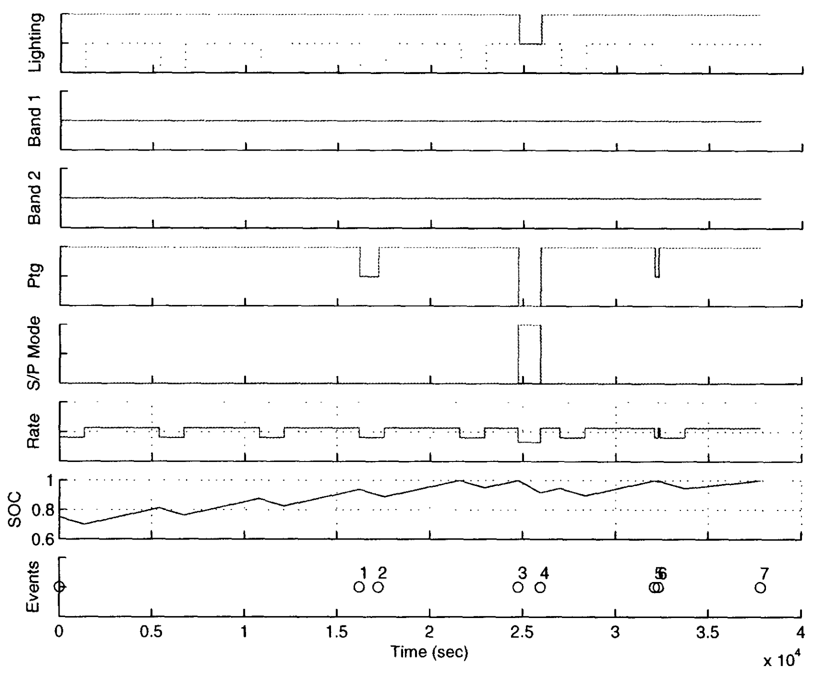
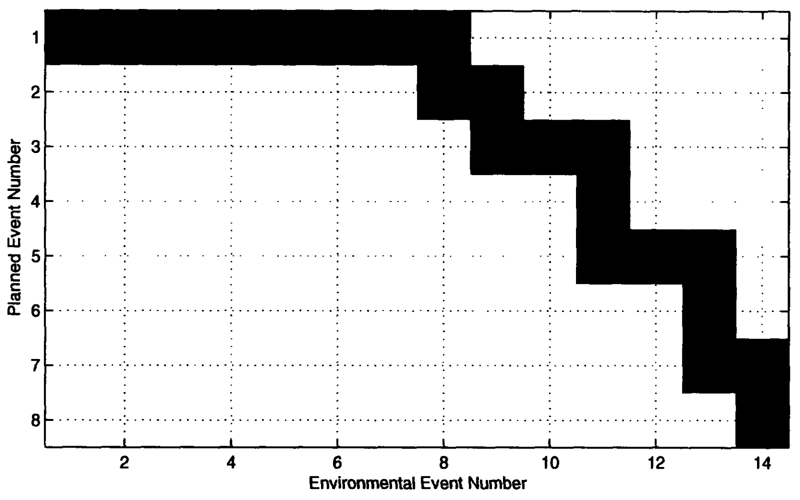


Figure A-5: Experiment 5: Optimal Path and Schedule for $\lambda_\beta = 1.0$



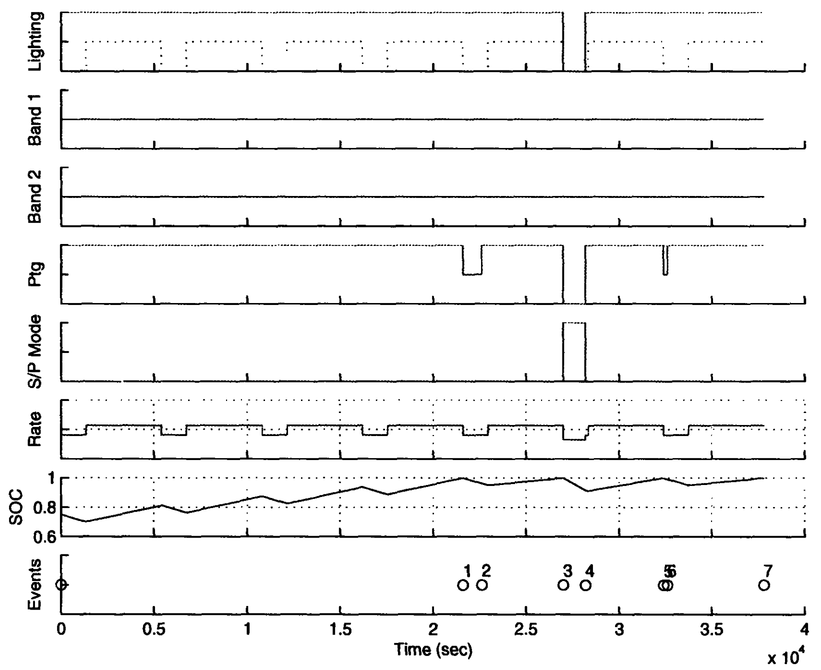
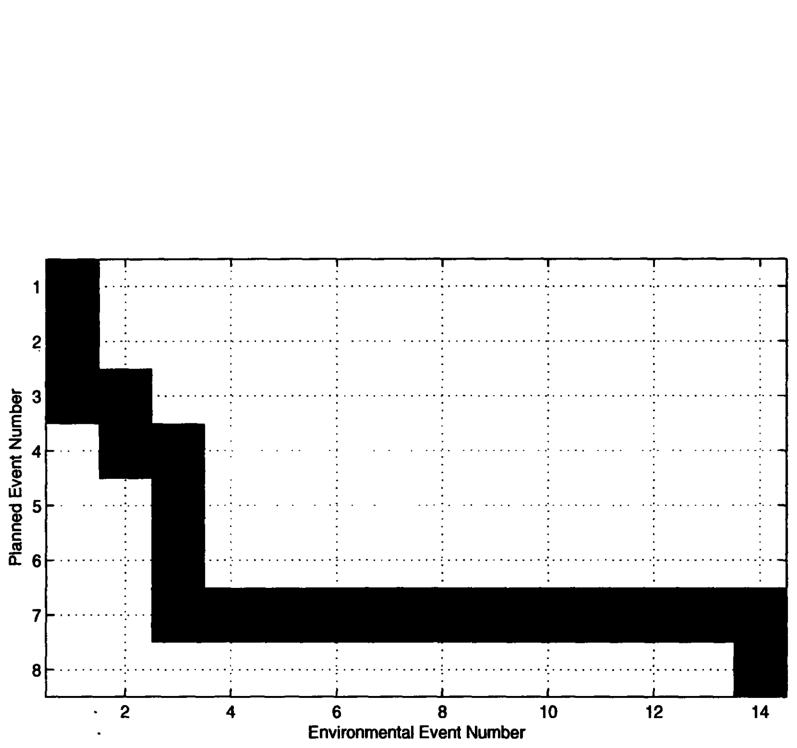


Figure A-6: Experiment 6: Optimal Path and Schedule for $\lambda_\beta = 1.0$



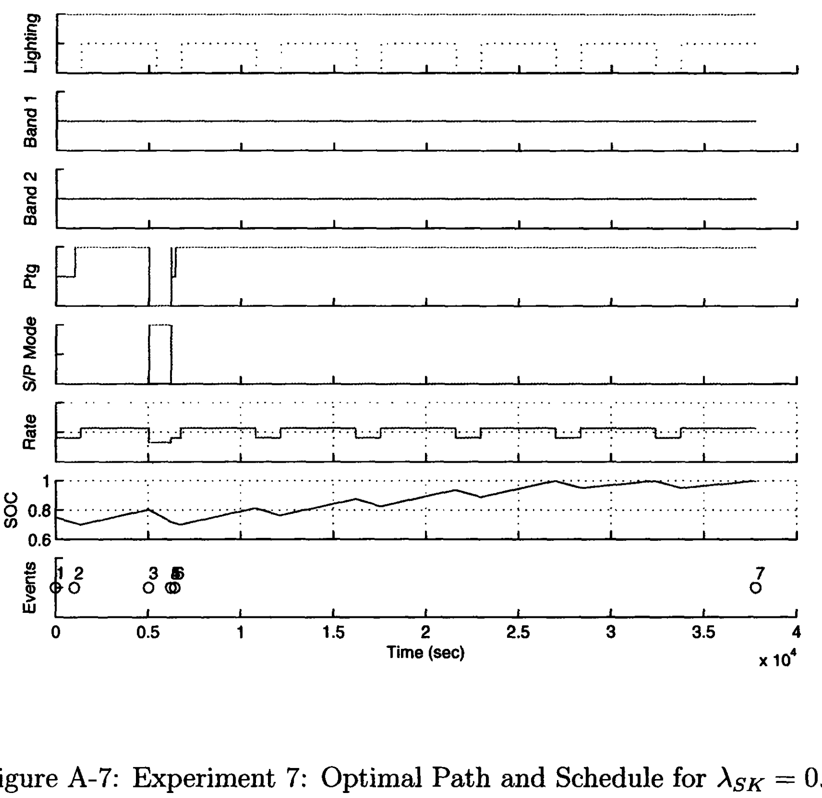
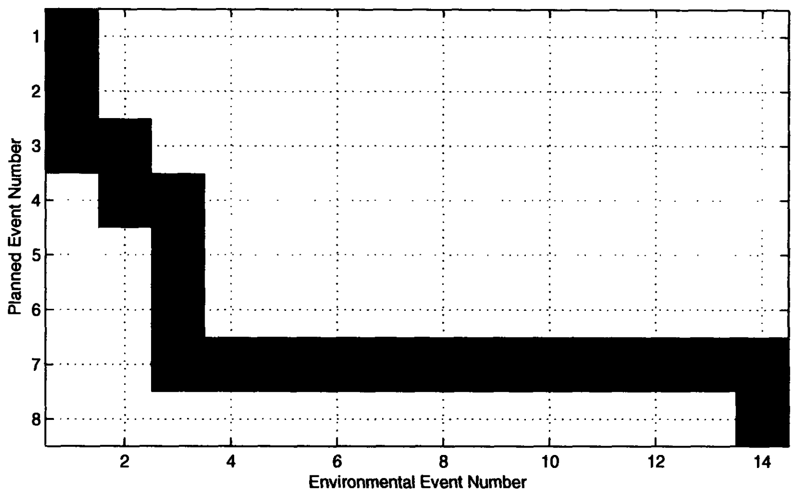


Figure A-7: Experiment 7: Optimal Path and Schedule for $\lambda_{SK} = 0.0$



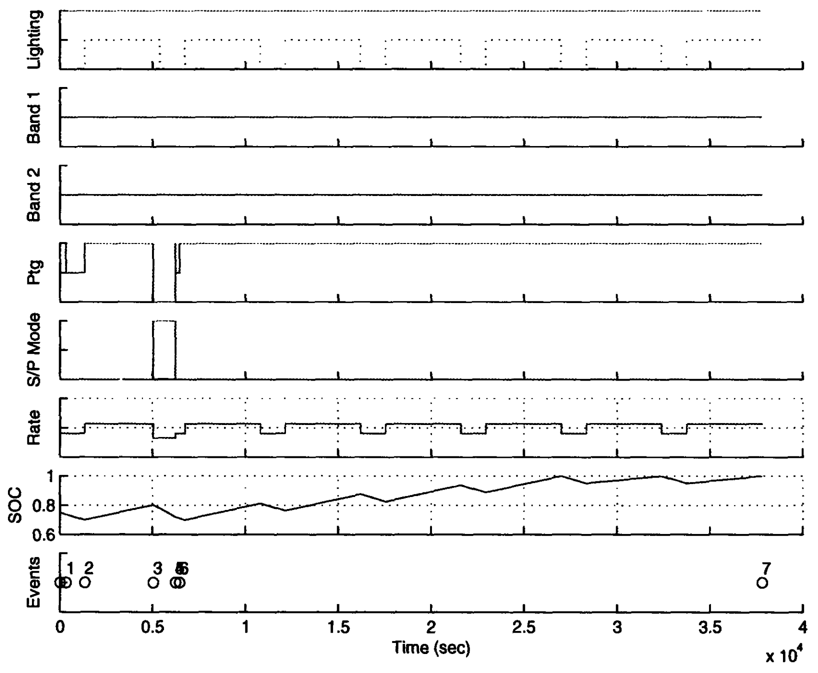
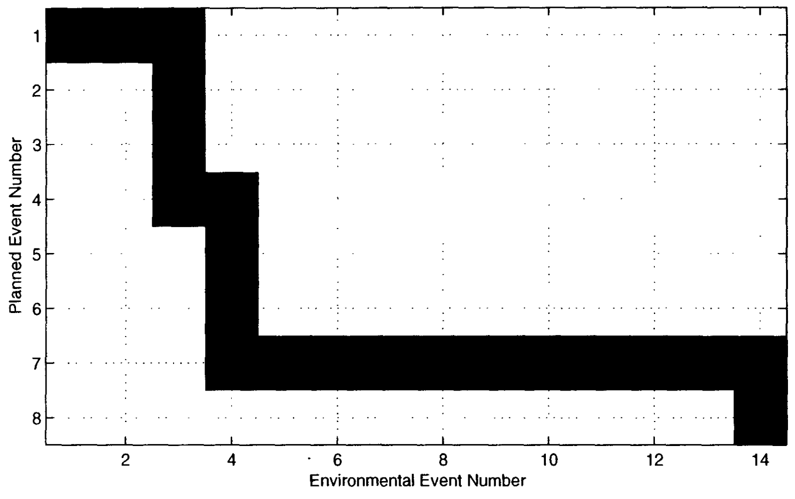


Figure A-8: Experiment 7: Optimal Path and Schedule for $\lambda_{SK} = 0.1$



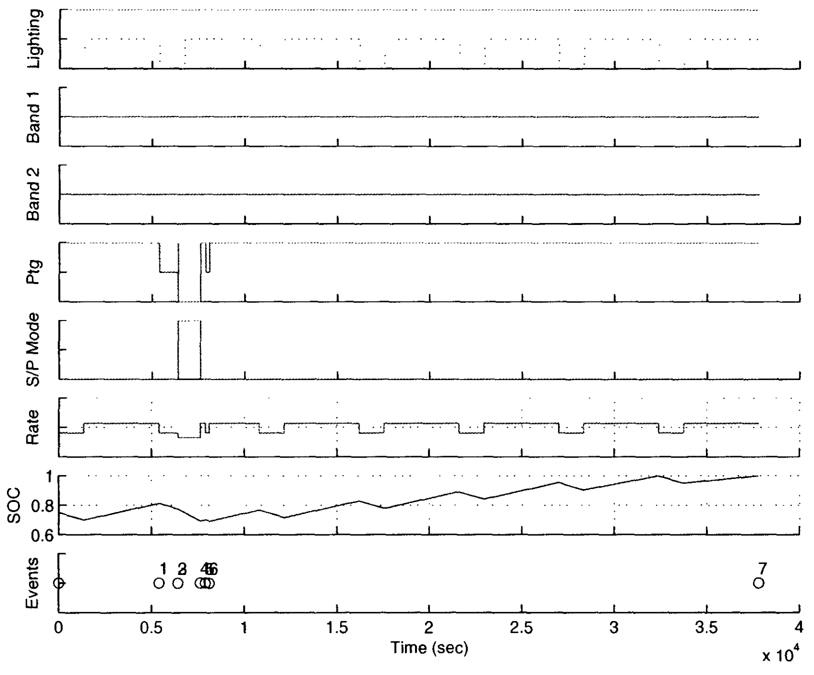
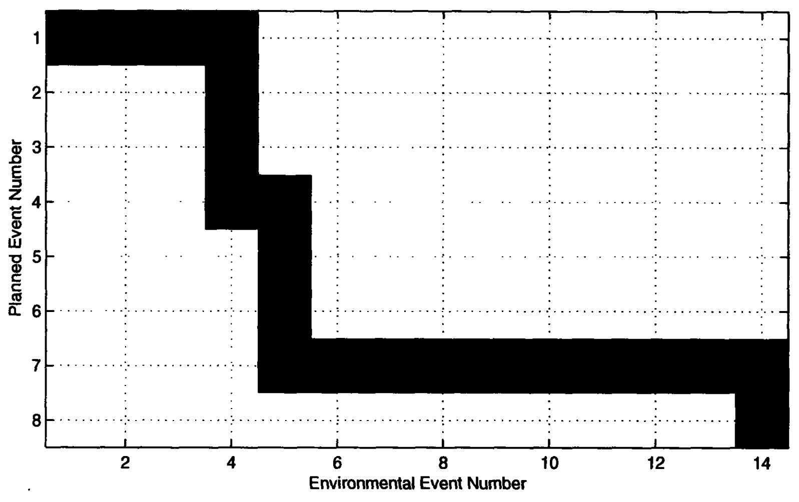


Figure A-9: Experiment 7: Optimal Path and Schedule for $\lambda_{SK} = 0.2,\ 0.3$



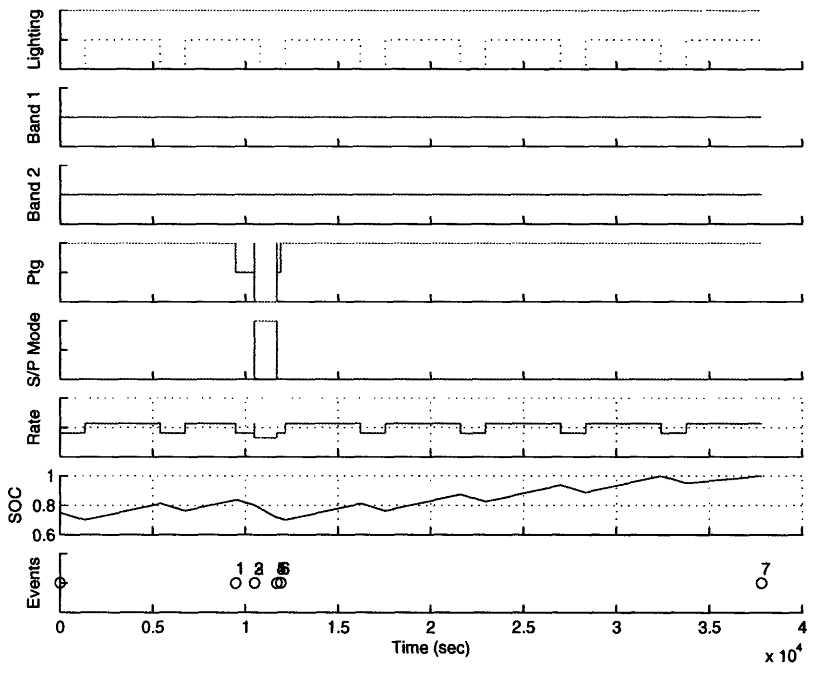
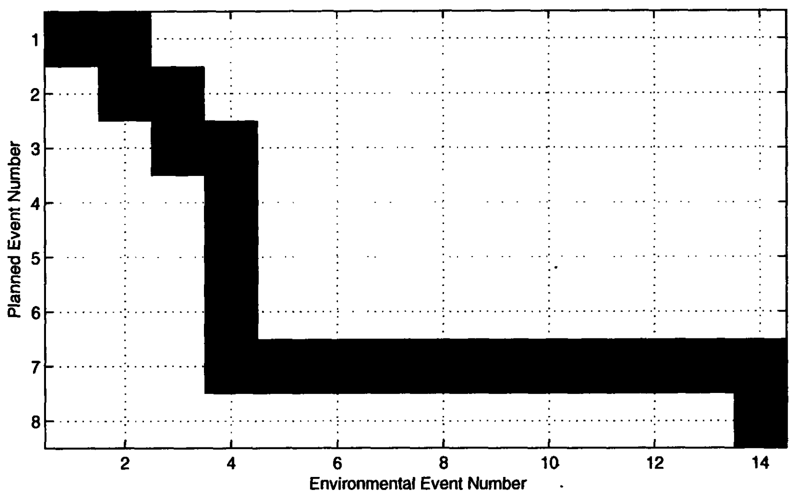


Figure A-10: Experiment 7: Optimal Path and Schedule for $\lambda_{SK} = 0.4, \ldots, 1.0$



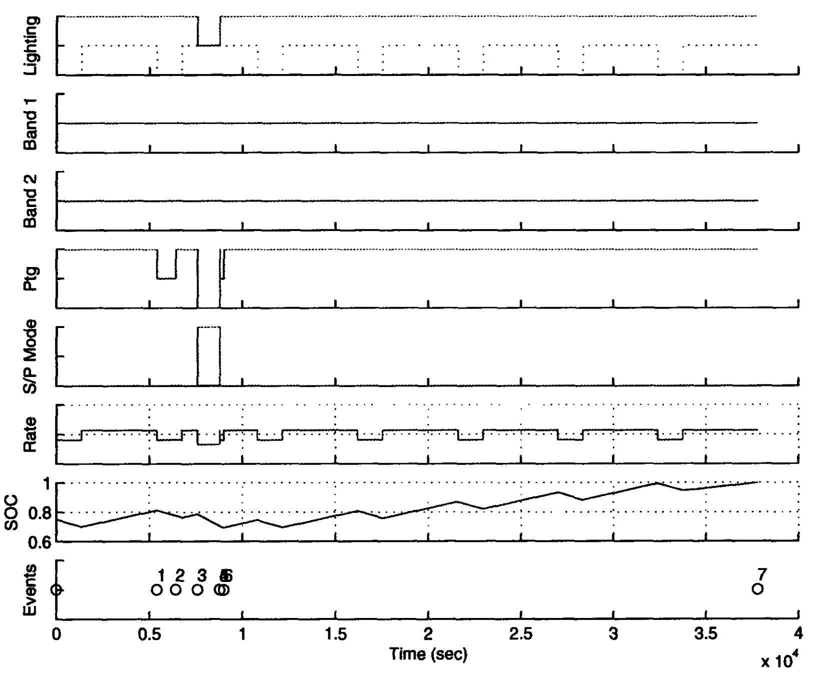
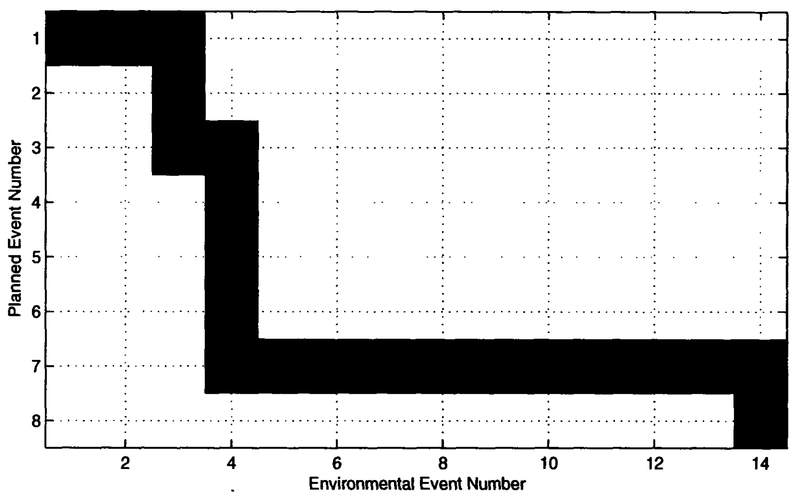


Figure A-11: Experiment 8: Optimal Path and Schedule for $\lambda_{SK} = 0.0$



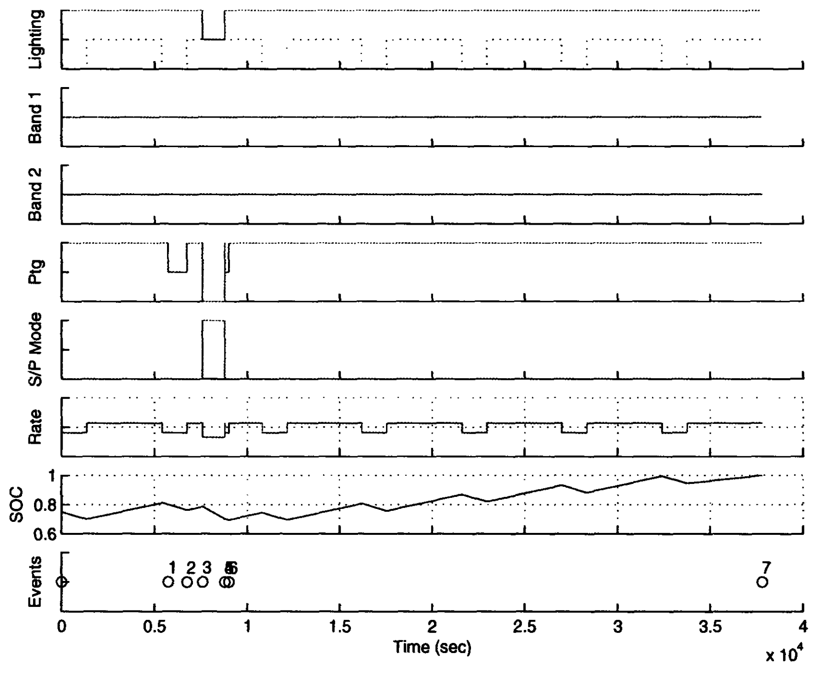
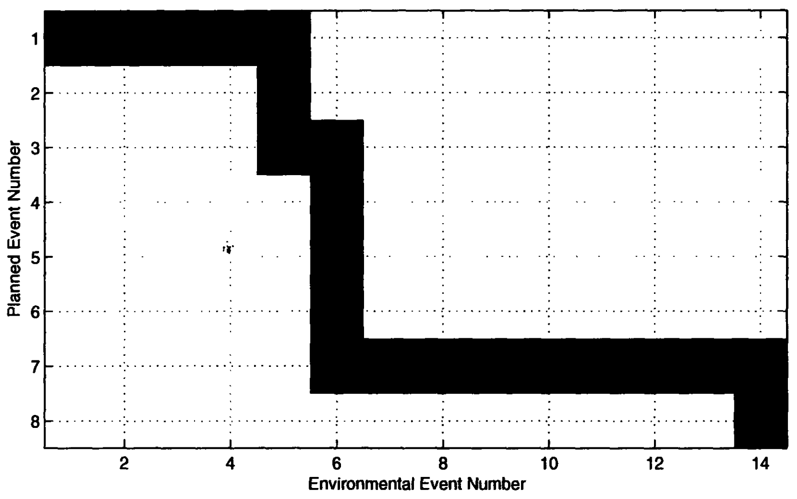


Figure A-12: Experiment 8: Optimal Path and Schedule for $\lambda_{SK} = 0.1,\ 0.2,\ 0.3$



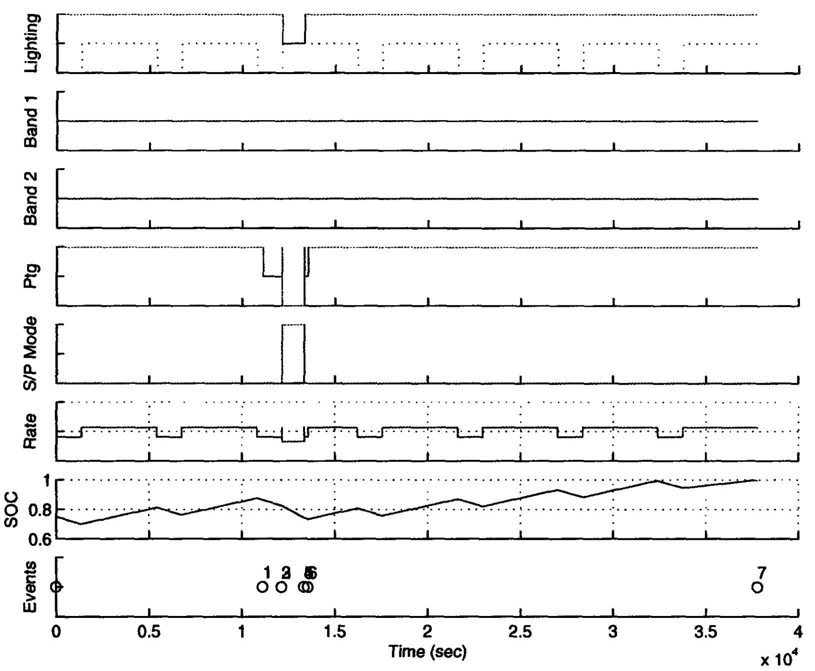
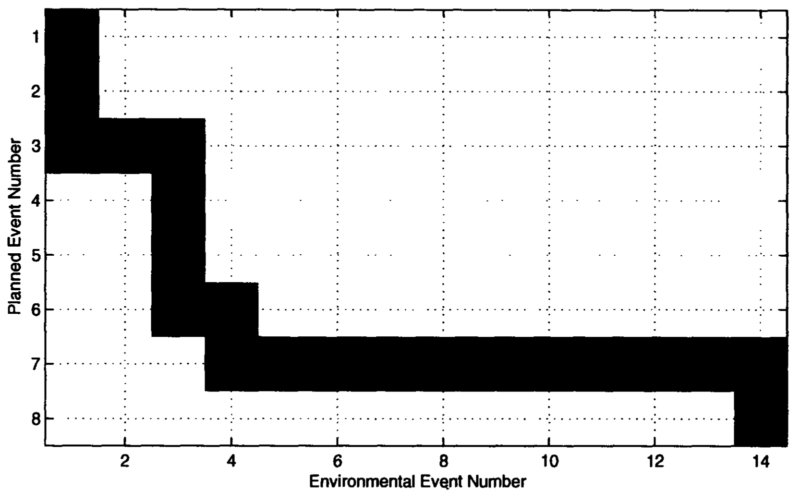


Figure A-13: Experiment 8: Optimal Path and Schedule for $\lambda_{SK} = 0.4, \ldots, 1.0$



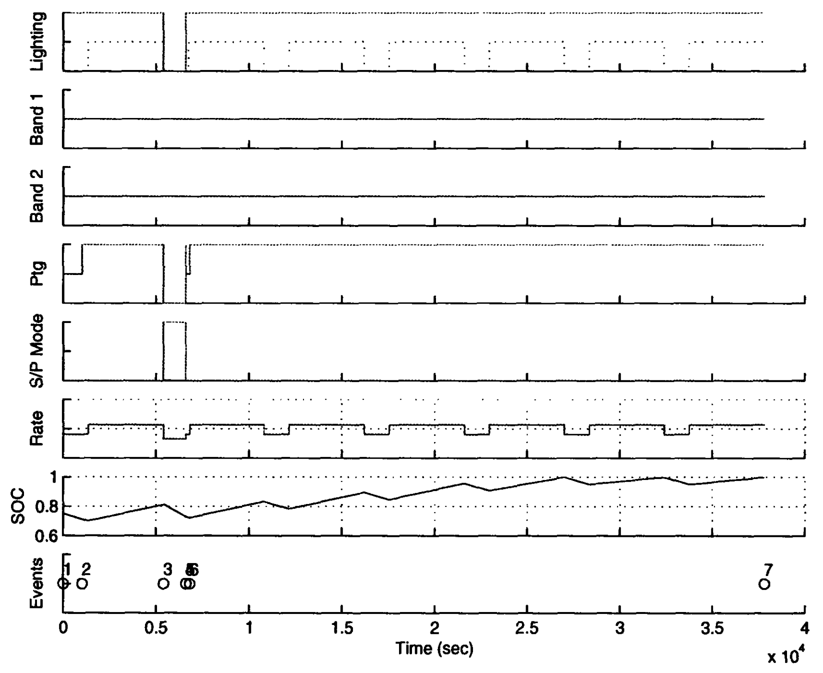
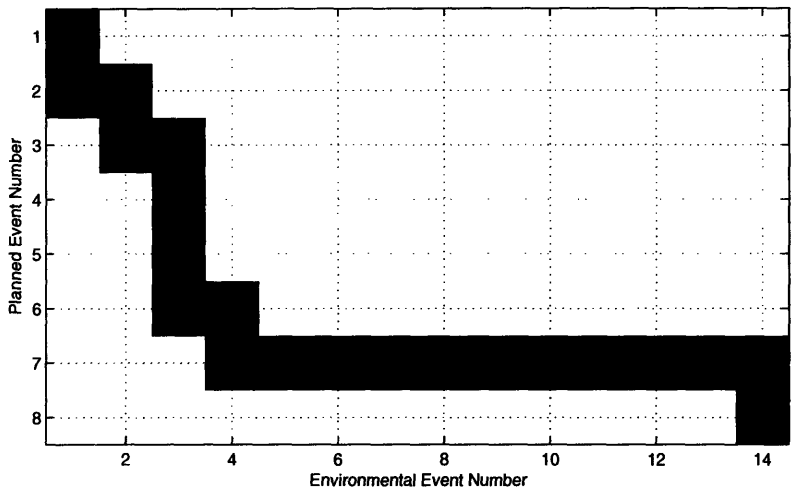


Figure A-14: Experiment 9: Optimal Path and Schedule for $\lambda_{SK} = 0.0$



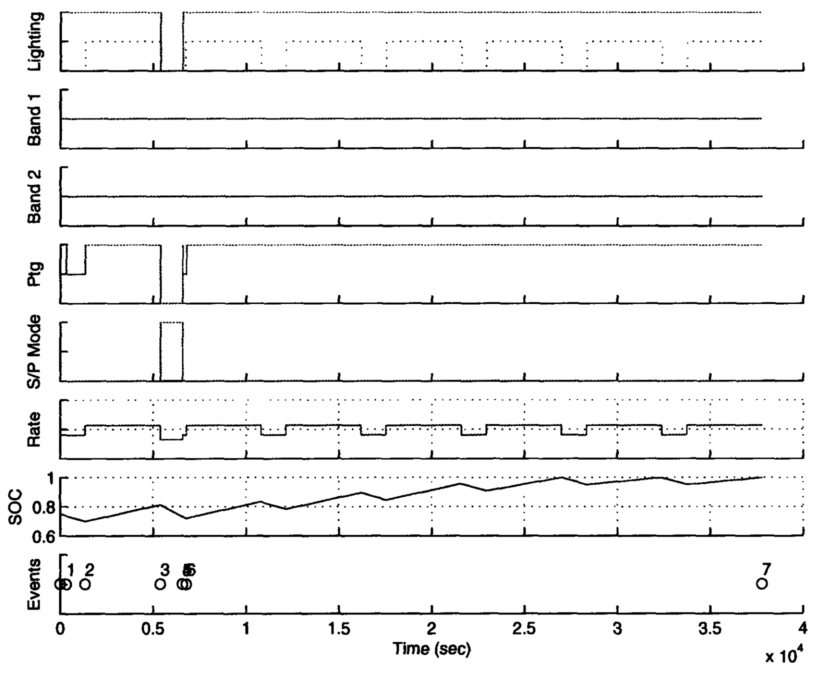
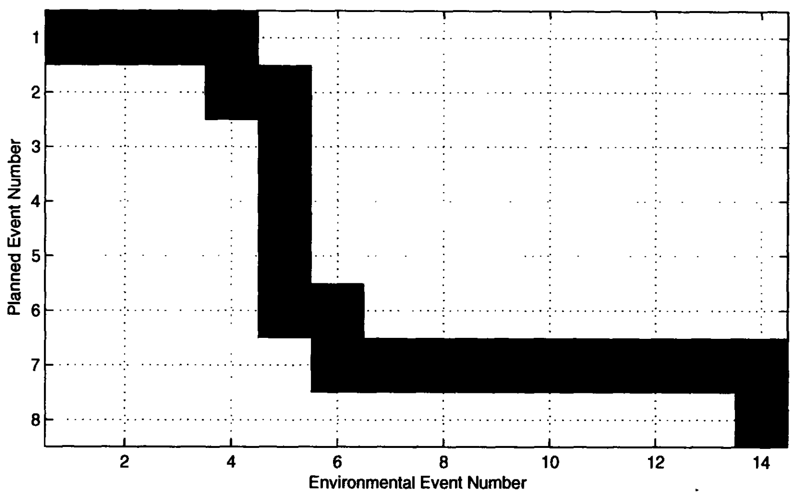


Figure A-15: Experiment 9: Optimal Path and Schedule for $\lambda_{SK} = 0.1$



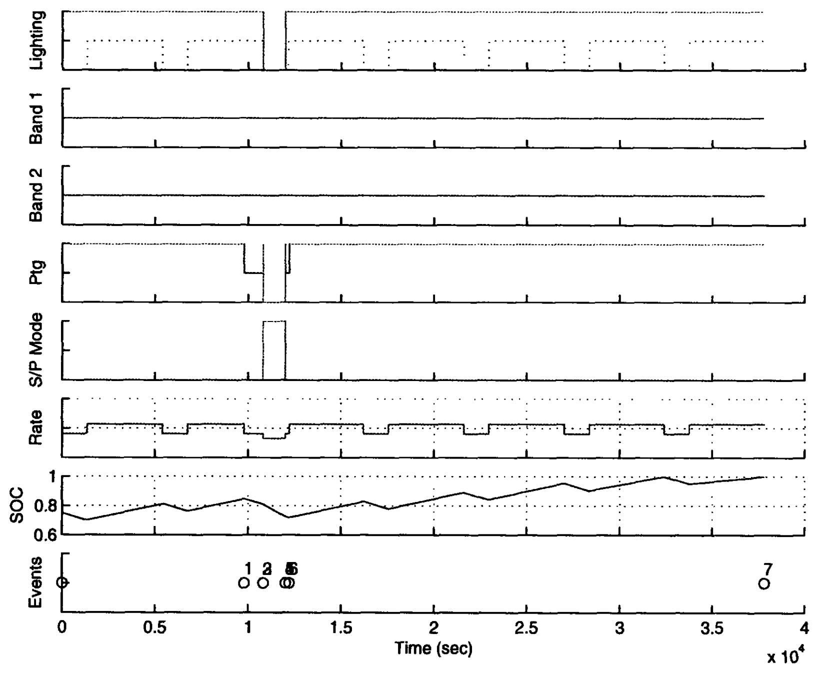


Figure A-16: Experiment 9: Optimal Path and Schedule for $\lambda_{SK} = 0.2, \ldots, 1.0$

[Except for this sentence, this page intentionally left blank.]

Appendix B

Example 2 in Depth

This section thoroughly examines application of the algorithm in Table 4.1 to the example of Section 5.2.2. The algorithm is reprinted here for convenience.

Table B.1: Implementing a MIP

1.	Specify planned modes, **M**, and environmental conditions, **E**
2.	Check for interval compatibility
3.	Compute resource rates
4.	Collect interval, temporal, and resource constraints
5.	Evaluate MIP with a solver package
6.	Review solution

Step 1 requires specifying **M** and **E**.

$$\mathbf{M} = \begin{bmatrix} 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 44 & 45 & 43 & 44 & 42 & 43 & 44 & 45 & 43 & 44 \\ 21 & 21 & 21 & 21 & 22 & 24 & 21 & 21 & 21 & 21 \end{bmatrix} \tag{B.1}$$

$$\mathbf{E} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{B.2}$$

These numbers should not be alarming because they are merely codes that denote a particular activity for a planned mode or environmental condition.

For example, examining the planned requirements (**M**) of $\vec{m}_6$, the first element, 2, indicates a *don't care* description for the lighting requirement. The 0 means that the user does not intend to use Band 1 for communication, but the 1 means that Band 2 is used. The 43 refers to the attitude mode that aims the thruster, and the 24 indicates a specialized sensor mode. For a visual description of **M** see Figure 5-7.

The environmental conditions are binary in **E**, but this is not a necessity. Here, a 1 in the first row indicates sunlight, whereas a 0 means darkness. The second row describes the Band 1 condition and the third, the Band 2 condition. Figure 5-8 describes these conditions.

Step 2 requires evaluating the interval compatibility. Applying the function of Equation 4.29 gives the space of valid overlapping intervals in Figure B-1. The shaded spaces represent overlapping intervals which are compatible. The optimal path can potentially pass through this point. The unshaded spaces indicate interval overlaps through which the path cannot pass.

Step 3 says to compute the resource rates. The values are stored in a matrix, $\mathbf{R} = \{\alpha_{i,j}\}$ for ease of reference. Since there are a large number of values, **R** is presented in three matrices, $\mathbf{R}_1, \mathbf{R}_2, \mathbf{R}_3$.

$$\mathbf{R} = \{\alpha_{i,j}\} = 10^{-3}\,[\mathbf{R}_1\ \mathbf{R}_2\ \mathbf{R}_3] \tag{B.3}$$

R is in the units of *normalized change in charge per second.* The function of Equation 3.20 takes the column vectors of **M** and **E** to compute the values in **R**. For example the first row and second column of $\mathbf{R}_1$ indicate that the overlapping case

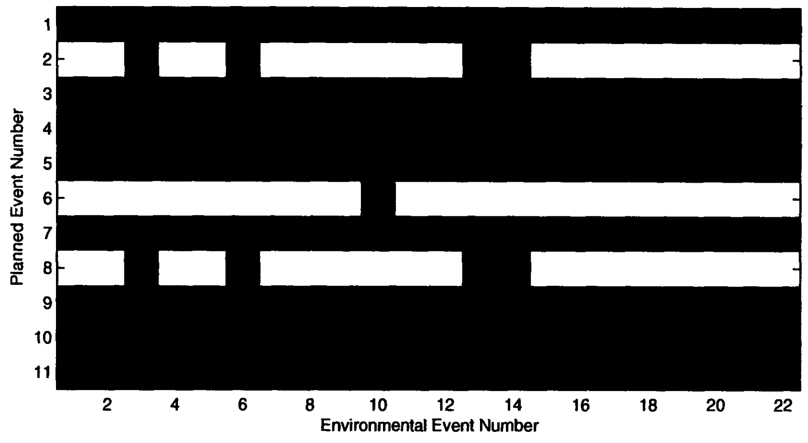


Figure B-1: Space of Valid Overlapping Intervals

of the planned mode described by $\vec{m}_1$ and the environmental conditions of $\vec{e}_2$ evaluates as $\alpha(1,2) = 10^{-3} \times 0.2778$. In this instance of overlap, the resource value should increase because its resource rate is positive. Where the planned mode and environmental condition are not allowed to overlap, the resource rate is shown as 0.

$$\mathbf{R}_1 = \begin{bmatrix} -0.3704 & 0.2778 & 0.0000 & 0.2778 & -0.3704 & -0.6481 & -0.3704 \\ 0 & 0 & -0.6481 & 0 & 0 & -0.6481 & 0 \\ -0.3704 & -0.3704 & -0.6481 & -0.3704 & -0.3704 & -0.6481 & -0.3704 \\ -0.3704 & 0.2778 & 0.0000 & 0.2778 & -0.3704 & -0.6481 & -0.3704 \\ -0.6620 & -0.6620 & -0.9398 & -0.6620 & -0.6620 & -0.9398 & -0.6620 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.3704 & 0.2778 & 0.0000 & 0.2778 & -0.3704 & -0.6481 & -0.3704 \\ 0 & 0 & -0.6481 & 0 & 0 & -0.6481 & 0 \\ -0.3704 & -0.3704 & -0.6481 & -0.3704 & -0.3704 & -0.6481 & -0.3704 \\ -0.3704 & 0.2778 & 0.0000 & 0.2778 & -0.3704 & -0.6481 & -0.3704 \\ -0.3704 & 0.2778 & 0.0000 & 0.2778 & -0.3704 & -0.6481 & -0.3704 \end{bmatrix}$$

$$\mathbf{R}_2 = \begin{bmatrix} 0.0278 & -0.0370 & -0.0370 & -0.0370 & 0.0278 & 0.0000 & -0.0648 & -0.0370 \\ 0 & 0 & 0 & 0 & 0 & -0.0648 & -0.0648 & 0 \\ -0.0370 & -0.0370 & -0.0370 & -0.0370 & -0.0370 & -0.0648 & -0.0648 & -0.0370 \\ 0.0278 & -0.0370 & -0.0370 & -0.0370 & 0.0278 & 0.0000 & -0.0648 & -0.0370 \\ -0.0662 & -0.0662 & -0.0662 & -0.0662 & -0.0662 & -0.0940 & -0.0940 & -0.0662 \\ 0 & 0 & -0.1190 & 0 & 0 & 0 & 0 & 0 \\ 0.0278 & -0.0370 & -0.0370 & -0.0370 & 0.0278 & 0.0000 & -0.0648 & -0.0370 \\ 0 & 0 & 0 & 0 & 0 & -0.0648 & -0.0648 & 0 \\ -0.0370 & -0.0370 & -0.0370 & -0.0370 & -0.0370 & -0.0648 & -0.0648 & -0.0370 \\ 0.0278 & -0.0370 & -0.0370 & -0.0370 & 0.0278 & 0.0000 & -0.0648 & -0.0370 \\ 0.0278 & -0.0370 & -0.0370 & -0.0370 & 0.0278 & 0.0000 & -0.0648 & -0.0370 \end{bmatrix}$$

$$\mathbf{R}_3 = \begin{bmatrix} 0.2778 & -0.3704 & 0.2778 & -0.3704 & 0.2778 & -0.3704 & 0.2778 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.3704 & -0.3704 & -0.3704 & -0.3704 & -0.3704 & -0.3704 & -0.3704 \\ 0.2778 & -0.3704 & 0.2778 & -0.3704 & 0.2778 & -0.3704 & 0.2778 \\ -0.6620 & -0.6620 & -0.6620 & -0.6620 & -0.6620 & -0.6620 & -0.6620 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.2778 & -0.3704 & 0.2778 & -0.3704 & 0.2778 & -0.3704 & 0.2778 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.3704 & -0.3704 & -0.3704 & -0.3704 & -0.3704 & -0.3704 & -0.3704 \\ 0.2778 & -0.3704 & 0.2778 & -0.3704 & 0.2778 & -0.3704 & 0.2778 \\ 0.2778 & -0.3704 & 0.2778 & -0.3704 & 0.2778 & -0.3704 & 0.2778 \end{bmatrix}$$

Step 4 says to collect the interval, temporal, and resource constraints. The number of constraints (just over one thousand) is intractable to print in this document. The following is a partial list of the constraints found in $\tilde{\mathbf{V}}$. The following are duration constraints that are input to the scheduler which correspond to the second list of Section 3.2.3 (p. 46). (Absent from this description of $\tilde{\mathbf{V}}$ are the constraints that encode Equations 4.31–4.36.)

$$\tilde{\mathbf{V}} = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 \end{bmatrix}, \vec{b}_{\tilde{\mathbf{V}}} = \begin{bmatrix} 599 \\ -599 \\ 420 \\ -420 \\ -600 \\ 2700 \\ 899 \\ -899 \\ 599 \\ -599 \\ 420 \\ -420 \end{bmatrix}. \tag{B.4}$$

The constraints in $\mathbf{Y}$, the policy and saturation constraints for the resource, may be written as

$$\mathbf{Y} = \begin{bmatrix} -\mathbf{I}_{M+N-1} \\ \mathbf{I}_{M+N-1} \end{bmatrix}, \vec{b}_{\mathbf{Y}} = \begin{bmatrix} -\text{minVal} \times \mathbf{1}_{M+N-1} \\ \mathbf{1}_{M+N-1} \end{bmatrix}. \tag{B.5}$$

Applying this data to the MIP solver results in a solution. This is *Step 5.*

In *Step 6* the solver responds with solution data. This includes the values of the $b_{m,n}$ which make the solution path. Additionally, the solution has the planned times, p_m, and the resource values r^p_m and r^e_n. The planned times are

$$p = \begin{bmatrix} 6000 & 6599 & 7019 & 10401 & 11001 & 11900 & 16000 & 16599 & 17019 & 37800 \end{bmatrix} \tag{B.6}$$

After arranging the resource values (SOC) in temporal order, the values are

$$r = \begin{array}{c} \Big[\; 0.7500 \;\; 0.7000 \;\; 0.7181 \;\; 0.7181 \;\; 0.7959 \;\; 0.7737 \;\; 0.7737 \;\; 0.7349 \cdots \\ 0.7348 \;\; 0.7292 \;\; 0.7193 \;\; 0.8132 \;\; 0.7868 \;\; 0.7735 \;\; 0.7734 \;\; 0.6665 \cdots \\ 0.6665 \;\; 0.6573 \;\; 0.7642 \;\; 0.7642 \;\; 0.7512 \;\; 0.7253 \;\; 0.7253 \;\; 0.7098 \cdots \\ 0.6901 \;\; 0.8026 \;\; 0.7526 \;\; 0.8651 \;\; 0.8151 \;\; 0.9276 \;\; 0.8776 \;\; 0.9901 \; \Big] \end{array} \tag{B.7}$$

Here, the first value is the initial resource value, r_0.

The rest of the solution including the path and schedule are in Figure 5-9.