Robust Planning for Effects-Based Operations

by

Corban Harrell Bryant

B.S. Operations Research
United States Air Force Academy, 2004

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

Masters of Science in Operations Research

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2006

Author .....
Sloan School of Management
Interdepartmental Program in Operations Research
May 18, 2006

Approved by .....
Mark Abramson
The Charles Stark Draper Laboratory
Technical Supervisor

Approved by .....
Stephan Kolitz
The Charles Stark Draper Laboratory
Technical Advisor

Certified by .....
Cynthia Barnhart
Professor, Civil and Environmental Engineering Department
Engineering Systems Division
Thesis Advisor

Accepted by .....
James B. Orlin
Edward Pennell Brooks Professor of Operations Research
Co-Director, Operations Research Center

Robust Planning for Effects-Based Operations

by

Corban Harrell Bryant

Submitted to the Sloan School of Management
on May 18, 2006, in partial fulfillment of the
requirements for the degree of
Masters of Science in Operations Research

Abstract

In this thesis, we introduce and analyze methods of creating theater-level robust mission plans for Effects Based Operations (EBO) of teams of Unmanned Aerial Vehicles (UAVs), and propose methods of effectively presenting the robust plan to an end user.

Recent conflicts have demonstrated the utility of UAVs in performing Intelligence, Surveillance, and Reconnaissance (ISR) and strike missions. As UAVs become more common, high-level pre-planning and task delegation will increase in complexity, requiring computer aided planning. Traditional planning methods, based on deterministic input data, generate plans that become infeasible in uncertain environments. Because military operations tend to contain substantial amounts of uncertainty and re-planning at a theater level is costly, plans should be robust to uncertainty yet still accomplish desired effects.

We present an effects-based planning framework in which we connect end effects to tasks, enabling planners to value task assignments based on their ability to achieve desired effects. We apply two robust planning techniques to this framework (Bertsimas/Sim and Chance Constrained Programming) and analyze their performance. We demonstrate how robust planning increases the length of time that a plan remains feasible in execution and achieves better overall value by avoiding re-planning costs. We analyze strengths and weaknesses of each model and suggest when their use is appropriate. Finally, we apply Human Machine Collaborative Decision Making (HMCDM) concepts to propose methods to facilitate human interaction with a robust effects-based planner.

Technical Advisor: Andrew Armacost
Lt Col, United States Air Force
Associate Professor and Director of Operations Research

Technical Supervisor: Mark Abramson
The Charles Stark Draper Laboratory

Technical Advisor: Stephan Kolitz
The Charles Stark Draper Laboratory

Thesis Advisor: Cynthia Barnhart
Professor, Civil and Environmental Engineering Department
Engineering Systems Division

Acknowledgments

I would like to thank sincerely all of the people who have helped in the development of this thesis. Mark Abramson and Steve Kolitz, my advisors at Draper Laboratory, thank you for your expert advice and guidance and for taking time for our weekly meetings. Your input has been invaluable. Professor Cynthia Barnhart, my MIT Faculty Advisor, thank you so much for taking time out of your crazy schedule (even during a sabbatical semester) to meet with us regularly. I truly feel like I've gotten to work with the best! Lt Col Armacost, thank you so much for taking time out of your research semester to read and edit my work. It has been a privilege to work with you on this and I hope we get to work together in the future. Thank you for your recommendations for all of us USAFA OR types who are at MIT. We would not be here without you.

I would also like to thank the members of draper staff who have helped me write my thesis. Francis Carr, thank you for all of your computer help. I'd still be de-bugging without you. Ramsay Key, thank you for developing the RPUU Monte Carlo Simulation and for devoting time to teaching Phil and I how to use it. Laura Major-Forest, thank you for your input on the human-factors side of my writing. I appreciate the expertise you all have brought to my thesis.

Philemon Sakamoto and Lavanya Marla, it has been a pleasure studying robust planning with you both. Dan, Phil, Tyler, Kendrick, Lucile, Tom, Christen, Steve, and Lavanya thanks for all the homework help along the way. Andrew, Drew, and Captain Bartolomei, I have really enjoyed our lunch discussions. To all the amazing friends I have made here in Boston, thanks for your support and prayers. You have helped make this time a joy. To my family, thank you for your prayers, love, and support in all that I do.

Finally to Katrina, my beautiful wife, you are the best friend I could ever have. Thank you for your prayers, encouragement, support, and love. You have made this experience about a million times better than it would have been without you. I cannot believe that I ever thought waiting to get married until after grad school was ever an option. I love you so much!

This thesis was prepared at The Charles Stark Draper Laboratory, Inc. under the independent research and development project Robust Planning Under Uncertainty, project number 20332-001. Publication of this thesis does not constitute approval by Draper of the findings or conclusions contained therein. It is published for the exchange and stimulation of ideas. The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government.

Corban H. Bryant, 2d Lt, USAF

May 18, 2006

Assignment

Draper Laboratory Report Number T-1544

In consideration for the research opportunity and permission to prepare my thesis by and at The Charles Stark Draper Laboratory, Inc., I hereby assign my copyright of the thesis to The Charles Stark Draper Laboratory, Inc., Cambridge, Massachusetts.

Corban H. Bryant, 2d Lt, USAF

May 18, 2006

THIS PAGE INTENTIONALLY LEFT BLANK

Contents

THIS PAGE INTENTIONALLY LEFT BLANK

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 1

Introduction

Recent conflicts have demonstrated the benefits of Unmanned Aerial Vehicles (UAVs) in performing Intelligence Surveillance and Reconnaissance (ISR) missions and strike missions. UAVs have long loiter times and ranges, and can perform missions in high-risk areas without placing people in harm's way. As UAV use increases, using them effectively will require more sophisticated planning methods. Because UAVs operate in a dynamic, uncertain environment, mission plans can be rendered infeasible or meaningless if the environment changes from expectations used when making plans. Re-planning costs can be significant, especially in large systems involving human interaction. In this thesis, we investigate methods of creating robust mission plans for UAVs.

Mission plans need to focus on the objectives they are designed to accomplish. Past planning techniques have suffered from failing to connect individual tasks to the end objectives. We use Effects-Based Operations (EBO) to develop a framework in which we connect tasks to resulting effects. Mission planning for EBO can involve very large problems, which can become intractable. These problems can be divided into a planning hierarchy to ensure problem tractability. In this thesis, we focus on the theater-level planning problem. We develop a mixed-integer linear program (MILP) formulation for creating mission plans for EBO. We also apply several robust optimization methods to this formulation to be able to generate robust plans.

Successful mission planning also requires that a human planner remain in the planning loop. Human interaction with computerized planners ensures trust in the computer generated solution, allows the human to validate the plan, and can utilize human's strengths in the planning process. In this thesis, we propose a method for human interaction with a robust, theater-level mission

planner.

1.1 Contributions

With the goal of creating robust mission plans for EBO for UAVs, we make the following contributions in this thesis:

- We introduce the *EBO Framework* as an approach for establishing the relationships between individual tasks, the effects that they cause, and the end objectives that a commander wants to achieve with the plan. The EBO Framework is a convenient approach for assigning groups of tasks that will cause effects with some probability and establishing the different options that might be used to cause the effects.

- We apply the EBO Framework to create the *Deterministic EBO Model*, a MILP Formulation for the theater-level planning problem that uses the nominal values of all uncertain data. We demonstrate the performance of the Deterministic EBO Model on a realistic-sized theater-level planning problem.

- We modify the Deterministic EBO Model using two approaches to design robust plans, Chance-Constrained Programming and the Bertsimas/Sim Formulation. We call the resulting models the *Chance-Constrained EBO Model* and the *Bertsimas/Sim EBO Model.* We analyze these models using Monte Carlo simulation, comparing the plans that we generate using them to: 1.) estimate the performance of current planning methods done by humans; 2.) compare them against plans that we generate using the Deterministic EBO Model; and 3.) test the performance of the robust EBO models against each other. The robust plans created using both models outperform: 1.) the plans generated using current planning method plans and 2.) plans generated using the Deterministic EBO Model in performance metrics measuring the expected time until the plan fails, and the expected value achieved by the plans.

- We discuss how a human planner might interact with the Robust EBO Models. *We apply principles from Human Machine Collaborative Decision Making (HMCDM) to make suggestions on how to implement the Robust EBO Models in the planning process* while capitalizing on the strengths the human planners bring to the planning process and allowing the human to gain trust in the solution.

1.2 Thesis Structure

This thesis is structured to emphasize first the justification for creating the EBO Framework and its application to a robust planner. Second, we emphasize the performance of the robust plans against other planning options. Third, we discuss how we might implement the robust planning methods with the human in the loop.

In *Chapter 2* we give a full description of the theater-level planning problem and justification for using EBO and robust optimization techniques. It contains a review of current military planning processes and background information on EBO. It also contains a literature review of robust optimization.

With the terminology and intent of EBO established, we develop the EBO Framework in *Chapter 3*. We are then able to apply it to introduce the Deterministic EBO Model. We demonstrate the performance of the Deterministic EBO Model. We then develop an algorithm that estimates current planning operations so we can compare it to the performance of the EBO Models. Finally, we develop the Chance-Constrained and Bertsimas/Sim EBO Models, giving pertinent references to the literature as we introduce the robust formulations.

After introducing the various models, we test them using a Monte Carlo Simulation and analyze the results in Chapter 4. We first give a detailed description of the performance of the Robust EBO Models; then we compare them to the estimate of current planning methods and to each other.

In Chapter 5 we review pertinent principles of human decision making and information presentation. We apply these principles to make suggestions for possible implementation of the robust models in a human-in-the-loop planning process.

Finally, in Chapter 6 we summarize pertinent results and contributions discussed in the thesis. Equally important is the identification of future areas of research. This research presents methods that have possible beneficial implications for military planning, but require more development to be implemented effectively.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 2

Problem Background and Literature Review

Recent conflicts have demonstrated the utility of Unmanned Aerial Vehicles (UAVs) in Intelligence, Surveillance, and Reconnaissance (ISR) missions and strike missions. UAVs have long loiter times and can accomplish high-risk missions without placing people in harm's way. As UAVs mature and their numbers increase in proportion to overall air assets, planning for their use will become both more complex and more important. In this chapter, we give some background of current military operations for UAV mission planning and Effects-Based Operations (EBO), introduce the robust theater-level UAV mission planning problem, introduce several approaches that can be used to solve the problem, and present a literature review of these approaches.

2.1 Current Military Operations

"As of September 2004, some twenty types of coalition UAVs, large and small, have flown over 100,000 total flight hours in support of Operation ENDURING FREEDOM (OEF) and Operation IRAQI FREEDOM (OIF)"[19]. The Army predominantly uses its UAVs for tactical support. Many of the Army's UAVs are Micro Air Vehicles (MAVs), carried by soldiers and launched on the battlefield when needed. The majority of the UAVs used by the Air Force has longer range and loiter ability and are maintained, launched, and recovered at specific airfields. This research will focus on the planning for these larger Air Force assets such as the Predator and Global Hawk

instead of the tactical uses of MAVs.

The Air Force outlines its planning doctrine in Air Force Doctrine Documents (AFDD) 1, 2, and 2-1.9. The Air Force's planning system supports joint operations planning as outlined in Joint Publications (JP) 1 and 5. In most respects, the joint planning system; has the same structure as the Air Force planning system, only it incorporates planning across all services.

The Joint Forces Commander (JFC) is responsible for the development of theater plans, with responsibilities spanning all military operations. Component commanders of land, sea, and air aid the JFC in the planning process. The Joint Forces Air Component Commander (JFACC) commands all air forces in the theater and is ultimately responsible to the JFC for effectively using air assets in support of the mission as summarized in AFDD 1[3]:

The Joint Air Operations Center (JAOC) is the primary planning house for military air operations. The JAOC is divided into four primary teams as shown in Figure 2-1: strategy, combat plans, combat operations, and air mobility. The primary functions of the JAOC are to: [2]

Figure 2-1: JAOC Organization Structure [2]

1. develop aerospace operations strategy and planning documents that integrate air, space, and information operations to meet JFACC objectives and guidance;

2. task and execute day-to-day aerospace operations; provide rapid reaction, positive control, and coordinate and de-conflict weapons employment as well as integrate the total aerospace effort;

3. receive, assemble, analyze, filter, and disseminate all-source intelligence and weather information to support aerospace operations planning, execution, and assessment;

4. issue airspace control procedures and coordinate airspace control activities for the Air Control Authority (ACA) when the JFACC is designated the ACA;

5. provide overall direction of air defense, including Theater Ballistic Missile Defense (TMD), for the Area Air Defense Commander (AADC) when the JFACC is designated the AADC;

6. plan, task, and execute the theater ISR mission;

7. conduct operational-level assessment to determine mission and overall aerospace operations effectiveness as required by the JFC to support the theater combat assessment effort;

8. produce and disseminate an Air Tasking Order (ATO) and changes;

9. and provide for the integration and support of all air mobility missions.

This thesis primarily focuses on aiding items 2, 4, 5, 6, and 7 in the list above. Most of these tasks fall in the Combat Plans Division of the JAOC. Combat Plans determines the optimal combination of target, platform, weapon, and timing for missions included in the ATO; ensures aerospace tasking supports the overall Joint Task Force (JTF) campaign; produces and disseminates an operationally and tactically sound ATO; and generates special instructions (SPINS) and the daily airspace control order (ACO)[2].

AFDD 2 summarizes the theater air planning process done by the members of the Combat Plans Division. This process is also depicted in Figure 2-2.

> Planning is an iterative process. Tasks and targets are nominated to support the objectives and the commander's priorities. All potential tasks and targets are developed through the planning cross-functional teams, which will identify, prioritize, and select specific tasks while considering available resources. In accordance with the commander's objectives and coalition or component requirements, the operations staff will develop the necessary plans to employ capabilities and forces. During weaponeering and force allocation, tasking and targeting personnel *quantify the expected results using modeling and simulation methods.* The final prioritized tasking and targets are then included in

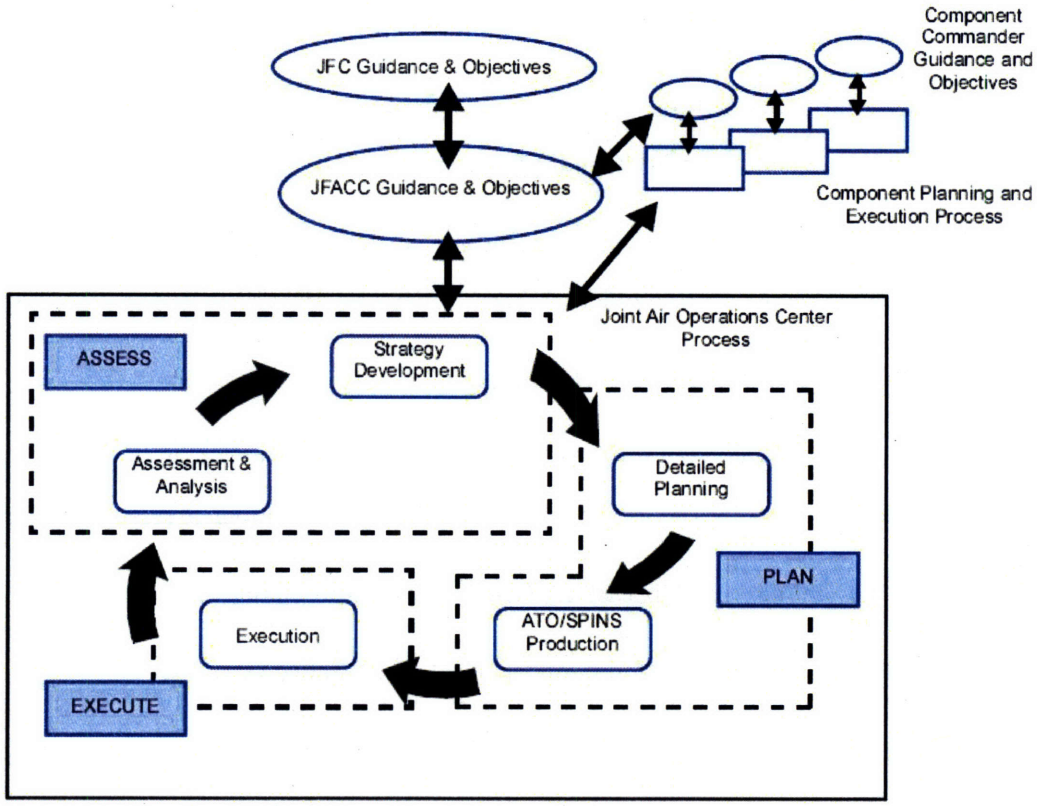


Figure 2-2: JAOC Planning Process[2]

a Master Air Attack Plan (MAAP) that forms the foundation of the ATO. After the commander approves the MAAP, teams finalize the ATO, special instructions (SPINS), and the airspace control order (ACO). The end product of the planning phase is an ATO, an air defense plan, and a prioritized list of tasks and targets with planned time of execution[2].

This thesis focuses on the "iterative" process of connecting the JFACC guidance and objectives to strategy and detailed plans, providing an effects-based model for this process, specifically in reference to planning for UAVs.

2.2 Effects-Based Operations

In the forward of AFDD 2-5.3, Major General Bentley Rayburn, Commander of the Air Force Doctrine Center says, "America's national security rests on a strategy of full spectrum dominance supported by effects-based planning and operations"[5]. The Department of Defense (DOD) views EBO as essential to the advancement of the military's strategic planning ability, and (as indicated by the numerous revisions of its primary doctrine documents) is trying to make EBO part of main-stream thought in the armed forces.

Military strategists have realized that a major challenge throughout the history of air planning has been how to connect top-level objectives to individual tasks. To a large extent, the EBO movement and the passion of its advocates stem from the wartime experiences of young U.S. Air Force officers who were appalled by the frequently mindless and ineffective use of air power in Vietnam[8]. AFDD 2.1 states, "Failure to properly analyze the mechanism that ties tactical results to strategic effects has historically been the shortcoming of both airpower theorists and strategists"[1].

EBO highlights the importance of connecting effects to tasks and stresses considering the full range of outcomes that a particular action can cause. The DOD has recently begun incorporating EBO into all areas of its planning doctrine. JP 1 (Joint Warfare), JP 5 (Plans), JP 5-1 (Joint Task Force Planning), AFDD 1 (Air Force Basic Doctrine), AFDD 2 (Organization and Employment of Aerospace Power), AFDD 2-1 (Air Warfare), and AFDD 2-1.9 (Targeting) are all being revised to incorporate EBO.

The January 2006 draft of the revised AFDD 2[6], defines some basic EBO terminology.

> EBO are operations that are planned, executed, and assessed in order to create specific effects that contribute directly to desired military and political outcomes. The basic methodology of EBO encompasses objectives, effects, and actions...*Objectives* are clearly defined, decisive, attainable, and measurable goals toward which every military operation should be directed. *Effects* are the full range of outcomes, events, or consequences that result from a particular action or set of actions. *Actions* are individual deeds or acts of will that can be either kinetic (physical, material) or non-kinetic (logical, behavioral).

In this thesis we use the term *tasks* as AFDD 2 defines *actions. If tasks are accomplished; they will cause effects, which we hope achieve objectives.* For a thorough discussion of the taxonomy of EBO terminology, see McCrabb's "Explaining Effects: A Theory for an Effects-Based Approach to Planning, Executing, and Assessing Operations"[43].

Although the concerted push to incorporate EBO into the military planning process has happened in the past several years, the ideas of EBO are not "new." Air warfare theorists such as Douhet, Mitchell, and Warden have differed in their preferred mechanism for forcing their will on the enemy, but each realized that the ultimate determinant lay not in destroying targets but in causing effects[8].

The latest drafts of Air Force Doctrine on planning (specifically AFDD 2-1.9 Targeting)[4] fully embraces and centers itself on EBO. It states:

> Planning, employment, and assessment should be inextricably linked and an effects-based approach should attempt to meld them as seamlessly as possible.
>
> Operations are built "from the top down," starting with the end state, through objectives at the highest levels, determining subordinate objectives needed to support those, then determining the effects needed to accomplish the objectives, and finally determining the actions necessary to create those effects.
>
> Objectives and the end state are products of commander's guidance, strategy development and planning, and while targeting efforts must always aim toward achieving them, they are not determined through the targeting process itself.

EBO highlights the process by which national objectives turn into detailed plans. "Effects-based operations provide the ideal means to execute this strategy-to-task framework because it forces planners to consciously link efforts with objectives and lower-level objectives with higher ones"[8]. The process by which national objectives are passed down to become air strategy can be demonstrated by a "Z-diagram" shown in Figure 2-3. There are multiple levels connecting end objectives and actions. Likewise, there can be multiple levels of effects. Causing one effect might subsequently cause another. Without venturing into taxonomy, these terms serve to identify those effects that cause others. In general, it is important to note that effects might be linked through many levels, might cause unintended effects, and might require more than one effect on more than

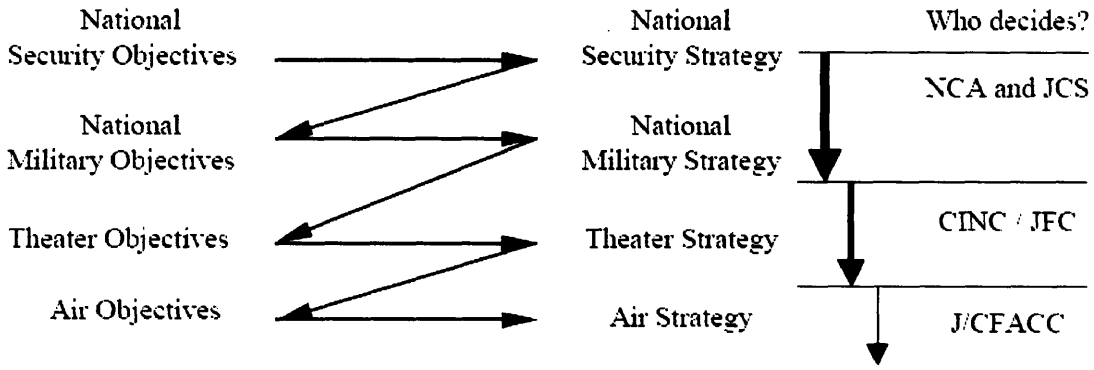


Figure 2-3: "Z-Diagram" of National Objectives to Air Strategy[8]

one level to cause a higher order effect. The relationship of how an action can cumulate into higher effect is shown in Figure 2-4.

In his 2001 RAND Study, Paul Davis asserted "Current methods of analysis and modeling are inadequate for representing EBO"[28]. He proposed a list of principles necessary for the incorporation of EBO into planning models. Among those principles that this research seeks to address are:

- EBO analysis should fully confront the scope and magnitude of uncertainty and should deal explicitly with probability and randomness;
- dealing with uncertainty will require low-resolution exploratory analysis for breadth;
- modeling should be organized around adaptive systems for command and control and other matters;
- a key element of analytical work should be qualitative modeling, including cognitive modeling of the decision-making and behavior of commanders.

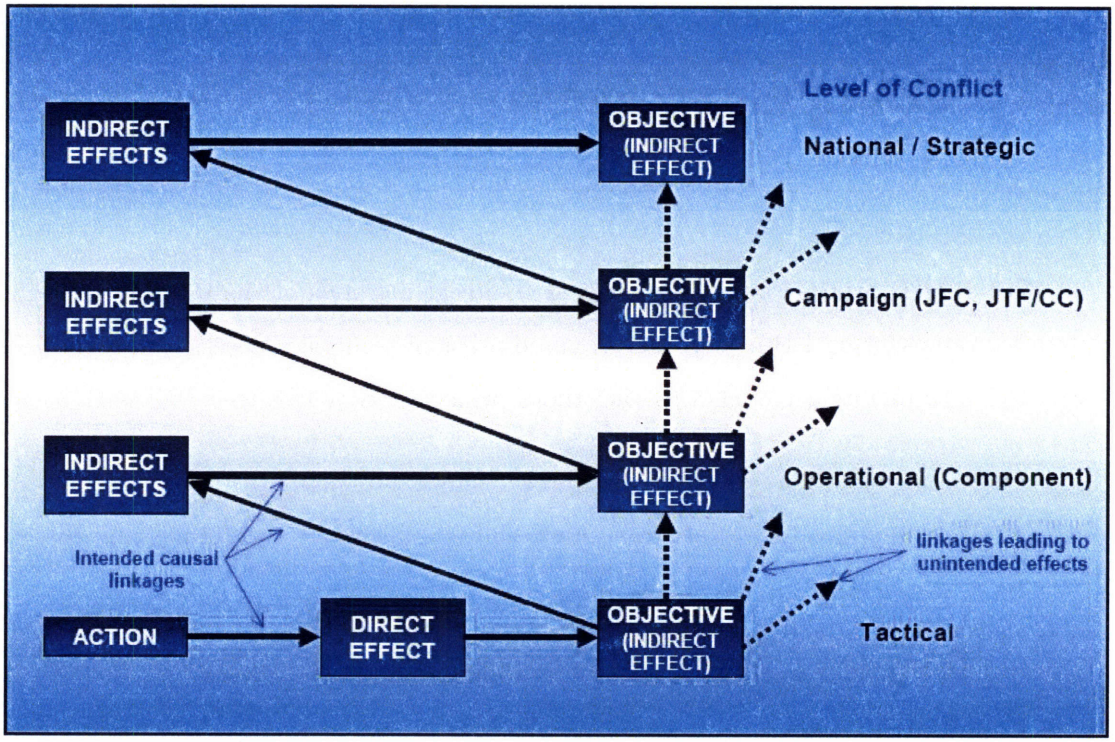


Figure 2-4: Hierarchy of Effects and Objectives[6]

2.3 Problem Description and Motivation

As UAVs become more common, high-level pre-planning and task delegation will increase in complexity requiring computer aided planning. To foster understanding and trust of computer-generated plans, humans must remain in the planning loop. The rapid and sometimes unexpected changes that occur in military operations can cause mission plans to become impossible to execute or ineffective in causing desired effects; therefore, mission plans need to be robust to uncertainty.

This research primarily focuses on planning for UAVs that requires significant pre-planning. Because the AOC currently makes plans for these UAVs as well as for all manned assets, there is significant interplay between manned asset planning and UAV planning. We focus on planning for UAVs and assume a future battlespace where UAVs are far more prevalent and utilize more autonomy.

2.3.1 The UAV Planning Hierarchy

UAV mission planning is a large problem that can be decomposed into small problems to attain tractability. UAV mission planning involves determining which individual tasks will achieve objectives, assigning those tasks to vehicles, and creating the flight plan for individual vehicles. Planners should focus on achieving end-objectives. In doing so the planners must consider the vehicle's aptitude or capability to perform a task, the risk involved, the proximity to the task, and the ability to cooperate with other aircraft. Attempting to solve all of these problems using a single monolithic formulation can create a problem so large that it is intractable. To maintain tractability, we can decompose the planning process into a hierarchical structure in which each level encompasses decreasing numbers of tasks, aircraft, and responsibilities, but increasing detail. We can decompose the planning process into a theater-level planner that assigns tasks to squadrons, a squadron-level planner that assigns aircraft in a squadron to tasks and create routes to those tasks, and a vehicle-level planner that plans individual vehicle trajectory and controls. *This research focuses on the theater-level planning problem.*

2.3.2 The Theater-Level Planning Problem

The theater-level planning problem presents the challenge of how to maximize benefit gained by causing effects while ensuring squadrons have a feasible task assignment. Task assignments must be

made amidst uncertainty created by a dynamic environment, an intelligent adversary, and various performance uncertainties. Thus, a good task assignment must not only maximize benefit gained from causing desired effects but must also be robust to uncertainty. This planning function is currently accomplished in the JAOC. The theater-level planner developed in this research can aid JAOC planners as they connect effect to individual tasks.

Air Force doctrine [2] specifies that JAOC planners should use modeling, simulation, and math programming methods to help create the ATO, which is a daily updated schedule of all air operations in the theater. Optimization techniques such as modeling, simulation, and math programming are approaches that have the potential to help the user generate task assignments. In many planning models, it is difficult to connect commander's intent to individual tasks, which could create doubt about the plan's quality. Furthermore, a good planning model must present clear rationale to decision-makers about why assignments were made in the plan.

2.4 Approaches to Solving the Theater-Level Planning Problem

We address three aspects of the theater-level planning problem:

1. We create a framework for formulating the problem and ensuring the plan accomplishes desired effects; for this we use an Effects-based Operations (EBO) approach.

2. We look at ways to make plans that last longer and require less re-planning; for this we use robust optimization techniques.

3. We look at ways to help a user interact with the planner; for this we use principles of Human Machine Collaborative Decision Making (HMCDM).

2.4.1 Effects-Based Operations

A significant challenge in the Theater-Level Planning Problem is how to connect theater objectives to individual taskings. Knowing which tasks are assigned to which squadrons in a plan does not necessarily indicate whether the plan will be good. EBO are "operations conceived and planned in a systems framework that considers the full range of direct, indirect, and cascading effects-effects that may, with different degrees of probability, be achieved by the application of military diplomatic,

and economic instruments"[28]. Through EBO, planners have a tool to value tasks based on their ability to achieve desired effects.

In the case of UAV mission planning, EBO allows a planner to organize tasks according to the effects they cause. He can then assign tasks to UAVs in order to achieve the maximum benefit from the resulting effects. In such cases, there is uncertainty associated with the certainty that a set of tasks will cause an effect. This uncertainty highlights the need for an EBO model to deal explicitly with uncertainty to achieve a robust plan.

2.4.2 Robust Optimization

Traditionally mission planning is an iterative process following four defined steps. Although the terminology changes among organizations, the steps are generally the same. In Draper Lab's All-Domain Execution and Planning Technology (ADEPT) Planning and Control Architecture[49], they are called: monitoring, diagnosis, generation, execution (depicted in Figure 2-5). Military members might be more familiar with John Boyd's OODA Loop: observe, orient, decide, and act. Air Force Doctrine Document 2 (AFDD 2) labels the planning steps: guidance and objectives, plan, execute, and assess[2]. Regardless of terminology used, these steps describe the general process of gaining a broad understanding of the situation, formulating a plan, executing the plan, and adjusting based on the effects resulting from actions executed in the plan.

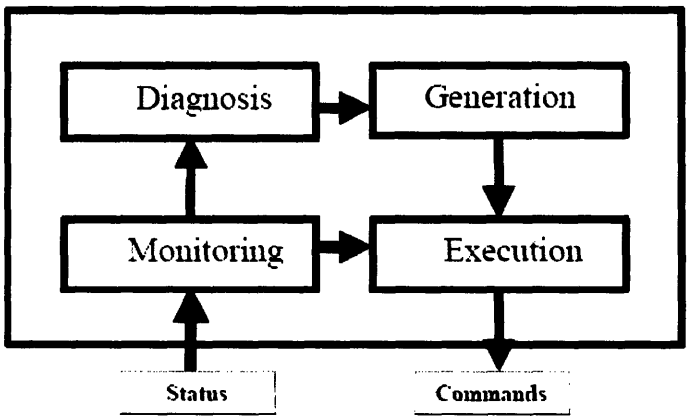


Figure 2-5: A General Planning Cycle Structure[61]

In robust optimization we seek a plan that "backs off" from optimality so that, when executed,

the plan will be feasible more often and for a longer period of time, in light of an uncertain, dynamic planning environment. In terms of the planning steps outlined in Figure 2-5, we intentionally create a plan that we can "Execute", decreasing the amount of times we must "Diagnose" and "Generate" a new plan. For UAV planning, a robust plan means that when the scenario's realization differs from expectations we must re-plan less often. The reduction of re-planning reduces the planning costs, which can be significant in systems involving human coordination, such as time spent generating and approving a new plan, disseminating plans among squadrons, preparing aircraft for flight, staffing flight operations, and backtracking from already accomplished tasks. When the need to re-plan is reduced, this improves the planner's confidence that the execution of the plan will accomplish desired objectives. When using EBO in UAV planning, a robust plan allows the human planner to be more certain that the plan will cause the desired effects and the squadron taskings will be feasible with greater probability.

Many planning methods involving uncertain data use dynamic programming, which handles uncertainty well, but can become intractable as the problem size increases. To maintain tractability, we look at math programming optimization methods. We avoid non-linear methods, because we are seeking to solve very large problems using math programming, where non-linearity tends to affect tractability negatively.

2.4.3 Human Machine Collaborative Decision Making

It is important that commanders and other decision makers understand and approve of the plan generated by computerized planners. Assigning tasks to multiple squadrons is a complicated process, especially when trying to maximize benefit received from achieved effects while also trying to make a robust plan. Computerized planners can handle problems of this magnitude; however, they might return solutions that are difficult to justify to a commander. Elements of Human Machine Collaborative Decision Making (HMCDM) offer promising techniques to help validate and improve the quality of the plans.

2.5 UAV Task Assignment Literature

The UAV task assignment problem can be modeled as a general knapsack problem. The objective of the knapsack problem is to maximize the benefit received by selecting objects to go into the knapsack

without exceeding the knapsack's capacity. Likewise, the objective of the task assignment problem is to maximize expected benefit received from assigning tasks without exceeding the workload capacity of the UAVs. The range of the aircraft, amount of ordinance, duration of mission, and other factors can determine workload capacity. If the problem also includes the routes taken by the aircraft, the problem becomes a vehicle routing problem (VRP), which is known for its intractability issues when trying to solve large problems to optimality.

In this thesis, we separate the routing problem from the tasking problem, delegating routing to the squadron-level planner. For an in-depth analysis of robust planning at the squadron level, see Sakamoto[50].

When using an EBO framework, we gain value by causing effects. The effects we can cause are limited by the squadrons' capacities. Once we determine which effects we are going to try to cause and hence which tasks we will do, we must delegate tasks among the squadrons. Unfortunately, without considering the individual vehicle routings, it becomes difficult to judge the relative value of the assignment based on the location of the tasks. In our models presented in Chapter 3, we delegate tasks based on distance to squadrons with a simple distance calculation. Ideally, when assigning squadrons to tasks we would cluster tasks to help make better routes at the squadron level. Several heuristic algorithms, which we present here, are available to cluster groups of tasks. These could be incorporated as a second stage in a theater-level planning algorithm after an EBO model has decided the effects and tasks to be put in the plan.

2.5.1 K-Means Clustering

The K-means clustering algorithm is a simple iterative algorithm that creates k groups based on their members' distance from the group's center of mass[37]. It is a convenient tool to partition tasks into clusters to create easily solvable routing problems, sometimes generally referred to as "cluster-first, route second" methods. When dealing with task locations, we start by dividing all tasks randomly into k groups of equal size. We then find the center of mass of each group. We then alternate between the following two steps:

- for each center, we identify the subset of points that is closer to it than any other center,
- and we compute the means of each feature for the data point in each cluster, and this mean vector becomes the new center for that cluster.

We continue iterating until there are no changes. This very general clustering algorithm does not consider the number of tasks per group, the relative difficulty, or the value of accomplishing those tasks. For a more sophisticated method of partitioning tasks into small routing problems, we look at Wei Zhao's Sweep Algorithm.

2.5.2 Sweep Algorithm

Wei Zhao [61] addresses the problem of routing aircraft from one airbase to many tasks, or "goal-points" that are assigned to the base. Realizing that routing many aircraft to many tasks would present an intractable problem, we partition the problem into groups. This is similar to the planning hierarchy discussed in Section 2.3.1, except Zhao's work focuses on partitioning tasks in the squadron-level and passing results to the vehicle-level; whereas this thesis focuses on planning in the theater-level and passing results to the squadron-level. Zhao's algorithm would have to be extended to model more than one squadron to be applied to the EBO model which we present in Section 3.1.

Zhao presents a sweep algorithm. It groups tasks in terms of both their geometric locations and mission values (a value assigned to the task by planners representing its importance or the benefit received by accomplishing the task.)

The algorithm starts by generating two sorted task lists: all the tasks sorted in increasing polar coordinate angle with respect to the base, and all the tasks sorted in decreasing mission value. It divides the tasks into groups of approximately equal mission value for each participating vehicle. It then sweeps in a forward direction, sorting the tasks in an increasing order of the polar coordinate angles.

Beginning with the smallest angle, the algorithm adds the tasks to the current group according to the polar angle list and adds the task's mission value to that group, as shown in the Figure 2-6. When the total mission value for the group reaches the total task mission value divided by K, the algorithm completes the current group and starts a new group. This process continues until completing all the tasks.

The algorithm then checks the vehicle travel distance constraint within each group. Using the algorithm uses a fast routing algorithm to estimate the length of a tour, which provides an upper bound for the tour length to check for feasibility. If the tour is infeasible, then there are two possible

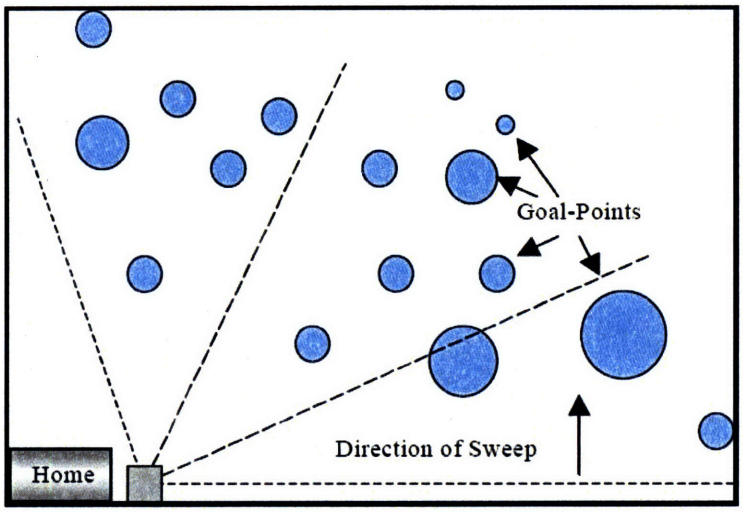


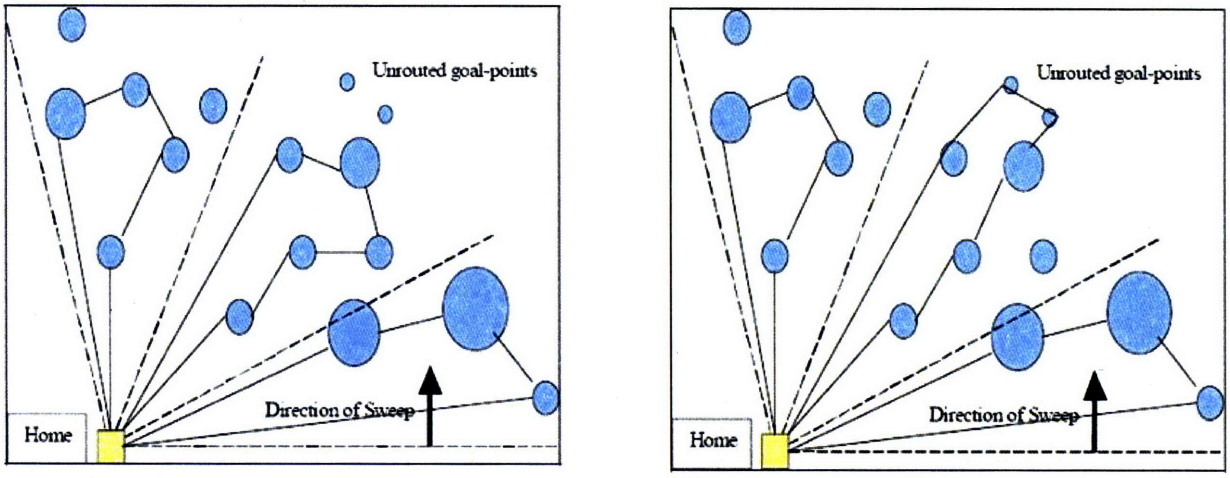
Figure 2-6: Sweep Algorithm (3 Groups)[61]

outcomes and the algorithm will handle them as follows.

- The task the furthest distance from home has the smallest value. In this case, the algorithm will remove the furthest task and continue to check the tour length constraint.
- The tour has at least one task with smaller value than the furthest one. In this case, the algorithm labels all such tasks as a test group, removes the task with the least mission value, and checks the tour length constraint. It continues removing tasks from the test group until it finds a feasible solution. Any goal points left out of the solution are added back in (considered un-routed) and considered later.

If a partitioned group contains un-routed tasks, the routed tasks do not necessarily represent the optimal routing for the group. To improve the total mission value of each group, Zhao implements a task swapping procedure between the un-routed tasks and the routed tasks within the same group. An example of task swapping is shown in Figure 2-7. The algorithm checks for tour length feasibility after the swapping. If the resulting solution is feasible, then the solution is kept. Otherwise, it reverses the swap. The process continues until no task with smaller mission value is available to swap.

Zhao's sweep algorithm partitions a large multi-vehicle routing problem into K smaller routing problems, and outputs routing solutions to each of these. If extended to handle multiple squadron's



(a) Situation A: Before Swapping

(b) Situation B: After Swapping

Figure 2-7: Example of Task Swapping[61]

Zhao's sweep algorithm, might be a useful method to partition tasks to squadrons after the appropriate tasks have been selected using an EBO model.

2.6 Robust Optimization Literature

The classic paradigm of optimization using mathematical programming assumes that all input data is known with certainty. If the data takes on values different from the nominal case, the solution might no longer be optimal and might not be feasible. "In real world applications of linear programming, one cannot ignore the possibility that a small uncertainty in the data can make the usual optimal solution completely meaningless"[12]. UAV mission planning using math programming is no exception.

There are many proposed methods and definitions for robust optimization. Robust optimization objectives might be similar to the following: maximizing an objective function subject to guaranteeing feasibility with some probability, maximizing an objective function while minimizing variance in the objective function, and maximizing the protection of the solution subject to a specified acceptable objective function level. Each considers uncertainty in the input data differently and can model it in different parts of the problem.

In the case of mission planning for UAVs, we define robust optimization as *creating a plan that remains feasible for a longer period of time during execution than the plan obtained when only*

deterministic input data is used. This can be done by using several of the objectives proposed in the literature. We review them here.

2.6.1 Protecting Against the Worst-Case

In 1973, Soyster proposed a linear optimization model to construct a solution that is feasible for all data belonging to a convex set[51]. As a result, Soyster's formulation gives ultra-conservative results, and according to Ben-Tal and Nemirovski, it gives up too much from the optimal solution to ensure feasibility[12]. Soyster starts with the general linear optimization problem given in (2.1)-(2.3), where there exists column-wise uncertainty in $\boldsymbol{A}$ (i.e. each column of $\boldsymbol{A}$ belong to a convex set K_j):[1]

$$\text{maximize} \quad \sum_{j \in J} c_j x_j \tag{2.1}$$

$$\text{subject to} \sum_{j \in J} a_{ij} x_j \leq b_i \quad \forall i \in I, \ \boldsymbol{a}_j \in K_j, \tag{2.2}$$

$$x_j \geq 0, \quad \forall j \in J \tag{2.3}$$

Soyster shows that this problem is equivalent to pushing all values in $\boldsymbol{A}$ to their worst-case values ($\bar{a}$) and re-solving, in which case the formulation looks as follows:

$$\text{maximize} \quad \sum_{j \in J} c_j x_j \tag{2.4}$$

$$\text{subject to} \sum_{j \in J} \bar{a}_{ij} x_j \leq b_i \quad \forall i \in I, \ \boldsymbol{a}_j \in K_j, \tag{2.5}$$

$$x_j \geq 0, \quad \forall j \in J \tag{2.6}$$

The Soyster method can also be applied to uncertainty in the c vector. We can push the values of the c vector in (2.1)-(2.3) to their-worst case values, to get a solution that maximizes the worst-case situation. When applied to mission planning, this formulation produces plans that assign tasks that guarantee a certain value or better.

[1] A note on notation: in this thesis we represent matrices as capitalized, bold, and italic ($\boldsymbol{A}$); vectors as lowercase, bold, and italic ($\boldsymbol{c}$): random variables as capitalized and non-italic (B); individual values as lowercase and italic (x): sets of data as capitalized and italic (J); and indices as lowercase, italic subscripts (j).

Consider the value of performing task j as having some expected benefit c_j. If there is uncertainty in c_j, we can model that uncertainty in different ways. First we might assume that c_j falls into some window, with maximum and minimum possible values. Second we might assume that c_j is generated from some probability distribution with standard deviation σ. In this case, we might seek a maximum benefit solution with the least expected variability. Bertuccelli[17] has recently proposed robust planning formulations for UAVs in this way. Bertuccelli's planning algorithm focuses on the squadron-level of the planning problem, but his application of robust optimization techniques to UAV planning makes it worth summarizing here.

2.6.2 Modified Protection against the Worst-Case Approach

Bertuccelli proposes a method for robust task assignment for heterogeneous UAVs using a Modified Soyster Method[17]. The method allows for analysis of uncertainty in the objective function where each benefit coefficient c_{ki} has its own standard deviation σ_{ki}. The robust objective function is the benefit coefficient minus its respective standard deviations times a scalar μ which represents a modeler's risk aversion rate. The formulation takes the form:

$$\max_x J_k = \sum_{i=1}^{|N_T|} (\bar{c}_{ki} - \mu\sigma_{ki}) x_{ki} \tag{2.7}$$

$$\text{subject to:} \sum_{i=1}^{|N_T|} x_{ki} = |N_v| \tag{2.8}$$

$$x_i \in 0, 1 \tag{2.9}$$

$|N_T|$ is the number of targets, $|N_v|$ is the number of vehicles, and x_{ki} is a binary variable representing whether a vehicle is assigned to target i at time k.

Bertuccelli enhanced this formulation by modifying it to also account for the ability to reduce the σ_{ki} by assigning an ISR vehicle to perform reconnaissance on a target. If an ISR vehicle performs reconnaissance on a target, it eliminates the penalty of σ_{ki}. This modified vehicle formulation looks

like:

$$\max_{x,y} J_k = \sum_{i=1}^{|N_T|} (\bar{C}_{ki} - \mu\sigma_{ki})x_{ki} + \mu\sigma_{ki}y_{ki} \tag{2.10}$$

$$\text{subject to: } \sum_{i=1}^{|N_T|} x_{ki} = |N_{VS}| \tag{2.11}$$

$$\sum_{i=1}^{|N_T|} y_{ki} = |N_{VR}| \tag{2.12}$$

$$x_{ki}, y_{ki} \in 0, 1 \tag{2.13}$$

x_{ki} is 1 if a strike vehicle is assigned to target i in time k. y_{ki} is 1 if a reconnaissance vehicle has visited target i in time k and 0 otherwise. $|N_{VS}|$ is the number of strike vehicles, and $|N_{VR}|$ is the number of reconnaissance vehicles.

This formulation does not consider cooperation between a strike and ISR vehicle. For example, it would be beneficial for the reconnaissance vehicle to do more than just update the knowledge of the environment by visiting the most uncertain targets. Since the ultimate goal is to achieve the best possible mission score, the reconnaissance mission should be modified to account for the strike mission, and vice versa. This can be achieved by coupling the mission objectives of decreasing the variability in the benefit received by striking a target by visiting the target with an ISR vehicle and receiving benefit by striking targets.

If an ISR vehicle visits a target before the target is struck, the target's score will remain the same, but its uncertainty (given by σ) will decrease from σ_k to $\sigma_{k+1|k}$. However, x and y can be

coupled into a separate variable v to form the following linear formulation:

$$\max_{x,y} J_k = \sum_{i=1}^{|N_T|} (\bar{c}_{ki} - \mu\sigma_{ki})x_{ki} + \mu(\sigma_{ki} - \sigma_{k+1|ki})v_{ki} \tag{2.14}$$

$$\text{subject to: } \sum_{i=1}^{|N_T|} x_{ki} = |N_{VS}| \tag{2.15}$$

$$\sum_{i=1}^{|N_T|} y_{ki} = |N_{VR}| \tag{2.16}$$

$$x_{ki}, y_{ki}, v_{ki} \in 0, 1 \tag{2.17}$$

$$v_{ki} \leq x_{ki} \tag{2.18}$$

$$v_{ki} \leq y_{ki} \tag{2.19}$$

$$v_{ki} \geq x_{ki} + y_{ki} - 1 \tag{2.20}$$

The formulation can further be improved by assigning any reconnaissance vehicles not used in the coupling to other targets with the greatest standard deviations. Doing so, decreases the uncertainty in the problem, enabling better decisions in future iterations. This can be done by adding an extra term to the objective function:

$$\max_{x,y} J_k = \sum_{i=1}^{|N_T|} (\bar{c}_{ki} - \mu\sigma_{ki})x_{ki} + \mu(\sigma_{ki} - \sigma_{k+1|ki})v_{ki} + K\sigma_{ki}(1 - x_{ki})y_{ki} \tag{2.21}$$

Where for a small K, the cost function makes the strike objective primary, while the small weight in the latter part of the formulation causes the assigning of left-over reconnaissance vehicles to be a secondary objective.

2.6.3 Chance-Constrained Programming

In 1959, Charnes and Cooper introduced a method for robust optimization called chance-constrained programming[21]. Charnes and Cooper were among the first to address the problem of robust planning under uncertainty. Because uncertain input can lead to constraint violations once a plan is put into operation, they decided to try to regulate the chance that any constraint is violated, hence the name "chance-constrained programming." They define chance-constrained programming as follows: "Select certain random variables as functions of random variables with known distributions in such

a manner as to maximize a functional of both classes of random variables subject to constraints on these variables which must be maintained at prescribed levels of probability"[21]. We give a detailed description of several of Charnes and Coopers chance-constrained models and apply them to the EBO Model in Section 3.2.1.

Since Charnes and Cooper initially introduced chance-constrained programming in the early 1960s, researchers have applied it to numerous applications, such as research and development projects [26], networks [25], and critical path analysis [23].

Unfortunately, chance-constrained programming encounters serious computational issues as we try to add multiple uncertain coefficients per constraint. For most of their models, Charnes and Cooper limited uncertainty to one random variable per constraint (right-hand-side value). However, if we try to add uncertainty into the left-hand-side, we must calculate a joint probability distribution for all uncertain coefficients in a constraint in order scale the uncertain values in the constraints to be feasible with some probability. Miller and Wagner discuss methods of having multiple random variables per constraint generated by a multinomial distribution [45]. However, most chance-constrained programming has been limited to having uncertainty only in the right-hand-side, due to the difficulties of associated with multiple random variable.

2.6.4 Bertsimas/Sim

In their 2001 paper, "The Price of Robustness", Bertsimas and Sim propose a robust optimization formulation with the intention of avoiding the overly conservative tendencies of the Soyster formulation while remaining in the linear domain[14]. Bertsimas and Sim do not assume a probability distribution for uncertain coefficients; rather they only assign a windows in which the uncertain coefficients can vary. The Bertsimas/Sim model is a Mixed Integer Linear Program (MILP) that protects against parameterized number Γ_i of uncertain coefficients going to their worst case value. Bertsimas and Sim show how this formulation can be applied to a portfolio optimization problem, a knapsack problem, supply chain management [16], and network flows [13, 15]. We describe this model in detail and apply it to the EBO Model in Section 3.2.4.

Bertsimas and Sim establish several probability bounds for how often a constraint will be violated if all coefficients a_i vary according to any symmetric distribution in the range by $[a_i - \hat{a}_i, a_i = \hat{a}_i]$. Although we do not use these bounds in our analysis, they are worth mentioning here to show how

uncertain data behaves in the Bertsimas/Sim model. The tightest of these bounds is given by the following:

If $n_{ij}, j \in J_i$ are independent and symmetrically distributed random variables in $[-1, 1]$, then

$$Pr\left(\sum_{j \in J_i} \gamma_i n_{ij} \geq \Gamma_i\right) \leq B\left(n, \Gamma_i\right) \tag{2.22}$$

where

$$B\left(n, \Gamma_i\right) = \frac{1}{2^n}\left\{(1-\mu)\sum_{l=\lfloor\nu\rfloor}^{n}\binom{n}{l} + \mu\sum_{l=\lfloor\nu\rfloor+1}^{n}\binom{n}{l}\right\} = \frac{1}{2^n}\left\{(1-\mu)\sum_{l=\lfloor\nu\rfloor}^{n}\binom{n}{\lfloor\nu\rfloor} + \mu\sum_{l=\lfloor\nu\rfloor+1}^{n}\binom{n}{l}\right\} \tag{2.23}$$

Where $n = |J_i|, \nu = \frac{\Gamma_i + n}{2}$ and $\mu = \nu - \lfloor\nu\rfloor$ This bound can be difficult to calculate and can be approximated using:

$$B\left(n, \Gamma_i\right) \leq (1-\mu)C(n, \lfloor\nu\rfloor) + \sum_{l=\lfloor\nu\rfloor+1}^{n}\binom{n}{l} C(n, l) \tag{2.24}$$

where

$$C(n,l) = \begin{cases} \frac{1}{\sqrt{2\pi}} if\ l = 0\ or\ l = n, \\ \frac{1}{\sqrt{2\pi}}\sqrt{\frac{n}{(n-l)l}}\exp\left(n\log\left(\frac{n}{2(n-l)}\right) + l\log\left(\frac{n-l}{l}\right)\right) otherwise. \end{cases} \tag{2.25}$$

For $\Gamma_i = \Theta\sqrt{n}$,

$$\lim_{n\to\infty} B\left(n, \Gamma_i\right) = 1 - \Phi(\Theta), \tag{2.26}$$

$$\text{and } \Phi(\Theta) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\Theta}\exp\left(-\frac{y^2}{2}\right) dy \tag{2.27}$$

The proof for the above probability bounds can be found in Bertsimas and Sim "The Price of Robustness"[14].

2.6.5 Ellipsoidal Model of Uncertainty

Ben-Tal, Nemirovski, and El-Ghaoui et al.[9, 32, 10, 11, 12, 33] have made significant advances in robust optimization using an ellipsoidal model of uncertainty. There are many motivating factors for assuming this type of uncertainty set, the principal one being that measurement errors are typically distributed in an ellipsoid centered at the mean of the distribution. With this model of uncertainty, linear programs have robust counter-parts that are conic quadratic problems. Conic problems are very computationally expensive, especially when they have integer constraints. Since realistic mission planning problems are very large, we do not focus on robust optimization methods which do not have linear objective functions.

2.7 Summary

The United States Military is clearly moving toward EBO as their primary planning paradigm. In the past, connecting objectives to actual plans has been a major challenge especially in planning models. Our goal in making robust plans is to create plans that last longer than they would otherwise, and not to create some guarantee that we will be feasible. We are also seeking to solve very large theater-level problems. For these reasons, we chose to apply chance-constrained programming and the Bertsimas/Sim formulation to an effects-based operations framework to create a theater-level UAV mission planner. In the following chapter, we present an EBO framework for UAV mission planning. We apply this framework in a math programming mission planner. We then present two options chance-constrained programming and the Bertsimas/Sim Formulation, for making robust plans.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 3

EBO Model Formulations

As discussed in Chapter 2, EBO (Effects-Based Operations) are "operations conceived and planned in a systems framework that consider the full range of direct, indirect, and cascading effects-effects that may, with different degrees of probability, be achieved by the application of military, diplomatic, and economic instruments"[28]. The Department of Defense and particularly the Air Force are currently in the process of incorporating EBO into their planning doctrine. Both recognize EBO as an effective approach to ensure that plans focus on tasks that will cause effects that will eventually cause end-objectives. In this chapter, we present a method of applying EBO to the theater-level planning problem, which we call the *EBO Framework*. We present a Mixed-Integer Linear Program (MILP) formulation for the EBO approach to the theater-level UAV planning problem. For ease of terminology, we will refer to this model as the *EBO Model*. We present several versions of the EBO Model. We begin with the *Deterministic EBO Model* and then we present several robust optimization versions: *Chance-Constrained EBO Model*, *Extended Chance-Constrained Model (ECCF)*, the *Bertsimas/Sim EBO Model*, and the *Delta Formulation*.

3.1 EBO Model Description

Our objective for the theater-level UAV planning problem is to assign tasks to squadrons in order to maximize the benefit received by executing the plan. An EBO approach can be used to create a framework in which we evaluate the quality of a plan based on its ability to cause effects. In the following sections, we describe our implementation of an EBO approach to the theater-level UAV

planning problem. This problem can be solved using many different methods. As we discussed earlier, we avoid dynamic programming and non-linear math programming methods because they tend to be intractable for large problems. We chose to formulate this model using a Mixed Integer Linear Program (MILP), because it is tractably solvable to a provable bound of optimality. We call the MILP of the EBO approach to the theater-level planning problem the *EBO Model*.

End objectives usually consist of non-task-specific terminology, such as destroying the will of the enemy to continue fighting or eradicating enemy forces from a particular region. Usually, there exist many effects that eventually cause the objectives. These effects are often more specific and can be caused in a shorter time than of end objectives. We cause effects by performing tasks, which are individual actions. If tasks are accomplished, they will cause effects, which we hope achieve objectives.

Often, there are several different ways to cause each effect. For instance, if our desired effect is denying enemy forces the ability to move into a city, we could destroy all the enemy forces, blockade all routes into the city, directly defend the city, or do some combination of these tasks. Although they cause the same effect, each option involves very different tasks. Usually, desired effects require accomplishing many tasks that can be grouped into sets. Thus, we assign *task-sets* to squadrons in such a way as to maximize benefit received from the resulting effects.

We propose an *EBO Framework* that connects effects to the different options for achieving the effects. We call these options task-sets, which consist of one or more individual tasks. The EBO Framework is shown in Figure 3-1.

Figure 3-1 shows only three levels: effects, task-sets, and tasks. As discussed in Section 2.2 and shown in Figure 2-4, there can be more than one level of effect and objectives. Effects that are directly caused by tasks are usually referred to as direct effects. Effects that are caused by direct effects are usually referred to as indirect effects. For simplicity of discussion, demonstration, and tractability, the EBO Model presented in Formulation (3.19)-(3.21) only incorporates the three levels shown in Figure 3-1 (direct effects, task-sets, and individual tasks). It can be extended to model indirect effects for all levels of objectives. We can do this by adding indirect effects and objectives and connecting them to the direct effects, similarly to how effects in Figure 3-1 are connected to task-sets. We demonstrate this in Figure 3-2.

We consider task-sets to be "all-or-nothing" when calculating the value of assigning them in

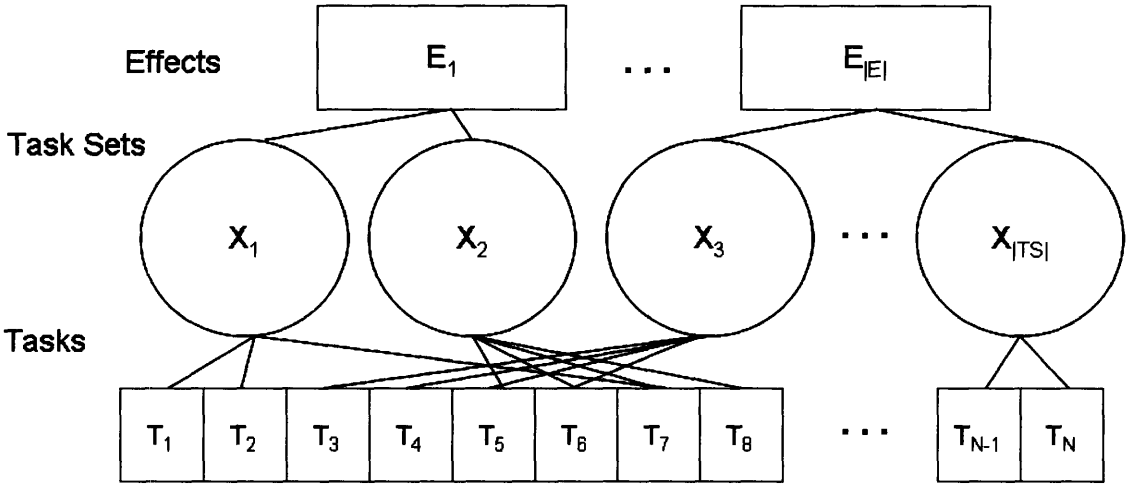


Figure 3-1: EBO Framework

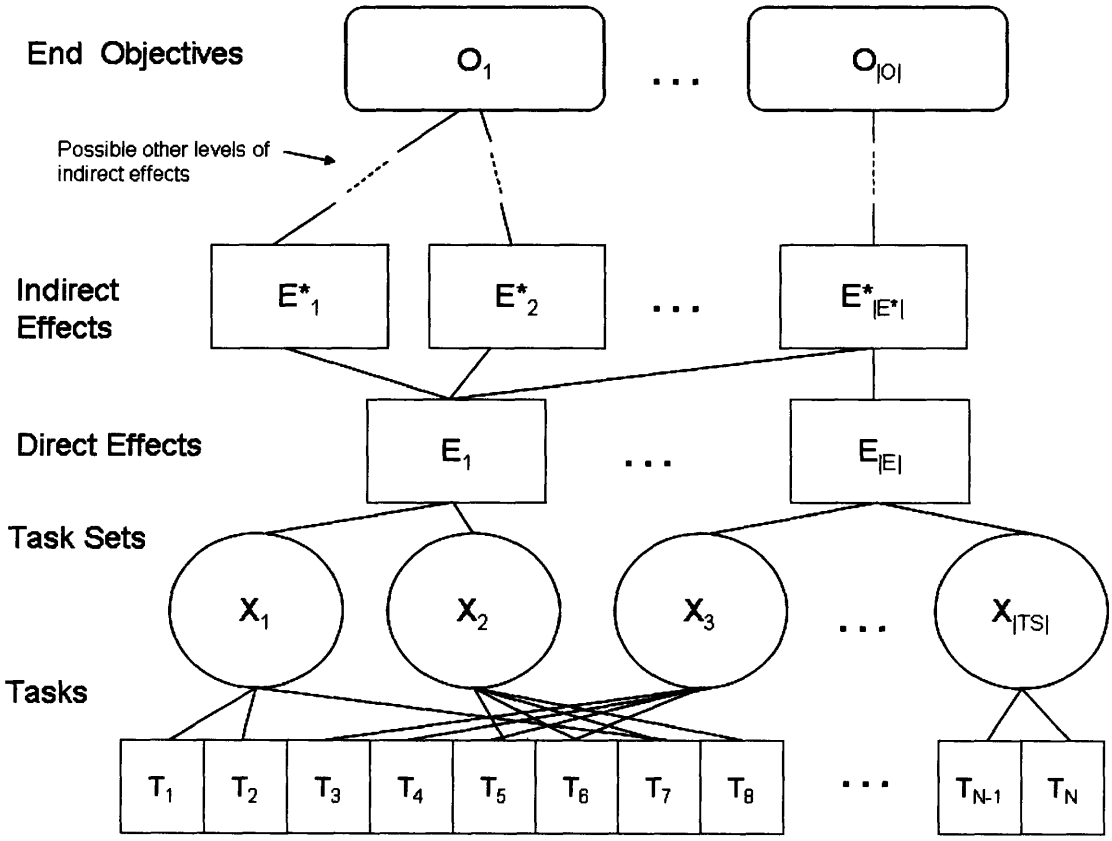


Figure 3-2: EBO Framework Extended to Encompass Multiple Effect Levels

the plan, meaning all tasks in the task-set must be assigned or we do not achieve any value. We do this because the value is derived from the effect caused by performing the tasks and not from performing the tasks themselves. If we cannot perform all the tasks of a particular task-set, we might not cause the desired effect. For instance, suppose the desired effect is to prevent enemy forces from advancing toward a neutral village, and the task-set with which we choose to cause the effect is to destroy a bridge that the enemy must cross. The tasks involved are performing ISR of the bridge, striking the bridge, and then performing BDA to ensure that the bridge is destroyed. If we fail to assign the strike task but assign the other two tasks, we will receive no value because the bridge will not be destroyed; we cannot cause the effect. If we fail to assign the ISR task, we might still be able to destroy the bridge but our chances of missing or encountering unexpected risks greatly increase. Thus, in the EBO model we require that all tasks in a task-set must be assigned to achieve value.

However, assigning all of the tasks in a task-set does not guarantee that we will cause an effect. Even if there is enough estimated capacity to assign all the tasks in a task-set, when the plan is executed it is possible that attempts to execute tasks will fail. Furthermore, the relationship between tasks and effects is estimated by strategists. Determining what tasks will actually cause subjective effects is difficult and prone to errors. Thus, each task-set also has an estimated probability that if all of its tasks are assigned it will cause the linked effect. We then maximize the expected value of achieving effects. Thus, assigning all tasks in a task-set achieves the value of the probability of causing the effect times the value of the effect; whereas failure to assign all the tasks in a task-set achieves no value. We will further elaborate on this and the case where multiple task-sets that cause the same effect are assigned in Section 3.1.1.

3.1.1 EBO Model Inputs

The EBO Framework of Figure 3-1 and subsequently the EBO Model require inputs about effects, detailed information about the current state of the world, and possible task requirements. The input data can be divided into three categories: effect information, squadron capabilities, and task information.

Effects Information

When creating a plan, the planners have end-objectives they want to achieve. There are many lower-level effects, which cascade to the ultimate end-objective. We decided to focus on the effects that a human planner might want to achieve in a short time-period, perhaps a day or couple of days. These effects might cascade to higher-level effects later; however, to maintain tractability and stay within the reasonable expectations of the effects UAVs can cause, we decided to focus on lower-level effects.

Each effect has an expected benefit value and is linked to a group of task-sets. Each task-set contains a list of tasks, which, if assigned and the plan is executed, will cause the linked effect with some probability, $\Pr_j$. There can be more tasks in the scenario than can be accomplished with available resources.

We assume that a plan can include more than one task-set that can cause the same effect. If more than one task-set linked to the same effect is assigned, we increase the probability that the effect is caused when the plan is executed. We assume task-sets cause effects independently. For instance, assume task-set k has probability $\Pr_k$ of causing effect e and task-set $k+1$ has probability $\Pr_{k+1}$ of causing effect e. If we assign both task-sets we now have probability $1-(1-\Pr_k)(1-\Pr_{k+1})$ of causing effect e.

In most cases, the assumption of independence is not valid. There are some situations where performing the tasks of a particular task-set can eliminate the ability to perform another task-set. For instance, if we want to cause the effect of preventing enemy forces from entering a neutral village we could destroy the primary route that the enemy takes into the village, or we could patrol that route and confront the enemy on the route. If the route is destroyed, then patrolling it is impossible. Thus, performing the first option prevents doing the second option. However, we will use a probability value that assumes task-set independence as an approximation despite the fact that in practice the task-sets might not be independent.

The actual effect data that we input includes a list of all effects ($e \in E$) and their associated benefit values (c_e). We input a list of all task-sets including which tasks belong to each set, which effect the task-set causes, and the probability that the task-set will cause the effect. This information comprises the top two levels of Figure 3-1.

In the EBO Model, we group combinations of task-sets that accomplish the same effect into a

task-set group (TSG), which is an enumeration of all task-set combinations that cause an effect. If there are two task-sets that can cause effect e, there will be three TSG variables: one for doing task-set 1, one for doing task-set 2, and one for doing both task-set 1 and 2. We model each TSG as a binary decision variable (x_j). We have two reasons for enumerating combinations of task-sets, even though it increases the number of decision variables by $\sum_{n=1}^{N} \binom{N}{n}$, where N is the number of task-sets that cause a particular effect. First, we expect that 3 or 4 different, realistic options for accomplishing a particular effect will suffice in most cases and thus N will usually be small. Second, we want to preserve linearity in the objective function. If we did not group task-sets that accomplish the same effect into TSGs, we would have an objective function that looks as follows:

$$\max \sum_{e \in E} c_e (1 - \prod_{k \in K | k \in K_e} (1 - y_k Pr_k)), \tag{3.1}$$

where c_e is the benefit received by causing effect e, K is the set of all task-sets, Pr_k is the probability that task-set k causes effect e, and y_k is a binary decision variable equal to 1 if we include task-set k in the plan and 0 if we do not. K_e is the set of all task-sets that cause effect e. This non-linear objective function is significantly more difficult to solve than linear objective functions.

If we group combinations of task-sets that cause the same effect into their own decision variables we can preserve linearity. For instance if we have task-sets 1 and 2 in a scenario with one effect, instead of having decision variable y_1 and y_2, we can have binary decision variables x_1, x_2, and x_3; where x_1 and x_2 represent doing only task-set 1 or 2 respectively and x_3 represents doing both task set 1 and 2. Now the objective looks as follows:

$$\sum_{e \in E} \sum_{j \in J | j \in J_e} c_e x_j Pr_j, \tag{3.2}$$

where c_e is the benefit received by causing effect e; J is the set of all TSGs; Pr_j is the probability that TSG j causes effect e; and x_j is a binary decision variable, 1 if we include TSG j in the plan and 0 if we do not. J_e is the set of all TSGs that cause effect e. For the linear version, the probability of a TSG causing an effect will need to be computed in advance; however, this calculation is simple due to the independence assumption.

Figure 3-3 shows the relationship between task-sets and TSGs. Notice that x_1 and x_2 represent doing just the individual task sets 1 and 2 respectively. Whereas x_3 represents doing both task set

1 and 2. The probability of achieving effect 1 when we do x_3, is the combined probability of success if both task-sets, 1 and 2, are assigned.

Squadron Capability Information

In order to determine how many effects we can achieve, we need to measure the capacity of our squadrons to perform tasks. Because our goal is not to create the actual routes for individual aircraft, but rather to assign tasks to a squadron as a whole, we want to base our decisions on aggregate measures of the squadron's ability. We assume that all of our UAVs are stationed within the theater to avoid modeling the extremely long travel times of traveling inter-theater.

We decided to model two types of UAVs. We have ISR UAVs (for example a Global Hawk), which we consider to be very good at performing ISR and have long range. We also model strike UAVs (possibly a predator fitted with hell-fire missiles), which are able to carry some number of munitions and have a limited ISR capability.

For each squadron ($s \in S$), located at $[lat_s, lon_s]$ we assign a certain number of ISR UAVs (ρ_s) and strike UAVs (υ_s). We use a parameter for the capability for each type of aircraft to perform tasks: the number of ISR tasks an ISR UAV can perform in a time-period (γ), the number of ISR tasks a strike UAV can do in a time-period (λ^{isr}), and the number of strike tasks a strike UAV can do in a time-period (λ^{st}). Based on these numbers we determine the squadron's aggregate capacity to perform tasks.

Task Information

We decompose the input task-sets to enumerate all component tasks ($i \in I$). For task data, we need to know what type of task is being performed (k_i); the task's location $[lat_i, long_i]$; how many resources are required to complete the task (r_i) (i.e. how many vehicles must cooperate to perform the task); the number of time-periods the task spans (q_i); the task precedence relationship (i, i'), and the set of time-periods t in which we can assign task i, T_i.

We model tasks as being performed during a time-period. We do not care when the task is accomplished within the time-period, because we leave the specific route planning to the squadron-level planner. We assume an aircraft can perform multiple tasks in one time-period. We regulate the number of tasks that can be performed in each time-period with the parameters: λ^{st}, λ^{isr}, and

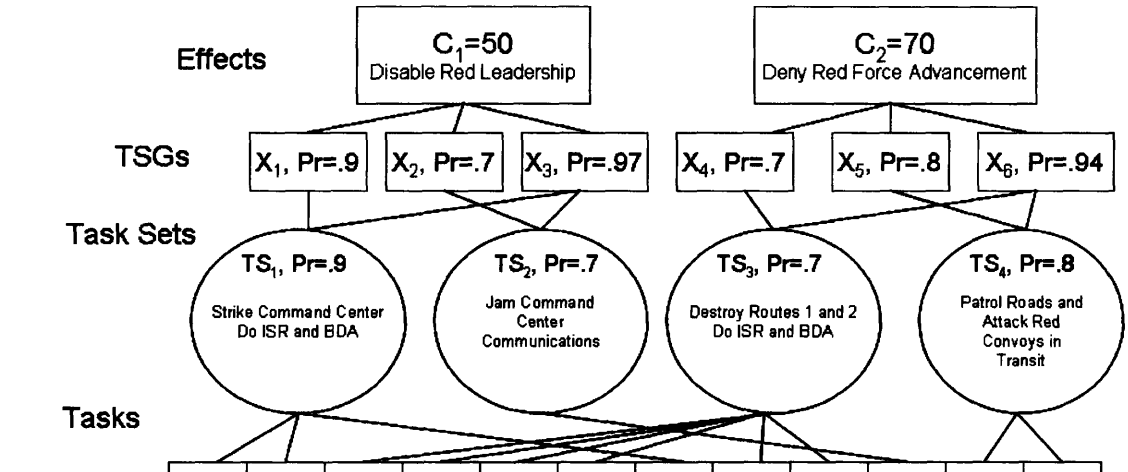
γ. Our nomenclature holds for any length of time being assigned as the time-period. For instance, we could create a plan for the next week where each time-period represents a day, or a plan for the next day where each time-period represents an hour. Obviously, we would have to adjust the λ^{st}, λ^{isr}, and γ parameters appropriately given the time-period.

We divide tasks into four types: ISR, strike, loiter tasks for strike UAVs, and loiter tasks for ISR UAVs. ISR and strike tasks can be accomplished quickly. They include ISR tasks of just a single flyover, or strike tasks where munitions are dropped and the task is completed. Loiter tasks are those that require extended mission time (long enough to occupy a UAV for a whole time-period), such as scanning a large area, or patrolling a road. We consider loiter tasks as keeping a UAV busy for an entire time-period, whereas we can accomplish multiple other tasks in a single time-period. We can also have loiter tasks that last longer than one time-period, as indicated by q_i, the number of time-periods that the loiter task spans.

Tasks also have specified time-windows. A task cannot be assigned outside of this time-window. Time-windows match up with the time-periods. For instance if there are 7 time-periods in the scenario, we might set the time-window for a particular task to be any of the first 3 time-periods. We indicate the time-windows by T_i, which is the set of time-periods t in which we can assign task i.

Many tasks require sequencing with other tasks. For instance, an ISR task may be required before a strike, which may be followed by another ISR task to perform battle damage assessment (BDA). We create a precedence value i' for each task i. The precedence value, i', identifies a task that we have to assign at the same time-period or later time-period as task i. If task i has no precedence relationship, we set i' equal to i. When creating a scenario, we must ensure that the time-windows allow tasks to be feasibly assigned according to their precedence relationships.

Figure 3-3 shows an example of a very small scenario put into the EBO Framework with the input data as described in this section. The scenario models two effects in a situation where red forces are expected to advance against a neutral village and our objective is to prevent them from doing so.



Task ID	1	2	3	4	5	6	7	8	9	10	11	12
Name	Recon Command Center	BDA Command Center	BDA Route 1	BDA Route 2	Recon Route 1	Recon Route 2	Strike Command Center	Strike Route 1	Strike Route 2	Jam Command Center	Patrol Route 1	Patrol Route 2
Type (Prec) [Lat, Lng] R D	ISR (7) [2.4, 5.7] 1 1	ISR (2) [2.4, 5.7] 1 1	ISR (3) [3.7, 6.2] 1 1	ISR (4) [2.5,5.0] 1 1	ISR (8) [3.7,6.2] 1 1	ISR (9) [2.5,5.0] 1 1	Strike (2) [2.4,5.7] 2 1	Strike (3) [3.7,6.2] 2 1	Strike (4) [2.5,5.0] 4 1	Loiter/ISR (10) [2.4,5.7] 2 3	Loiter/stk (11) [3.7,5.9] 2 2	Loiter/stk (12) [2.5,5.0] 2 2
Time Windows: $T_{i,1}$	1	1	0	0	1	1	1	1	1	1	0	0
$T_{i,2}$	1	1	1	1	1	1	1	1	1	1	1	1
$T_{i,3}$	0	0	1	1	0	0	0	1	1	1	1	1

$T_{i,t}$=1 if task i allowed in time t

Figure 3-3: Small EBO Model Example

3.1.2 EBO Model Objective and Constraints

The objective of the EBO Model is to maximize the benefit received by causing effects, subject to constraints on the number of tasks we can perform, based on the capabilities of the squadrons, and the demands of the tasks. We assume that in most scenarios, there are more desired effects than we have capacity to cause. We consider TSGs to be all-or-nothing similar to how we described task-sets in Section 3.1. Thus, the plan is constrained such that a TSG is assigned only if all of its tasks can be accomplished. Our primary objective is to maximize the benefit of effects, but we also want to accomplish a second objective of minimizing the distance squadrons have to travel in order to perform the tasks that cause the effects. For Formulation (3.3)-(3.15), the objective function value increases for effects we expect to cause and is slightly penalized for the distance the UAVs have to travel to accomplish the required tasks. Maximizing the effects value is the primary goal and minimizing the distance traveled is the secondary goal. As such, the value of the effects c_e is weighted more heavily than the penalty for the distance to the tasks.

We constrain task assignment in several different ways: tasks cannot be assigned beyond the squadron's capacity to perform the specific type of task; tasks must be assigned within their time-windows; tasks must be assigned according to their precedence order; and tasks that must be performed in a precedence order must all be assigned to the same squadron.

We cannot assign tasks beyond a squadron's capacity for a particular time-period and type of task. We define capacity to perform different types of tasks as the number of UAVs in the squadron times the average number of those kind of tasks we think each UAV can accomplish in one time-period. These constraints are task-type specific and require a different constraint for each of the four types of tasks: ISR, strike, loiter tasks for strike UAVs, and loiter tasks for ISR UAVs. Furthermore, if we assign loiter tasks for a particular squadron and time-period, we have to reduce the squadron's capacity to perform other non-loiter tasks in that time-period.

As discussed, in 3.1.1 each task in the plan must be assigned in an allowed time-window. Also discusses in 3.1.1, certain tasks must be assigned according to a precedence order. The set of tasks that must be assigned according to a precedence order is called a string.

We enforce that the same squadron must perform all tasks that belong to a string. The theater-level plan is designed to pass assignments to the squadron-level where UAV route planning is done. This hierarchy divides the very large planning problem into smaller sections. The benefit of the

hierarchy is weakened if squadrons have to cooperate in the squadron-level planning process to ensure precedence relationships are maintained. We ensure that the squadron-level planners do not have to plan with other squadrons by restricting that strings are assigned to only one squadron.

3.1.3 EBO Model Outputs

The EBO model solution includes a value for the expected benefit received by causing effects, which effects we intend to cause by executing the plan, and what tasks we assign that will cause these effects. We represent which effects in the plan are assigned with the binary decision variable x_j equal to 1 if TSG j is in the plan and 0 otherwise. We model task assignments with the binary decision variable z_{its} equal to 1 if task i is assigned to squadron s in time-period t and 0 otherwise.

3.1.4 Deterministic EBO Model Formulation

We present a deterministic version of the EBO Model here. We first list the input data, parameters, and decision variables. Then we give the mathematical formulation, followed by explanations of the individual constraints of the formulation.

Input Data

- E: Set of all effects, $e \in E$
- J: Set of all TSGs, $j \in J$
- J_e: Set of all TSGs $j \in J$ that are members of Effect e
- I: Set of all tasks, $i \in I$
- I_j: Set of all tasks $i \in I$ that are members of TSG j
- T: Set of all time-periods, $t \in T$
- T_i: Set of all time-periods $t \in T$ in which task i is allowed to be assigned
- S: Set of all squadrons, $s \in S$
- Pr_j: Probability that TSG j causes effect e

- i': Task i must be assigned in a time-period before or equal to the assigned time-period of the task represented by i'
- k_i: Designates a task's type:
  - ISR (isr)
  - Strike (st)
  - Loiter tasks for ISR UAVs (lisr)
  - Loiter tasks for strike UAVS (lst)
- lat_i, lon_i: Latitudes and longitudes of all tasks
- lat_s, lon_s: Latitudes and longitudes of all squadrons
- d_{is}: The great circle distance in kilometers from task i to squadron s, calculated by the following equation:

$$d_{is} = 6378.8 \arccos\left[\sin(lat_i)\sin(lat_s) + \cos(lat_i)\cos(lat_s)\cos(lon_s - lon_i)\right]$$

Parameters

- c_e: Value achieved by causing effect e
- r_i: Number of UAVs required to complete task i
- q_i: Number of time-periods task i spans
- υ_s: Number of strike UAVs in squadron s
- ρ_s: Number of ISR UAVs in squadron s
- λ^{isr}: Number of ISR tasks a strike UAV can do per time-period
- λ^{st}: Number of strike tasks a strike UAV can do per time-period
- γ: Number of ISR tasks an ISR UAV can do per time-period
- δ: Number small enough to ensure the sum of all task distances from their assigned squadrons times this value is less than the benefit received from any one effect

Decision Variables

- x_j: equals 1 if TSG j is assigned, 0 otherwise
- z_{its}: equals 1 if task i is assigned to squadron s in time-period t

Formulation

$$\text{Max} \sum_{e\in E}\sum_{j\in J_e} c_e Pr_j x_j - \delta \sum_{i\in I}\sum_{t\in T}\sum_{s\in S} d_{is} r_i z_{its} \tag{3.3}$$

$$\text{ST:} \sum_{j\in J_e} x_j \leq 1 \quad \forall\, e \in E \tag{3.4}$$

$$\sum_{i\in I_j}\sum_{t\in T_i}\sum_{s\in S} z_{its} \geq \sum_{i\in I_j} q_i x_j \quad \forall\, j \in J \tag{3.5}$$

$$\sum_{t\in T_i}\sum_{s\in S} z_{its} \leq q_i \quad \forall\, i \in I \tag{3.6}$$

$$\sum_{s\in S} z_{its} = 0 \quad \forall\, i \in I, t \in T | t \notin T_i \tag{3.7}$$

$$\sum_{i\in I|k_i=isr} r_i z_{its} + \gamma \sum_{i\in I|k_i=lisr} r_i z_{its} + \lambda^{isr} \sum_{i\in I|k_i=lst} r_i z_{its} \leq \lambda^{isr} \upsilon_s + \gamma \rho_s \quad \forall t \in T, s \in S \tag{3.8}$$

$$\sum_{i\in I|k_i=st} r_i z_{its} + \lambda^{st} \sum_{i\in I|k_i=lst} r_i z_{its} \leq \lambda^{st} \upsilon_s \quad \forall\, t \in T, s \in S \tag{3.9}$$

$$\sum_{i\in I|k_i=lst} r_i z_{its} \leq \upsilon_s \quad \forall\, t \in T, s \in S \tag{3.10}$$

$$\sum_{i\in I|k_i=lisr} r_i z_{its} \leq \rho_s \quad \forall\, t \in T, s \in S \tag{3.11}$$

$$\sum_{t\in T} t z_{its} \leq \sum_{t\in T} t z_{i'ts} \quad \forall\, i \in I, s \in S \tag{3.12}$$

$$\sum_{t\in T} z_{its} = \sum_{t\in T} z_{i'ts} \quad \forall\, i \in I, s \in S \tag{3.13}$$

$$x_j \in [0,1] \quad \forall j \in J \tag{3.14}$$

$$z_{its} \in [0,1] \quad \forall\, i \in I, t \in T, s \in S \tag{3.15}$$

- (3.3) Objective function: Maximize the benefit received from effects and minimize the distance between tasks and their assigned squadrons.
- (3.4) Restrict that we cannot do more than one TSG j per effect e.

- (3.5) In order for TSG j to be assigned, the correct tasks in j must be assigned. (The sum of all aircraft assigned to tasks that belong to TSG j in allowable time-periods must be greater than or equal to the sum of the total number of tasks times the number of time-periods the task requires in TSG j)

- (3.6) The number of aircraft assigned to task i in allowable time-periods cannot exceed the number of aircraft required to accomplish the task. This constraint ensures the correct tasks are assigned. It prevents us from assigning a task twice in place of another required task.

- (3.7) Tasks cannot be assigned in time-periods that are not allowed. Although this constraint is implied by 3.5, it greatly reduces solve time by cutting the solution space.

- (3.8) The number of ISR tasks assigned to a squadron during a specific time-period cannot exceed the squadron's capacity. (For squadron s, during time-period t, the number of aircraft, not including those assigned to loiter tasks, assigned to ISR tasks does not exceed the product of the number of strike UAVs times the strike UAVs' ISR-capacity plus the number of ISR UAVs times the ISR UAVs' ISR-capacity.)

- (3.9) The number of strike tasks assigned to a squadron in a specific time-period cannot exceed the squadron's capacity. (For squadron s during time-period t, the number of aircraft assigned to strike tasks minus those assigned to loiter tasks does not exceed the number of strike UAVs times the strike UAVs strike-capacity.)

- (3.10) The number of loiter tasks for strike UAVs assigned is less than the number of strike UAVs available. (For squadron s during time-period s, the number of loiter strike tasks does not exceed the number of strike UAVs in squadron s.)

- (3.11) The number of loiter tasks for ISR UAVs is less than the number of ISR UAVs. (For squadron s during time-period t, the number of loiter ISR tasks does not exceed the number of ISR UAVs in squadron s.)

- (3.12) Enforce precedence relationships. (Task i must be assigned in a time-period before or equal to the assigned time-period of task i'.)

- (3.13) Prevent tasks that must be performed in a dependent sequence from being assigned to multiple squadrons. (Task i' must be assigned to squadron s if task i was assigned to s.)

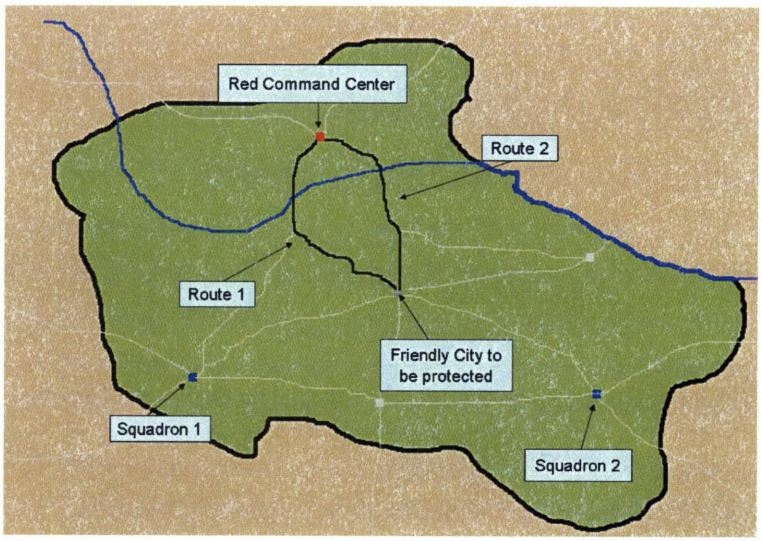


Figure 3-4: Small EBO Scenario Map

- (3.14) x_j is binary.
- (3.15) z_{its} is binary.

3.1.5 Deterministic EBO Model Example and Performance

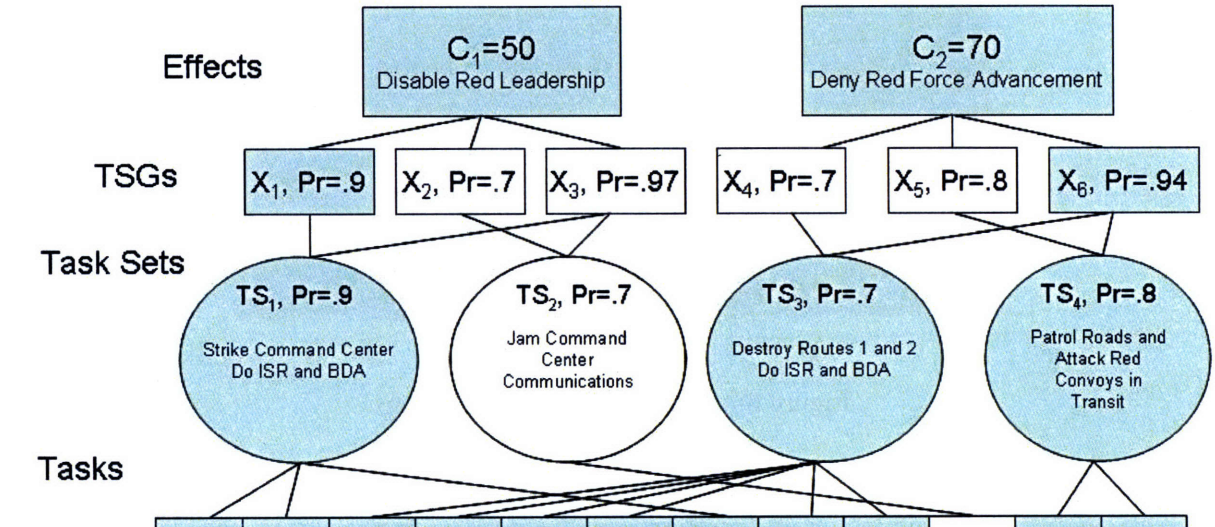
We ran the Deterministic EBO Model on the example scenario from Figure 3-3. The tasks are shown in a hypothetical country in Figure 3-4. Table 3.1 summarizes the parameter data for the scenario not given in Figure 3-5. The remainder of the effect, task-set, task, and time data is in

Table 3.1: Input Data for Small Example Scenario

Number of Squadrons	2
Strike UAVs per Squadron (υ)	2
ISR UAVs per Squadron (ρ)	1
Strike tasks per time-period for strike UAVs (λ^{st})	2
ISR tasks per time-period for strike UAVs (λ^{isr})	2
ISR tasks per time-period for ISR UAVs (γ)	4

Figure 3-5.

We solved the problem using Xpress Mosel Version 1.6.0, Optimizer Version 16.01.02 on a 3.2 GHz Pentium 4 with 1 GB of RAM. The model solved to an optimality gap of less than 0.005% in less than a second.



Task ID	1	2	3	4	5	6	7	8	9	10	11	12
Name	Recon Command Center	BDA Command Center	BDA Route 1	BDA Route 2	Recon Route 1	Recon Route 2	Strike Command Center	Strike Route 1	Strike Route 2	Jam Command Center	Patrol Route 1	Patrol Route 2
Type (Prec) [Lat, Lng] R D	ISR (7) [2.4, 5.7] 1 1	ISR (2) [2.4, 5.7] 1 1	ISR (3) [3.7, 6.2] 1 1	ISR (4) [2.5,5.0] 1 1	ISR (8) [3.7,6.2] 1 1	ISR (9) [2.5,5.0] 1 1	Strike (2) [2.4,5.7] 2 1	Strike (3) [3.7,6.2] 2 1	Strike (4) [2.5,5.0] 4 1	Loiter/ISR (10) [2.4,5.7] 2 3	Loiter/stk (11) [3.7,5.9] 2 2	Loiter/stk (12) [2.5,5.0] 2 2
$Z_{i,1,1}$	0	0	0	0	0	1	0	0	1	0	0	0
$Z_{i,1,2}$	1	1	0	0	1	0	1	1	0	0	0	0
$Z_{i,2,1}$	0	0	0	0	0	0	0	0	0	0	0	1
$Z_{i,2,2}$	0	1	1	0	0	0	0	0	0	0	1	0
$Z_{i,3,1}$	0	0	0	1	0	0	0	0	0	0	0	1
$Z_{i,3,2}$	0	0	0	0	0	0	0	0	0	0	1	0

Z_{its}

Figure 3-5: Small EBO Model Example Results

The solution to the problem in Figure 3-5 is depicted by the shaded effects, task-sets, and tasks. In this case, the plan was able to incorporate all effects but not all task-sets. The objective value received was 109.57, which is the value of causing each effect times the probability we expect to cause each effect, minus the small penalty for the distance each task is away from the assigned squadron, where δ is 0.0001.

The actual task assignment is shown in the lower portion of Figure 3-5. T_{it} indicates whether task i is allowed in time-period t. A shaded z_{its} block indicates if task i is assigned to squadron s in time-period t.

In this scenario, the solution utilizes the strike UAVs at capacity for every time-period. If some of the scenario data were to change (such as, decreasing the strike UAVs capacity to perform tasks or increasing the number of UAVs required to strike some of the targets) the current solution would not be feasible, thus requiring us to re-plan. We analyze and discuss the probability of infeasibility and the extent to which the plan is recoverable in Chapter 4. However, this vulnerability to infeasibility is our motivation for investigating methods for creating robust plans, which we discuss in Section 3.2.

We also tested the Deterministic EBO Model on a realistic-sized theater-level planning problem. This scenario's data is summarized in Table 3.2. We solved the large scenario on the same computer

Table 3.2: Input Data for Realistic Size Scenario

Number of Effects	25
Number of Task-Sets	56
Number of Tasks	550
Number of Time-Periods	7
Number of Squadrons	8
Strike UAVs per Squadron (υ)	5
ISR UAVs per Squadron (ρ)	1
Strike tasks per time-period for strike UAVs (λ^{st})	2
ISR tasks per time-period for strike UAVs (λ^{isr})	2
ISR tasks per time-period for ISR UAVs (γ)	4

as the small scenario given above (Xpress Mosel Version 1.6.0, Optimizer Version 16.01.02 on a 3.2 GHz Pentium 4 with 1 GB of RAM). The solver took 134.1 seconds to solve to an optimality gap of 0.0031%. We consider the roughly 2 minutes solve time to be acceptable for theater-level planning, especially given that the planning time-frame is for long periods of time like days or weeks.

3.1.6 Greedy Algorithm

We want to compare the performance of the EBO Model to an estimate of how a human planner might perform when creating the plan by hand. Humans are typically poor planners when dealing with large complex processes containing uncertainty[59]. When making decisions, humans tend to treat all cues (pieces of information contributing to their decision) as if they are equally rated. This is typically called the "as if" heuristic. Kahneman and Tversky demonstrated that even those well trained in statistical theory do not give proportionally more weight to more reliable cues, when making predictions[39]. We can assume that a human planner, although well trained, will probably use a similar heuristic and not give proportional weight to task assignment. A greedy algorithm does not take into account the overlap between effects or the value an effect achieves compared to the amount of tasks needed to accomplish it, but simply assigns tasks that achieve the highest expected value until all capacity is used. Thus, a greedy algorithm, in which tasks are assigned in order of the value of their achieved effect, is a decent approximation of how a human might perform the same assignment problem modeled by Formulation (3.3)-(3.15).

We created a greedy algorithm, that assigns tasks according to the EBO Framework we presented in Figure 3-1. We hypothesized that a human would start with the most valuable effects and assign tasks to achieve those effects and then work down until there is no more capacity to perform tasks. The algorithm is depicted in Figure 3-6 and runs as follows:

1. Calculate each squadron's initial capacity to perform ISR and strike tasks in each time-period. This capacity is the same as the right hand side values of constraints (3.8), (3.9), (3.10), and 3.11.

2. Create a list of all task-sets. Find the task-set j^* that gives the best additional value by causing an effect.

   - If the list is empty, the algorithm is complete.

3. Find all tasks that have not yet been assigned and are associated with task-set j^*. Call this set of tasks I.

4. Step through all tasks in I.

5. If task i is a loiter task in I,

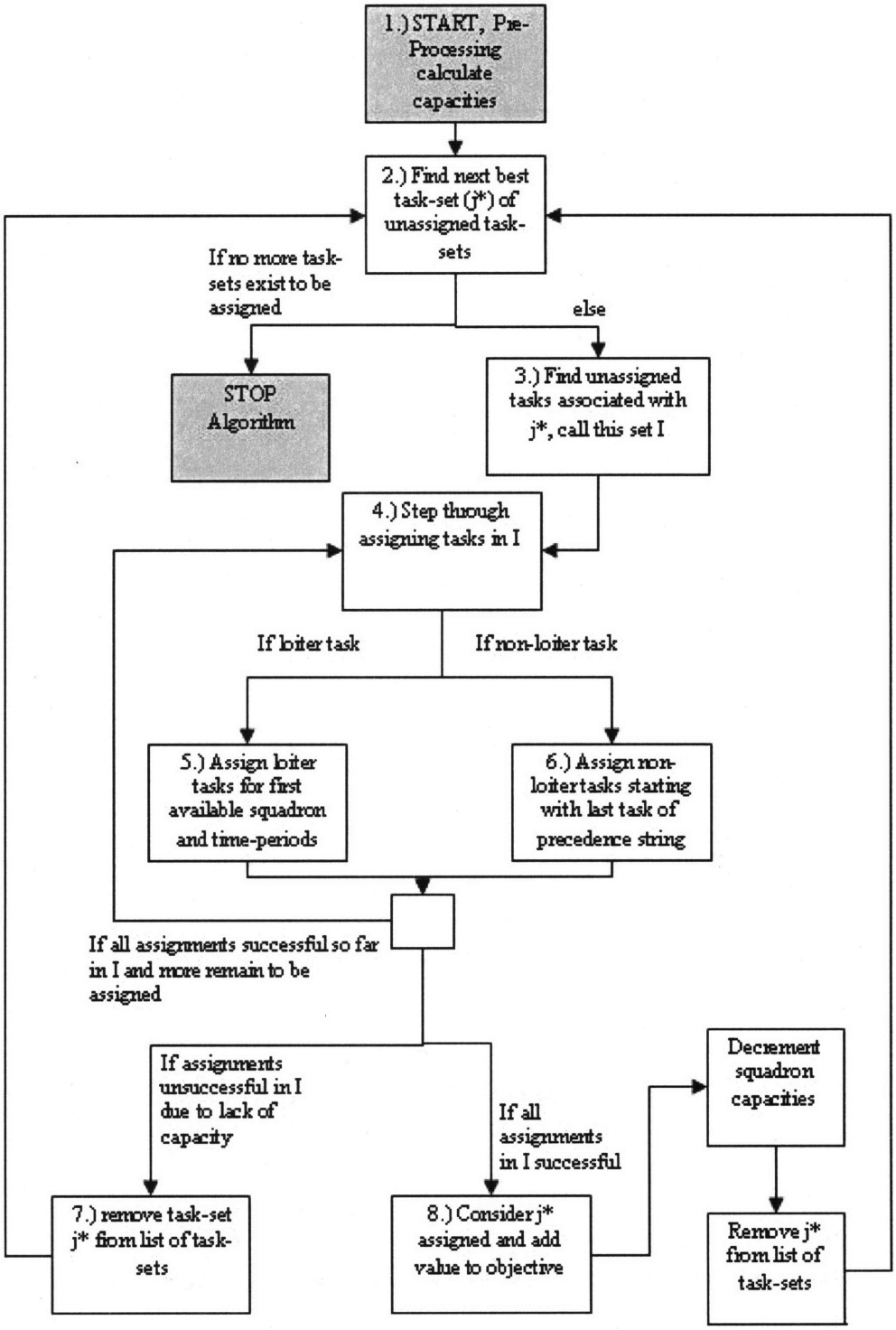


Figure 3-6: Greedy Algorithm Flow Diagram

- Find all time-periods that the task spans;

- Increment through the task's required time-periods for the squadron s;

- If there is capacity remaining in all required time-periods for s, consider the task assigned and track capacity to be decremented from s;

- Else check capacity of other squadrons to perform the task

- If there is no capacity in any of the squadrons across all time-periods, consider the task un-assignable.

6. Else if task i is a non-loiter task,

   - If the task has a precedence relationship where it must be done before another task, try assigning the succeeding task first;

   - Increment through the task's possible time-periods starting with the latest possible time, and increment through squadrons;

   - If there is enough capacity remaining in an available time-period for a single squadron, consider the task assigned and track capacity to be decremented from the squadron if the whole TSG gets assigned;

   - Else if there is no capacity in any of the squadrons across all time-periods, consider the task un-assignable;

   - If the task was successfully assigned and was part of a precedence string, go back and try to assign its predecessors.

7. If there is inadequate capacity to assign all tasks i in I, we do not have capacity to do task-set j^*;

   - Remove task-set j^* from the list of task-sets and return to second step.

8. Else if all tasks are successfully assigned, consider task-set j^* accomplished, and add task-set j^*'s expected value to our objective;

   - Decrement squadron capacities from all tasks assigned in j^*;

   - Remove task-set j^* from the list of task-sets and return to second step.

In Section 4.2, we compare the performance of the greedy algorithm to the performance of the robust models from 3.2.2 and 3.2.5.

3.2 Robust EBO Models

The Deterministic EBO Model presented in Section 3.1.4 assumes that there is no uncertainty in the input data. In reality, uncertainty can exist in the model in many of the data inputs. If we ignore this uncertainty and simply use deterministic expected values in the model, we may find that the plan is infeasible when we try to execute the plan and be forced to re-plan. The following list contains the data inputs that can contain uncertainty:

- c_ϵ: Commander's estimate of the worth of an effect;
- r_i: Number of aircraft to complete a task;
- v_s, ρ_s: Expected number of strike and ISR UAVs available in each squadron;
- Pr_j: Probability that TSG j causes an effect;
- λ^{isr}, λ^{st}, γ: Estimation of an aircraft's capacity to perform tasks in a time-period;
- lat_i, lon_i: Task locations.

In order to make the plans robust amidst uncertainty, we protect the plans using robust optimization techniques. In the following sections, we present two methods for incorporating robustness into the EBO Model: chance-constrained programming and a robust optimization formulation created by Bertsimas and Sim. We use these methods to create the Chance Constrained EBO Model, Extended Chance Constrained EBO Model, the Bertsimas/Sim EBO Model, and the Delta EBO Model.

3.2.1 Chance Constrained Programming

As discussed in Section 2.6.3, chance-constrained programming is a convenient method for protecting against constraint violations. Chance-constrained programming maximizes an objective function subject to constraints that must be satisfied with some probability. To expand on this

idea, consider a basic linear programming problem shown below:

$$\text{maximize} \sum_{j \in J} c_j x_j \tag{3.16}$$

$$\text{subject to} \sum_{j \in J} a_{ij} x_i \leq b_i \qquad \forall i \in I \tag{3.17}$$

$$x_j \geq 0 \qquad \forall j \in J \tag{3.18}$$

where J is the set of all variables and I is the set of all constraints. Assuming A_{ij} and B_i are random variables according to a known probability distribution for all constraints i, we can maximize the objective such that each constraint i is feasible with probability α_i. Each constraint can have a unique α_i. According to these assumptions, (3.16)-(3.18) can be transformed to the following::

$$\text{maximize} \sum_{j \in J} c_j x_j \tag{3.19}$$

$$\text{subject to } \Pr\left(\sum_{j \in J} \mathrm{A}_{ij} x_j \leq \mathrm{B}_i\right) \geq \alpha_i \quad \forall i \in I \tag{3.20}$$

$$x_j \geq 0 \quad j \in J \tag{3.21}$$

Charnes and Cooper introduced three variations of possible objective functions for chance constrained models which can take on a deterministic form: an expected value optimization (the "E model"), a minimum variance objective (the "V model"), and a maximum probability model (the "P model")[22]. In each of these three models, Charnes and Cooper assumed that the values of $\boldsymbol{b}$ and $\boldsymbol{c}$ can be uncertain. In the following paragraphs we will outline these three models from [22] to give examples of how chance-constrained programming can be implemented in a linear equivalent form.

The "E model" using (3.19)-(3.21) and maximizing the expected value of the objective function $E[\mathrm{C}'x]$, assuming C is a vector of random variables. For this formulation, we also assume uncertainty in the b_i's, making the vector B of random variables. We assume we know the probability function that generates the random variables for C and B. Therefore, each constraint has one uncertain coefficient B_i. Each B_i can be governed by its own probability distribution. We add the constraints $x_j = d_{ji}\mathrm{B}_i$, where d_{ij} is determined by reference to (3.20). We can select any probability α_i that

constraint i will not be violated. Thus (3.19)-(3.21) become:

$$\text{maximize} \sum_{j \in J} E[\mathrm{C}_j x_j] \tag{3.22}$$

$$\text{subject to } \Pr\left(\sum_{j \in J} a_{ij} x_j \leq \mathrm{B}_i\right) \geq \alpha_i \quad \forall i \in I \tag{3.23}$$

$$x_j = \sum_{i \in I} d_{ij} \mathrm{B}_i \quad \forall j \in J \tag{3.24}$$

To expand on the use of the d_{ij}'s as decision variables, we will take the "E model" in (3.22)-(3.24) and transition it to a deterministic equivalent. For this model, we assume that the uncertainty in the $\boldsymbol{c}$ vector is factored out of the problem by using its expected value. The uncertainty in the $\boldsymbol{b}$ vector is represented by the vector B which contains normally distributed random variables with known means and variances. In order to achieve a deterministic objective function, we do substitutions with constraint (3.24). For convenience, we represent the formulation in matrix notation for the rest of the example. Thus (3.24) becomes $x = \boldsymbol{D}\mathrm{B}$. We substitute it into the objective function to get $E[\boldsymbol{c}'\boldsymbol{D}\mathrm{B}) = (E[\boldsymbol{c}])'\boldsymbol{D}(E[\mathrm{B}])$. We can then define the vectors $\mu_c' = (E[\boldsymbol{c}])'$ and $\mu_{\mathrm{B}}' = (E[\mathrm{B}])'$. The formulation now becomes:

$$\text{minimize } -\mu_c' \boldsymbol{D} \mu_{\mathrm{B}} \tag{3.25}$$

$$\text{subject to } \Pr(\boldsymbol{A}\boldsymbol{D}\mathrm{B} \leq \mathrm{B}) \geq \alpha \tag{3.26}$$

Now, uncertainty only remains in the B vector. To facilitate subsequent developments, we introduce:

$$\hat{\mathrm{B}} = \mathrm{B} - \mu_{\mathrm{B}} \tag{3.27}$$

$$a_i' = (a_{i1}, ..., a_{in}) \tag{3.28}$$

$\hat{\mathrm{B}}$ is the deviation from the expected value for B and a_i' is the i^{th} row of $\boldsymbol{A}$. Because we assume that the elements of the B vector are normally distributed, we know that $(a_i'\boldsymbol{D}\mathrm{B} - \mathrm{B}_i)$ is also normally

distributed. Assuming that $E[\hat{\mathrm{B}}_i - ai'\boldsymbol{D}\hat{\mathrm{B}}]^2 > 0$ we can do the following:

$$\Pr\left(a_i'\boldsymbol{D}\mathrm{B} - \mathrm{B}_i \leq 0\right) = \Pr\left(\mathrm{B}_i - a_i'\boldsymbol{D}\mathrm{B} \geq 0\right) \tag{3.29}$$

$$= \Pr\left(\hat{\mathrm{B}}_i - a_i'\boldsymbol{D}\hat{\mathrm{B}} \geq -\mu_{\mathrm{B}_i} + a_i'\boldsymbol{D}\mu_{\mathrm{B}}\right) \tag{3.30}$$

$$= \Pr\left(\frac{\hat{\mathrm{B}}_i - a_i'\boldsymbol{D}\hat{\mathrm{B}}}{\sqrt{E[\hat{\mathrm{B}}_i - a_i'\boldsymbol{D}\hat{\mathrm{B}}]^2}} \geq \frac{\mu_{\mathrm{B}_i} - a_i'\boldsymbol{D}\mu_{\mathrm{B}}}{\sqrt{E[\hat{\mathrm{B}}_i - a_i'\boldsymbol{D}\hat{\mathrm{B}}]^2}}\right) \tag{3.31}$$

$$Z_i = \frac{\hat{\mathrm{B}}_i - a_i'\boldsymbol{D}\hat{\mathrm{B}}}{\sqrt{E[\hat{\mathrm{B}}_i - a_i'\boldsymbol{D}\hat{\mathrm{B}}]^2}} \tag{3.32}$$

so that Z_i has zero mean and unit variance. Then by substituting (3.32) into (3.31) we get:

$$\Pr\left(Z_i \geq \frac{\mu_{\mathrm{B}_i} - a_i'\boldsymbol{D}\mu_{\mathrm{B}}}{\sqrt{E[\hat{\mathrm{B}}_i - a_i'\boldsymbol{D}\hat{\mathrm{B}}]^2}}\right) \geq \alpha_i. \tag{3.33}$$

We can refer to (3.33) as F_i, where F_i is the cumulative density function of Z_i

$$F_i\left(\frac{\mu_{\mathrm{B}_i} - a_i'\boldsymbol{D}\mu_{\mathrm{B}}}{\sqrt{E[\hat{\mathrm{B}}_i - a_i'\boldsymbol{D}\hat{\mathrm{B}}]^2}}\right) \geq \alpha_i. \tag{3.34}$$

We can now use the symmetric properties of Z_i to achieve a deterministic equivalent . We can do so by creating K_{α_i} which is defined as:

$$\frac{\mu_{\mathrm{B}_i} - a_i'\boldsymbol{D}\mu_{\mathrm{B}}}{\sqrt{E[\hat{\mathrm{B}}_i - a_i'\boldsymbol{D}\hat{\mathrm{B}}]^2}} \leq F_i^{-1}(\alpha_i) = -K_{\alpha_i}. \tag{3.35}$$

Next we introduce ν_i defined by:

$$-\mu_{\mathrm{B}_i} + a_i'\boldsymbol{D}\mu_{\mathrm{B}} \leq -\nu_i \leq -K_{\alpha_i}\sqrt{E[\hat{\mathrm{B}}_i - a_i'\boldsymbol{D}\hat{\mathrm{B}}]^2}. \tag{3.36}$$

From this we have the following equations with $\nu_i \geq 0$.

$$-a_i'\boldsymbol{D}\mu_{\mathrm{B}} - \nu_i \geq -\mu_{\mathrm{B}} - K_{\alpha_i}^2 E\left[\hat{\mathrm{B}}_i - a_i'\boldsymbol{D}\hat{\mathrm{B}}\right]^2 + \nu_i^2 \geq 0. \tag{3.37}$$

We can now write the deterministic equivalent for (3.25)-(3.26), which is a convex programming

problem in the variables $\boldsymbol{D}$ and ν.

$$\text{minimize} - \mu_c; \boldsymbol{D}\mu_{\mathrm{B}} \tag{3.38}$$

$$\text{subject to } \mu_i \boldsymbol{D} - \nu_i \geq 0 \qquad \forall i \in I \tag{3.39}$$

$$-K_{\alpha_i}^2 \sigma_i^2(\boldsymbol{D}) + K_{\alpha_i}^2 \mu_i^2(\boldsymbol{D}) + \nu_i^2 \geq 0 \qquad \forall i \in I \tag{3.40}$$

$$\nu_i \geq 0 \qquad \forall i \in I \tag{3.41}$$

$$\text{where } \sigma_i^2(\boldsymbol{D}) = E\left[a_i' \boldsymbol{D}\mathrm{B} - \mathrm{B}_i\right]^2 \qquad \forall i \in I \tag{3.42}$$

$$\mu_i^2(\boldsymbol{D}) = \left(\mu_{\mathrm{B}_i} - a_i' \boldsymbol{D}\mu_{\mathrm{B}}\right)^2 \qquad \forall i \in I \tag{3.43}$$

The "V model" is set up similarly to yield a deterministic model that minimizes the variance of the objective function. In other words, we want to minimize the measure of the deviations about some given preferred value, annotated by $z^0 = (c^0)'x^0$. Thus, the model looks as follows:

$$\text{minimize } E\left[\boldsymbol{c}'x - z^0\right]^2 \tag{3.44}$$

$$\text{subject to } \Pr\{\boldsymbol{A}x \leq \mathrm{B}\} \geq \alpha \tag{3.45}$$

$$x = \boldsymbol{D}\mathrm{B} \tag{3.46}$$

As with the "E model," we can substitute in $\boldsymbol{D}$ and change the model to:

$$\text{minimize } V(\boldsymbol{D}) \tag{3.47}$$

$$\text{subject to } \Pr\{A\boldsymbol{D}\mathrm{B} \leq \mathrm{B}\} \geq \alpha \tag{3.48}$$

$$\text{where } V(\boldsymbol{D}) = E(\boldsymbol{c}'\boldsymbol{D}\mathrm{B} - z^0)^2 \tag{3.49}$$

The "P model" has a somewhat unorthodox objective function when compared to those of the previous two models. Charnes and Cooper call its results "satisficing" as opposed to optimizing. In this approach, z^0 is specified relative to some set of values which a human determines to be satisfactory. Of course, when there is uncertainty in the problem, a human cannot be sure that he will achieve these satisfactory levels among what he believes are the available alternatives. Thus, we now add a probability statement to the objective function and make our goal to maximize the probability that we achieve a certain satisfactory objective while also ensuring constraints are

satisfied with probabilities α. The formulation looks as follows:

$$\text{minimize } \Pr\left\{\boldsymbol{c}'x \geq z^0\right\} \tag{3.50}$$

$$\text{subject to } \Pr\left\{Ax \leq \mathrm{B}\right\} \geq \alpha \tag{3.51}$$

$$x = \boldsymbol{D}\mathrm{B}. \tag{3.52}$$

The above formulation can be made deterministic following similar transformations used to get (3.38)-(3.43). Charnes further details the structure and use of the "P model" in [24].

One of the major disadvantages of CCP is that in order to create these deterministic equivalent models presented by Charnes and Cooper, we have to determine how to restrict the uncertain coefficients so that the constraints are feasible with probability α. For the three models presented by Charnes and Cooper, uncertainty was assumed to be only in the objective function and in the right-hand side values. They handled the objective function uncertainty by using the expected value or minimizing the variance. They restricted the values in the $\boldsymbol{b}$-vector using the $\boldsymbol{D}$ matrix so that each constraint i would be feasible with probability α_i. Adding uncertainty into the left-hand side of the constraints will make much more difficult the determination of how to restrict the coefficients so that the constraints are feasible with some probability.

3.2.2 Chance-Constrained EBO Model

We applied the model given in (3.19-3.21) to the EBO Model in (3.3)-(3.15). Because we are maximizing the expectation of the objective function in (3.3)-(3.15), this is similar to the "E model." However, because we have already incorporated the expected value of causing effects into the Deterministic EBO Model, we can implement a simpler version of the "E model" than the $\boldsymbol{D}$ transformations that Charnes and Cooper presented. A basic chance-constrained model can be represented as follows:

$$\text{minimize } \sum_{j \in J} c_j x_j \tag{3.53}$$

$$\text{subject to } \sum_{j \in J} a_{ij} x_j \leq F^{-1}_{\mathrm{B}_i}(1 - \alpha_i) \quad \forall i \in I \tag{3.54}$$

$$x_j \geq 0 \quad \forall j \in J \tag{3.55}$$

where $F_{B_i}^{-1}(1-\alpha_i)$ is the value of B_i which is exceeded α_i percent of the time, which we can determine from the distribution of B_i. Before we can solve this model, we must know at least an approximate distribution for B_i. Then we can solve for $F_{B_i}^{-1}(1-\alpha_i)$ accordingly. We can apply this model to the EBO Model of (3.3)-(3.15) to account for much, but not all of the uncertainty.

The uncertain values discussed in Section 3.2 for the EBO Model, are in the objective function coefficients, and the constraint matrix coefficients and the right-hand side values of (3.8), (3.9), (3.10), and (3.11). We deal with uncertainty in the objective function by maximizing the expectation of the value attributed to causing each effect, $E[\boldsymbol{c}_e]$ which becomes $\sum_{e\in E}\sum_{j\in J_e} \Pr_j x_j \boldsymbol{c}_e$. We ignore the possible uncertainty in $[lat_i, lon_i]$ because we treat the minimization of distance we have to travel to perform tasks as a secondary objective and the uncertainty in task location is not likely to change which squadron is closest to the task. All other uncertainty resides in constraints (3.8), (3.9), (3.10), and (3.11). Coefficients in both the left and right-hand sides of these constraints can contain uncertainty. As stated before, using chance constraints to model uncertainty in both the left and right-hand sides of constraints is extremely difficult; therefore, we decided to model the uncertainty only in the right-hand-side of these constraints. They become as shown below:

$$\Pr\left(\sum_{i|K_i=isr} r_i z_{its} + \lambda^{isr}\sum_{i|K_i=lst} r_i z_{its} + \gamma \sum_{i|K_i=lisr} r_i z_{its} \le \lambda^{isr} v_s + \gamma\rho_s\right) \ge \alpha_{ts}^{isr} \quad \forall\, t\in T, s\in S \tag{3.56}$$

$$\Pr\left(\sum_{i|K_i=st} r_i z_{its} + \lambda^{st}\sum_{i|K_i=lst} r_i z_{its} \le \lambda^{st} v_s\right) \ge \alpha_{ts}^{st} \quad \forall\, t\in T, s\in S \tag{3.57}$$

$$\Pr\left(\sum_{i|K_i=lst} r_i z_{its} \le v_s\right) \ge \alpha_{ts}^{lst} \quad \forall\, t\in T, s\in S \tag{3.58}$$

$$\Pr\left(\sum_{i|K_i=lisr} z_{its} \le \rho_s\right) \ge \alpha_{ts}^{lisr} \quad \forall\, t\in T, s\in S \tag{3.59}$$

Where α_{ts}^{b} is the probability bound that constraint type b for time-period t and squadron s is feasible.

These four constraints contain all the uncertain coefficients discussed in Section 3.2 except those in the objective function. v_s, ρ_s are only in the right-hand-side. r_i is only in the left-land-side. λ^{isr}, λ^{st}, and γ are in both the right and left-hand sides. As discussed earlier, it becomes

extremely difficult to model uncertainty in more than one coefficient per constraint. This is because the λ^{isr}, λ^{st}, and γ coefficients in the left-hand-side only affect the constraint when certain tasks are in the plan. Thus, in order to model uncertainty in the left-hand-side we would need a joint probability distribution for all uncertain coefficients in a constraint that has conditional cases for each plan containing the different combinations of tasks that are in that constraint. This becomes an intractable problem.

As a result, we only model uncertainty in the right-hand-side. In the case of constraints 3.56 and 3.57, this causes us to be more conservative than we need to be. For instance, in constraint 3.56, if we want to be increase the probability that we will not violate the constraint, we restrict the right-hand-side. Since λ^{isr} is part of this restriction, we would also reduce it on the left-hand-side. Since we cannot do that, we end up planning too conservatively. This limitation of where uncertainty can be located in the model is a major limitation of CCP. To model uncertainty in the right-hand-side, we assign a random variable for each uncertain value. This random variable can be unique to each constraint, but makes more sense to be the same across constraints for each squadron. We determine the right-hand-side value $\lambda^{isr}\upsilon_s + \lambda^{st}\rho_s$, which represents the estimated aggregate number of ISR tasks squadron s can do per period t. We assume a probability distribution for each value $\lambda^{isr}, \upsilon_s, \lambda^{st}, \rho_s$ and then determine a joint probability distribution function (PDF) for the value $\lambda^{isr}\upsilon_s + \lambda^{st}\rho_s$. As in (3.53)-(3.55) we then convert the constraints in (3.56)-(3.59) to the following:

$$\sum_{i|K_i=isr} r_i z_{its} + \lambda^{isr} \sum_{i|K_i=lst} r_i z_{its} + \gamma \sum_{i|K_i=lisr} r_i z_{its} \leq F^{-1}_{\mathrm{B}^{isr}_{ts}}(1-\alpha_{ts}) quad \forall\ t \in T, s \in S \tag{3.60}$$

$$\sum_{i|K_i=st} r_i z_{its} + \lambda^{st} \sum_{i|K_i=lst} r_i z_{its} \leq F^{-1}_{\mathrm{B}^{st}_{ts}}(1-\alpha_{ts}) \quad \forall\ t \in T, s \in S \tag{3.61}$$

$$\sum_{i|K_i=lst} r_i z_{its} \leq F^{-1}_{\mathrm{B}^{lst}_{ts}}(1-\alpha_{ts}) \quad \forall\ t \in T, s \in S \tag{3.62}$$

$$\sum_{i|K_i=lisr} z_{its} \leq F^{-1}_{\mathrm{B}^{lisr}_{ts}}(1-\alpha_{ts}) \quad \forall\ t \in T, s \in S \tag{3.63}$$

When the EBO Model is set up according to these constraints, we call it the Chance-Constrained EBO Model.

Once we've determined a PDF for the right-hand-side values, we can set them so that the

constraint will be feasible with at least probability α_{ts}^{B} by determining the value of $F_{\mathrm{B}_{ts}}^{-1}(1-\alpha_{ts})$. For instance, if we assume $\lambda^{isr}\upsilon_s+\lambda^{st}\rho_s$ is a discrete uniform random variable from 10 to 20 and we want to be at least 90% sure that this particular constraint is not broken, then we'd set $F_{\mathrm{B}_{ts}}^{-1}(1-\alpha_{ts})$ to 11. We can perform similar calculations for all uncertain constraints.

Although different probability bounds can be used for each constraint, we find it convenient to set all α_{ts}^{B} to the same probability. We could then set the whole model to a particular "protection-level," meaning that each constraint is feasible with at least the probability of the protection-level.

We examine the performance of the Chance-Constrained EBO Model in 4.1.

3.2.3 Extended Chance-Constrained Model

The protection-level method of determining the robustness level is somewhat arbitrary. Because we are not guaranteed that all constraints are feasible with the probability of the confidence level, but only that each constraint is feasible with at least that probability, we have little useful information to signify exactly how robust the solution will be.

For this reason, Barnhart and Marla derived the Extended Chance-Constrained Formulation (ECCF)[7]. In this formulation, we parameterize a specified budget that we are willing to "give up" from the optimal deterministic solution, and then find the maximum probability of being feasible (α_i) for each constraint according to the budget. When applied to a basic MILP, the ECCF looks as follows:

$$\text{maximize } \gamma \tag{3.64}$$

$$\text{subject to } \boldsymbol{c}'\boldsymbol{x} \leq \boldsymbol{c}'\boldsymbol{x}^* + \delta \tag{3.65}$$

$$\Pr(\boldsymbol{A}\boldsymbol{x} \leq \mathrm{B}) \geq \alpha \tag{3.66}$$

$$\gamma \leq \alpha_i \quad \forall i \in I \tag{3.67}$$

Where γ is the protection-level (maximum probability each constraint is feasible that we can achieve for the budget), $\boldsymbol{x}^*$ is the optimal solution to the deterministic problem, and δ is the budget parameter. In this formulation we maximize γ, which we constrain as having to be less than or equal to every α_i. Thus we push α_i as high as possible for all constraints i or maximize the minimum protection-level of all constraints. The solution given has each constraint feasible with

at least probability γ.

We can extend this model beyond maximizing the minimum protection-level to maximizing the weighted α_i values based on their individual importance, or weights w_i. Now, we maximize γ_i times the weight assigned to each α_i.

$$\text{maximize} \sum_{i \in I} \gamma_i w_i \tag{3.68}$$

$$\text{subject to} \sum_{j \in J} c_j x_j \leq \sum_{j \in J} c_j x_j^* + \delta \tag{3.69}$$

$$\Pr\left(\sum_{j \in J} a_{ij} x_j \leq \mathrm{B}_i\right) \geq \alpha_i \quad \forall i \in I \tag{3.70}$$

$$\gamma_i \leq \alpha_i \quad \forall i \in I \tag{3.71}$$

As with the other chance-constrained models, we must find a deterministic equivalent. If we assume, as before, that all of the uncertainty resides in the vector B, and it is distributed according to a known distribution, (3.68)-(3.71) has the following deterministic equivalent.

Data Sets

- a_{ij}: constraint matrix coefficients specific to problem
- b_i: constraint coefficient specific to problem
- w_i: weight assigned to protection-level for constraint i
- I: set of all constraints, $i \in I$
- J: set of all variables, $j \in J$
- D: set of deterministic constraints, $i \in D$
- U: set of uncertain constraints, $i \in U$
- K: set of discretized protection-levels, $k \in K$

Decision Variables

- x_j: decision variable specific to problem

- y_i^k: 1 if constraint i can achieve protection-level k, 0 otherwise
- γ_i: protection-level achieved for constraint i

Formulation

$$\text{maximize} \sum_{i \in U} w_i \gamma_i \tag{3.72}$$

$$\text{subject to} \sum_{j \in J} c_j x_j \leq c_j x_j^* + \delta \tag{3.73}$$

$$\sum_{j \in J} a_{ij} x_j \leq b_i \qquad \forall i \in D \tag{3.74}$$

$$\sum_{j \in J} a_{ij} x_j \leq \sum_{k=1,...,K} b_i^k (y_i^k - y_i^{k+1}) \qquad \forall i \in U \tag{3.75}$$

$$y_i^{k+1} \leq y_i^k \qquad \forall i \in U, k = 1, ..., K-1 \tag{3.76}$$

$$y_i^1 = 1 \qquad \forall i \in U \tag{3.77}$$

$$y_i^{K+1} = 0 \qquad \forall i \in U \tag{3.78}$$

$$\gamma_i \leq \sum_{k \in K} k(y_i^k - y_i^{k+1}) \qquad \forall i \in U \tag{3.79}$$

$$x_j \geq 0 \qquad \forall j \in J \tag{3.80}$$

$$y_i^k \in 0, 1 \qquad \forall i \in I, k = 1, ..., K \tag{3.81}$$

$$\gamma_i \geq 0 \qquad \forall i \in I \tag{3.82}$$

- (3.72) Maximize the discretized probability that each uncertain constraint $i \in U$ is feasible according to the weight w_i for constraint i.
- (3.73) Maximize the function $\sum_{j \in J} c_j x_j$ as long as it is less than the optimal deterministic objective function $\sum_{j \in J} c_j x_j^*$ minus some budget δ we are willing to give up in order to be robust.
- (3.74) Constrain all constraints that do not contain uncertainty $i \in D$ as in the deterministic formulation.
- (3.75) Constrain all constraints that contain an uncertain B_i value $i \in U$, forcing the left-hand-side to be less than the value of b_i^k which is feasible with the selected maximum probability

k, as indicated by the y_i^k binary variables. (If we can achieve probability k, then y_i^k is one. For the largest probability we can achieve k^*, the next largest $k^* + 1$ will have $y_i^{k^*+1}$ equal to zero. Thus for all k's where we can achieve both k and $k + 1$, both y_i^k and y_i^{k+1} will be one and they will cancel out to zero, but for k^*, $y_i^{k^*} - y_i^{k^*+1}$ equals one. In this way, we only activate the highest probability b_i^k we can achieve.)

- (3.76) If we can achieve probability $k + 1$, then we can also achieve k.
- (3.77) We know that because we achieved a feasible solution in getting the deterministic optimal solution in which we used the expected value of each uncertain value, we should be able to achieve a feasible solution at the lowest k, which will usually be 0.5 to match the same probability as using the expected values.
- (3.79) We set $K + 1$ to zero so if we achieve the highest probability K, constraint (3.75) still has a right-hand-side value $y_i^K - y_i^{K+1}$ that equals one so we can activate b_i^K.
- (3.80) Set γ_i equal to the highest probability k achieved for constraint i using the same $(y_i^k - y_i^{k+1})$ rule as in constraint (3.75).
- (3.80) All x_j's are non-negative.
- (3.81) All y_i^k's are binary.
- (3.82) All γ_i's are non-negative.

We can apply the ECCF to our Deterministic EBO Model, to obtain a more convenient method of determining a protection-level for the plan. Because it is difficult to see how a particular protection-level affects the solution or how protected we really want to be, it might be easier for a human planner to set how much objective he is willing to give up and protect the plan as much as possible at that objective. If we want to have equal protection for all constraints, we can use (3.72)-(3.82), but not subscript y_i^k by i and only use y^k and change the objective to just maximize γ.

Unlike the Chance-Constrained EBO Model, if we apply ECCF to the Deterministic EBO Model, we do not just solve for the value of the right-hand-side for a particular probability, but solve the right-hand-side at multiple discretized levels $k \in K$. Depending on how many levels we

want this can greatly increase the problem size. In most cases, we want to start the smallest k at 0.5 since this is equivalent to using the expected values of the uncertain data.

Chance-constrained programming gives us a convenient method of representing the uncertainty in the model and measures of how protected the solution will be by applying a probability that each constraint must be feasible. However, chance-constrained programming requires that we can approximate the distribution of the uncertain data, and it also has difficultly incorporating uncertainty in more places than the right-hand-side values of the formulation. We now turn to the Bertsimas/Sim model, which addresses these issues, but has some shortcomings of its own.

3.2.4 Bertsimas/Sim Model

The Bertsimas/Sim formulation introduced in Section 2.6.4 is the second robust optimization technique we apply to the EBO planner. The Bertsimas/Sim formulation [14] provides more flexibility for modeling uncertain data in different parts of the problem formulation. To develop the Bertsimas/Sim formulation, we start with the basic linear programming problem:

$$\text{maximize} \quad \boldsymbol{c}'\boldsymbol{x} \tag{3.83}$$

$$\text{subject to} \quad \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \tag{3.84}$$

$$\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}. \tag{3.85}$$

In the Bertsimas/Sim formulation, all uncertainty is located in the coefficients of the $\boldsymbol{A}$ matrix. By a simple transformation, changing the objective function to maximize z and adding the constraint $z - \boldsymbol{c}'\boldsymbol{x} \leq 0$, we can roll the objective function into the $\boldsymbol{A}$ matrix, thus enabling us to consider uncertainty in the objective function. Similarly, if we have uncertainty in the $\boldsymbol{b}$-vector, we can subtract the $\boldsymbol{b}$-vector values into the left-hand-side and replace the right-hand-side with a zero.

The Bertsimas/Sim formulation assumes no probability distribution for the uncertain coefficient, but only sets a symmetric range around the deterministic value. Consider a particular row i of matrix $\boldsymbol{A}$ and let J_i be the set of uncertain coefficients in row i. Each value a_{ij}, where $j \in J_i$, is a symmetric, bounded random variable that takes on a value in $[a_{ij} - \hat{a}_{ij}, a_{ij} + \hat{a}_{ij}]$. For every row i there is a parameter Γ_i which takes a value in $[0, |J_i|]$, but is not necessarily an integer.

The Bertsimas/Sim formulation models the uncertain data by assuming that the majority of

deviation in the constraint can be accounted for by allowing a select number of uncertain coefficients to go to their worst-case values. The goal is to be protected in all cases where up to $\lfloor \Gamma_i \rfloor$ coefficients in constraint i go to their worst-case values and one coefficient changes by $(\Gamma_i - \lfloor \Gamma_i \rfloor)\hat{a}_{ij}$. In other words, we restrict the uncertainty in the constraint to only allow a subset of the uncertain coefficients to go to their worst-case values. The Bertsimas/Sim formulation guarantees feasibility if the uncertainty in the coefficients behaves in this manner. Furthermore, Bertsimas and Sim prove that if more than $\lfloor \Gamma_i \rfloor$ coefficients change, the solution will be feasible with high probability[14].

In order to create this formulation, Bertsimas and Sim start with the following non-linear formulation:

$$\text{maximize} \quad \sum_{j \in J} c'_j x_j \tag{3.86}$$

$$\text{subject to} \quad \sum_{j \in J} a_{ij} x_j + \max_{S_i \cup \{t_i\} | S_i \subseteq J_i, |S_i| = \lfloor \Gamma_i \rfloor, t_i \in J_i / S_i} \left\{ \sum_{j \in S_i} \hat{a}_{ij} y_j + (\Gamma_i - \lfloor \Gamma_i \rfloor) \hat{a}_{it_i} y_t \right\} \leq b_i \quad \forall i \in I$$

$$-y_j \leq x_j \leq y_j \quad \forall i \in I \tag{3.87}$$

$$l_j \leq x_j \leq u_j \quad \forall j \in J \tag{3.88}$$

$$y_j \geq 0 \quad \forall j \in J. \tag{3.89}$$

For each constraint i, the $\max_{S_i \cup \{t_i\} | S_i \subseteq J_i, |S_i| = \lfloor \Gamma_i \rfloor, t_i \in J_i / S_i} \left\{ \sum_{j \in S_i} \hat{a}_{ij} y_j + (\Gamma_i - \lfloor \Gamma_i \rfloor) \hat{a}_{it_i} y_t \right\}$ term finds the set of Γ_i pairs of $\hat{a}_{ij} x_j$ that increase the left-hand-side of the constraint the most. Thus the constraint is protected against the worst-case Γ_i coefficient changes. Note that when $\Gamma_i = 0$ the constraint is equal to the deterministic problem. If $\Gamma_i = |J_i|$ then all uncertain coefficients go to their worst-case and the constraint is equivalent to the Soyster formulation which we discussed in Section 2.6.1.

Bertsimas and Sim derived the following method to change (3.86)-(3.89) into a linear program. Given a vector $\boldsymbol{x}^*$, the protection function of constraint i is:

$$B_i(x^*, \Gamma_i) = \max_{S_i \cup \{t_i\} | S_i \subseteq J_i, |S_i| = \lfloor \Gamma_i \rfloor, t_i \in J_i / S_i} \left\{ \sum_{j \in S_i} \hat{a}_{ij} x_j^* + (\Gamma_i - \lfloor \Gamma_i \rfloor) \hat{a}_{it_i} x_j^* \right\} \tag{3.90}$$

$B_i(x^*)$ is equal to the objective function of the following linear optimization problem:

$$B_i(x^*, \Gamma_i) = \text{maximize} \sum_{j \in J_i} \hat{a}_{ij} x_j^* w_{ij} \tag{3.91}$$

$$\text{subject to} \sum_{j \in J_i} w_{ij} \leq \Gamma_i \quad \forall i \in I \tag{3.92}$$

$$0 \leq v_{ij} \leq 1 \quad \forall j \in J_i. \tag{3.93}$$

The optimal solution of the above problem consists of Γ_i variables at 1 and one variable at $(\Gamma_i - \lfloor \Gamma_i \rfloor)$. This is equivalent to the subset $S_i \cup \{t_i\} \, | S_i \subseteq J_i, |S_i| = \lfloor \Gamma_i \rfloor, t_i \in J_i / S_i$ with the same cost function $\left\{ \sum_{j \in S_i} \hat{a}_{ij} x_j^* + (\Gamma_i - \lfloor \Gamma_i \rfloor) \hat{a}_{it_i} x_j^* \right\}$ as given before. With this in mind (3.86)-(3.89) can be reformulated in the following way:

$$\text{maximize} \sum_{j \in J} c_j' x_j \tag{3.94}$$

$$\text{subject to} \sum_j a_{ij} x_j + w_i \Gamma_i + \sum_{j \in J_i} p_{ij} \leq b_i \qquad \forall i \in I \tag{3.95}$$

$$w_i + p_{ij} \geq \hat{a}_{ij} y_j \qquad \forall i \in I, j \in J_i \tag{3.96}$$

$$-y_j \leq x_j \leq y_j \qquad \forall j \in J \tag{3.97}$$

$$l_j \leq x_j \leq u_j \qquad \forall j \in J \tag{3.98}$$

$$p_{ij} \geq 0 \qquad \forall i \in I, j \in J_i \tag{3.99}$$

$$y_j \geq 0 \qquad \forall j \in J \tag{3.100}$$

$$w_i \geq 0 \qquad \forall i \in I. \tag{3.101}$$

In order to understand this reformulation, we must consider the dual of (3.86)-(3.89) which is:

$$\text{minimize} \sum_{j \in J_i} p_{ij} + \Gamma_i w_i \tag{3.102}$$

$$\text{subject to } w_i + p_{ij} \geq \hat{a}_{ij} \left| x_j^* \right| \qquad \forall i \in I, j \in J_i \tag{3.103}$$

$$p_{ij} \geq 0 \qquad \forall j \in J_i \tag{3.104}$$

$$w_i \geq 0 \qquad \forall i. \tag{3.105}$$

By strong duality, because (3.91)-(3.93) is feasible and bounded for all $\Gamma_i \in [0, |J_i|]$, then the dual problem is also feasible and bounded and its objective function value coincides with that of the primal. Because $B_i(x^*, \Gamma_i)$ is the protection limit of constraint i, it is also the objective function value of the dual problem. When we substitute this to (3.86)-(3.89), we get (3.94)-(3.101).

If $k = \sum_i |J_i|$, then (3.94)-(3.101) increases the problem size of the (3.86)-(3.89) from $i + 2k$ variables and $j+2k$ constraints to $i+k+1$ variables and $j+k+i$ constraints. However, it maintains the scarcity of the A matrix, which is advantageous for the performance of many algorithms.

3.2.5 Bertsimas/Sim EBO Model

We can add robustness to the Deterministic EBO Model by converting it to the Bertsimas/Sim formulation. To each constraint containing uncertainty, we add the Γ_i parameter, dual variables p_{uts}, and new decision variable z_u; and we constrain them as shown in (3.94)-(3.101), where $u \in U$ denotes the uncertain coefficients in each constraint. Representing the Bertsimas/Sim formulation applied to the EBO Model for all constraints requires indices indicating from which constraint each Γ_i, dual variables p_{uts} and z_u, and the right-hand-side values come. Notation in this representation becomes very confusing, so we just present the Bertsimas/Sim formulation applied to one constraint, (3.9):

$$\sum_{i|K_i=st} r_i z_{its} + \lambda^{st} \sum_{i|K_i=lst} r_i z_{ips} + w_{ts}\Gamma_{ps} + \sum_{u \in U} p_{uts} - \lambda^{st} v_s b \leq 0 \quad \forall\, t \in T, s \in S \tag{3.106}$$

$$w_{ts} + p_{uts} \geq \hat{a}_u y_u \quad \forall t \in T, s \in S, u \in U \tag{3.107}$$

$$b = 1 \tag{3.108}$$

$$p_{uts} \geq 0 \quad \forall t \in T, s \in S, u \in U \tag{3.109}$$

$$y_u \geq 0 \quad \forall u \in U \tag{3.110}$$

$$w_{ts} \geq 0 \quad \forall t \in T, s \in S \tag{3.111}$$

$\hat{a}_u$ is the range value of each uncertainty coefficients u. In practice we do not formulate the Bertsimas/Sim formulation in this way, but instead we find the complete $\boldsymbol{A}$-matrix, $\boldsymbol{b}$-vector, and $\boldsymbol{c}$-vector for the problem, and then formulate it according to (3.94)-(3.101). We create an $\hat{\boldsymbol{A}}$ matrix which contains the information about the uncertain coefficients and has a zero for all deterministic

values and a Γ_i vector which has zeros for all constraints not containing uncertainty.

Unlike chance-constrained programming, Bertsimas/Sim allows us to model uncertainty in any part of the problem formulation. Each uncertain constraint has its own unique Γ_i parameter. Bertsimas/Sim does not require that we know the distribution of the uncertain data, but that we just assume it comes from a bounded symmetric distribution. Beyond the complicated probability bounds given in Section 2.6.4, Bertsimas/Sim does not have intuitive probability statements like chance-constrained programming. Interpreting the amount of robustness a particular Γ_i achieves in the solution is difficult.

Like chance-constrained programming we also find it convenient to determine some overall protection-level for the Bertsimas/Sim formulation. We define the Bertsimas/Sim formulation protection-level as the percent of uncertain coefficients in each constraint that go to their worst-case values. For instance if there are 4 uncertain variables in a coefficient, the maximum Γ_i we can have for that constraint is 4, and its minimum is 0. If we were to set the protection-level for that constraint to 0.5, then we would set Γ_i to 2. If we set Γ_i similarly for all constraints, we would say that the protection-level for the entire formulation is 0.5.

3.2.6 Extended Bertsimas/Sim (Delta) Formulation

Like the ECCF, Barnhart and Marla also extended the Bertsimas/Sim formulation to maximize the protection of the solution, subject to maintaining some objective function value[7]. The Extended Bertsimas/Sim or Delta Formulation, works much like the ECCF and can also be applied to the EBO Model. As with the ECCF, the Delta Formulation was designed because it is difficult for a user to define the level of protection for every constraint. Furthermore, a human planner is generally more concerned with the benefit achieved by executing the plan than the amount of protection; thus, it is easier for a human to specify a benefit value to give up than a protection-level to attain.

In the Delta Formulation, the objective is to maximize the minimum protection-level of all uncertain constraints, which we call ν. To do this, we no longer use the parameter Γ_i but define a new variable Δ_i, which is the number of uncertain coefficients in constraint i that we plan on *not* deviating from their expected values. Δ_i now represents the opposite of Γ_i (that is, number of coefficients not changing versus the number of coefficients that go to their worst-case value.)

The Delta formulation requires that we first order all of the decision variables x_j in increasing

order of their respective $\hat{a}_{ij}$ values. The order of the j^{th} column in the i^{th} row is annotated by l_j^i. For example in constraint i, if variable j is has the smallest $\hat{a}_{ij}$ value its l_j^i value is 1. If, however it has the highest $\hat{a}_{ij}$ value, its l_j^i value is equal to the number of uncertain coefficients in constraint i.

Decision Variables

- x_j: binary decision variable specific to problem
- s_{ij}: decision variable, equal to 1 if coefficient a_{ij} can not go to its worst-case value or if the, 0 otherwise
- z_{ij}: binary decision variable, equal to 1 for row i and all $j = l_i^j$ indicating the set of variables, if in the solution, whose coefficients can take on their boundary values in the ith constraint. As such, the z_{il}s follow the function show in Figure 3-7

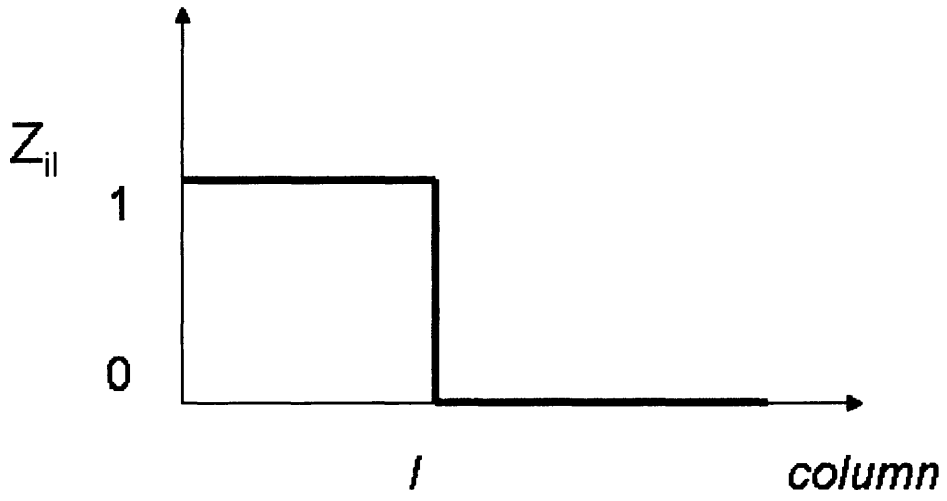


Figure 3-7: Function of z_{il} variables

- Δ_i: integer decision variable, the number of coefficients in constraint i that do not take on their worst-case value

Data

- a_{ij}: constraint coefficients specific to problem

- $\hat{a}_{ij}$: range a_{ij} can vary up or down

- b_i: constraint coefficient specific to problem

- I: set of all constraints

- J: set of all variables

- N: number of variables

- l_j^i: order index of variable j in constraint i

- x_j^*: solution to deterministic problem

- δ: budget specified by user that he is willing to give up from the deterministic solution value in order to have a robust plan

Formulation

$$\text{minimize} \quad \nu \tag{3.112}$$

$$\text{subject to :} \quad \sum_{j \in J} c_j x_j \geq \sum_{j \in J} c_j x_j^* - \delta \tag{3.113}$$

$$\nu \geq \Delta \qquad \forall i \in I \tag{3.114}$$

$$\sum_{j \in J} (a_{ij} + \hat{a}_{ij}) x_j - \sum_{j \in J} \hat{a}_{ij} s_{ij} \leq b_i \qquad \forall i \in I \tag{3.115}$$

$$\Delta \geq \sum_{l=0}^{N} \left[l(z_{il} - z_{i(l+1)}) - \left(s_{il_j^i} + z_{il} \right) \right] \qquad \forall i \in I \tag{3.116}$$

$$\Delta \geq \sum_{l=0}^{N} s_{il} \qquad \forall i \in I \tag{3.117}$$

$$s_{ij} \leq x_j \qquad \forall i \in I, j \in J \tag{3.118}$$

$$s_{ij} \leq z_{il_j^i} \qquad \forall i \in I, j \in J \tag{3.119}$$

$$s_{ij} \geq x_j + z_{il_j^i} - 1 \qquad \forall i \in I, j \in J \tag{3.120}$$

$$z_{i\phi} = 1 \qquad \forall i \in I \tag{3.121}$$

$$z_{i(N+1)} = 0 \qquad \forall i \in I \tag{3.122}$$

$$z_{il+1} \leq z_{il} \qquad \forall i \in I, l \in J \tag{3.123}$$

$$x_j \in \{0, 1\} \qquad \forall i \in I \tag{3.124}$$

$$z_{ij} \in \{0, 1\} \qquad \forall i \in I, j \in J \tag{3.125}$$

$$s_{ij} \in [0, 1] \qquad \forall i \in I, j \in J \tag{3.126}$$

- 3.112: Minimize the maximum number of coefficients that cannot go to their worst-case value out of all constraints.

- 3.113: The objective function value must be at least as good as the deterministic objective function value minus a user specified budget, δ.

- 3.114: Set (ν) as greater that or equal to the maximum number of coefficients that cannot go to their worst-case value out of all constraints.

- 3.115: For constraint i, for all coefficients j that cannot go to their worst-case value set s_{ij}

equal to 1.

- 3.116: Set Δ_i as greater than or equal to the number of coefficients in constraint i that cannot go to their worst-case value. All coefficients that cannot go to their worst-case value have z_{il} and $s_{il^i_j}$ set to 1. Thus, the function $\left[l(z_{il} - z_{i(l+1)}) - \left(s_{il^i_j} + z_{il}\right)\right]$ will only return a non-zero value in the two cases listed below and it counts the number of coefficients associated with the variables in the solution that cannot go to their worst-case value. For l where z_{il} is 1 and z_{il+1} is zero (all other coefficients with order greater than l can go to there worst-case values) the function $\left[l(z_{il} - z_{i(l+1)}) - \left(s_{il^i_j} + z_{il}\right)\right]$ will return the value l less the number of variables ranked ahead of the l^{th} variable in the ordered list for constraint i.

- 3.117: Ensures that the number of variables that cannot go to their worst-case values is at least equal to the number of variables with s value of 1 ($s = 1$ if $x = 1$ and the feasibility constraints are not satisfied unless coefficients remain at their nominal values).

- 3.118: Set s_{ij} to zero if x_j is zero in the solution.

- 3.119: Ensure $z_{il^i_j}$ is equal to to 1 if s_{ij} is equal to 1, otherwise the model could pick the variable with the largest coefficient $\hat{a}$ and only one variable could not go to its worst-case value. This ensures that we estimate the worst-case number of variables that cannot go to their worst-case value.

- 3.120: This ensures that if x and z are both 1, then s must be 1.

- 3.123: Ensure that the z_{il}s are monotonic.

- 3.121: Ensure that the zero ranked z_{il} value is 1.

- 3.122: Ensure the highest ranked z_{il} value is 0.

- 3.124: The x_{j_s} are binary.

- 3.125: The z_{ij}s are binary.

- 3.126: The s_{ij}s are in $[0, 1]$.

Unfortunately, the Delta Formulation is not amenable to column generation, which greatly reduces its tractability for large-scale problems. Because the theater-level problem can be a very

large-scale problem, it might be intractable for realistic sizes when attempted with the Delta Formulation.

3.3 Summary

In this chapter, we presented the EBO Framework and applied it to the Deterministic EBO Model. We ran the Deterministic EBO Model on a small example and explained the solution. We presented our greedy algorithm to which we will compare the performance of the Deterministic EBO Formulation. We applied the Deterministic EBO Formulation to chance-constrained programming and Bertsimas/Sim, giving two options for finding robust plans. We also presented the ECCF and Delta Formulation, which allow us to maximize the protection of a solution subject to an objective function level.

We now move to testing how well the models perform. We will compare the Deterministic EBO Formulation against the greedy algorithm. We will look at how often the deterministic formulation produces infeasible results under uncertainty. We will compare the performance of the robust models against that of the deterministic formulation and those of each other. We will also look for ways to recover a plan once it is determined to be infeasible without having to do an entire re-plan.

Chapter 4

Model Testing and Analysis

The primary goal of robust planning is to create plans that last longer and require fewer re-planning iterations than deterministic plans. In large planning problems, such as theater planning of air assets, re-planning costs can be significant. By avoiding re-planning costs, robust plans achieve more value than deterministic plans. We want to determine if robust planning formulations presented in Chapter 3 produce plans that will last longer and require fewer re-plans than deterministic plans. We also want to compare the performance of the EBO Model to the performance of current planning techniques. In this chapter, we analyze the ability of the Chance-Constrained EBO Model from Sections 3.2.2 and the Bertsimas/Sim EBO Model from Section 3.2.5 to make plans last longer and require fewer re-planning iterations; we compare the performance of these models with the performance of the greedy algorithm from Section 3.1.6; and we compare, in various scenarios and under different kinds of uncertainty, the performance of the Chance-Constrained EBO Model to the performance of the Bertsimas/Sim EBO Model.

4.1 Robust Model Testing

We want to determine how the robust EBO models perform. We step through several small scenarios to highlight specific performance attributes of the models. We hypothesize that robust plans created by both the Chance-Constrained and Bertsimas/Sim EBO Models encounter fewer constraint violations, encounter violations in later time-periods, and require less re-planning than plans generated by the Deterministic EBO Model. We also hypothesize that the achieved value of

a robust plan, created by either the Chance-Constrained or Bertsimas/Sim EBO Model, will be greater than the achieved value of a plan generated by the Deterministic EBO Model; because the robust plans encounter fewer constraint violations in execution, allowing more of the plan's value to be achieved without re-planning.

4.1.1 Simulation Approaches

We can analyze a plan's performance using several methods. One method is to generate a plan; simulate running the plan in a dynamic, uncertain environment; and re-plan when the deviations in the environment are such that the plan is no longer feasible. In doing so, we can measure how many times we have to re-plan, how long our plan lasts before requiring a re-plan, estimate the costs of re-planning, and compare the value we gain by performing the robust plan as opposed to the deterministic plan. For the theater-level planning problem, this method requires a closed loop simulation with lower-level planners that create individual aircraft routings. This kind of simulation is very difficult to create and tends to be computationally expensive.

A simpler method for analyzing a plan's performance is to first generate a plan, then generate realizations of the uncertain data that exists in the plan. We can test the plan against the realizations to determine the number of constraints that are violated, the frequency that we violate constraints, and the time-period in which the plan first encounters a violation. This kind of simulation is often called Monte Carlo Simulation. Monte Carlo Simulation is our primary tool for analyzing the performance of the models presented in Chapter 3.

4.1.2 Test Scenario Structure

To test the performance of the EBO Models we created the scenario shown in Figure 4-1. We use this scenario to demonstrate the performance of the Chance-Constrained and Bertsimas/Sim EBO Models based on several different metrics. This scenario demonstrates how different protection-levels affect the plan and demonstrates several non-intuitive aspects of the performance of the plans.

We use this scenario because it contains task-sets that share tasks, task-sets that are completely contained within other task-sets, solves quickly even when doing many test runs, is small enough to display the plan visually to gain insight into what is included in the plan, and is small enough to

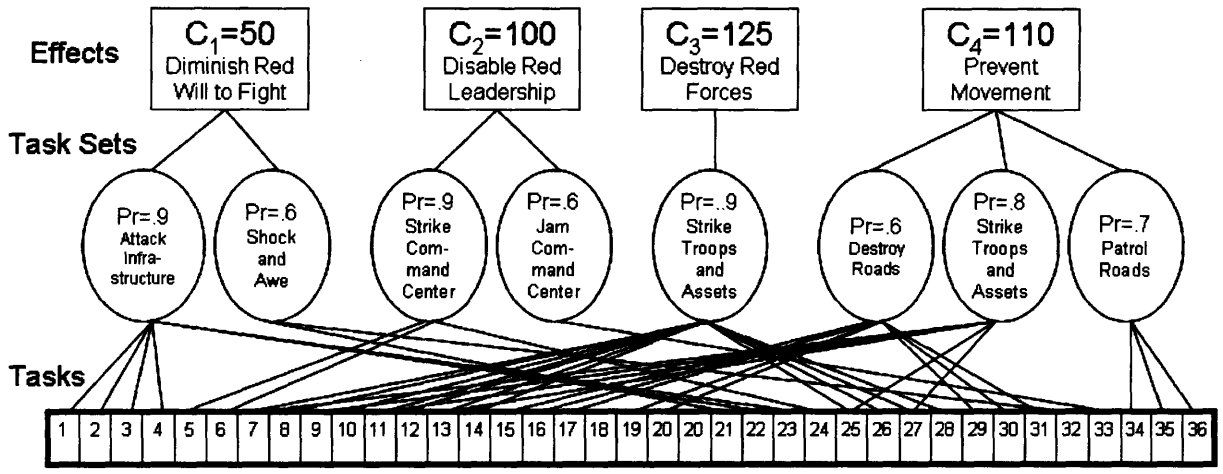


Figure 4-1: Test Scenario

stay within memory constraints of the computer when manipulating results and generating many realizations. This scenario contains 4 effects, 8 task-sets (which combine to form 14 TSGs), 36 tasks, and 6 time-periods. There are 3 squadrons each with 4 strike UAVs and 1 ISR UAV. We set the expected value of λ^{st} (the number of strike tasks a strike UAV can perform per time-period) as 2, λ^{isr} (the number of ISR tasks a strike UAV can perform per time-period) as 2, and γ (the number of ISR tasks an ISR UAV can perform per time-period) as 4.

Uncertain Coefficients

We considered uncertainty in the same coefficients when running both the Chance-Constrained and Bertsimas/Sim EBO Models on this scenario. Because the Chance-Constrained EBO Model allows uncertainty only in the right-hand-side of the constraints, we were limited to uncertainty in this area. The uncertain coefficients that we modeled were the number of each type of UAV available per squadron (ρ_s and υ_s) and the aggregate number of tasks a squadron can perform in a time-period (λ^{isr}, λ^{st}, γ). We set the maximum variation from the expected value of each uncertain coefficient to be one-half of the expected value. For instance, if the expected value of λ^{isr} is 4, we allowed it to vary between 2 and 6. We modeled the uncertain coefficients as uniform random variables within these windows. Modeling the uncertain data in this way might assume uncertain values that vary more than they would in reality; however, the large variations serve to highlight the performance of the models at different protection-levels.

Protection-Levels

We solve the Chance-Constrained and Bertsimas/Sim EBO Models at increasing protection-levels across the range of protection possible in both models. Because the Chance-Constrained and Bertsimas/Sim EBO Models protect the solution differently, the definition of a protection-level for each model is different.

For the Chance-Constrained EBO Model, we define a protection-level as setting the right-hand-side so that each constraint will be feasible with probability equal to the protection-level. For instance, setting a protection-level of 75% means that we solve the model where the right-hand-side is such that each uncertain constraint will be feasible with probability 0.75. A 50% protection-level is the same as using the expected value of every uncertain coefficient in the model which is also the same as the deterministic case, assuming that the uncertain coefficients vary according to a symmetric distribution. Thus, it makes little sense to plan at a protection-level less that 50% using chance-constrained programming, because we would be planning for the data to take on values higher than their expected value creating a less robust plan than the deterministic plan. A 100% protection-level means that we set the data such that we guarantee none of the constraints will encounter any violations in the realization of the uncertain data.

For the Bertsimas/Sim EBO Model, we define a protection-level as the percent of uncertain coefficients in each constraint containing uncertainty that take on their worst-case values. For instance, if we set the Bertsimas/Sim model to a 50% protection-level 50% of the uncertain coefficients in each constraint take on their worst case value. Recall that the Bertsimas/Sim EBO Model finds the Γ_i coefficients in each constraint i that are the most detrimental to the solution if they go to their worst-case value. For Bertsimas/Sim, a protection-level of 0% is the same as the deterministic plan, as opposed to 50% for chance-constrained programming. It then solves the problem with those coefficients at their worst-case values. Thus, if we have a constraint with 4 uncertain coefficients, setting a protection-level of 75% would mean that Γ_i would be 3 for this constraint, and the solution would reflect planning for the case where the 3 most detrimental coefficients are at their worst-case values. Like, the Chance-Constrained EBO Model, 100% protection-level for Bertsimas/Sim means that each constraint is guaranteed to be feasible in the realization of the uncertain data.

4.1.3 Simulation Function

To solve all of the models at each protection-level we used Xpress Mosel Version 1.6.0, Optimizer Version 16.01.02 on a 3.2 GHz Pentium 4 with 1 GB of RAM. We generated realizations of the uncertain data for each plan generated to test the models. We generated all realizations independently (i.e. a realization that affects the performance of a squadron in time-period 1 has no effect on other realizations in subsequent time-periods or for other squadrons.) For each constraint and realization, we then multiplied the solution variables with the left-hand-side coefficients and compared this to their respective right-hand-side realized value. This process required the use of several different programs to handle processing between data files, data parsing, formulating the models, solving the models, and generating results. We outline the simulation process in Figure 4-2.

There was little run-time difference between the Deterministic EBO Model, Chance-Constrained EBO Model, and Bertsimas/Sim EBO Model when solving the scenario shown in Figure 4-1. Each protection-level iteration for both the Chance-Constrained EBO Model and Bertsimas/Sim EBO Model solved to within an optimality gap of 0.001% in less than 10 seconds. However, generating 3000 realization for each protection-level and testing them against constraints took significantly longer than finding the solutions to the models; this process took roughly 45 min for this scenario.

For the simulations in this section, we chose to use 3000 iterations: because we found no significant change in the resulting statistics when using more than 1000 realizations, and the realizations were the primary factor affecting simulation run time. At 3000 iterations, we were able to run each simulation within a few hours and ensure that we did enough iterations to avoid anomalous behavior due to outlying realizations. We used the same realizations on both the Chance-Constrained EBO Model and Bertsimas/Sim EBO Model for similar protection-levels. For instance, the same 3000 realizations were used for 50% protection-level for the Chance-Constrained EBO Model and 0% protection-level for the Bertsimas/Sim EBO Model.

4.1.4 Robust Model Results

The results of running the Chance-Constrained and Bertsimas/Sim EBO Models on the scenario from Figure 4-1 are shown in Tables 4.1 and 4.2. For each protection-level we generate 3000 realizations of the uncertain data and compare the performance of the plan at each protection-level

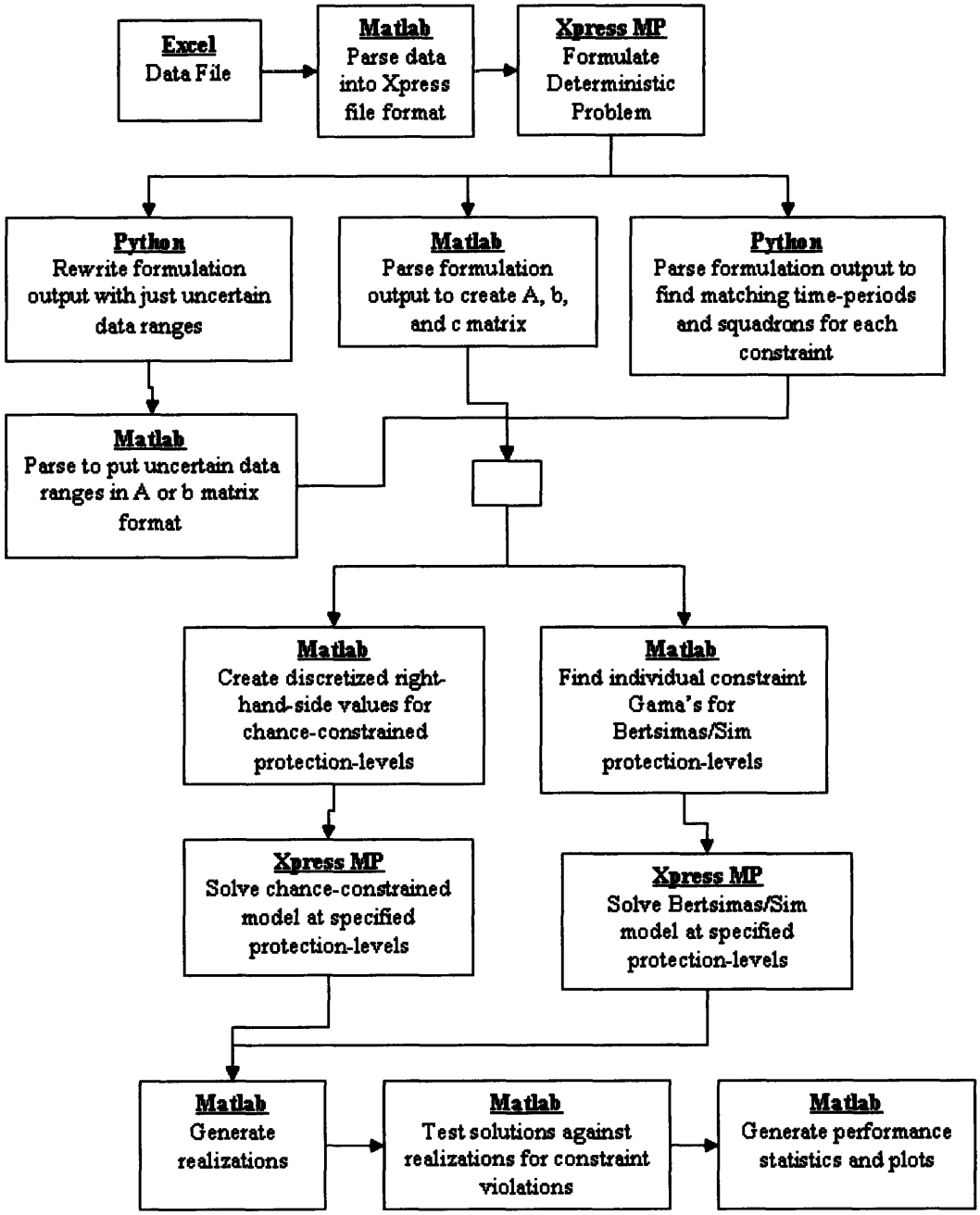


Figure 4-2: Flow Chart of Monte Carlo Simulation for EBO Models

to the realizations to determine the mean frequency of constraint violation (percent of realizations that cause a constraint to be violated, averaged over all constraints); the mean percent of constraints violated (percent of uncertain constraints that are violated in one realization, averaged over all realizations); and the mean time-periods until the plan fails (minimum time-period of a constraint that is violated per realization, averaged over all realizations).

Table 4.1: Chance-Constrained EBO Model Results

Protection-Level	Obj. Funct. Value	Mean freq. of Cstr Viol.	Mean Time-Period Until Plan Fails
50	346.3	0.1721	2.167
55	343.76	0.0476	3.576
60	343.76	0.0479	3.536
65	343.76	0.0489	3.520
70	343.76	0.0480	3.531
75	343.76	0.0481	3.479
80	300.16	0	7
85	300.16	0	7
90	300.16	0	7
95	300.16	0	7
100	300.16	0	7

3000 realizations, uniformly distributed uncertain coefficients, 50% variation from expected value

Table 4.2: Bertsimas/Sim EBO Model Results

Protection-Level	Obj. Funct. Value	Mean freq. of Cstr Viol.	Mean Time-Period Until Plan Fails
0	346.27	0.1107	1.6607
10	343.75	0.0160	4.5540
20	343.75	0.0243	4.1243
30	343.75	0.0435	3.7647
40	343.75	0.0429	3.5947
50	343.75	0.0272	3.7580
60	300.17	0.0063	6.1750
70	300.17	0	7
80	300.17	0	7
90	300.17	0	7
100	300.17	0	7

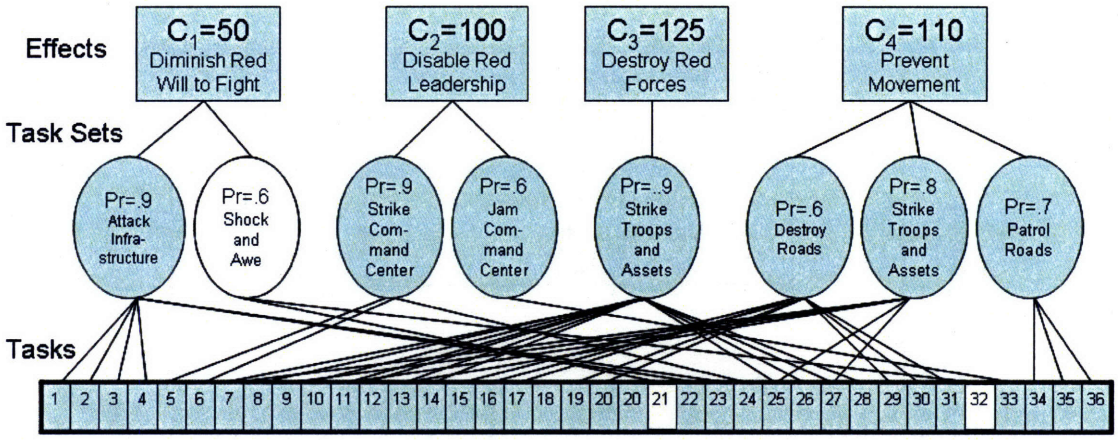
3000 realizations, uniformly distributed uncertain coefficients, 50% variation from expected value

4.1.5 Objective Function Behavior

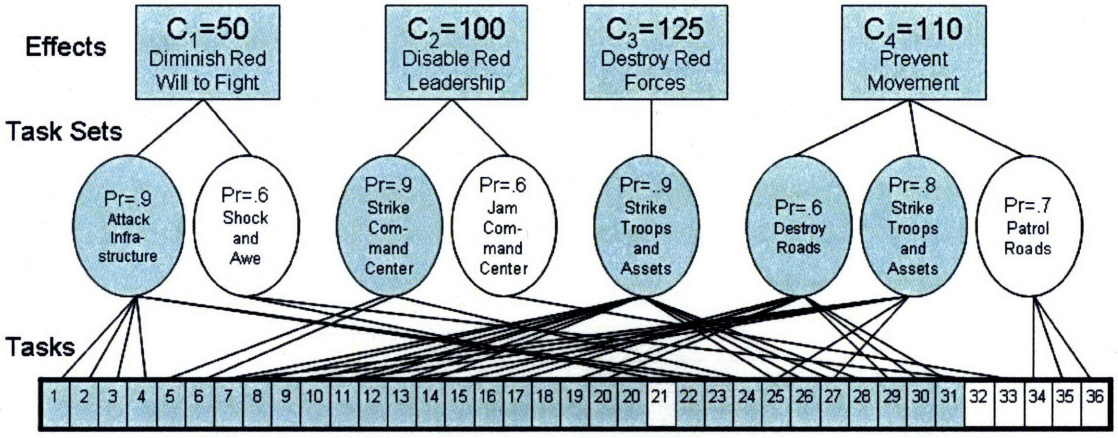
We notice that for both models the objective functions move down in steps as the protection-level increases. The Chance-Constrained and Bertsimas/Sim EBO Models generate the same objective function steps for similar protection-levels. Protection-levels 50 and 0 for the Chance-Constrained and Bertsimas/Sim EBO Models respectively, represent no protection of the problem and are the same as the deterministic problem. The first step down includes protection-levels 55-75 and 10-50 for the Chance-Constrained and Bertsimas/Sim EBO Models respectively, and the second step includes protection-levels 80-100 and 60-100 respectively. The slight difference between the objective function values of the Chance-Constrained and Bertsimas/Sim EBO Models in these steps is due to the different assignments between squadrons which occur because we only solve within a bound of optimality instead of all the way to optimality; but the same tasks, task-sets, and effects are assigned in both plans. Figure 4-3 shows which tasks, task-sets, and effects are being assigned in the plans used that generated the data in Tables 4.1 and 4.2.

In most cases, there are multiple solutions that achieve the same value (have the same tasks, task-sets, and effects assigned to the same squadrons) that differ in the time-periods that tasks are assigned to squadrons. This difference in time-periods does not affect the objective function value, but does affect the tightness of constraints and hence the frequency and number of constraints violated when the uncertain data is realized. We will discuss this phenomenon further in Section 4.1.6.

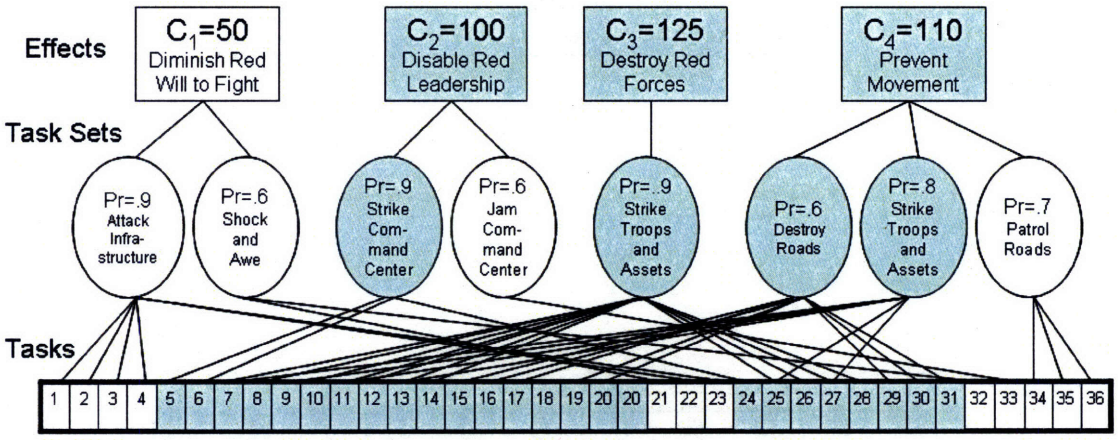
The stepwise movement of the objective function shows the all-or-nothing nature of EBO. In order to cause an effect, the plan must assign all tasks in a task-set. If we lose the capacity to perform just one task in a task-set, we lose the value of the effect it causes. As we incrementally increase the protection-level, we plan to have less capacity to do tasks with each squadron. As such, we cannot do as many tasks and task-sets. The steps in the objective function value can be clearly seen in Figure 4-4. The solid line at the top of the graphs is the objective function value. Notice that it steps down as we increase the protection-level and the steps are almost identical for both the Chance-Constrained and Bertsimas/Sim EBO Models.



(a) Protection-Level 0 for Bertsimas/Sim and 50 for Chance-Constrained



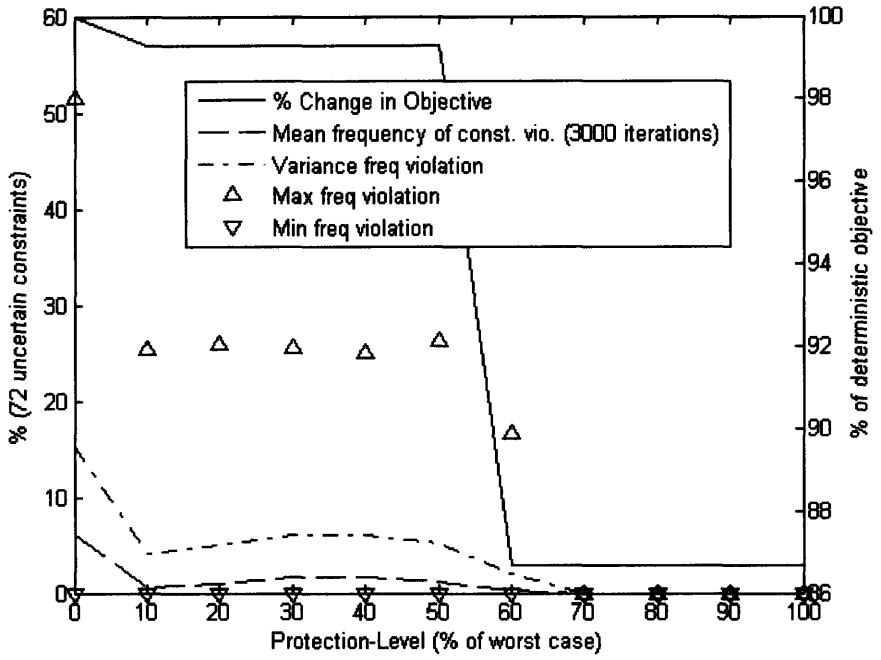
(b) Protection-Levels 10-50 for Bertsimas/Sim and 55-75 for Chance-Constrained



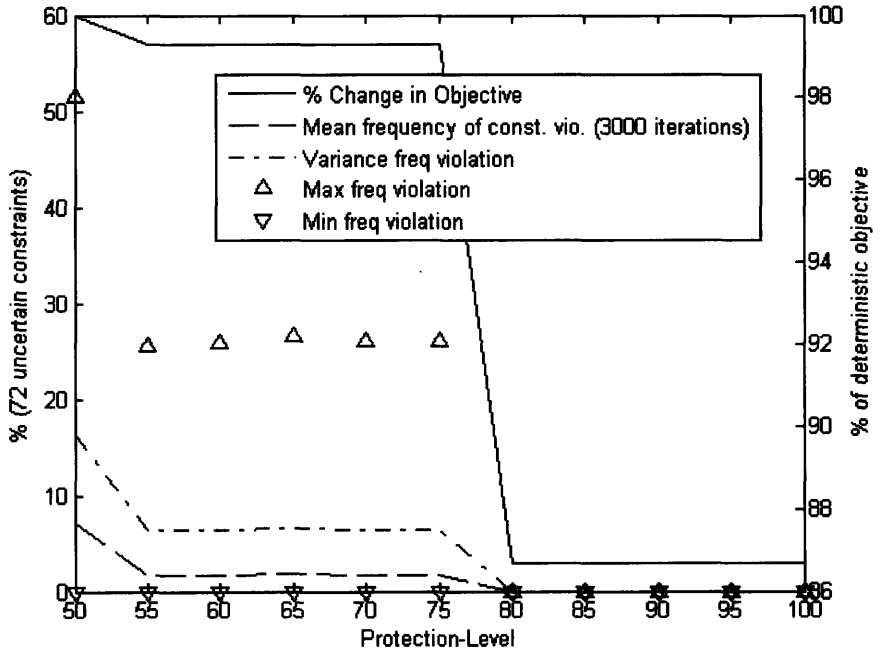
(c) Protection-Levels 80-100 for Bertsimas/Sim and 60-100 for Chance-Constrained

Shading of tasks, task-sets, and effects means they are assigned in the plan.

Figure 4-3: Plans Generated for Medium Scenario



(a) Bertsimas/Sim



(b) Chance-Constrained

Figure 4-4: Objective Value and Frequency of Constrain Violation

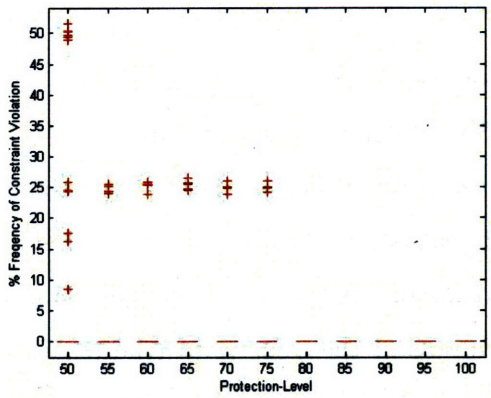
4.1.6 Frequency of Constraint Violation and Percent of Constraints Violated

Figure 4-4 also shows the mean frequency of constraint violation for constraints containing uncertainty for the plans generated by the Chance-Constrained and Bertsimas/Sim EBO Models. This data is also in Tables 4.1 and 4.2. Notice that the mean frequency of constraint violation tends to step down with the objective function; however, as shown in Figure 4-4 (a), the decrease is not always monotonic like the objective function's decrease.

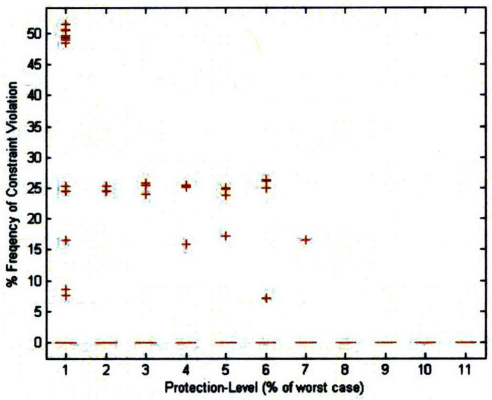
The mean frequency of constraint violation can be a confusing and somewhat misleading statistic. The mean frequency of constraint violation is the percent of realizations that cause a constraint to be violated, averaged over all constraints containing uncertainty. This is a different statistic than the mean percent of uncertain constraints violated in each realization. The mean frequency of constraint violation equals the mean percent of constraints violated, but the variance of each statistic is different. This is because both statistics are double means of the same data (the means are calculated in reverse order.) The mean frequency of constraint violation is the mean number of realizations causing a violation averaged over all constraints. The mean percent of constraints violated is the percent of uncertain constraints violated per realization averaged over all realizations. Although these statistics are related, they give different information about the performance of a plan.

We demonstrate the difference between the frequency of constraint violation and percent of constraints violated in Figure 4-5. Each data point in the box plots in (a) and (b) represent the frequency of constraint violation for each uncertain constraint; each box plot in (a) and (b) contains one data points for each uncertain constraint in the formulation. Each box plot in (c) and (d) contains 3000 data points, one for each realization.

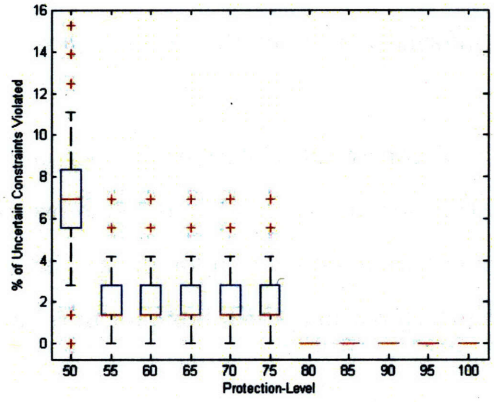
In graphs (a) and (b) of Figure 4-5, the median frequency of violation across all protection-levels is zero. In each plan, there tends to be a relatively small percentage of the uncertain constraints that are tight enough to be violated. These constraints tend to have high frequencies of violation, whereas the rest of the constraints are never violated. Many of the non-violated constraints are not in the plan, meaning they have a left-hand-side value of zero. We can get a better picture of the distribution of the frequency of violation by looking only at those constraints which are in the plan. which we call "active" constraints. An example of an active constraint might be (3.9) from Section 3.1.4 where we introduce the Deterministic EBO Model. (3.9) enforces that we cannot



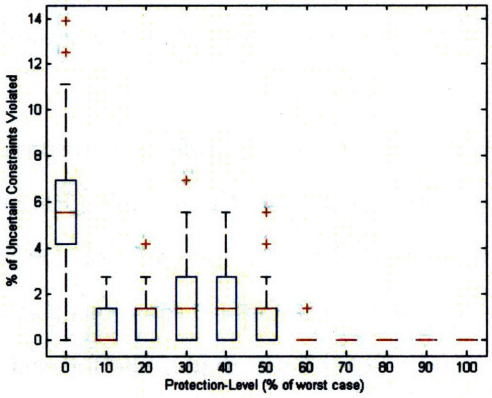
(a) Chance-Constrained Frequency of Constraint Violation



(b) Bertsimas/Sim Frequency of Constraint Violation



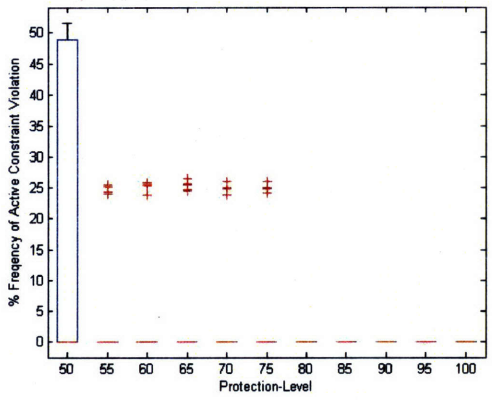
(c) Chance-Constrained Percent of Constraints Violated



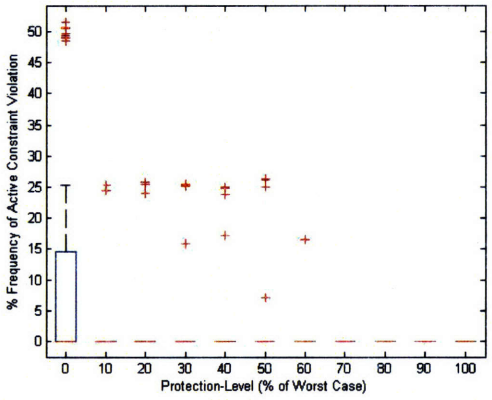
(d) Bertsimas/Sim Percent of Constraints Violated

Figure 4-5: Frequency and Percent Constraint Violation Box plots

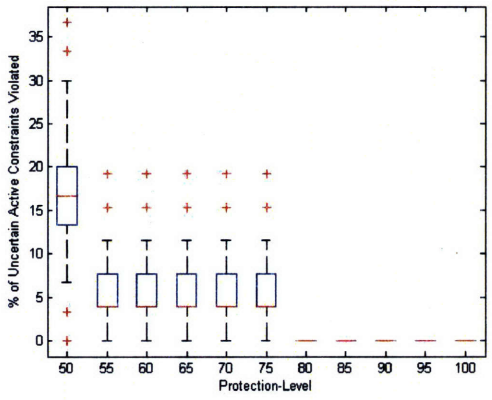
assign squadron s to do more strike tasks in time-period t than the squadron's capacity. If our plan dictates that squadron s does at least one strike tasks in time-period t then the constraint is active. If, however, the plan has no strike tasks assigned for squadron s in time-period t, then the constraint can never be violated and we call it non-active. Figure 4-6 plots the frequency of violation and number of constraints violated for only active constraints.



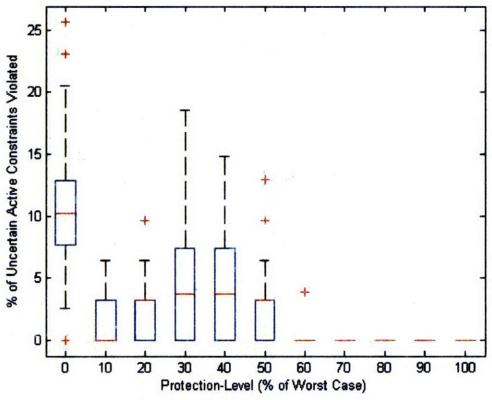
(a) Chance-Constrained Frequency of Active Constraint Violation



(b) Bertsimas/Sim Frequency of Active Constraint Violation



(c) Chance-Constrained Number of Active Constraints Violated



(d) Bertsimas/Sim Number of Active Constraints Violated

Figure 4-6: Frequency and Percent Active-Uncertain Constraint Violation Box Plots

In graph (d) of Figure 4-6 we notice that the median percent of constraints violated is non-monotonic across the protection-levels. Graph (c) does not demonstrate this behavior. Upon close inspection, we find that this is not a function of the way the Bertsimas/Sim EBO Model solves the problem, but more a function of which of the multiple optimal solutions the solver settles upon.

Recall from Figure 4-3 (b) that the same tasks and task-sets are assigned in plans across a broad range of protection-levels. Although these plans contain the same effects, task-sets, and tasks for each protection-level, the time that tasks are assigned to squadrons changes. These changes do not result in a change in the objective function, but might results in significant changes in how many constraint violations occur when we generate realizations.

In general, we might assume that increased protection causes fewer constraint violations. This is not always the case, as we see in Figure 4-7 where the curve for mean percent of constraints violated is non-monotonic. This non-monotonic behavior is because the time-period in which a task is assigned does not affect the objective function. Thus, there can be multiple optimal solutions at each protection-level. Each optimal solution assigns the same tasks to the same squadrons, but might assign those tasks in different time-periods, which sometimes creates tighter constraints and more constraint violations. We can see this by looking at the plans from the protection-levels that demonstrate the non-monotonic behavior.

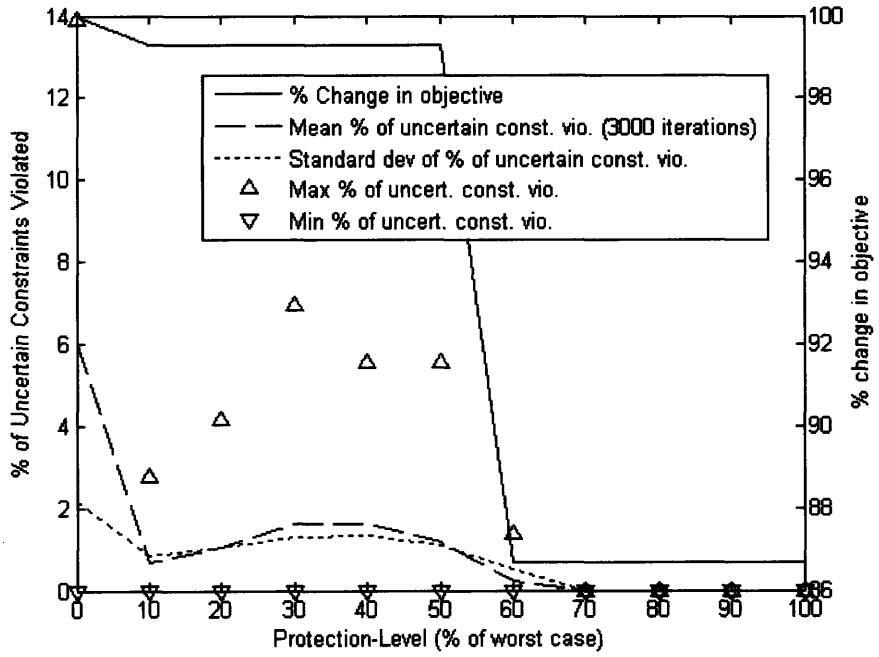


Figure 4-7: Bertsimas/Sim Objective Value and Percent of Uncertain Constraints Violated

To highlight this, we focus only on those constraints that regulate strike tasks. There is one

Table 4.3: Bertsimas/Sim Strike Constraint Left-Hand-Side Values

Protection-Level	10	20	30	40	50
Constraint(time,squad)	LHS Value	LHS Value	LHS Value	LHS Value	LHS Value
(1,1)	6	6	0	6	6
(1,2)	4	4	4	4	4
(1,3)	4	4	4	4	4
(2,1)	0	0	6	0	0
(2,2)	4	4	4	4	4
(2,3)	6	6	6	6	6
(3,1)	2	2	0	0	0
(3,2)	4	0	6	2	6
(3,3)	4	4	4	4	4
(4,1)	2	0	2	0	2
(4,2)	0	4	0	4	0
(4,3)	4	6	4	6	4
(5,1)	2	2	0	0	0
(5,2)	0	0	0	0	2
(5,3)	0	0	2	6	0
(6,1)	2	2	0	0	2
(6,2)	4	4	6	2	4
(6,3)	4	4	4	4	4
Num. Possibly Violated	2	3	4	4	3
Total Assigned	52	52	52	52	52
Obj. Funct Value	343.76	343.76	343.76	343.76	343.76

strike task constraint for each squadron s in each time-period t, 18 total for this scenario. Each squadron has 4 strike UAVs with a capacity to perform 2 strike tasks per time-period. Therefore, the expected value of strike tasks each squadron can perform per time-period is 8, which is the right-hand-side value of the strike task constraints. We consider this value uncertain, and as we said before we modeled it as a uniform random variable that can vary up or down 50% of the expected value, in this case between 4 and 12.

Table 4.3 shows the left-hand-side value of the strike constraints at protection-levels 10 through 50, in which the planned achieved objective value is the same but the percent of constraints violated varies considerably. These values show that different time assignments cause changes in the number of constraints that can be violated.

In Table 4.3, any value that is larger than the minimum possible value of the right-land-side (which is 4) indicates a constraint that can be violated. The number possibly violated increases from 2 to 3 to 4 and then decreases back to 3 across the protection-levels, which matches the

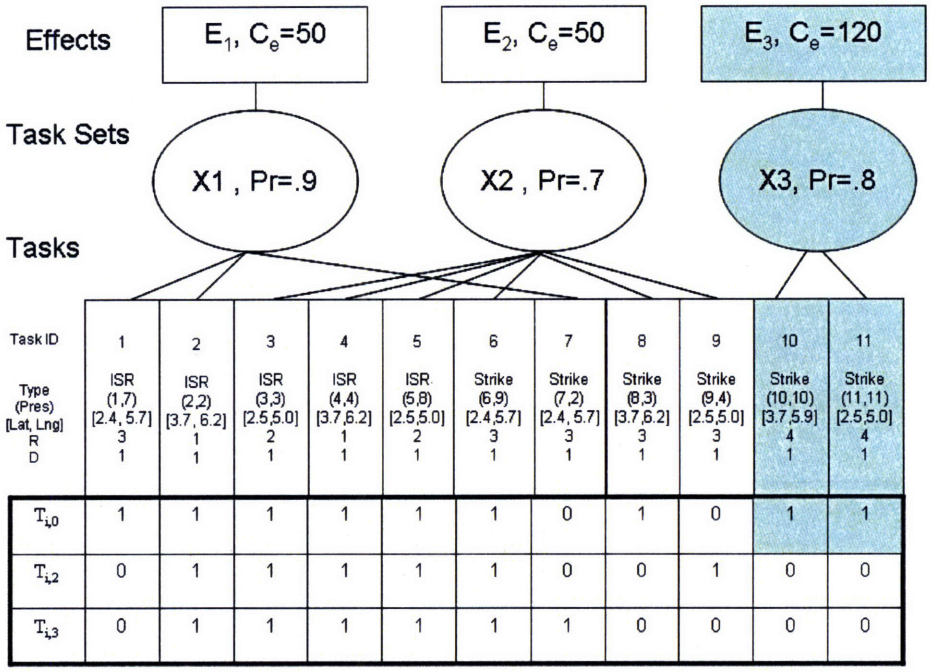
behavior shown in Figure 4-7. Notice that for all protection-levels the same number of strikes are assigned (52); these strikes are simply distributed differently among the time-periods.

The difference in which time-periods tasks are assigned is not a function of the formulation, but a function of which optimal solution the solver finds first. Thus, this behavior also sometimes occurs in the Chance-Constrained EBO Model results, though it did not do so in this scenario. To avoid this behavior, we need to force the model to distribute tasks equally among time-periods without loosing value in the objective function after finding the optimal objective function value.

Further complicating the results, some scenarios display non-monotonic frequency of violation and percent constraints violated behavior for a different reason. If there is a very demanding task (such as a strike mission that requires many UAVs to attack at once) that is connected to a very valuable effect, we may plan to do this task at the expense of performing many other less demanding tasks that achieve less value. If we increase the protection-level, we may plan to have too little capacity to do the one demanding task and choose to do the many less-demanding tasks. Because the one demanding task only activates one constraint, the plan using the other tasks may actually generate a mean frequency of constraint violation and a mean percent of constraints violated greater than those of the first plan even though we have increased the protection-level.

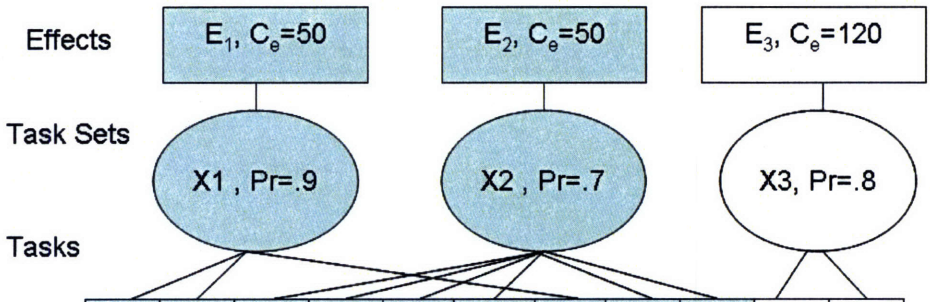
We demonstrate this phenomenon in the scenario shown in Figure 4-8. For the plan in (a), we plan to cause the third effect because it has a value higher than the combined value of effects 1 and 2. Effect 3 however, causes us to use all of our strike capacity for time-period 1. This prevents us from causing effects 1 or 2. As we increase the protection-level to 10 for Bertsimas/Sim, we limit the capacity that we plan to do for each squadron and time-period. This prevents us from doing the demanding strike tasks of effect 3 that require 4 UAVs each. However, we can assign the tasks of effects 1 and 2, which never require more than 3 UAVs per time-period per squadron. In doing effects 1 and 2 instead of effect 3, we now have more protected constraints and we have a greater number of active constraints. None of the constraints in plan (b) is violated as often as those in plan (a), but since there are a greater number of them that can be violated the mean frequency of constraint violation and the mean percent of constraints violated are higher.

This is a case where we also need to look at the statistics for only *active* constraints. In Figure 4-9 we compare the performance for all uncertain constraints versus active uncertain constraints when solved with the Bertsimas/Sim EBO Model. In (a) and (b), we see an increase in both



Task ID	1	2	3	4	5	6	7	8	9	10	11
Type (Pres) [Lat, Lng] R D	ISR (1,7) [2.4, 5.7] 3 1	ISR (2,2) [3.7, 6.2] 1 1	ISR (3,3) [2.5,5.0] 2 1	ISR (4,4) [3.7,6.2] 1 1	ISR (5,8) [2.5,5.0] 2 1	Strike (6,9) [2.4,5.7] 3 1	Strike (7,2) [2.4, 5.7] 3 1	Strike (8,3) [3.7,6.2] 3 1	Strike (9,4) [2.5,5.0] 3 1	Strike (10,10) [3.7,5.9] 4 1	Strike (11,11) [2.5,5.0] 4 1
$T_{i,0}$	1	1	1	1	1	1	0	1	0	1	1
$T_{i,2}$	0	1	1	1	1	1	0	0	1	0	0
$T_{i,3}$	0	1	1	1	1	1	1	0	0	0	0

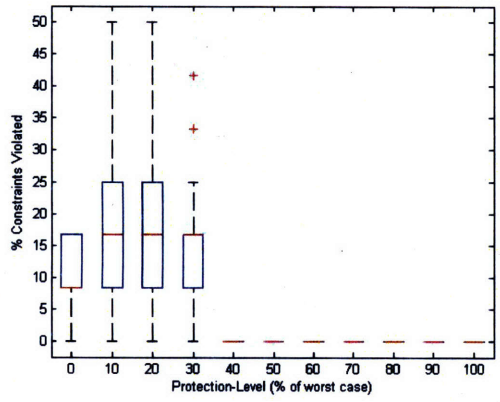
(a) Plan at Protection-Level 0



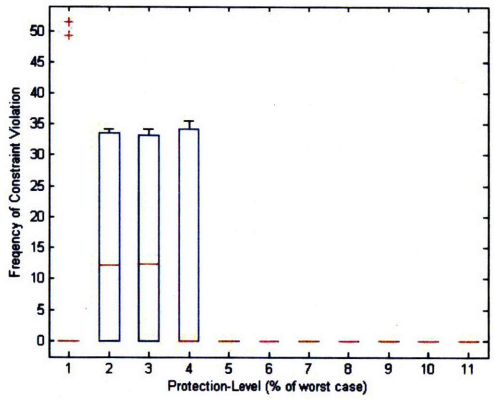
Task ID	1	2	3	4	5	6	7	8	9	10	11
Type (Pres) [Lat, Lng] R D	ISR (1,7) [2.4, 5.7] 3 1	ISR (2,2) [3.7, 6.2] 1 1	ISR (3,3) [2.5,5.0] 2 1	ISR (4,4) [3.7,6.2] 1 1	ISR (5,8) [2.5,5.0] 2 1	ISR (6,9) [2.4,5.7] 3 1	Strike (7,2) [2.4, 5.7] 3 1	Strike (8,3) [3.7,6.2] 3 1	Strike (9,4) [2.5,5.0] 3 1	Strike (10,10) [3.7,5.9] 4 1	Strike (11,11) [2.5,5.0] 4 1
$T_{i,0}$	1	1	1	1	1	1	0	1	0	1	1
$T_{i,2}$	0	1	1	1	1	1	0	0	1	0	0
$T_{i,3}$	0	1	1	1	1	1	1	0	0	0	0

(b) Plan at Protection-Level 10

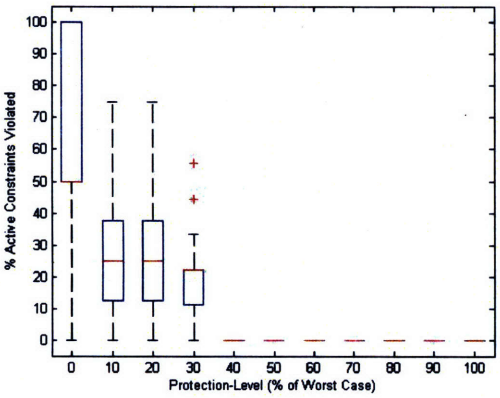
Figure 4-8: Plans Demonstrating Effect of High-Demand Task Scenario



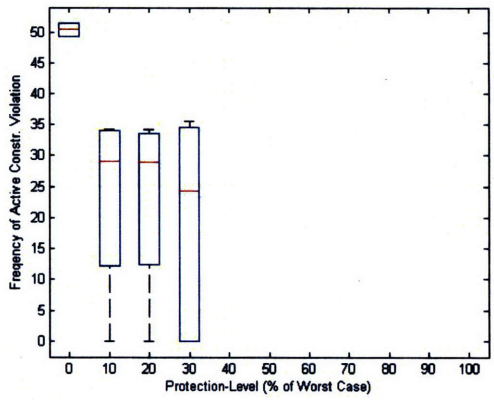
(a) Percent of ALL Uncertain Constrains Violated



(b) Frequency of ALL Constraint Violation



(c) Percent of ACTIVE Constraints



(d) Frequency of ACTIVE Constraint Violation

Figure 4-9: Bertsimas/Sim Violations: All Uncertain Constraints vs. Active Uncertain Constraints

the mean frequency of violation and mean percent uncertain constraints violated as we raise the protection-level from 0 to the 10-30 range. In (c) and (d) we see that if we only consider the active constraints, the increased protection decreases the mean frequency of violation and mean percent of constraints violated.

This behavior causes us to conclude that the mean frequency of violation and the percent constraints violated can be inconsistent measures of robustness and the quality of a plan cannot be determined with these two metrics alone. Although (c) shows that the few active constraints in the plan at protection-level 0 are violated often, it does not necessarily mean that it is a worse plan than those for protection-levels 10 and 20. To actually determine which plans will last longer and in the end give us more value we will need to look at other metrics, particularly the expected time

until the plan fails.

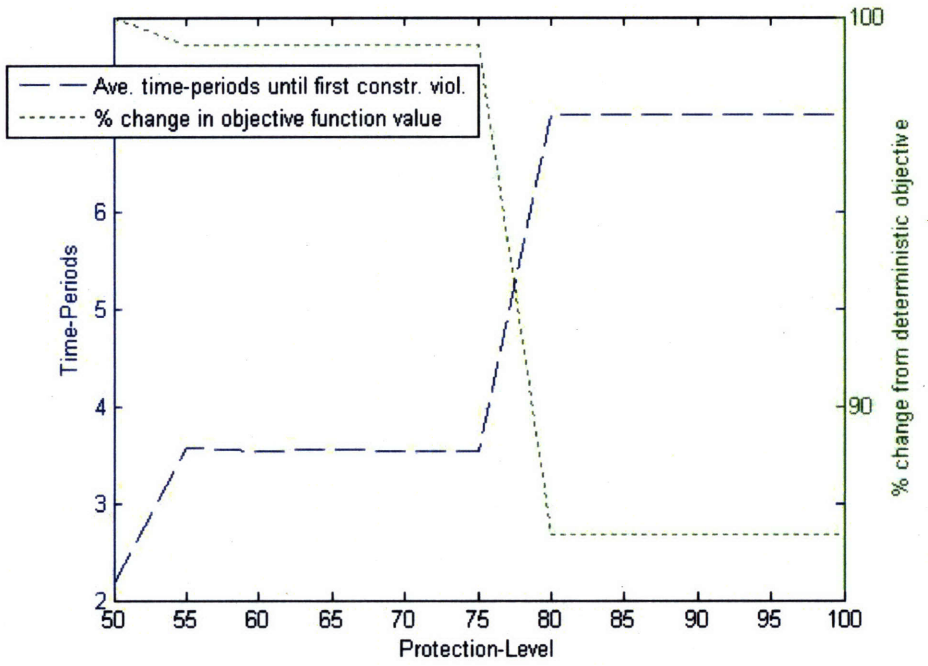
4.1.7 Time until Plan Fails

A more accurate measure of the performance of a plan is the minimum time-period in which a violation occurs. This first constraint violation gives us the time until the plan fails. We hypothesize that as we increase the protection-level for a plan, the mean time until failure will also increase. We expect this because there should be fewer violations for more protected plans; and with fewer violations, the average time-period until we come across the first violation with be later.

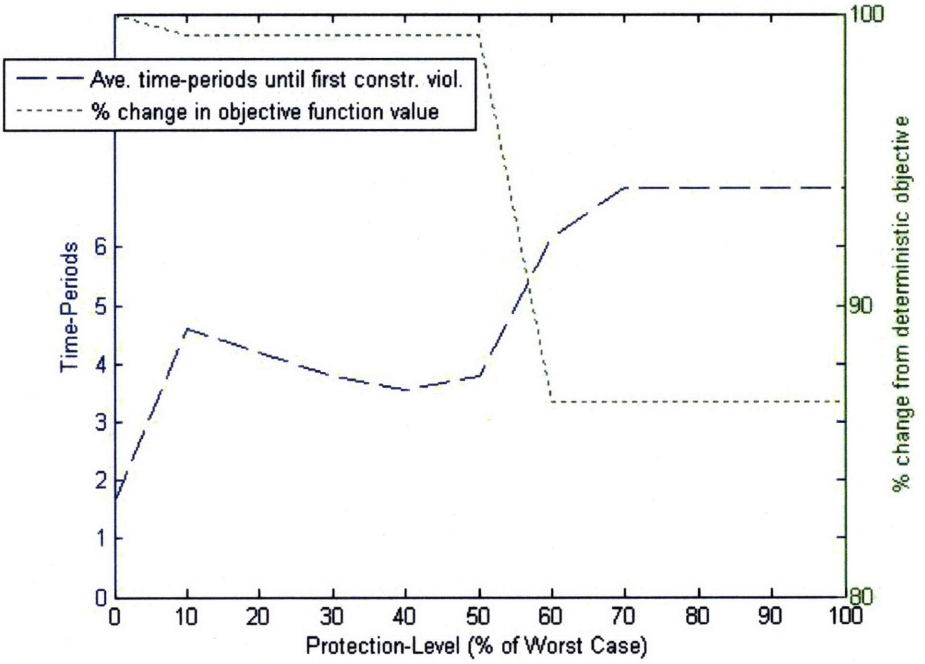
Using the same results from the simulation runs given in Section 4.1.4, we can generate the graphs in Figure 4-10. Again, we see non-monotonic behavior in Bertsimas/Sim as we saw with frequency of constraint violation and percent constraints violated. The reasons discussed in Section 4.1.6 (differing time assignment while assigning the same tasks and task-sets) play a role in creating the non-monotonic behavior in time until failure. However, time until failure is also affected by the time-windows in which we can assign tasks. If we have a scenario with many tasks required in early time-periods and few in later time-periods, we will likely have a low mean time-period until the plan fails. However, in scenarios with relatively evenly spread task time-windows we get a general upward trend in mean time-periods until the plan fails.

To determine which factors are causing the non-monotonic behavior, we look closer at when the first constraint violations occur, using the hazard rate plots in Figure 4-11. In these plots, the shading represents the percent of realizations that cause the plan's first constraint violation at each time-period. Darker shaded squares means a greater percent of realizations caused violations, whereas lighter means a smaller percent of violations. The numbers in each square show the number of realizations that cause the plan to encounter its first violation in that time-period. The numbers at the top of the graph represent the total number of realizations that do not encounter a violation for the whole length of the plan.

The shading is based on the percent of all realizations that cause the plan to encounter its first violation out of all that have not yet encountered a violation. For instance, in plot (a) for protection-level 55 in time-period 1, there were 733 realizations that caused a constraint violation out of 3000 realizations. This is 24% of the realizations, so the square is shaded accordingly as per the color scale on the right-hand-side of the graph. In time-period 2, 588 realizations caused

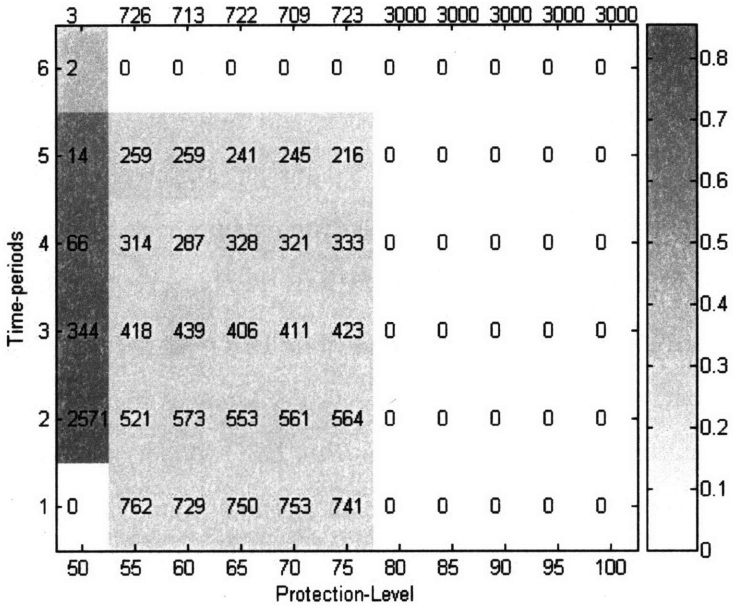


(a) Chance-Constrained Objective Function and Time until Failure

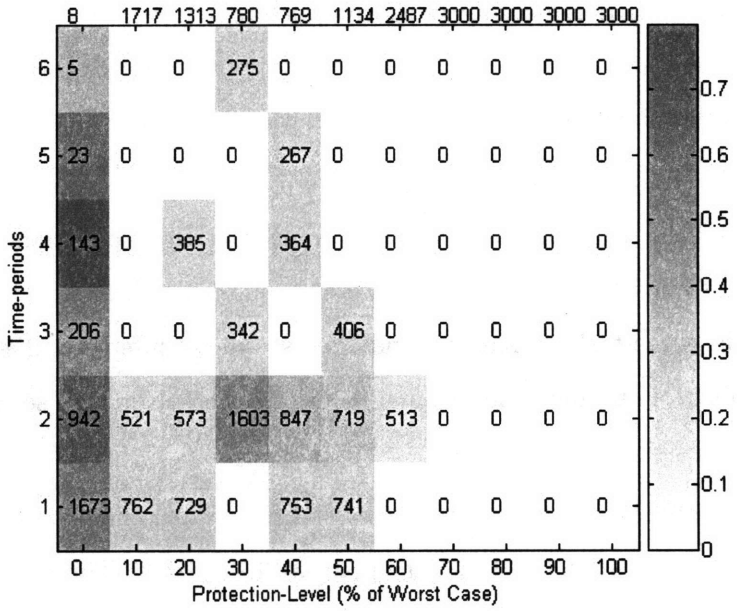


(b) Bertsimas/Sim Objective Function and Time until Failure

Figure 4-10: Chance-Constrained and Bertsimas/Sim EBO Model Time until Plan Failure



(a) Chance-Constrained



(b) Bertsimas/Sim

Numbers inside of the plot represents the number of realizations that encounter their first constraint violation for the specified time-period and protection-level. The numbers at the top of the plot represent the number of realizations that encountered no constraint violations in any time-period. The shading represents the percent of realizations that encounter a violation in the specified time-period out of all realizations that have not yet encountered a violation.

Figure 4-11: Hazard Rate Plots

a violation out of the remaining 2267 realizations that have not yet failed. This is 25.9% of the remaining 2267 realization and the cell is shaded accordingly.

The hazard rate plots help to illuminate when and how often violations occur and which protection-level actually gives the best plan. All plans after protection-level 75 for the Chance-Constrained EBO Model and 60 for the Bertsimas/Sim encounter no violations. For no protection in both plots, we see that only a handful of realizations do not encounter any violations.

For protection-levels 10-50 for the Bertsimas/Sim EBO Model and 55-75 for the Chance-Constrained EBO Model, we can now clearly see the differences in the plans that results in the different constraint violation rates. The Bertsimas/Sim EBO Model plans change significantly over this range. Each protection-level reveals that the tight constraints that are causing the violations have been moved to different time-periods. Protection-level 10 is clearly the best plan in this range, having 1679 realizations that never encounter a violation.

The Chance-Constrained EBO Model's plans are all the same across protection-levels 55-75. The Chance-Constrained models actually perform worse than the Bertsimas/Sim Model for in all similar protection-levels. This does not indicate that the Chance-Constrained model typically performs worse than Bertsimas/Sim, but for this scenario, it happens to solve to solutions, which cause more violations.

The primary reason for the difference in performance between the Chance-Constrained EBO Model and Bertsimas/Sim EBO Model is the different time-period assignments that occur across plans that include the same tasks, task-sets, and effects. These different time-period assignments are a function of which of the multiple optimal solutions the solver first settles upon and not a function of the model itself.

The hazard rate plots show that as we protect the plans, we tend to decrease the number of violations and thus increase the chances that the plan will last longer. There can be variation in the performance of plans within the protection-levels that achieve the same objective function value.

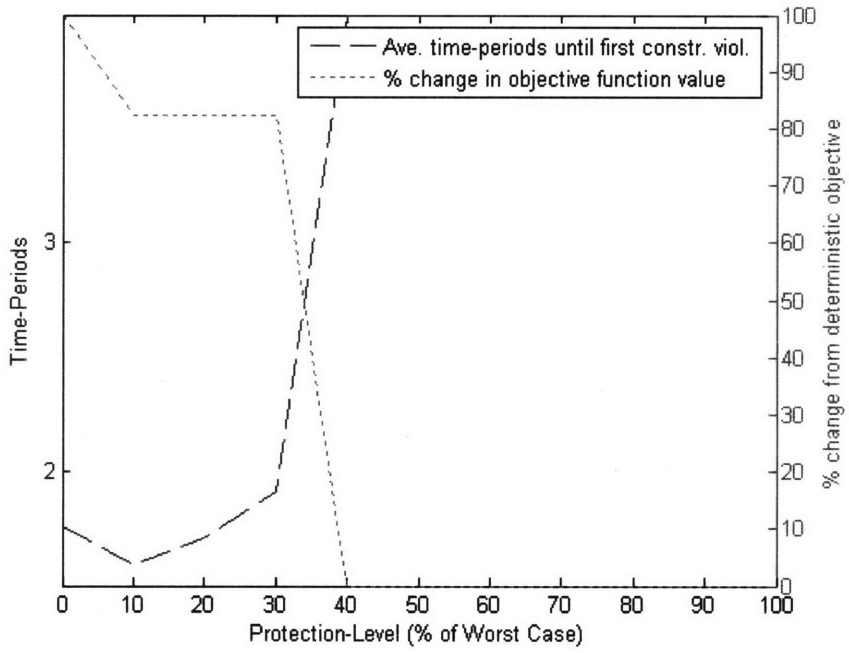
This scenario shows that we do not have to use a protection-level of 100 to achieve no violation, even though this is the only protection-level where encountering no violations is guaranteed. The plans never encounter a violation after protection-level 75 for the Chance-Constrained EBO Model 60 for the Bertsimas/Sim EBO Model. This is due to two reasons: the models use integer constraints and many of the tasks require multiple vehicles.

An example of how the integer constraints affect the plan is that we may constrain that a squadron's ability to perform strike tasks varies uniformly between 4 and 12. To be 90% certain that we will not violate this constraint, we cannot assign more than 4.8 tasks to that squadron. Obviously, we cannot assign fractions of a task, so we must round down. As a result, we end up protecting more than the protection-level indicates.

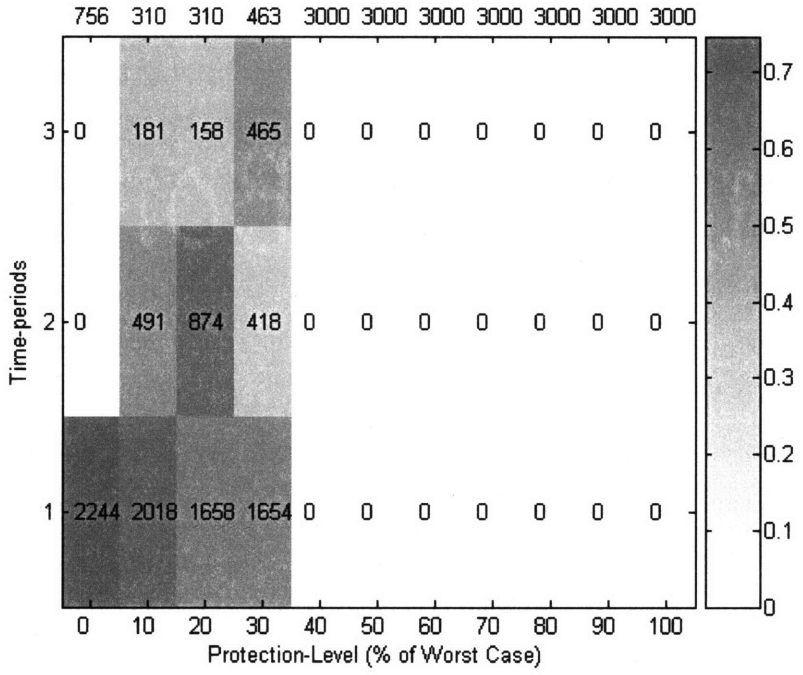
This behavior is compounded by many tasks that require more than one aircraft to be assigned to them. In this scenario, we have several tasks that require 4 strikes and the expected capacity of squadrons to be able to do strike tasks in one time-period is 8 strikes. For a higher protection-level we might be constraining that we can only do 7 strikes in a time-period, but since the tasks cost 4 strikes, we'll only be able to assign one of these tasks to a squadron per time-period. The only case in which we can assign two of these strike tasks to one squadron in the same time-period is the deterministic case.

For the scenario introduced in Figure 4-8, if we look at the time until failure and the hazard rate plot we are able to make a better judgment of the quality of the plan than just looking at the constraint violation rates. In Figure 4-12 (a), we see a slight decrease in the expected time until failure from protection-level 0 to 10, but in Figure 4-12 (b) we also see that more than twice as many plans do not encounter violations in protection-level 0 as in 10. Although a large proportion of realizations encounter violations in time-period 1 of protection-level 0, the fact that no other time-periods have violations allows more plans to run their full duration than the plans generated for protection-levels 10-30.

For protection-level 0, the plan in Figure 4-12 has the highest planned objective function value, a longer expected time until failure, and more plans that encounter no violations than the plans for protection-levels 10-30; we do not yet know which plan will actually achieve the most value. The plan at protection-level 0 includes just one effect, which encounters a significant number of violations in the first time-period. It might be that these are very costly violations that prohibit the plan from achieving any value from the planned effect. Or it might be that the value of the effect is worth the risk of violation when compared to the risk associated with the other plans. In order to determine which plans are actually the best and whether incorporating robustness actually gives better plans, we developed several methods of estimating the achieved objective value.



(a) Time Until Failure and Obj Value



(b) Hazard Rate Plot

Figure 4-12: Time to Failure and Hazard Rate Plot for High-Demand Task Scenario

4.1.8 Estimated Value Achieved

One of the primary weaknesses of using Monte Carlo Simulation to test the robust planning algorithms is that it is difficult to estimate what actual value executing the plan achieves. Obviously, we can count constraint violations, and determine in which time-period the first violation occurs; however, we do not have a good way to determine re-planning costs, how well our re-plan can perform, if we can make minor adjustments and continue the original plan, or if we can drop some tasks from the plan and move ahead with a decreased expectation of achieving the desired effects.

Because the EBO Model is intended to be used at a theater-level, which has large planning problems involving multiple organizations and human operators; we assume that the costs of planning are very high and that frequent re-plans are not an option. We also realize that we are planning for squadrons and they have flexibility to make minor changes to regain feasibility of a plan. They might drop certain tasks from the plan and keep those that give the most value. Instead of creating a closed-loop simulation and estimating re-planning costs; we created an algorithm to estimate value we will achieve with our original plan, based on which constraints are violated by the realizations in the Monte Carol Simulation. We can then compare this value to what we might expect to receive from other protection-levels to help us determine how much protection generates the best plan.

There are two simple methods of estimating a plan's achieved value. We have named them the *Set Method* and the *Percent Method.*

For the Set Method, we assume that in order to cause an effect all of the tasks in a task-set must be accomplished. If we cannot do a task of a task-set, we cannot cause the effect. When the plan encounters a constraint violation, we assume we are not able to accomplish whatever tasks are represented by the violated constraint. Subsequently, we lose any value we planned to achieve by task-sets that contain these tasks. This method represents the worst-case scenario.

We demonstrate this method in the plan from Figure 4-3 (a). If we assume a realization where we can no longer perform tasks 7, 9, 10 and 24, we lose their connected task-sets' value as shown in Figure 4-13. Notice, the set method causes the complete loss of value from effect 3.

The Percent Method makes a more liberal estimate of achieved value. We assume that for most task-sets if we fail to do a task in the task-set, there is still some probability that the remaining tasks will cause the desired effect. This is dependant on the scenario. If we fail to strike a communications

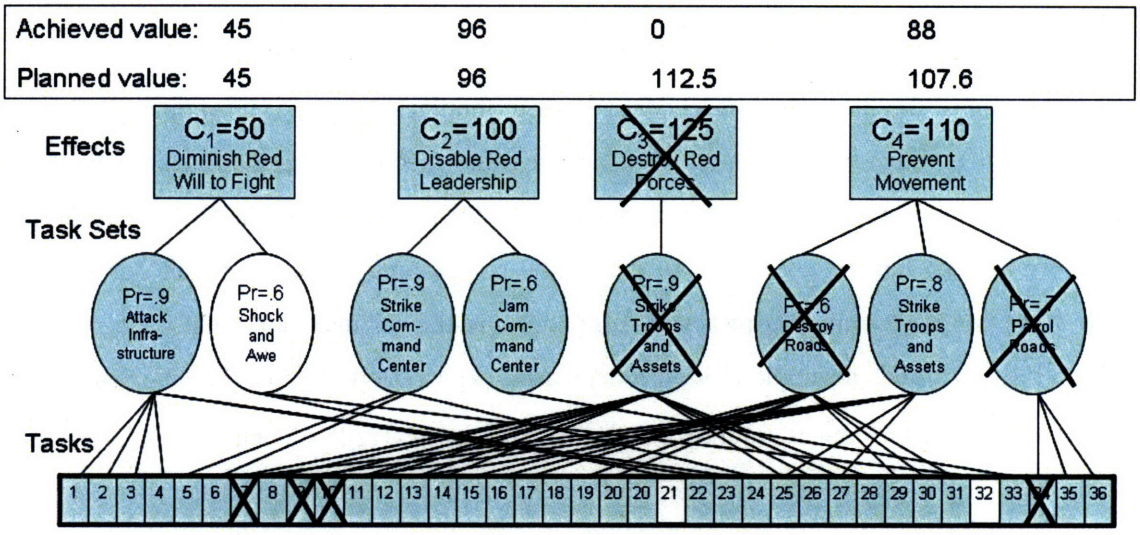


Figure 4-13: Set Method Performance

center and our desired effect was to disable the enemy's communications, it does not matter if we do the other tasks in the set (such as ISR or BDA); we will achieve no value. However, if in the same case we failed to be able to do the BDA task, we likely caused the effect, but we do not have updated information from the BDA.

Determining the ability to cause an effect if we do not do all of the tasks in a task-set is based on subjective judgment of a planner and the particularities of the scenario. We simplify the determination by assuming all tasks are equally weighted. This is a generous assumption, because some tasks must be done or we achieve zero value; however, we want to avoid making a planner review each task-set in order to determine new probabilities. We demonstrate the Percent Method in Figure 4-14 and we compare its values to the set method in Table 4.4.

Table 4.4: Estimate Achieved Value Calculations

	Effect Value	Planned Value	Set Method Value	Percent Method Value
Effect 1	50	45	45	45
Effect 2	100	96	96	96
Effect 3	125	112.5	0	75
Effect 4	110	107.6	88	102.9
Total Value	385	361.1	229	318.9

In most cases, if we trade a few tasks among the squadrons we can make an infeasible plan feasible again. This trading however would involve a re-plan, which we wish to avoid. It is feasible

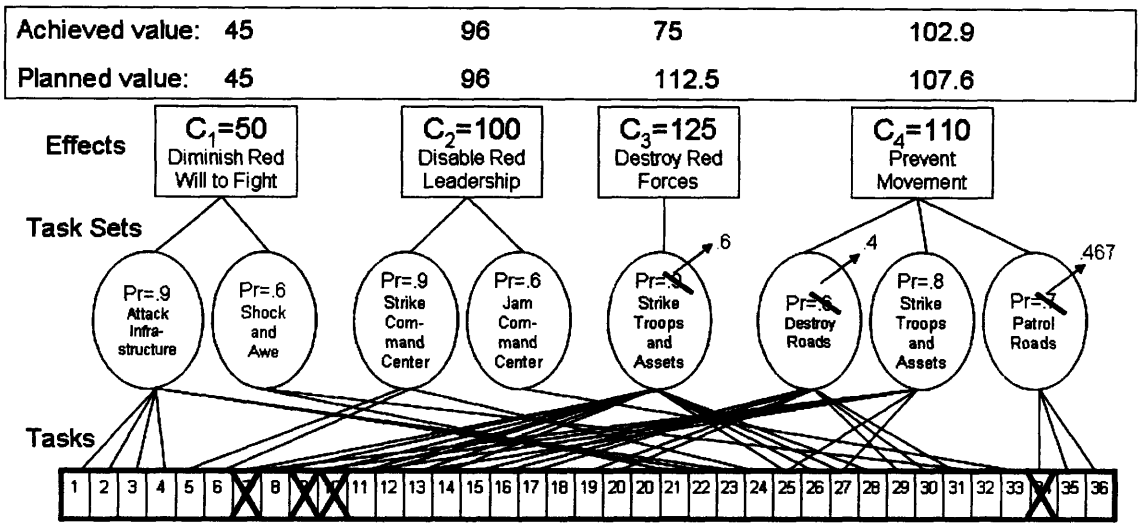


Figure 4-14: Percent Method Performance

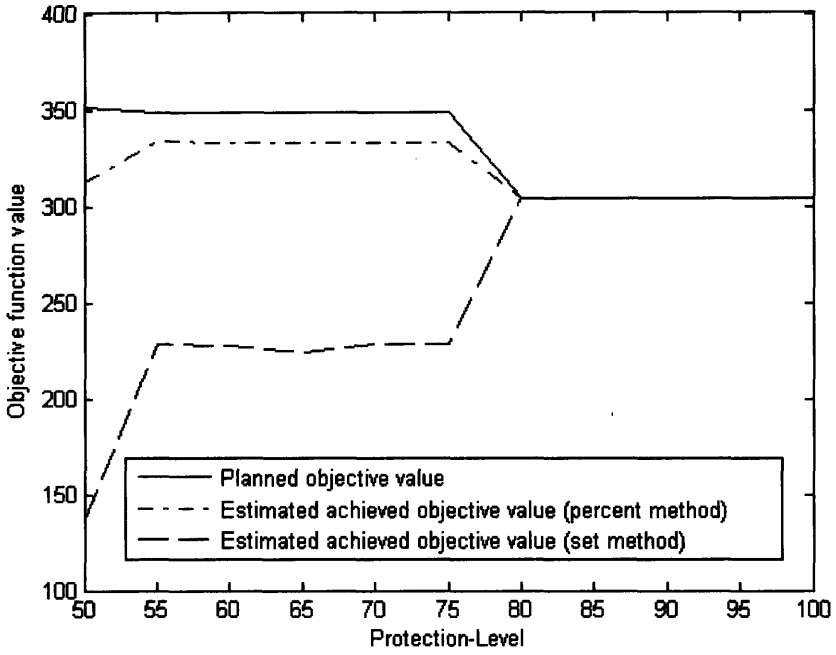
though that at a lower level, the squadron planner may find an easy change that makes the plan feasible without causing a theater re-plan. Thus, the Percent Method seems to represent reality better than the Set Method.

We used the Percent and Set Methods on the results from the simulation data given in Tables 4.1 and 4.2. The mean estimated achieved values from both methods are presented in Figure 4-15.

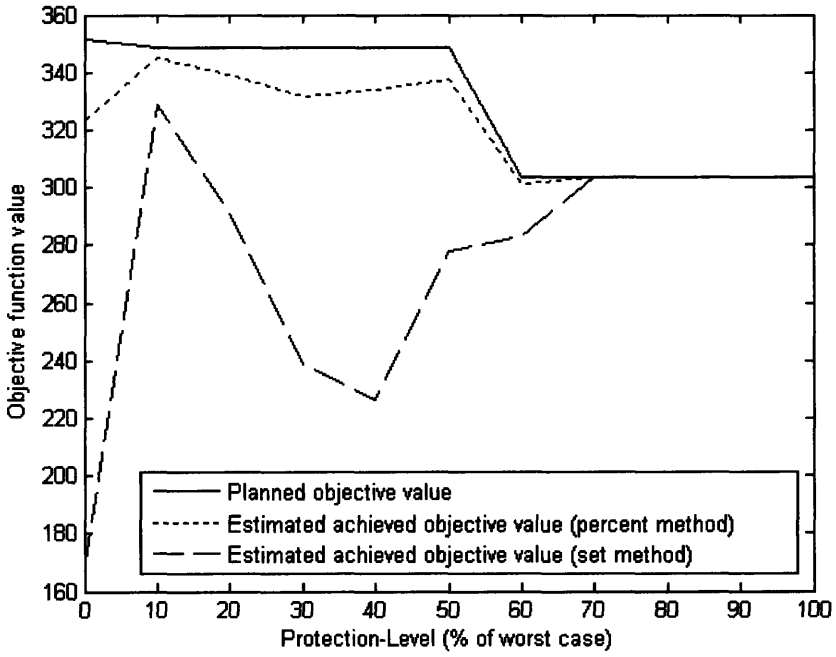
The set method is obviously more susceptible to constraint violations, as shown in (b) with the sharp decrease in the mean expected achieved value at protection-levels 20, 30 and 40. This is because one constraint violation nullifies the value of an entire task-set. If constraint violations are in multiple task-sets, the plan can lose nearly all of its value. This particular scenario highlights how different time assignments affect a plan's value.

For protection-levels 55-75 for the Chance-Constrained EBO Model and 10-50 for the Bertsimas/Sim EBO Model, the percent method value increases from the deterministic case and then decreases for higher protection-levels. This indicates that the violations we avoid by protecting the plan at these protection-levels allows us to gain more value that if the plan was not protected. However, if we protect the plan too much, we decrease the expected achieved value, indicating we have planned too conservatively.

The expected achieved value of the scenario from Figure 4-8 is in Figure 4-16. In Figure 4-16 we see that the plan generated at protection-level 0 has a lower mean expected achieved value than



(a) Chance-Constrained



(b) Bertsimas/Sim

Figure 4-15: Expected Achieved Objective Value

those at 10, 20, and 30. Recall that the plan at protection-level 0 has only a few very demanding tasks that cause violations often and the plan for 10, 20, and 30 has more tasks, which are connected to less valuable effects. These are not as demanding so they are violated less often. Because there are only a few tasks and one effect in the plan for protection-level 0, if we have a violation in this plan, we lose most of the value using the Percent Method, and all of the value using the Set Method. Because there are two effects being cause in the plans for protection-levels 10, 20, and 30, and many tasks involved: if we lose a task we lose a smaller portion of the effects value and we still get the value from the second effect. In Figure 4-12 (b), we see that the plan at protection-level 0 had more realizations that did not fail than for 10, 20, and 30. However, a failure in protection-level 0 is far more costly than in protection-levels 10, 20, and 30. Thus, the plans in protection-levels 10, 20, and 30 on average achieve more value than the plan at protection-level 0.

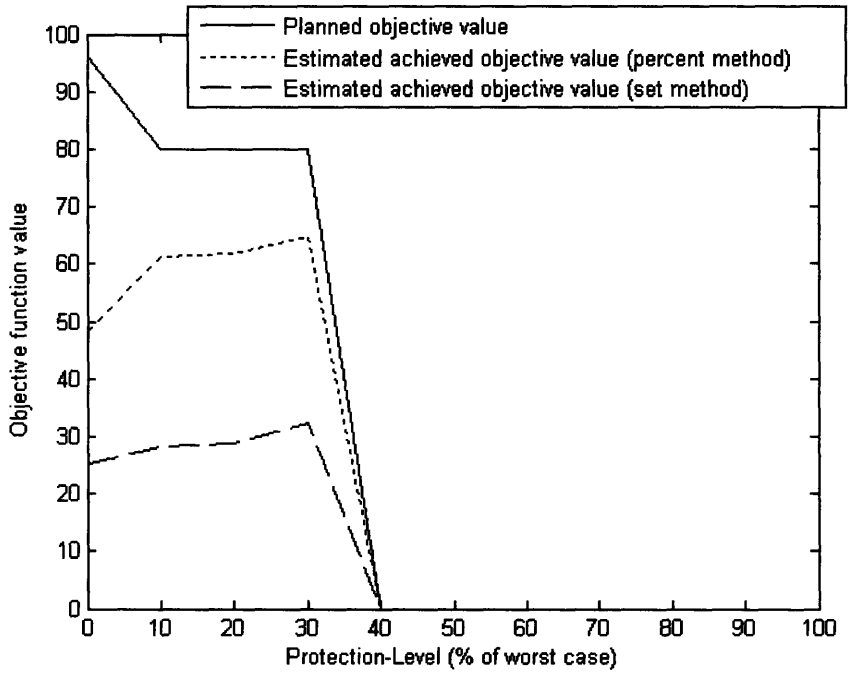


Figure 4-16: High-Demand Task Scenario Estimated Achieved Objective Value

4.1.9 Conclusions on Robust EBO Model Performance

In Section 4.1 we stepped through several small scenarios to explain the performance of the Robust EBO Models. We wanted to determine if the robust plans actually last longer and require fewer re-planning iterations than deterministic plans, hence achieving better overall value. We hypothesized that robust plans created by both the Chance-Constrained and Bertsimas/Sim EBO Models encounter fewer constraint violations, encounter violations in later time-periods, and require less re-planning than deterministic plans. We also hypothesized that the achieved value of a robust plan created by either the Chance-Constrained or Bertsimas/Sim EBO Model will be greater than the achieved value of a deterministic plan.

Deterministic plans (protection-level 50 for the Chance-Constrained EBO Model and protection-level 0 for the Bertsimas/Sim EBO Model) fail frequently. For all the scenarios tested in this section, all of the deterministic plans had mean time-periods until first constraint violation in the first or second time-periods. They also never had more than 1% of realizations that encountered no violations. This frequent failure rate resulted in significantly lower estimates of achieved value as shown in Figures 4-15 and 4-16. *The frequent failures of the deterministic plans demonstrate the need for robust planning.*

The Monte Carlo simulations from this section highlight many trends in the performance of the Robust EBO Models. For both models, we found that *as we increased the protection-level, the planned objective function value moves down in a stepwise manner.* This is due to the all-or-nothing nature of planning based on accomplishing effects. Also for both models, we see a general decrease in constraint violations as we increase the protection-level; however, the frequency of constraint violation and the percent of constraints violated is not guaranteed to decrease monotonically. The non-monotonic nature of the constraint violation statistics is due to two reasons. First, there are usually multiple optimal solutions for each plan, which simply involve assigning tasks to the same squadrons, but at different time-periods. Some time assignments can cause tighter constraints, which cause more frequent violations. Second, some scenarios can have higher-value, more-demanding task-sets, which activate few constraints. An increase in the protection-level can cause the plan to drop task-sets but add more less-demanding tasks than the previous plan, resulting in more active constraints and more violations. *In general, the frequency of constraint violation and the percent of constraints violated are not consistent measures of robustness; they do*

not necessarily indicate how long a plan will last or how much value a plan will achieve.

Because of the inconsistencies of constraint violation in indicating how a plan will perform, we looked at the time until failure. We found that this is best shown in the hazard rate plots, which give indications of what percentage of the time a plan will fail in each time-period and what percentage of the time the plans will not fail. We also developed two methods of estimating the value a plan will achieve based on its simulated performance, the set-method (worst-case estimate) and the percent-method (generous estimate). Using the hazard rate plots and the estimated value achieved plots can give a good indication of the performance of the plans across the range of protection-levels.

We found that the differences in task-time assignments can have significant impacts of the time until failure and estimated value achieved for plans that have the same planned objective value. *This indicates the need for post-processing of the plans to ensure time-assignments are made in such as way as to spread capacity among constraints as much as possible, once the best objective value has been obtained.*

In general we found that the robust plans generated by both the Chance-Constrained EBO Model and the Bertsimas/Sim EBO Model perform far better than the deterministic plans. The robust plans have longer mean time-periods until failure and higher estimated achieved objective function values. *As we increase the protection, we see a general increase in performance to a point where we begin to plan too conservatively and the estimated achieved value starts to decrease because we are planning to do less. This point is difficult to determine due to the changes in plan performance caused by the differing task-time assignments, and because the best point must be subjectively determined balancing the risk involved with possible plan failure versus the possible value achieved by the plan.*

In the following sections, we want to compare the performance of the robust EBO models against an estimate of current planning techniques. We also want to see if our performance conclusions hold true in planning scenarios of different sizes and structures. Finally, we want to determine if either the Chance-Constrained EBO Model or the Bertsimas/Sim EBO Model performs better as the method to create robust plans.

4.2 Greedy Algorithm versus EBO Models

We want to determine how the EBO models perform relative to a human planner. In Section 3.1.6, we introduced a simple greedy algorithm that assigns tasks according to the EBO Framework. We use it as an approximation for the performance of a human planner making a plan using the EBO Framework. We hypothesize that the greedy algorithm will not achieve objective function values as high as the Deterministic EBO Model's values because it does not solve to optimality. We also hypothesize that the greedy algorithm will have a shorter time until failure and more frequent constraint violations than the Deterministic EBO Model because it greedily assigns tasks starting with the latest possible time-periods. Finally, we hypothesize that the greedy algorithm will have a higher planned objective function value than that of some robust plans but will have a significantly lower time until failure and estimated realized objective function value because it is not robust.

The greedy algorithm does not necessarily solve to optimality, but can sometimes find a solution that has a planned objective function value equal to the optimal solution found by the Deterministic EBO Model. Because the greedy algorithm assigns tasks, starting with the latest available time-period, it is unlikely that the greedy algorithm finds the exact same plan as the Deterministic EBO Model. It might, however, find a plan that uses the same task-sets and tasks as the Deterministic EBO model, but with different time assignments. It cannot find a plan with a planned objective function value higher than the Deterministic EBO Model.

The greedy algorithm does not consider robustness in finding a solution, but simply assigns the task-sets in order of marginal benefit. Because it greedily assigns tasks to squadrons until a squadron is at capacity, then moves to another squadron, its plans can be fragile. Because it does not find the optimal solution, it can produce plans that involve fewer tasks than the plans produced by the Deterministic EBO Model. Therefore, it can demonstrate more robust behavior than plans generated by the Deterministic EBO Model.

4.2.1 Test Scenarios and Simulation

To test the performance of the greedy algorithm against that of the robust EBO models, we want to compare the planned objective function values, constraint violations, the mean times until failure, and the estimated achieved objective function values from these models. We introduce three new scenarios to test the models. In conjunction with those presented previously, they span a range of

Table 4.5: Test Scenarios

Scenario	Tasks	Effects	TSGs	Time Periods	Squadrons	Strike UAVs/Squad	ISR UAVs/Squad
1	36	4	14	6	3	4	1
2	103	6	24	7	3	4	1
3	80	2	46	7	3	4	1
4	78	9	13	7	3	4	1

problem sizes and scenario types. We summarize them in Table 4.5. Scenario 1 is the same scenario we introduced in Figure 4-1. We include it for comparison against the plans generated by the EBO models discussed in Section 4.1. Scenario 2 has a similar effect to TSG ratio as Scenario 1 but has about three times more tasks. Scenario 3 has few effects, but many options to accomplish each effect (i.e. many TSGs per effect). Scenario 4 is the opposite, having many effects and few options to accomplish each effect.

We create plans for each of these scenarios using the robust EBO models and the greedy algorithm. We test the plans using the Monte Carlo Simulation described in Section 4.1.1. For each scenario, we model uncertainty the same as we did in Section 4.1: all uncertainty is limited to the right-hand-side values, each random variable is modeled as a uniform random variable that varied from one half of the expected value up to one and a half times the expected value. As in Section 4.1, we set the expected value of λ^{st} (the number of strike tasks a strike UAV can perform per time-period) as 2, λ^{isr} (the number of ISR tasks a strike UAV can perform per time-period) as 2, and γ (the number of ISR tasks an ISR UAV can perform per time-period) as 4.

Scenarios 2, 3, and 4 have nearly three times as many tasks as Scenario 1. They also have seven time-periods instead of six. This greatly increases the problem size. When running the Monte Carlo Simulation described in Section 4.1.1, the EBO models and the greedy algorithm solved within a few seconds for each scenario. However, the increase in problem size increases the number of random variables that must be generated, which is the slowest part of the Monte Carlo Simulation. Thus, for Scenario 2, 3, and 4 we limited the number of iterations to 1000 instead of 3000 as previously done with Scenario 1. We ran the simulations on the same computer, and each run took approximately one and a half hours to run, which includes running the greedy algorithm, solving both robust EBO Models, and simulating realizations at each protection-level.

4.2.2 Greedy Algorithm versus Deterministic EBO Model Performance

The results of the simulations for the greedy algorithm and the deterministic case protection-levels of both robust models are given in Table 4.6. Because the Chance-Constrained EBO Model and the Bertsimas/Sim EBO Model solve to within .001% of optimality they can arrive at slightly different squadron to task assignment solutions. Thus, we present the deterministic case for both (protection-level 50 for the Chance-Constrained EBO Model and protection-level 0 for the Bertsimas/Sim EBO Model).

Table 4.6: Deterministic EBO Model vs. Greedy Algorithm Results

Plan	Planned Obj. Funct. Value	Percent Method Obj.	Set Method Obj.	Mean Freq. of Cstr. Viol.	Mean Time-Period Until Plan Fails
Scenario 1					
Greedy	332.0	323.1	200.3	.0269	2.652
Chance Det.	346.3	312.8	137.9	0.1730	2.180
Bert/Sim Det.	346.3	323.5	169.7	0.1115	1.649
Scenario 2					
Greedy	203.3	172.9	110.7	0.0551	3.035
Chance Det.	208.9	185.0	126.5	0.1846	1.932
Bert/Sim Det.	209.1	175.7	121.1	0.1777	1.871
Scenario 3					
Greedy	388.2	374.2	325.6	0.0808	2.174
Chance Det.	388.9	380.0	331.4	0.170	2.251
Bert/Sim Det.	389.1	377.5	296.4	0.201	2.233
Scenario 4					
Greedy	649.1617	496.4970	225.7790	0.1479	2.5560
Chance Det.	655.3	481.7	170.6	0.2197	1.6690
Bert/Sim Det.	655.3	485.3	149.0	0.1833	1.8260

3000 realizations for Scenario 1 and 1000 realization for all others, uniformly distributed uncertain coefficients, 50% variation from expected value

Because the greedy algorithm does not solve to optimality, the greedy algorithm creates a plan with a planned objective function value less than that of the deterministic EBO Model, for all of the scenarios. The greedy algorithm generates planned objective function values that are very close to those of the deterministic EBO models (average 2% difference). For Scenario 3, the greedy algorithm finds a plan that uses the same tasks and task-sets as the plan generated by the Deterministic EBO Model. For Scenarios 1, 2, and 4, the greedy algorithm finds plans that use different task-sets than those generated by the Deterministic EBO Model. The greedy algorithm does not create

squadron to task assignments that are as good as the Deterministic EBO Model. In each case, the distance between the tasks and their assigned squadrons is greater for the plans generated by the greedy algorithm than those from the Deterministic EBO Model. If the greedy algorithm is a good approximation for a human planner, this indicates that the Deterministic EBO Model will usually create plans that have a higher planned objective value than the greedy algorithm; and the greedy algorithm's plan, at its best case, can only match the deterministic plan.

Although the plans generated by the Deterministic EBO Model have higher planned objective value functions, this does not mean that they are better plans. Actually, because the deterministic plans include more tasks than the greedy plans; they usually have tighter constraints, encounter more violations, and have lower mean times until failure. For some of the scenarios, this results in plans generated by the Deterministic EBO Model that have percent method and set method values less than those of the plans from the greedy algorithm. Because the greedy algorithm does not solve to optimality, it leaves more slack in the plans than the Deterministic EBO Model plans, resulting in less fragile plans that can achieve more value under the realized data. *This indicates that it is possible to replace a human planner with a computerized planner that solves for "optimal" plans that have higher planned value than those of the human planner; but because the computerized planner creates plans with less slack, the plans end up achieving less value when executed due to frequent failures. This highlights the need for robust planning.*

4.2.3 Robustness versus Greedy

If the incidental slack from the greedy algorithm, due to not finding an optimal solution, causes better estimated achieved value; intelligently adding slack with the Robust EBO Models should provide even better plans. If the robust EBO models create plans at certain protection-levels that do not perform better than those generated by the greedy algorithm, then they are not an improvement on human planning methods. To investigate this, we look at the hazard rate plots, times until failure, and estimated achieved objective function value of plans generated by the robust EBO models across the range of protection-levels compared to the performance of plans generated by the greedy algorithm.

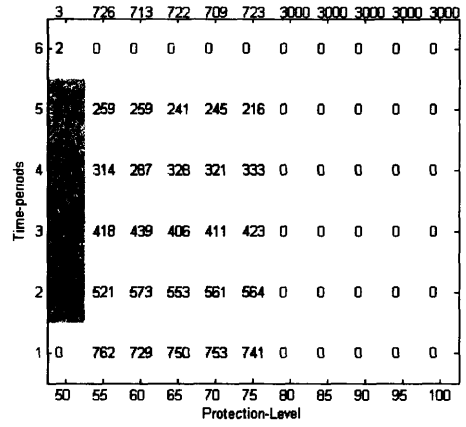
Figure 4-17 shows the comparison of the plans for Scenario 1. Notice in the hazard rate plots ((a). (b), and (c) of Figure 4-17) that for the deterministic case (i.e. protection-level 50 for chance-

constrained and 0 for Bertsimas/Sim) the greedy algorithm has 252 realizations that encounter no constraint violations, but the EBO models only have 8 and 3 respectively. Subfigures (d), (e), (f), and (g) in Figure 4-17) show that the greedy algorithm has a higher mean time until the first violation, percent method estimate, and set method estimate. At protection-level 55 for the Chance-Constrained EBO Model and 10 for the Bertsimas/Sim EBO Model, the robust EBO models outperform the greedy algorithm in all categories. In fact, at protection-level 10 for Bertsimas/Sim, the set method estimate (worst-case estimate) actually surpasses the percent method estimate (optimistic estimate) of the greedy algorithm. The robust EBO models perform better because they protect each constraint as we increase the protection-level; whereas, the greedy algorithm incidentally protects certain constraints because it greedily pushes others to capacity.

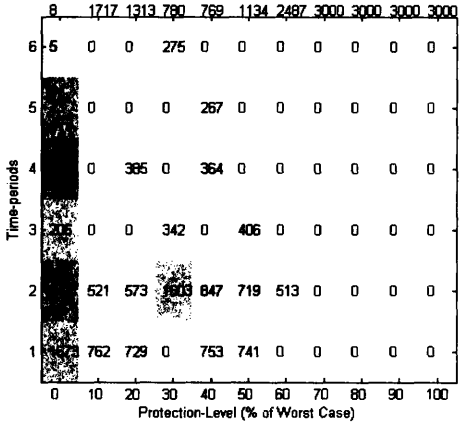
For protection-level 80 for the Chance-Constrained EBO Model and 60 for the Bertsimas/Sim EBO Model and all higher protection-levels, the plans are more conservative than the one generated by the greedy algorithm. In these cases, the planned objective function value of the robust models is lower than that of the greedy algorithm; however, these plans experience no failures. Depending on the acceptability of risk, in some situations, this lower planned level with no risk might be preferable.

The other scenarios performed similarly to Scenario 1. In general, the robust models create plans at the deterministic protection-level that plan to achieve more value than the greedy algorithm, but encountered more violations, resulting in lower estimated achieved values. When the protection-level is increased slightly, the robust models outperform the greedy algorithm in all categories except the planned objective function value. Figures 4-18 show the results for Scenario 2. Notice in Figure 4-18 (f) and (g) that with the higher protection-levels we can achieve nearly the same planned objective function value as the greedy algorithm with no constraint violations.

Figure 4-19 shows the results for Scenario 3. Scenario 3 only has two effects, but has 31 TSGs for effect 1 and 15 TSGs for effect 2. Therefore, most plans of Scenario 3 involve many task-sets that accomplish the same effect. The marginal benefit of a task-set decreases as we plan to do more. Thus, for plans that have several task-sets that accomplish the same effect, if constraint violations occur that cause us to lose ability to accomplish a task-set, the loss is not as great as in other scenarios because the remaining task-sets will still give a relatively high expected value for the effect. We see in Figures 4-19 that constraint violations are not as costly as in other models;

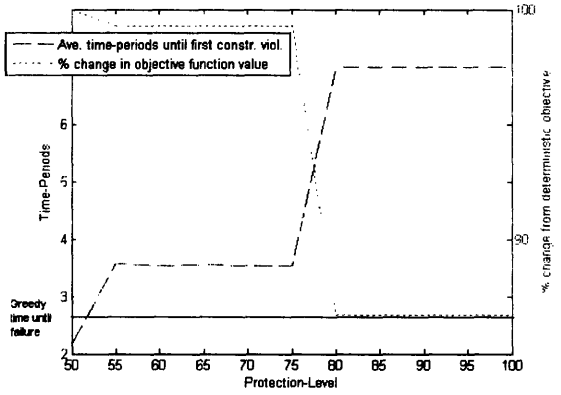


(a) Chance-Constrained

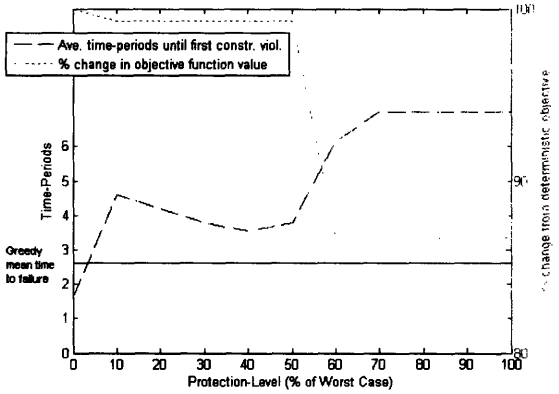


(b) Bertsimas/Sim

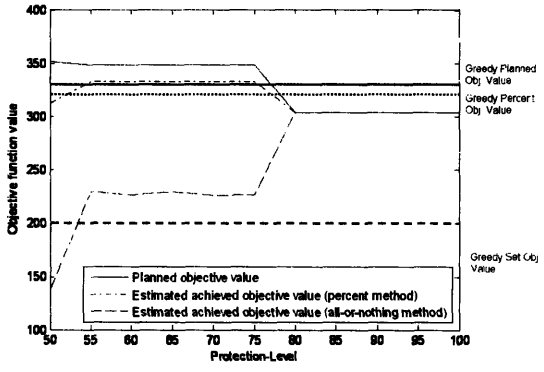
(c) Greedy



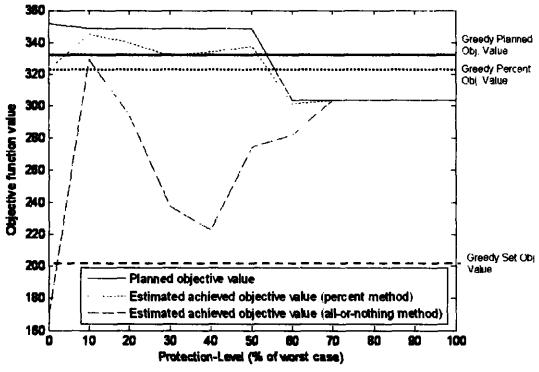
(d) Chance-Constrained



(e) Bertsimas/Sim

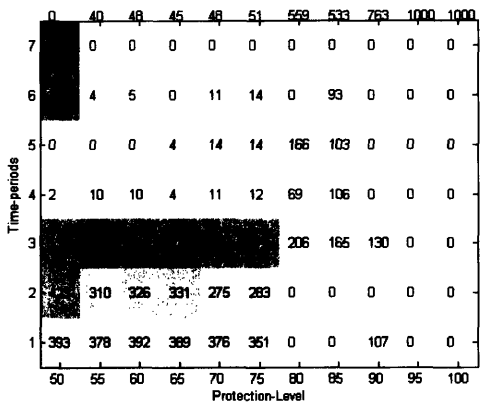


(f) Chance-Constrained

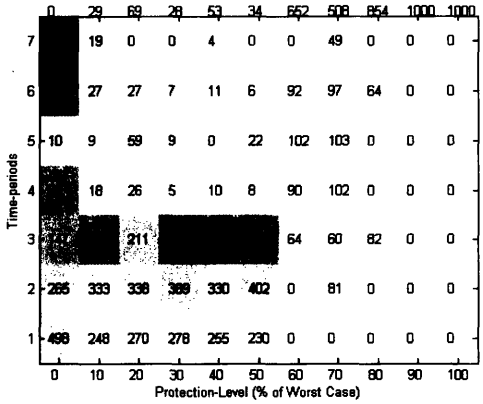


(g) Bertsimas/Sim

Figure 4-17: Scenario 1, Robust vs. Greedy Results

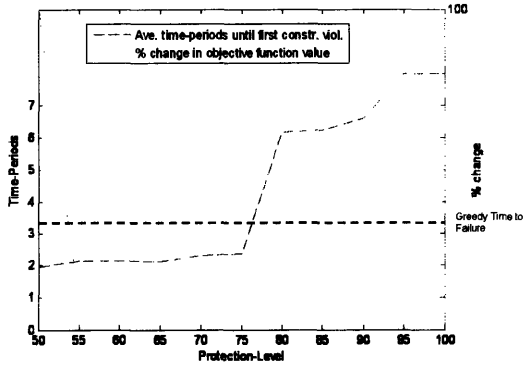


(a) Chance-Constrained

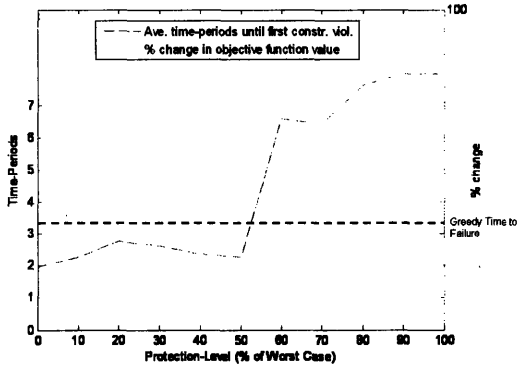


(b) Bertsimas/Sim

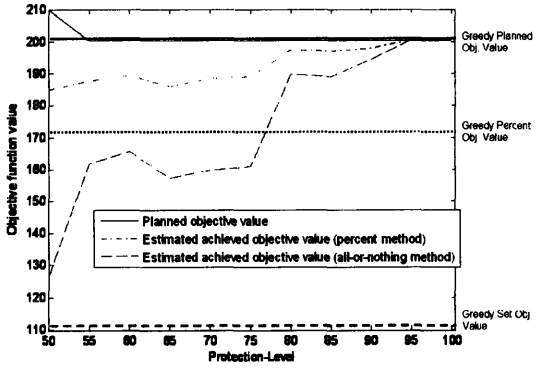
(c) Greedy



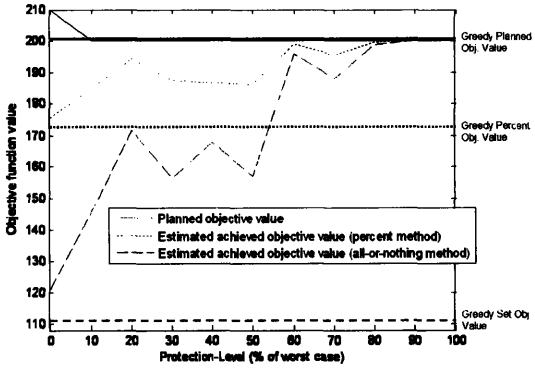
(d) Chance-Constrained



(e) Bertsimas/Sim



(f) Chance-Constrained



(g) Bertsimas/Sim

Figure 4-18: Scenario 2, Robust vs. Greedy Results

the set method value for the deterministic case is about three quarters of the planned objective, whereas it is less than half of the planned value for all other scenarios. Furthermore, this is the only scenario where the percent method value decreases as we increase the protection-level from the deterministic case to the next higher protection-level.

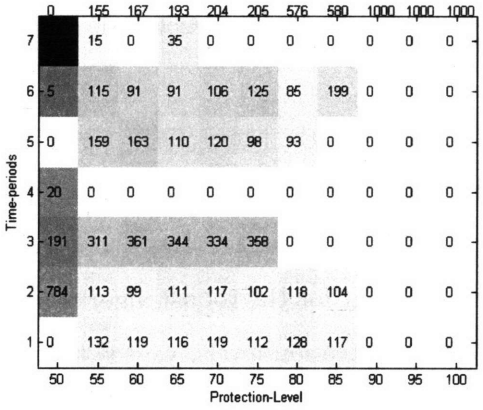
Because of the protection given by many task-sets that are causing the same effect, the greedy algorithm has a percent method estimate that is higher than that of the robust model's up until the last few protection-levels, where the robust models encounter no violations. Even though the percent method estimate for the greedy algorithm is higher than that for the robust algorithms, if we look at the hazard rate plots in Figure 4-19, for the middle protection-levels, we see that the robust algorithms have far more realizations that do not encounter violations. Also, Figure 4-19 (d) and (e) show that the robust models have longer times until the first violations.

Scenario 4 is the opposite of Scenario 3 in structure. Scenario 4 has many effects and few options to cause each effect, making the loss of a task-set due to constraint violations very costly to the plan. We can see this in Figure 4-20. The plans generated by the robust EBO models at their deterministic equivalent protection-levels and the plan generated by the greedy algorithm all have set method estimates of more than two-thirds less than their planned objective function values. This is in stark contrast to the slight losses seen in Scenario 3. In Scenario 4 the plans at higher protection-levels encounter no violations with only a small amount of loss from the deterministic planned objective value.

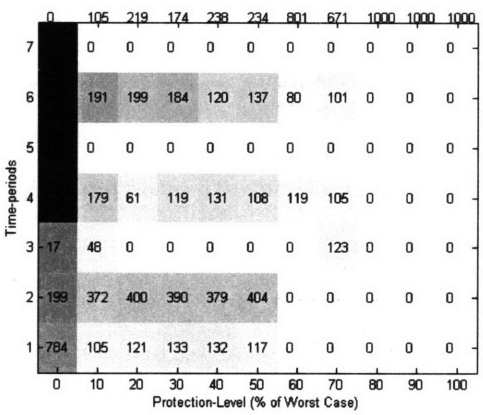
As with the other scenarios, the plan generated by the greedy algorithm for Scenario 4 obtains a planned objective function value near that of the plan generated by the deterministic EBO models with fewer constraint violations. As the protection-level increases, the plans created by robust EBO models encounter fewer violations and achieve a guaranteed value higher than the estimated achieved value of the greedy algorithm plan.

4.2.4 Conclusions on Greedy Algorithm Performance

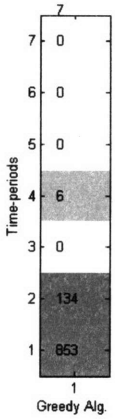
By comparing the greedy algorithm to the EBO models, we can make several conclusions. *First, even incidental slack added to the system because the algorithm does not solve to optimality will cause the performance of the plan to be more robust.* This is demonstrated by the fact that the greedy algorithm plans have longer mean times until failure and higher estimated achieved objective values



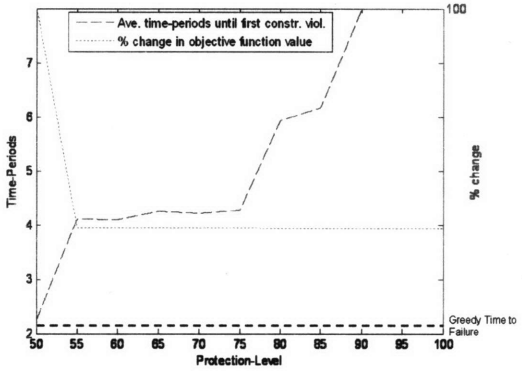
(a) Chance-Constrained



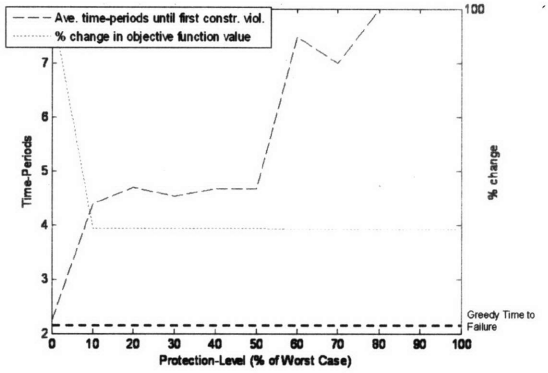
(b) Bertsimas/Sim



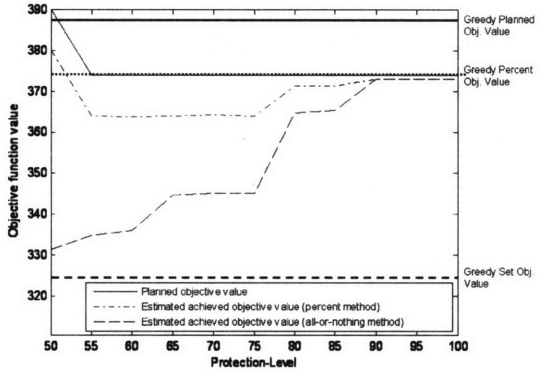
(c) Greedy



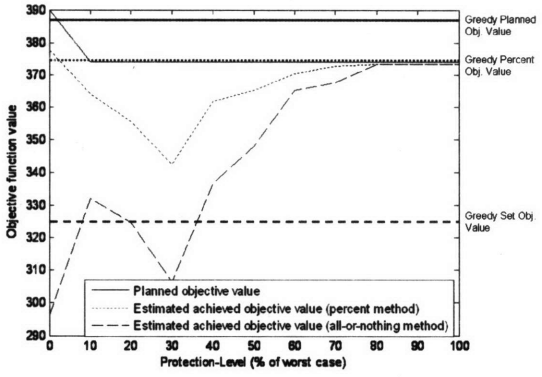
(d) Chance-Constrained



(e) Bertsimas/Sim

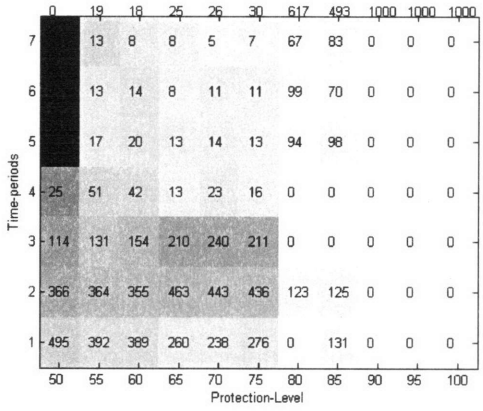


(f) Chance-Constrained

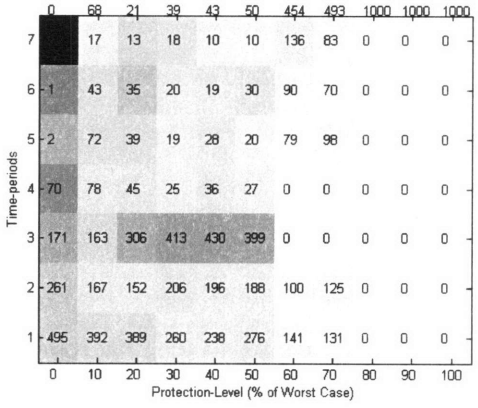


(g) Bertsimas/Sim

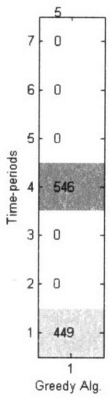
Figure 4-19: Scenario 3, Robust vs. Greedy Results



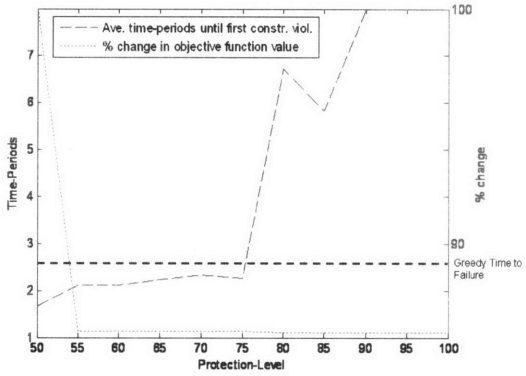
(a) Chance-Constrained



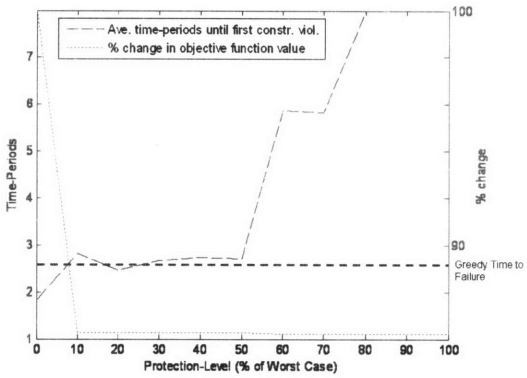
(b) Bertsimas/Sim



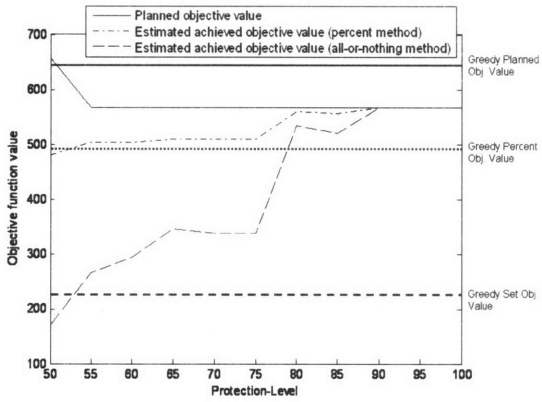
(c) Greedy



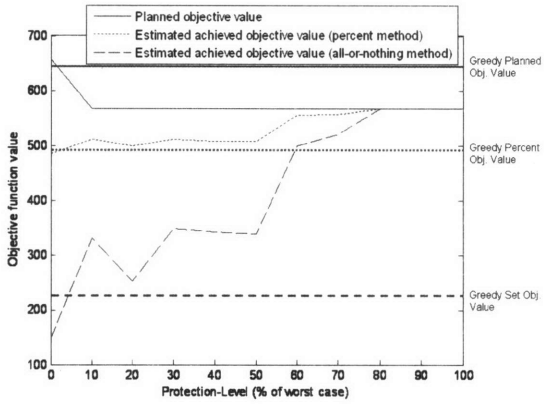
(d) Chance-Constrained



(e) Bertsimas/Sim



(f) Chance-Constrained



(g) Bertsimas/Sim

Figure 4-20: Scenario 4, Robust vs. Greedy Results

than the deterministic EBO model plans. Assuming the greedy algorithm is a good estimate of the performance of a human planner, *this shows that solving to optimality using a deterministic model often performs worse than the human generated plan when executed.* This is a strong indication for the need for robust planning. *The robust planning models all had certain protection-levels that outperform the greedy algorithm in all performance metrics* (set method value, percent method value, and time until failure). This shows that intelligently adding slack by using one of the robust EBO models is better than the incidental slack caused by a human planner who does not arrive at the solution with the highest planned objective function value. If the robust EBO models are used, a human planner will still have to decide how much protection is appropriate for the plan. We will discuss how a human planner does this in Chapter 5.

4.3 Robust EBO Model Comparison

We not only want to determine if the plans generated by the robust EBO models perform better than those from the Deterministic EBO Model and the greedy algorithm, but we also want to determine which of the robust formulations creates the best plans. As discussed in Section 4.1.9, as the protection-levels increase, the amount of constraint violations tend to decrease, which, on average, causes the plans to last longer and have a higher estimated achieved objective function value. There are several exceptions to these general trends: plans can achieve the same planned objective function value but the tasks involved in the plan can be assigned in different time-periods resulting in different constraint violation rates and a plan created at a particular protection-level might use heavily weighted tasks that activate few constraints; whereas, plans created using higher protection-level, might have more active constraints resulting in more frequent constraint violations.

With the exception of performing differently due to finding different task-time assignments, the Chance-Constrained EBO Model and Bertsimas/Sim EBO Model create plans that perform similarly on the scenarios presented in Section 4.1. We want to determine if when tested under different scenarios and different models of uncertainty, if either the Chance-Constrained EBO Model or the Bertsimas/Sim EBO Model stand out as the better method for creating robust plans. We hypothesize that, with the exception of differing performance based on which of the multiple-optimal solutions the solver finds, plans generated with both models will perform similarly regardless of scenario structure when realizations are generated using uniform random variables and all uncer-

tainty is located in the right-hand side. We also hypothesize that when the uncertain variables are modeled with a truncated normal distribution, the Bertsimas/Sim EBO Model will create more protected plans at lower protection-levels than the Chance-Constrained EBO Model because it will be less likely for coefficients to approach their worst-case value.

4.3.1 Performance under Different Scenario Structures

We tested the plans generated by the Chance-Constrained EBO Model and the Bertsimas/Sim EBO Model for the Scenarios 2, 3, and 4 described in Section 3.1.6. These scenarios have significantly different scenario structures. Scenario 2 has two or three task-sets that can accomplish each effect, translating to seven or fewer TSGs for each effect. It is similar in structure to the Scenario 1, presented in Figure 4-1 but has about three times more tasks. Scenario 3 only has two effects, but has four and five task-sets for each effect. This translates to up to 31 TSGs for one effect. Scenario 4 has nine effects, more effects than the other scenarios, but uses only one or two task-sets for each effect. These differences in structure affect the amount of value lost as we increase the protection-level of a plan, but they affect the plans generated by the Chance-Constrained EBO Model and the Bertsimas/Sim EBO model similarly.

For the scenarios shown in Figures 4-17 through 4-20 the Chance-Constrained EBO Model and Bertsimas/Sim EBO Model create plans that perform similarly. Differences in performance can only be attributed to different task-time assignments as discussed in Section 4.1.6 and a few different squadron-task assignments due to the fact that the models only solve to a bound of optimality (0.001%). The reason that the Chance-Constrained EBO Model and the Bertsimas/Sim EBO Model create plans that perform similarly when the uncertainty is all in the right-hand-side and modeled using a uniform distribution, is that the models plan at exactly the same amount of protection if we match each protection-level interval of 5 for the Chance-Constrained EBO Model to each protection-level interval of 10 of the Bertsimas/Sim EBO Model (i.e., 50 is the same as 0, 75 is the same as 50, and 100 is the same as 100 for Chance-Constrained and Bertsimas/Sim respectively.)

To demonstrate this, recall that when we model uncertainty only in the right-hand-side, the

Chance-Constrained EBO Model protects constraints using the following:

$$\sum_{j \in J} a_{ij} x_j \leq F_{\mathrm{B}_i}^{-1}(1 - \alpha_i) \quad \forall i \in I, \tag{4.1}$$

where $F_{\mathrm{B}_i}^{-1}(1 - \alpha_i)$ is the density function of B_i and α_i is the protection-level or percent of the time that constraint i will be feasible when we generate realizations for the random variables. For example, assume B is a uniform random variable with mean 8, and it varies between 4 and 12. If α_i is 0.50, then we set the right hand side of the constraint to 8. If α_i is 0.75, then we set the right-hand-side of the constraint to 6.

When the random variables are uniform and uncertainty only exists in the right-hand-side, the Bertsimas/Sim EBO Model works exactly the same way, though for a different reason. Recall that Bertsimas/Sim protects constraints with the following:

$$\sum_{j \in J} a_{ij} x_j + \max_{S_i \cup \{t_i\} | S_i \subseteq J_i, |S_i| = \lfloor \Gamma_i \rfloor, t_i \in J_i / S_i} \left\{ \sum_{j \in S_i} \hat{a}_{ij} y_j + (\Gamma_i - \lfloor \Gamma_i \rfloor) \hat{a}_{it_i} y_t \right\} \leq b_i. \tag{4.2}$$

The Γ_i parameter finds the Γ_i coefficients that are most detrimental to the objective function if they go to their worst-case value ($a_{ij} \pm \hat{a}_{ij}$), and solves for the case for which the solution is feasible if these coefficients behave as such. In the case where all of the uncertainty is in the right-hand-side coefficient, there is only one coefficient per constraint that can vary. Thus, Γ_i can range from 0 to 1 for each constraint i. When Γ_i is non-integer, the decimal values indicate that one constraint is moving $\Gamma_i - \lfloor \Gamma_i \rfloor$ percent of the way to its worst case value. Thus, for the example given for the Chance-Constrained constraint with right-hand-side value with expected value of 8 and a range up or down of 4, Γ_i set to 0 (corresponding to a protection-level of 0) leaves the coefficient at its expected value, Γ_i set to 0.50 (corresponding to a protection-level of 50) moves the coefficient 50% of the way to its worst case, and Γ_i set to 1 (corresponding to a protection-level of 100) moves the coefficient all the way.

This shows that the Bertsimas/Sim EBO Model performs exactly like the Chance-Constrained EBO Model when uncertainty is uniformly distributed and only located in the right-hand-side. Thus, the difference in performance of the plans from the robust models tested against the plans from the greedy algorithm in Section 4.2.3 in Figures 4-17 through 4-20 can only be attributed to

differing multiple optimal solutions or solving only to within a bound of optimality.

4.3.2 Performance under Different Kinds of Uncertainty

Although the Chance-Constrained EBO Model and the Bertsimas/Sim EBO Model perform the same when the uncertainty is all in the right-hand-side and the random variables are uniform, if we change the distribution, they will no longer perform the same. To determine the differences that non-uniform random variables might have on the models, we tested the four scenarios with truncated-normal random variables.

Truncated Normal Random Variables

We have to use a truncated-normal distribution to generate the random variables, because a regular normal distribution has no maximum or minimum values. Maximum and minimum values are necessary for the Bertsimas/Sim EBO Model, and right-hand-side values cannot be negative without creating an infeasible problem. Thus, we had to truncate the normal random variables.

To use a normal random variable for the Chance-Constrained EBO Model, we made the function $F_{B_i}^{-1}(1-\alpha_i)$ a normal cumulative distribution function (CDF). For any α_i value that causes $F_{B_i}^{-1}(1-\alpha_i)$ to be less than the minimum right-hand-side value, we simply held the right-hand-side at the minimum value for the plan. The Bertsimas/Sim EBO Model does not rely on distributions of the uncertain coefficients, and needs only the uncertain data's maximum and minimum values. If the maximum and minimun values of the uncertain data are the same as in the tests of the Bertsimas/Sim EBO Model with a uniform distribution, then the model will solve exactly the same, but will encounter different violation rates because the realization will now be generated with a truncated-normal distribution.

To generate the truncated-normal random variables for the testing, we simply generated a normal random variable with a known mean and variance. We make the variance a parameter as a function of the percent of the distance from the mean value of the coefficient to the worst-case value. For instance, if we have a coefficient that has a mean of 4 and a minimum value of 2 and we want the variance of the normal distribution to be 1, we give an input value of .5 (i.e. half the distance from 4 to 2 is 1, the desired variance). If the random number generator using a normal distribution with defined mean and variance generates a value outside the allowable data range, we

simply re-generate that value until it is in the data range. This changes the CDF of the normal distribution as shown in Figure 4-21.

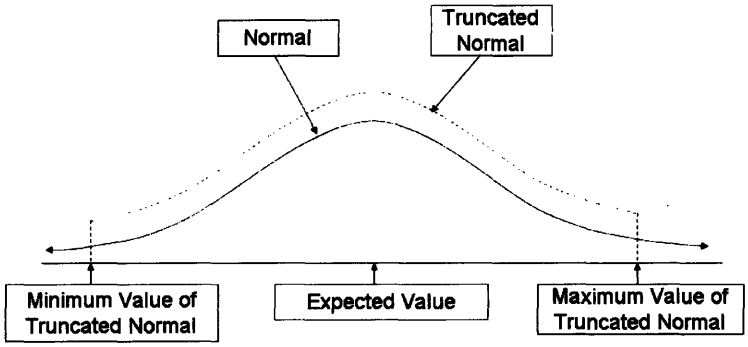


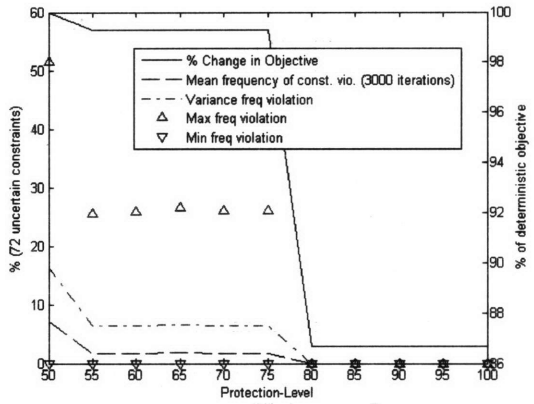
Figure 4-21: Normal vs. Truncated Normal Distribution

We ran the same Monte Carlo Simulation as before, but with the new distribution, to test the performance of the plans generated by the robust EBO models in the same scenarios but now using normal random variables. As stated in Section 4.2.1, the random number generator is the portion of the simulation that takes the longest. Generating the truncated-normal random variables takes up to three times longer than generating the uniform random variables from previous sections. Thus, for all runs with normal random variables, we only simulated 1000 iterations. We ran the simulations on the same computer, and all runs took between one and a half to three hours. The EBO models still solved within a few seconds. We set the variance of the normal data to be 25% of the distance from the expected value to the minimum value of each uncertain coefficient.

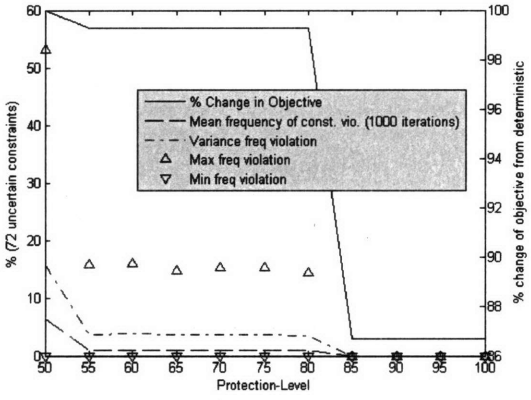
Figures 4-22 and 4-24 show the performance of Scenario 1 for Chance-Constrained EBO Model and the Bertsimas/Sim EBO Model. The figures on the right-hand-side show the models for the runs using normal random variables and the figures left show the performance of the runs previously done with uniform random variables.

For the plans generated by the Chance-Constrained EBO Model, there is little change in performance. In general, we see a decrease in the frequency of constraint violation for the middle protection-levels of the model with normal data. We also see that the plans generated under normal uncertainty maintain the first objective function step for one more protection-level. Otherwise, the plans across all protection-levels look very similar.

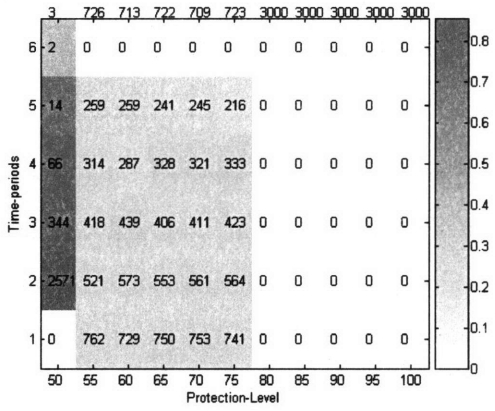
The reason for the decrease in the frequency of constraint violation, is that the normal coeffi-



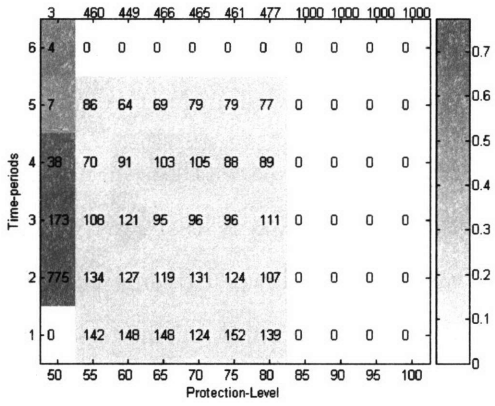
(a) Uniform Obj. Function and Freq. Viol



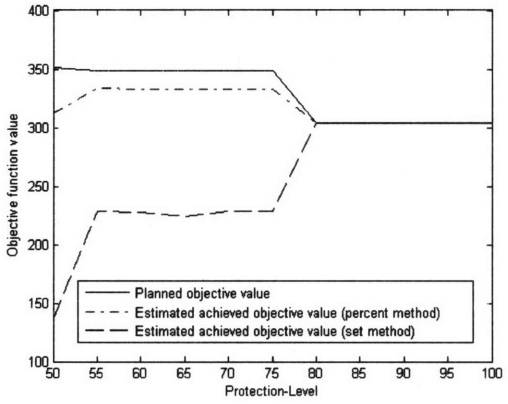
(b) Normal Obj. Function and Freq. Viol



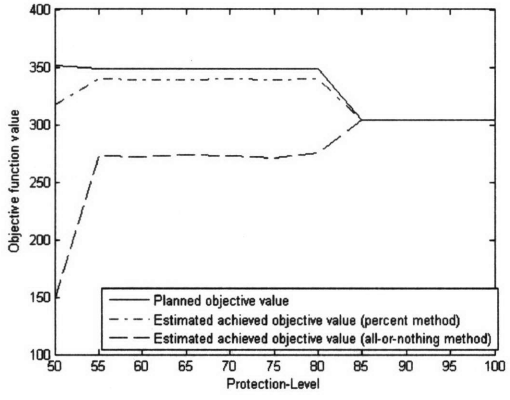
(c) Uniform Hazard Rate Plot



(d) Normal Hazard Rate Plot



(e) Uniform Estimated Achieved Obj. Func.



(f) Normal Estimated Achieved Obj. Func.

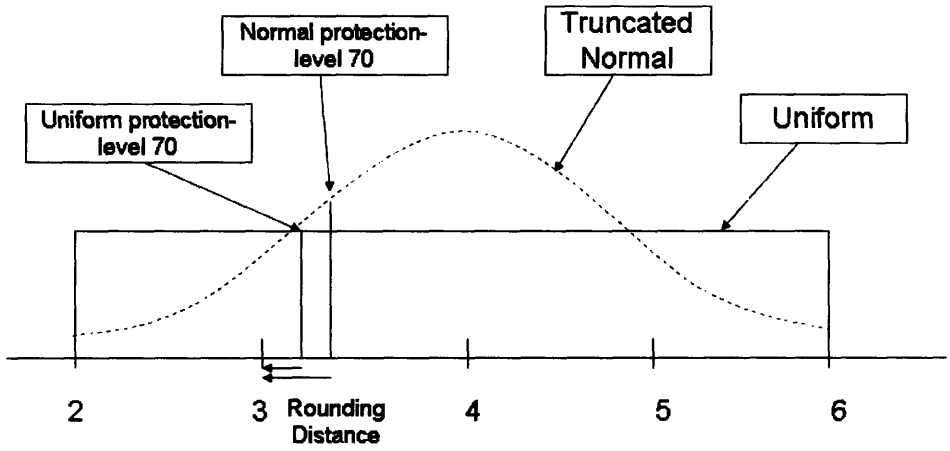
Figure 4-22: Scenario 1 Chance-Constrained Performance, Uniform vs. Normal

cients are slightly less variable than the uniform coefficients; meaning the normal coefficients are less likely to be at their extremes than the normal coefficients. Therefore, the uniform data is less likely to cause constraint violations. Figure 4-23 shows an example of why this happens. In this example, we assume the right-hand-side value have an expected value of 4 and can range from 2 to 6. The right-hand-side value protected for the normal case will always be larger than when protected for the uniform case when comparing equal protection-levels. Because the constraints have integer requirements, the right-hand-side will be treated as though it were rounded to the next lowest integer value. In most cases, the normal and uniform protected right-hand-sides will round to the same value, as shown for protection-level 60. In a few cases, they will round to different values, as shown for protection-level 80. This shows that the protection-levels for the normal case are less restrictive at higher protection-levels than for the uniform case. Thus, we see in Figure 4-22 (b) that the normal case maintains the first step down in the objective function value for one more protection-level than for the uniform case in (a).

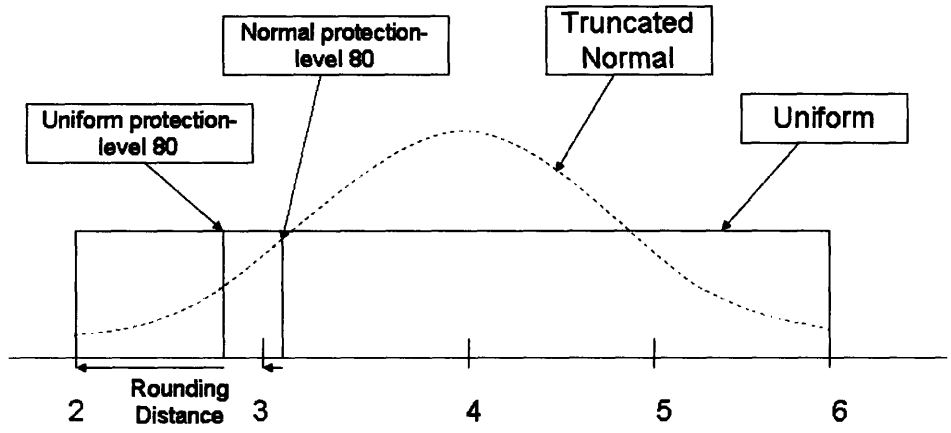
Figure 4-23 also shows that the normal CDF has more mass around the expected value than the uniform CDF. Thus, it is less likely when we generate normal coefficients that they will be realized with values at the lower extreme than for uniform coefficients. Because the right-hand-sides are treated as the same number, due to the integer rounding, the less-variable normal coefficients cause fewer violations. We see this in 4-22 (a) and (b). The fewer violations allow the plan to encounter no violations for more realizations as shown in the hazard rate plots ((c) and (d)). The fewer violations also push the estimated achieved values higher, as shown in (e) and (f).

Figure 4-24 shows the performance of the Bertsimas/Sim EBO Model with normal and uniform uncertain coefficients. As we stated before, when uncertainty exists only in the right-hand-side, the Bertsimas/Sim formulation protects by decreasing the right-hand-side by of each constraint i by Γ_i times the range that the coefficient can vary, protecting like the Chance-Constrained EBO Model with uniform uncertainty. Since the Bertsimas/Sim EBO Model does not use the distribution of the data to determine how it will protect, it finds the exact same plans with normal uncertain data as it does with uniform uncertain data.

When tested against realized data the plans have fewer constraint violations. This is because the normal data varies less than the uniform data. The plots in Figure 4-24 show the fewer violations and resulting performance.

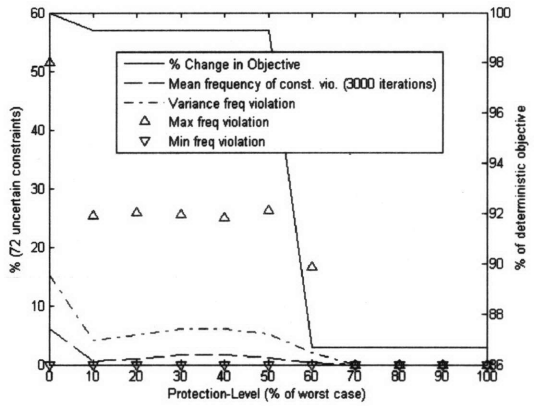


(a) Protection-Level 70

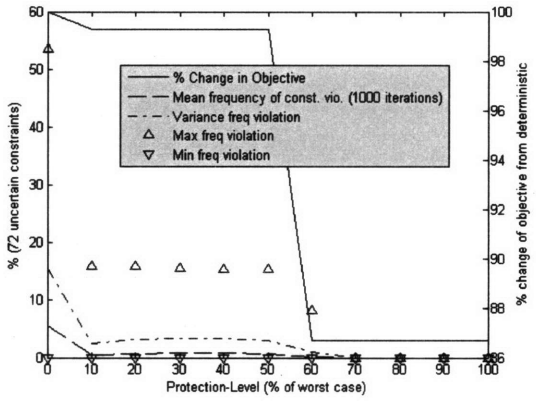


(b) Protection-Level 80

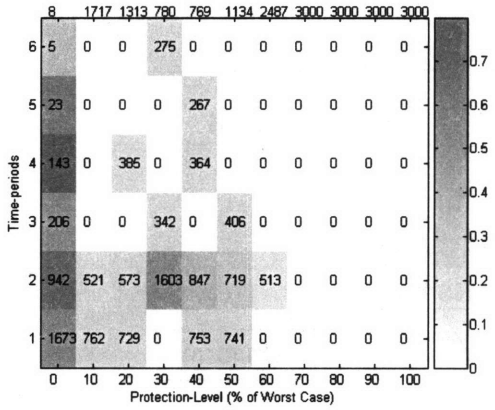
Figure 4-23: Chance-Constrained Normal vs Uniform Right-Hand-Sides



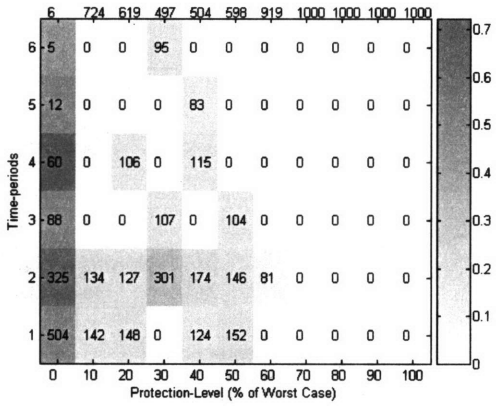
(a) Uniform Obj. Function and Freq. Viol



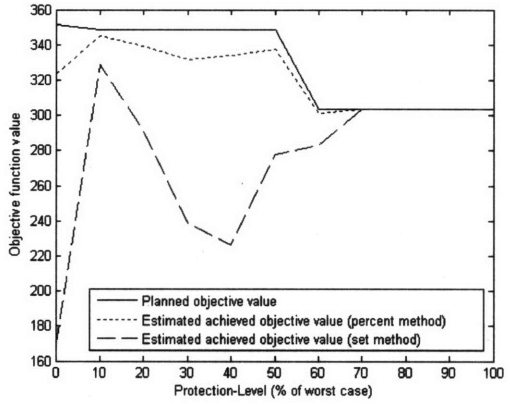
(b) Normal Obj. Function and Freq. Viol



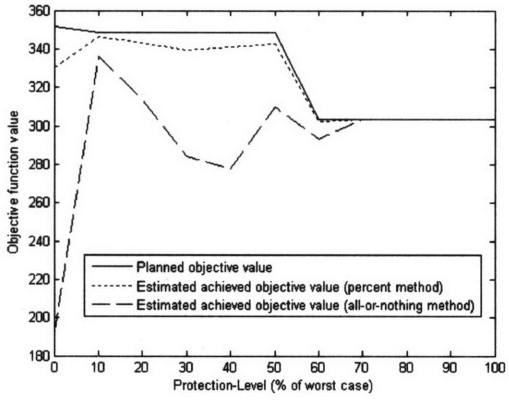
(c) Uniform Hazard Rate Plot



(d) Normal Hazard Rate Plot



(e) Uniform Estimated Achieved Obj. Func.



(f) Normal Estimated Achieved Obj. Func.

Figure 4-24: Scenario 1 Bertsimas/Sim Performance, Uniform vs. Normal

On average in the simulated environment, the normal case cannot perform worse than the uniform case because the truncated normal distribution can never have more variability than the uniform distribution. Thus, there will always be fewer violations for the normal case.

4.3.3 Conclusions on the Performance of the Chance-Constrained EBO Model versus the Bertsimas/Sim EBO Model

In this Section 4.3, we compared the performance of plans generated by the Chance-Constrained EBO Model to those generated by the Bertsimas/Sim EBO Model. We tested plans from both models on a range of scenarios and used different kinds of uncertainty. We hypothesized that plans from both models perform similarly under all scenario structures when uniform uncertainty exists in the right-hand-side only, and that the Bertsimas/Sim EBO Model creates plans that are more protected than those from the Chance-Constrained EBO Model when normal uncertainty exists.

We found little significant difference in the performance of plans from the Chance-Constrained EBO Model and plans from the Bertsimas/Sim EBO Model when only right-hand-side uncertainty exists. We showed that when the right-hand-side uncertainty is uniformly distributed, the models use the same values in the right-hand-side for corresponding protection-levels. *All differences in model performance with right-hand-side uniform uncertainty can be attributed to the different task-time assignments in the solutions that are one of the multiple optimal solutions.*

When we use normally distributed right-hand-side uncertainty, the Bertsimas/Sim Model uses the same protection-level values as with uniform uncertainty and generates the exact same plans. Because the normal uncertainty varies less, the plans are more protected in the normal case than in the uniform case. The Chance-Constrained EBO Model adjusts for the normal uncertainty, but due to the effects of the integer constraints the right-hand-side values are treated as though they are rounded down to the next integer. The rounding causes the model with normal uncertainty to protect similarly to the uniform case for many of the protection-levels, as shown in Figure 4-23. *Thus, the Chance-Constrained EBO Models generates plans that encounter slightly fewer violations in the normal case than in the uniform case because the normal data varies less while the model protects almost the same.*

In general, we cannot say that either model generates plans that perform better when right-hand-side uncertainty exists. We did not test for uncertainty in both the right-hand and left-

hand sides of the formulation. We hypothesize that the Bertsimas/Sim EBO Model will generate plans that perform better than those from the Chance-Constrained EBO Model because the Bertsimas/Sim EBO Model can consider uncertainty in the constraint matrix, while the Chance-Constrained EBO Model cannot. This should be tested in future research. Choosing which model is preferable might be primarily determined by the user's preference of how to consider protecting the plan. We discuss user interaction with the models in Chapter 5.

4.4 Summary

In this chapter, we tested the performance of plans generated by the robust EBO Models. We analyzed their specific performance, then tested them against plans generated by the greedy algorithm and tested them against each other. In general, we found that both the Chance-Constrained EBO Model and the Bertsimas/Sim EBO Model can be used to find plans with fewer constraint violations, longer mean times until failure, and better estimated achieved values than plans from the Deterministic EBO Model and greedy algorithm. Neither robust model stood out as the better method to create robust plans. Plans from both models perform the same under uniform right-hand-side uncertainty, and very similarly under normal right-hand-side uncertainty. Future tests should be done for left-hand-side uncertainty. In most cases as we increased the protection of the plans we can find a point where we achieve the most estimated value, though in some cases we can protect too much and achieve less value not because of constraint violations, but because we planned to do too little. Finding this best protection-level is a subjective process involving balancing the risk of encountering a failure with the benefits of achieving more value. We discuss this process in Chapter 5.

Chapter 5

Human Interaction with Robust EBO Models

In order to implement the robust EBO models presented in Chapter 3 and analyzed in Chapter 4, a human planner must interact with the computer-based models. As we demonstrated in Chapter 4 it is often difficult to determine the proper amount of protection needed to make the best plan, estimate how well a plan will perform, and understand how the plan's structure can cause performance that is not obvious to a human operator. Furthermore, the robust EBO models rely on input data generated by a human planner. If this data is poorly determined, the model output will also be poor. To facilitate human interaction with the robust EBO models, we look at principles of human information perception, decision-making, and cognition. The cooperation of humans and computer-based models to make complex decisions is sometimes called Human Machine Collaborative Decision Making (HMCDM). In this chapter, we review principles from the literature of HMCDM. We apply these principles to possible ways to facilitate human interaction with the robust EBO models.

5.1 HMCDM Overview and Motivation

It is important that commanders and other decision makers understand and approve of the plan generated by computerized planners and that the computerized planner makes a plan based on an accurate model of reality. Unfortunately, human knowledge and preferences cannot be fully designed

into the computers. As a result computerized planners can generate plans outside the context of the situation, and commanders might have little trust in or acceptance of the solution. HMCDM seeks ways to allow a human operator to augment the plan generated by the computer. This allows the human to guide the machine towards context-appropriate solutions and have visibility into the plan generation process, thereby reaching a trusted solution[41]. One goal of HMCDM is to capitalize on the strengths of both humans and computers to mitigate some of the respective weaknesses and make better decisions possible.

Researchers, primarily from psychology and human factors engineering fields, have studied how humans process information and make decisions. In the following sections, we look at pertinent findings in the literature about human performance that is applicable to human interaction with the robust EBO models. We review principles of how humans interact with information displays and principles of human decision-making (both what heuristics people use and what biases people have in making decisions).

5.2 Information Presentation Principles

For a planner to use the EBO models presented in Chapter 3, he must be able to understand the input data, model outputs, and structure of the problem. A theater-level plan can include thousands of tasks, hundreds of effects, and dozens of squadrons. Tasks can belong to multiple task-sets, which can cause multiple effects. The output of the model is the assignment of tasks to squadrons for specified time-periods; but this data is meaningless to a planner unless it is related to the effects that the tasks will cause and the probability of causing the effects.

User interaction with the EBO models will require that the user understand large amounts of information to get a clear picture of what the plan is going to accomplish and why. User interaction will require information displays that aid in inputting data, setting up effect to task relationships, determining how protected to make the solution, and understanding the solution generated by the model.

Some of the most established research into human interaction with information displays comes from human factors research of aviation displays. Pilots must quickly process massive amounts of information while performing complex motor tasks under high stress. As such, researchers have worked to find information displays that best communicate the information pilots need to aviate,

navigate, and communicate safely. We review some of the applicable findings from this research here.

5.2.1 Display Principles

Much research has been done into humans' ability to perform search and monitoring tasks, such as a pilot identifying another aircraft to avoid collision or a nuclear power plant operator monitoring the status of a reactor. Tsang and Vidulich present seven principles of display design. Although they do so in an aviation context, these principles are applicable to any human interaction with complex information displays[53].

- *Information Needed*: Due to the complexity of the EBO planning problem and large size, users can experience information overload. Information displayed should be limited to only information needed for the planner to make planning decisions. If certain kinds of information are more frequently needed, they should be displayed in locations that are more accessible.

- *Legibility*: It is self-evident that displays need to be legible in order to be useful. Displays need to be of adequate size, contrast, and brightness.

- *Discriminability*: Displayed elements that represent different kinds of information should not look similar to another element that could occur in the same display context. Humans take in visual information in two phases: the preattentive phase, which is carried out automatically and organizes the visual world into objects and groups of objects; and the selective phase, which is used to attend to certain objects of the preattentive array for further elaboration[59]. Information should not be grouped so that the information a user perceives from the preattentive phase contradicts the information from the selective phase. This can lead to misinterpretation of the information. For instance, Figure 5-1 shows an experiment performed by Navon[46], in which subjects were asked to state the name of the large letter. The large letter is perceived by the preattentive phase and the small letters that make up the large one are perceived by the selective phase. When subjects were asked to say the large letter in (a), there is a conflict because both phases lead to a different response. In (b) there is no conflict.

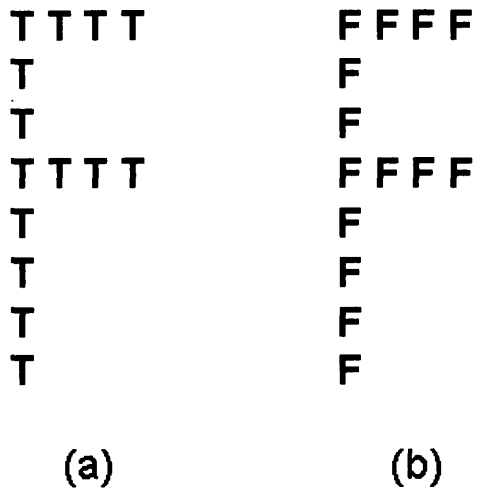
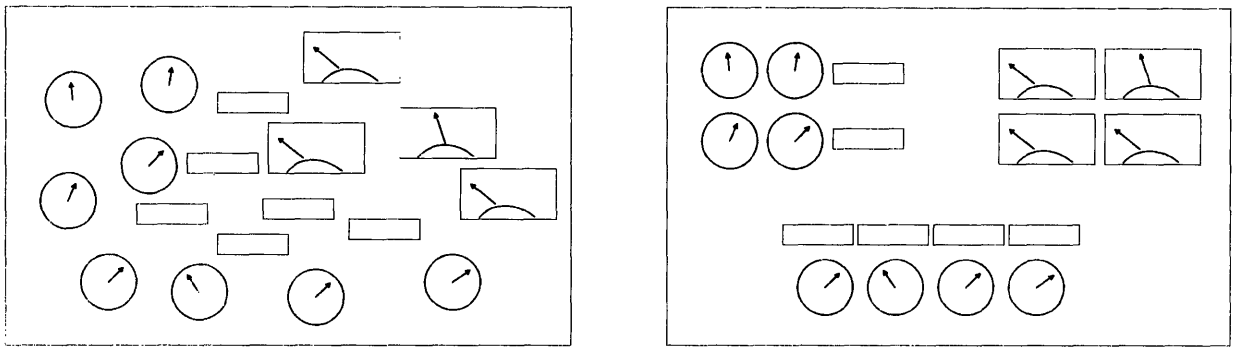


Figure 5-1: Preattentive and Selective Perception

- *Compatibility*: Scanning separate sources of information demands mental effort to direct the visual scan to the appropriate location at the right time. The amount of effort increases when destinations are farther apart and when pairs of destinations contain information that must be related, compared, or integrated in performing the task. Thus, it is important to keep frequently used information in a prominent location, so that minimal long-distance scanning is required to access it repeatedly. It is also important that when information sources need to be integrated or compared they should be positioned close together on the display. There should be "compatibility" or agreement between closeness or relatedness in the mind and closeness on the display.

  Grouping displays to allow a human to better perceive information is called the Gestalt principle[59]. There must be proper compatibility among displays. For instance, if the operator often compares the upper-left gauges with the bottom row gauges, he might have trouble because of their different layout and distance apart. In this case if compatibility is not achieved, and he might perform worse than when the gauges are scattered as in (a).

- *Pictorial Realism*: A display should be a pictorial representation of the information it represents. For instance, the attitude indicator of an aircraft shows the aircraft in relation to the horizon, a similar picture as the pilot sees as he looks forward out of the cockpit.



(a) (b)

Figure 5-2: Gestalt Principles of Display Organization

- *The Moving Part*: Moving elements on a display should correspond with elements that moves in the viewer's mental model and should move in the same direction.

- *Predictive Aiding*: Active prediction of the future behavior of dynamic systems is often a cognitively demanding task and becomes more demanding the farther into the future that such prediction is needed. Users usually benefit from any source of predictive information as long as that prediction is reasonably accurate and understood.

5.2.2 Color-Coding

There are many benefits to color-coding; however, they can be offset by some of its limitations. We list some of the benefits and limitations here [59].

Benefits

- Color stands out from a monochromatic background, as such color-coding is very effecting in aiding a person to find objects, such as highlighting an important item on a menu[35]. Researchers have found that search time for a uniquely color-coded object in a cluttered search field is independent of the size of the field[27].

- Certain colors have pre-determined cultural significance which can be useful in making significance of displays intuitive. For instance, in America red typically means danger, emergency, or the command to stop. Green typically means it is safe to proceed.

- Color-coding can associate specially separated display elements, which is most effective when colored items also need to be integrated as part of a task (i.e. color-coding of similar temperature ranges on a weather map[58]).

- Color-coding enhances the value of information when coded redundantly with shape, size, or location. For instance, traffic lights use the redundant coding of location and color (the top light means stop, the red light means stop, together top and red makes the signal stronger).

Limitations

- Color-coding is subject to the capacity of human's to take in and process information. To avoid color-to-meaning misidentification, no more than five or six colors should be used in a display[20].

- Color does not naturally define an ordered continuum. If people are asked to rank order a set of colors from least to most, no predominant order will emerge. Quantitative variables should be denoted by saturation or brightness, rather than color changes (i.e., in an ocean map darker blue means deeper and lighter blue means shallower[54]).

- Color stereotypes can cause confusion when the cultural significance of a color is different from the intended meaning in the display. For instance green can be associated with both cool temperature and safe operating range. In a system where low temperature is bad, using green to denote the low temperature can be confused as the safe operating range.

- Irrelevant color-coding can be confusing. It is important that colors be connected to distinctions in the display that are meant to be interpreted by the viewer.

- Roughly 3% of humans have some form of color blindness. The most common is red-green color blindness. Displays need to be made such that a user can discriminate between information sources without relying on color alone[59].

5.3 Human Decision-Making

In Section 5.2 we discussed general principles that help humans better find, perceive, and interpret information. We are not only interested in displaying information, but also interested in helping a

human make decisions. While Section 5.2 is about *how* to display information, this section is about *what* information we should display so that a human can make the best decision possible.

Decision making is conventionally characterized as the act of choosing between alternatives under conditions of uncertainty[47]. The simplest decisions are go-no-go, while more difficult decisions involve multiple options and ordering. Apart from the complexity of the decision, there are three other major factors that affect the decision's difficulty: uncertainty, familiarity and expertise, and time[59]. Decisions become increasingly difficult with increased uncertainty and risk. Under uncertainty, the decision maker not only has to sort through all the options for the decision, but also predict how each will perform under uncertainty. Familiarity and expertise with the decision, generally tends to make the decision easier and allows it to be made more quickly. Familiarity and expertise; however, do not always guarantee better accuracy. Time affects not only time-pressure to make a decision, but also whether or not the decision is an evolving decision or a one-time decision. With an evolving decision, a decision maker can observe the effects of a decision and change decisions as he sees results. This is not the case with one-time decisions.

Decision making is generally studied and evaluated through three different frameworks: rational or normative, cognitive or information processing, and naturalistic[59]. In the rational or normative framework, it is assumed that people make decisions based on a gold standard: maximizing expected profit or minimize the expected loss. Most efforts of research in this framework focus on departures from these optimal prescriptions. The cognitive or information processing framework focuses on the biases and processes used in decision making that can be related to limitations in human attention, working memory, or strategy choice as well as familiar decision routines called heuristics. The naturalistic framework places the emphasis on how people make decisions in real environments (i.e. not in the laboratory) with expertise and where decisions are very complex.

Our goal is to take elements from each of the frameworks that describe how humans make decisions, how we can avoid typical mistakes humans make in their decision processes, and what information humans need to make good decisions. We then apply them, along with the display principles from Section 5.2, to facilitate user interaction with the robust EBO models presented in Chapter 3.

5.3.1 Information Processing

Humans have a finite capacity to filter and comprehend cues. A cue is a signal to provide information to the decision maker about the decision to be made. Although cues provide information, more cues do not necessarily facilitate better decisions by the decision maker. In practice when the number of cues from different sources grows to more than two, the decision maker typically does not perform better as the number of cues increases[48]. Under time stress, a decision maker's performance tends to decrease as more cues are introduced[60]. When too many cues are present, the decision maker filters the cues. More cues lead to more effort required for filtering and less effort deciding. Although there are decreasing marginal returns on the amount of information used to make a decision, people have a tendency to seek far more information than they can handle.

Humans tend to be poor decision makers when cues are missing. Good decisions can be made by knowing what is missing and seeking the missing information. For instance, many troubleshooting problems in computer programs or fixing mechanisms can be identified by realizing what is *not* happening. This failure to capitalize on information given by the lack of cues can be attributed to the fact that humans tend to pay attention to the most salient cues.

Humans tend to struggle when there are outlying cues, such as information that is very unusual, or data that are above the 95^{th} or below the 5^{th} percentile. Humans estimate the mean value of many observations well if the data has few outliers. When there is data in the extremes (above the 95th or below the 5th percentiles), humans tend to weight these values too heavily or overestimate the frequency of these extreme cases occurring[57].

Humans have difficulty estimating variability. In general, humans will estimate less variability as the mean of the data increases. For instance, in Figure 5-3 most people will estimate greater variability in (a) than (b) even though the variability is the same[42]. Variance also tends to be disproportionately influenced by the extreme values in a distribution, causing higher estimates of variance. Humans are also poor estimators of correlation. They tend to underestimate high correlations and over-estimate low correlations[44]. They are also more likely to make linear extrapolations than higher order extrapolations.

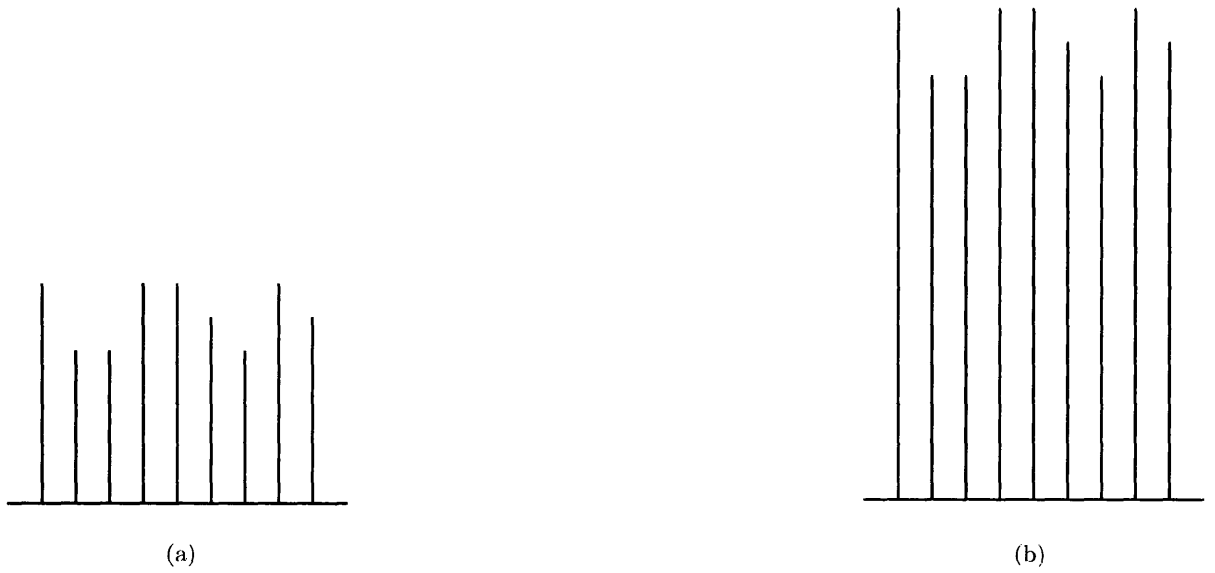


Figure 5-3: Estimation of Variance

5.3.2 Heuristics

Given the limitations humans have in processing information, humans tend to rely on heuristics to help simplify the decision process. We present here a list of common heuristics used in decision-making.

- *"As if" Heuristic*: Humans tend to treat all cues as if they were of equal value[38]. The "as if" heuristic works adequately most of the time, but if a salient, low-value cue is present, it can lead to incorrect diagnosis of the situation. For instance, evaluators of job applications tend to give more weight to the tone or enthusiasm of a letter of recommendation than the credibility or reliability of the evidence[36]. Unfortunately, even those who are well-trained in statistical theory do not give proportionally more weight to more reliable cues[39]. This inability to properly weight cues, has shown in many studies that computers are better at predicting outcomes or diagnosing situations when the prediction or diagnosis involves multiple cues of different information value[59]. Some have recommended that the role of humans in prediction should be limited to identifying relevant predictor variables, determining how they should be measured and coded, and identifying the direction of the relationship to the criterion[29].

- *Representativeness*: A decision maker diagnoses the situation by evaluating the extent to

which a set of cues, symptoms, or perceptual evidence corresponds with the set that is representative of the hypothesis created based on experience[56]. Representativeness tends to work well, unless there is an ambiguous or uncommon cue, in which case the decision maker often relates it to a case he has encountered without considering the likelihood of that being the case.

- *Anchoring Heuristic*: The initially chosen hypothesis gets more weight (mental anchor attached) than any other hypothesis or options[56]. Recency tends to trump primacy in complex problems.

- *Elimination by aspect*: When trying to choose among several options, the decision maker inspects the most important aspect of each option first. Any option that does not score well in this aspect is eliminated. Then the decision maker reviews the second most important aspect and so on until one option is left[55]. This heuristic can eliminate good options that may have very strong second and third-ranked aspects that would outrank others, but since it is eliminated at the beginning, these options are not considered.

5.3.3 Biases

Human decision making also suffers from bad assumptions, or biases, often made by decision makers. We discuss some of these biases here.

- *Salience Bias*: the salience of a cue, its attention-attracting properties or ease of processing, affect the weight that humans assign to the information it provides when making decisions. "Loud sounds, bright lights, underlined or highlighted information, abrupt onsets of intensity or motion, and spatial positions in front or top of a visual display are all examples of salient stimulus[59]". Top locations are more salient to users, who assume they are more important and thus process top information first.

  Salience does not predict the quality of information. For instance, a loud alarm is very salient and good for telling you something is wrong, but the alarm does not necessarily tell you what is wrong and what you should do. Information that can be very useful is often ignored or underrated if it is difficult to interpret. For instance, one study found that decision maker

paid more attention to pictorial representations of risk (easy to understand) than quantitative data of the risk (more difficult to interpret)[52].

- *Overconfidence Bias*: Decision makers are not likely to seek more information as necessary because they hold too much confidence in their state of knowledge of the situation[18]. One study shows that when subjects are asked to predict the outcome of a future event (such as a sporting event or election) and also give their confidence as to how likely they will be correct, the confidence exceeds accuracy by up to 30 percentage points[34].

- *Confirmation Bias*: The decision maker's tendency is to seek information that confirms his initial hypothesis rather than disproves it[31]. This bias tends to produce cognitive tunnel vision, in which decision makers fail to pay attention to cues that contradict their initial hypothesis.

- *Utility Theory (Distortion of Gains and Losses)*: People tend to hold a non-linear valuation of gains and losses. In general, researchers have found that people value gaining something (such as money or points) as having decreasing marginal benefit. They also value losing something as having decreasing marginal loss. This phenomenon can be explained if we view people as trying to maximize their utility instead of the actual thing being gained or lost[30]. Utility is the subjective value of "goodness" or "badness" people associate with a gain or loss. As the amount of utility gained or lost increases, the marginal utility gained or lost decreases. People tend to weight gains less than losses. Utility as a function of actual value gained or lost is as shown in Figure 5-4.

- *Perception of Probability*: Not only do humans not perceive value linearly, they also do not perceive probability linearly. Humans tend to overestimate low probabilities and under estimate high probabilities[40]. The subjective estimates of probability as compared to the actual probability are shown in Figure 5-5, where the dark curved line represents a human's subjective valuation of probability and the dashed line represents the actual probability.

- *Framing Effect*: Because humans tend to weight losses more heavily than gains, it is possible to "frame" an event so that humans value the same event differently, based on whether they consider it a gain or a reduction in loss. For instance a tax cut may be perceived a reduction in loss if the neutral point is "paying no taxes," or a positive gain if the neutral point is the

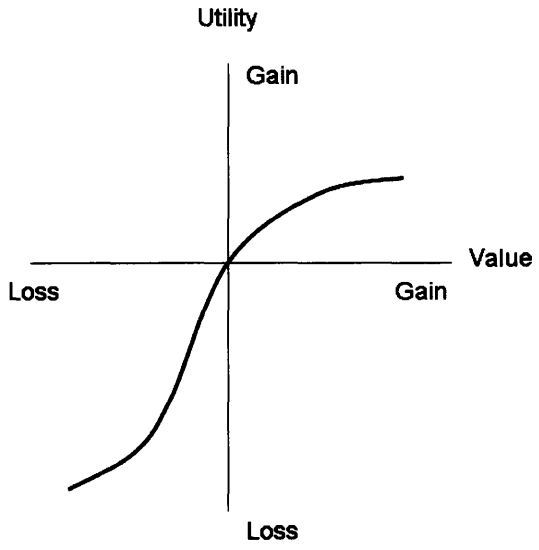


Figure 5-4: Hypothetical Relationship between Value and Utility

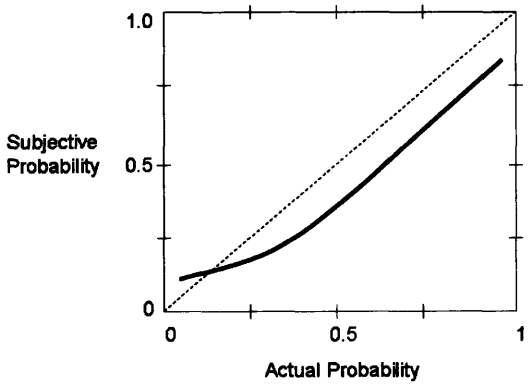


Figure 5-5: Hypothetical Probability Weighting Function

amount paid on last year's taxes. As shown in Figure 5-4, the same change in value in the gain portion of the vertical axis is given less utility than on the loss portion.

5.4 Examples of Possible User Applications with Robust EBO Models

Human Machine Collaborative Decision Making (HMCDM) looks at methods to maximize the synergies created when humans interact with computers. The goal of HMCDM applied to mission planning is to identify what humans do well and computers do well and to create a methodology for dividing the planning process between the human operator and the computer so that better decisions can be made.

In general, humans are good at identifying patterns and dealing with the subjective aspects of a plan, but they also demonstrate the biases discussed in section 5.3.3. Computers are good at integrating large amounts of information from multiple sources, doing marginal analysis, and performing calculations very quickly. Computers tend to be poor at subjective analysis. Computers also have to use whatever input information is given to them, which can be incorrect or biased by humans.

To capitalize on the strengths of humans and computers, in making mission plans, we outline a collaboration framework shown in Figure 5-6. This framework maximizes the human operator's ability to determine objectives and make subjective valuations. The computer does the detailed calculations and returns plans to the human. The computer can also perform simulations based on user-defined uncertainty to test the robustness of plans. The user can review plans and their simulated performance and guide the machine towards a plan that makes sense to him. The computer then creates a final plan, which the human can modify as necessary.

5.4.1 Stage 1: Problem Definition Input

One of the most important aspects of planning is to ensure that the information used to form the plan is accurate. It is very difficult to program computers to observe reality and generate planning objectives and options. Humans are better suited to determine objectives and possible options. Furthermore, objectives can be dynamic, changing over time. Humans can understand the reasons

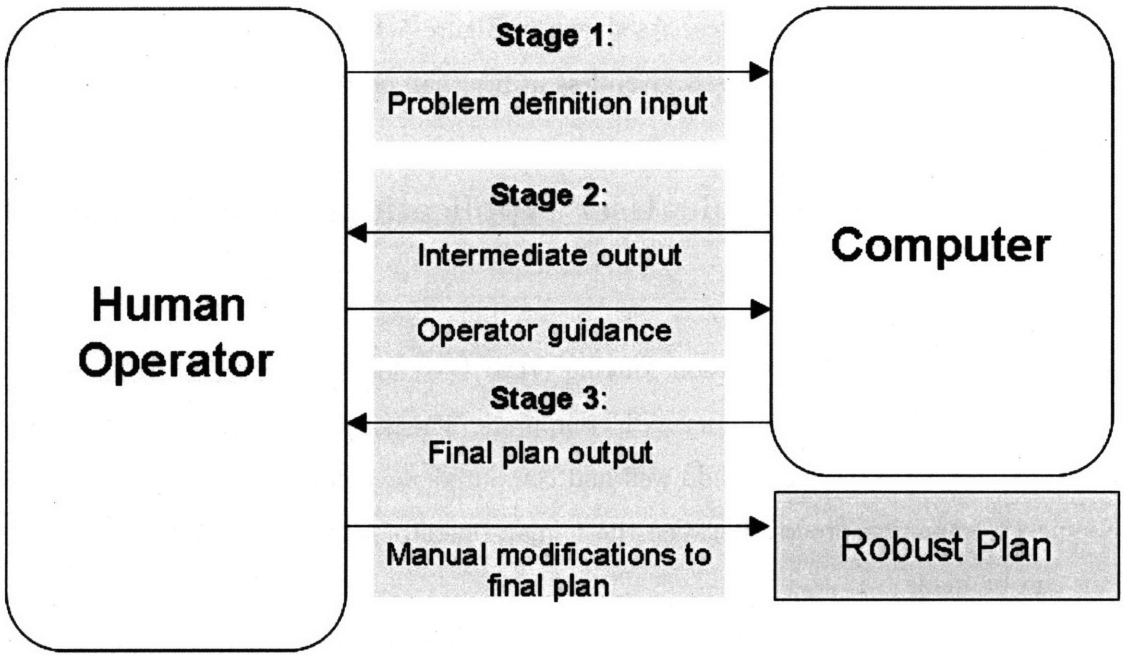


Figure 5-6: Human-Machine Collaboration Framework

for changing objectives. The computer simply determines the best option for accomplishing the objectives. If a computer is given poor information, the plan it makes with the information will also be poor.

We consider the input data required for the EBO model originally presented in Section 3.1.1. We classify the input data into three categories: effects data, squadron capabilities, and task information. Some of this data can be generated automatically by the computer from stored historic information. This data might include the squadron capabilities (such as how many UAVs the squadron has or the average number of tasks each type of UAV can do per time-period) and much of the task information (such as location or number of UAVs required to do a task). The subjective information requires the planner to input his knowledge and valuations (such as how much value an effect achieves, what tasks he thinks will cause effects, and how much uncertainty he thinks there might be in the data.) Stage 1 should focus on aiding the human to input the most accurate data possible for this information.

As discussed in Section 5.3.3, humans do not make valuations based on true value, but based on perceived utility. Losses are weighted more heavily than gains, and decreasing marginal returns

come into play. If a human is asked to assign a value subjectively to a group of effects, he will likely assign too much value to effects that avoid losses (such as defensive military actions, security actions, or eliminating enemy threats) and assign too little value to effects that will achieve value but not cause much loss if they are not accomplished. He will also value very important tasks too little because of the perceived decreasing marginal returns. The effects of salience bias and the "as if" heuristic may also come into play. If a planner is forced to value effects based on their relation to other tasks, he may avoid some of these biases.

We propose to frame the effects-valuing process as a percentage of the total value that the human planner wants to achieve over the course of the plan. We assume the planner has a predetermined set of all effects that the human wants to cause by executing the plan (i.e., there can be more effects than capacity to perform). If executing a plan causes all of these effects, the plan achieves 100% of the human planner's objectives. The human planner must weight each effect according to what percent of the total value the effect can cause. This can be done graphically, such as the pictorial representation of effect value in Figure 5-7. In this way, the planner has to compare effects against one another and against how much of the total value the effect achieves. The visual representation of the value of an effect compared to other effects and the plan's possible total value will hopefully mitigate the effects of human bias in making valuations by representing the causal relationship between weights.

The human not only has to estimate the value for the effects, but also estimate what tasks can be performed to cause those effects, group the tasks into task-sets, and estimate the probability that a particular task-set will achieve its linked effect. We can apply the display principles discussed in Section 5.2.1 to help human planners set up the relationships between the effects, task-sets, and tasks. A graphical user interface for this process should incorporate pictorial realism representing the EBO framework, so that a user can see the relationships between effects and tasks at a glance. As problem size grows, legibility, discriminability, and compatibility will suffer. To maintain these, color-coding and grouping must be used effectively. A "zoomed in" view that can isolate only those tasks connected to a particular effect as well as a "zoomed out" view to put these tasks in context would be helpful. An example of this is shown in Figure 5-8.

Unfortunately, humans struggle at estimating variability. If the uncertainty parameters are not estimated well, the performance of robust plans will perform differently than estimated by

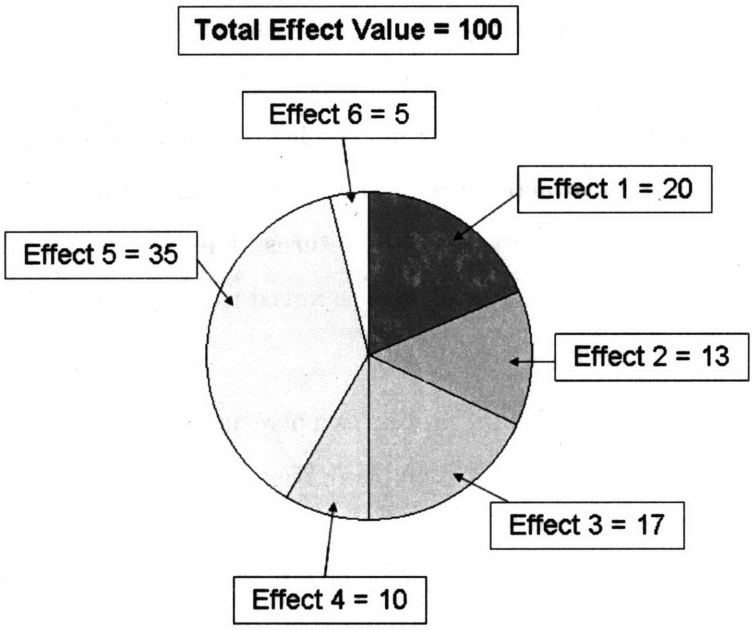


Figure 5-7: Valuation of Effects

the computer. Preferably, the computer would estimate the ranges and distributions of uncertain coefficients based on historical data, perhaps using previous plan's performance deviations from expected values. However, if we cannot use the computer to estimate the uncertainty, the user should estimated the real distance from the expected value rather than a percentage as we did in chapter 4. This will curtail some of the bias humans usually demonstrate when they assume higher variance for data with a lower mean as shown in Figure 5-3.

5.4.2 Stage 2: Intermediate Output and Operator Guidance

In Stage 2, the user is presented with intermediate output from the computer and returns guidance to the computer for the final plan. In Stage 2 the user will decide the amount of protection used in the final plan and possibly adjust input data based on the intermediate plan's performance.

To help the user find the best amount of protection for the plan, we need to help the user avoid the biases discussed in Section 5.3.3. Particularly, we want the user to avoid using representativeness and the anchoring heuristic. Past performance of particular protection-levels does not predict future performance; and what seems like an obvious first choice, might not be the best option. Users should also avoid using the "elimination by aspects" heuristic. We are looking for the best overall plan.

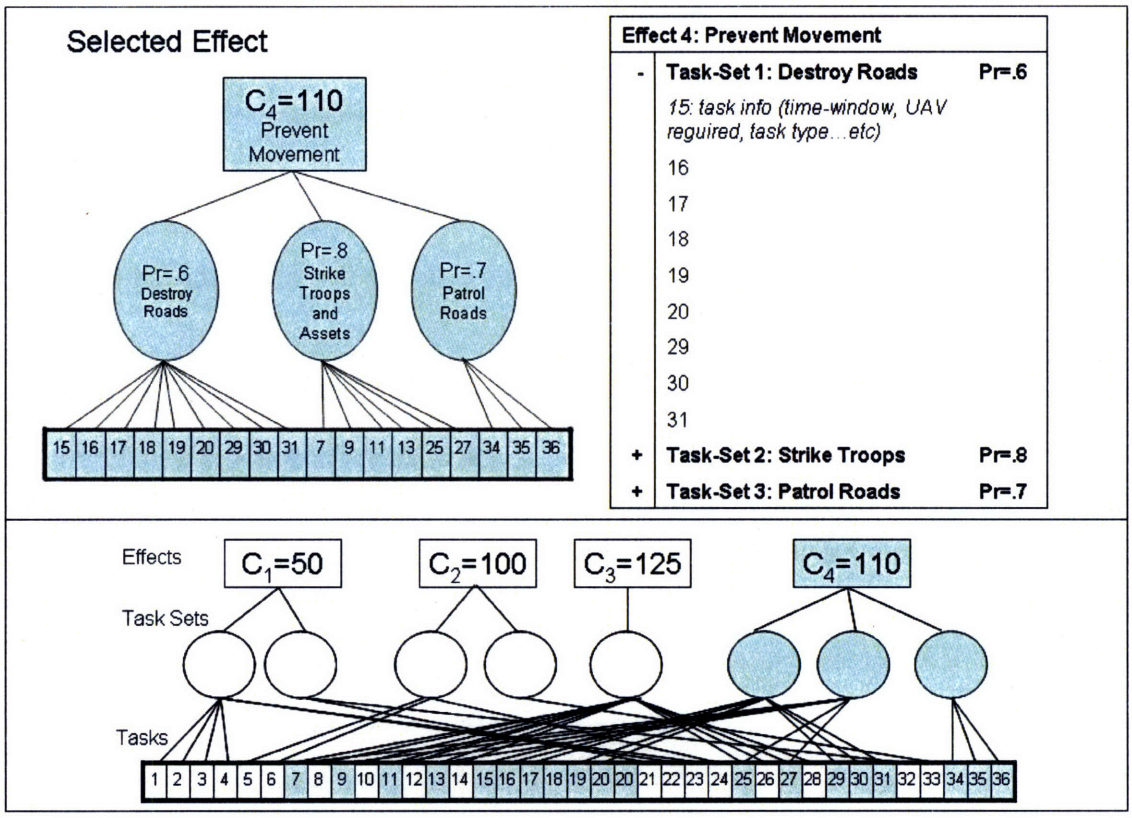


Figure 5-8: Example of User Interface for Building Effect-Task-set Relationships

Plans that exhibit a favorable trait, such as a high objective function value usually have poor performance in other areas, such as high failure rates.

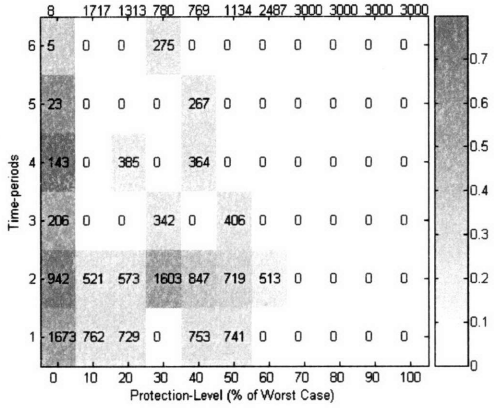
Avoiding these biases requires a thorough examination of the expected performance of plans at each protection-level; however, we do not want to display all of the data to the user. Humans tend to seek more information than they can mentally process, and sometimes seek more information instead of better information. Therefore, to facilitate human interaction with the robust EBO planner effectively, we want to display the best information required for diagnostics.

The most useful information will explain the probable performance of the plan, both its expected value and information about when and why it might fail. The hazard rate plots and estimation of value achieved plots introduced in Chapter 4 give the best summaries of these performance metrics. When used together, they match the planned value to conservative and liberal estimates of what value will be achieved when the plan is executed, and give information about the failure rate of the plan.

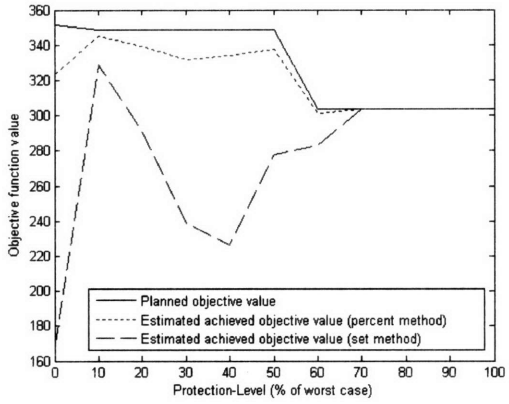
The hazard rate plots also give the information pictorially, enabling the user to estimate the performance of the plan across all protection-levels at a glance. The hazard rate plots can also be enhanced using color, maximizing of cultural assumption of color meanings (green for fewer failures and red for many failures.) The hazard rate plots show how many plans fail, what time-period the failures occur, and how many plans are successfully completed. The hazard rate plots integrate different kinds of information into one chart, a task that humans find difficult.

This integrated information is more useful to a user than looking at plots of the time of failure and violation rates separately. For instance in Figure 5-9, if we look at (a) and (b) compared to (c) and (d), we get a sense of when failures occur, how often, and what value we might achieve amidst the failures. To get this same information from the raw data presented in (c) and (d) requires significant calculations that are difficult for humans.

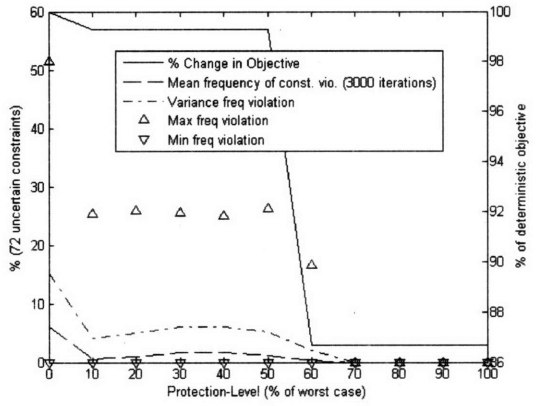
Using information such as that in Figure 5-9, the user can select the most preferable protection-level. Choosing the best protection-level requires deciding how much risk the planner wants to accept in order to achieve an objective value. In this case, protection-level 10 has the highest estimate of achieved objective value; however, nearly half of its simulated realizations encounter a violation over the course of the plan. Thus, a user would have to weigh the benefits of the higher objective function value of protection-level 10 against a lower objective function value with no risk



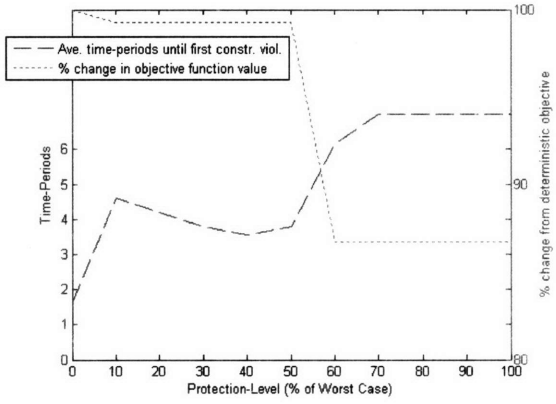
(a) Hazard Rate Plot



(b) Expected Achieved Objective Value



(c) Planned Objective Value and Freq. Constr. Viol.



(d) Mean Time Until Failure

Figure 5-9: Possible Information Displays For User Comparison

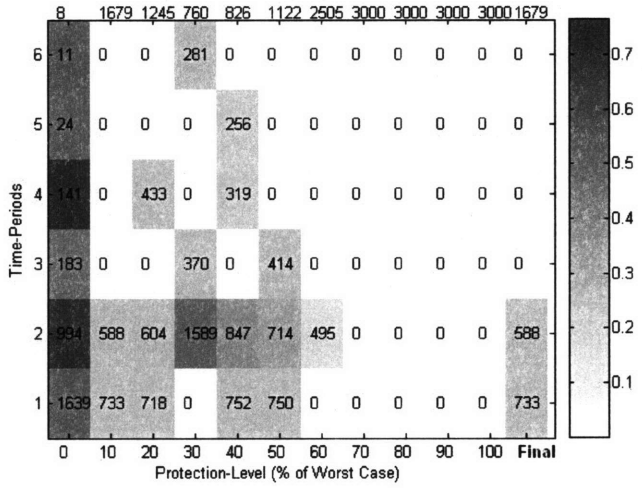
such as protection-levels 70-100.

The user can use the planned objective function value from the selected plan and input it as the satisfactory objective function value for one of the extended robust models introduced in Sections 3.2.3 and 3.2.6. Recall from Chapter 4 that the objective function value across the protection-levels decreases in a stepwise manner as the protection-level increases. Because the performance of a plan can vary even though the planned objective value is the same (in the same step), it is easier for the user to select a step and allow the computer to find the most robust solution at that step, than to have the user try to find the best performing protection-level for a satisfactory objective function value. Thus, in Stage 2, the computer can output information about the performance of plans at different protection-levels; the human then uses the information to determine the best range of protection-levels and inputs the planned objective function value at this level. The computer then finds the most protected plan possible while still achieving this value and returns this plan in Stage 3.

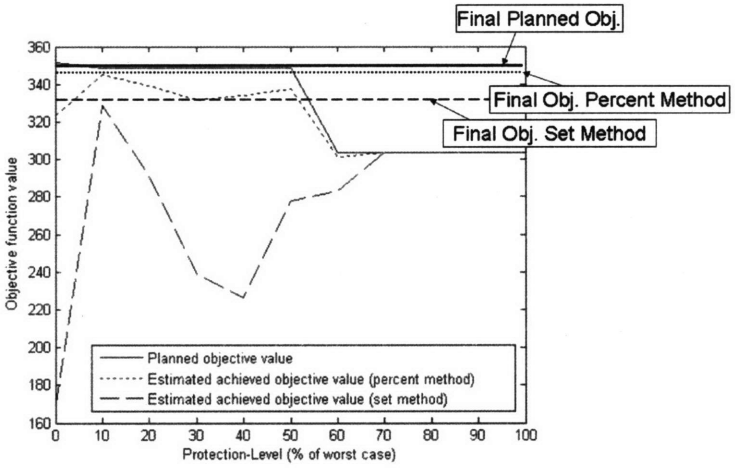
5.4.3 Stage 3: Final Plan Output and Manual Modifications to Final Plan

In Stage 3, the computer returns a final plan and simulation results for the final plan using one of the extended robust models from Sections 3.2.3 and 3.2.6 and the Monte Carlo Simulation. As we showed in Table 4.3, plans that achieve the same objective function value can have very different performance, not based on their protection-level, but based on which of the multiple optimal solutions the solver settles upon. Using the extended models eliminates some of the post processing required to ensure the best solution, as discussed in 3.2.3. Any remaining post-processing will involve detailed analysis of constraints and the trade-offs related in which time-periods tasks are assigned. Preferably, we would create a computer algorithm that does the constraint specific post-processing.

The better role of a human in Stage 3 is validation of the plan. Using pictorial representations of the plan similar to those used in Stage 1, the user should view the plan, ensure the tasks assigned are those that can cause the desired effects, and validate that the projected performance of the final plan is better than other options from Stage 2. The user should be able to compare the same statistics from the final plan against options in Stage 2. The final plan output should be presented next to the Stage 2 output in the same graphs for easy comparison, as shown in Figure 5-10



(a) Hazard Rate Plot



(b) Expected Achieved Value

Figure 5-10: Information Displays for User Comparison

5.5 Conclusions

The framework for the implementation of Stages 1, 2, and 3 is depicted in Figure 5-11. Through this framework, we hope to facilitate human interaction with the robust EBO models to create the best plans possible. Stage 1 uses displays to aid the human to input accurate valuations of effects and relationships between effects and tasks. Stage 2 presents the user with the necessary information to determine the best protection-levels so the user can input the desired objective level for Stage 3. In addition, in Stage 3, the user validates the solution. Through this process, we capitalize on the synergies that exist between the human and computer planners, ensuring that the computer is given accurate input data, and the protection-levels are based on user preference. This process will increase the likelihood that there will be buy-in of the computer-generated plan by the user, because the user will have a better understanding of the computer process and the computer will be working with data that the user has validated.

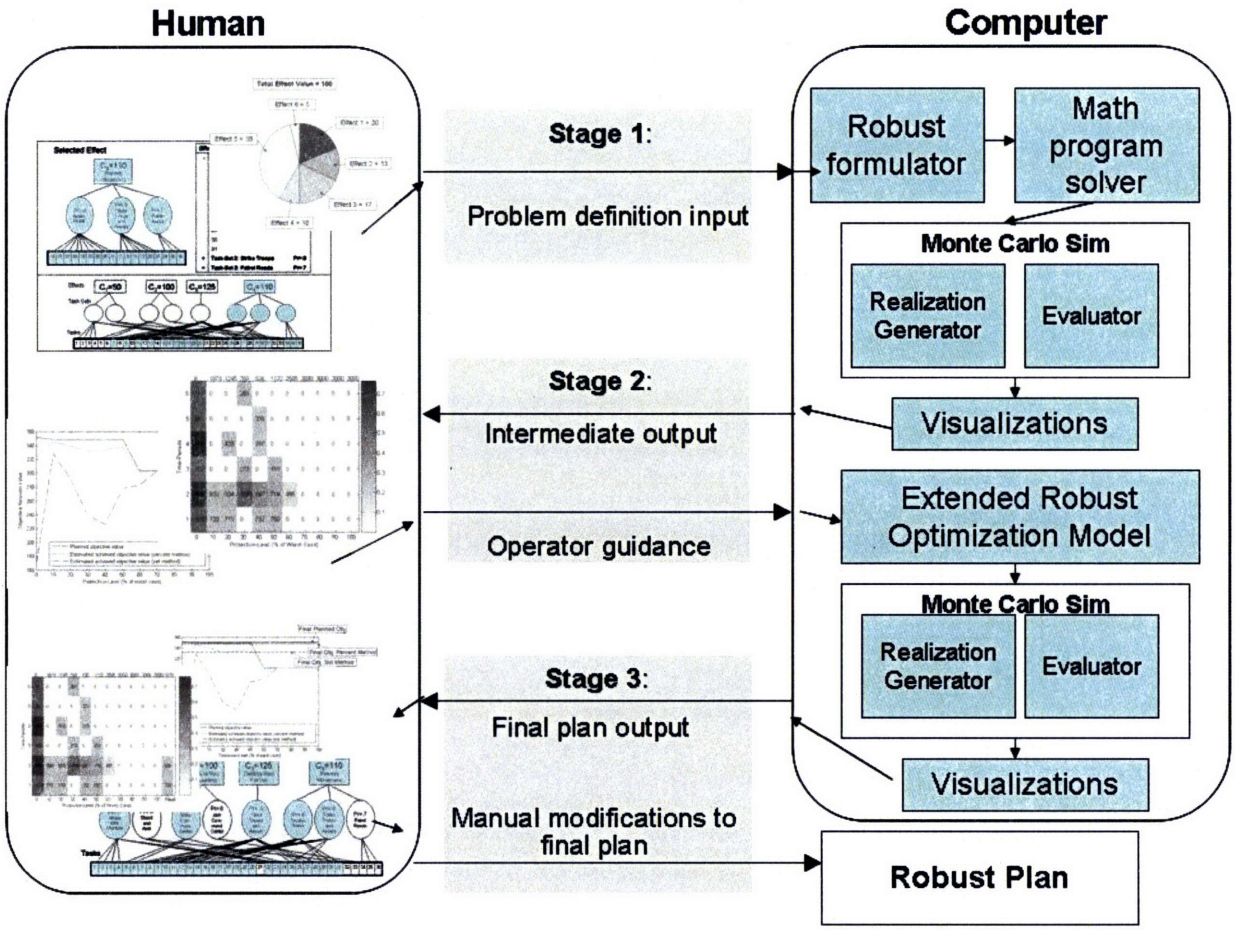


Figure 5-11: Framework for Human Interaction with Robust Models

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 6

Conclusions and Future Work

Robust planning for EBO presents promising results in creating plans that achieve end-objectives and are robust to uncertainty. We have demonstrated that the robust techniques of chance-constrained programming and the Bertsimas/Sim Formulation help create plans that are expected to last longer and have higher estimated achieve values (i.e. set method and percent method values). This research made contributions in three major areas: developing an EBO Framework that can be applied to math programming planning techniques; applying robust optimization techniques to the math programs based on the EBO Framework and testing the performance of plans generated by the robust EBO Models; and suggesting possible implementation of the robust EBO models in a human-in-the-loop planning environment, using principles of HMCDM.

6.1 Summary of Results and Contributions

We summarize the major results and contributions from this thesis in the following:

- We introduced the *EBO Framework* as an approach for establishing the relationships between individual tasks, the effects that they cause, and the end-objectives that a commander wants to achieve with the plan. The EBO Framework is a convenient approach for assigning groups of tasks that will cause effects with some probability and establishing the different options that might be used to cause the effects.

- We applied the EBO Framework to create the *Deterministic EBO Model*, a mixed integer linear program (MILP) formulation for the theater-level planning problem that uses the nominal

values of all uncertain data. We demonstrated the performance of the Deterministic EBO Model on realistic-sized theater-level planning problems. We demonstrated that we can solve realistic-sized problems (8 squadrons, 550 tasks, and 25 effects) to an optimality gap of 0.001% within two minutes using the Deterministic EBO Model.

- We applied two approaches, Chance-Constrained Programming and the Bertsimas/Sim Robust Optimization Formulation, to the Deterministic EBO Model to design robust plans. We call the resulting models the *Chance-Constrained EBO Model* and the *Bertsimas/Sim EBO Model.* We found that the frequency of constraint violation is an inconsistent measure of robustness and that mean time until failure and estimated value achieved give better information about the plan's possible performance. The Chance-Constrained EBO Model and the Bertsimas/Sim EBO Model perform similarly for uniformly distributed uncertain data, but have slight differences in performance for other types of uncertainty. Unfortunately, the models can generate dissimilar plans due to multiple solutions that achieve the same planned objective function value, but have different time assignments that result in different failure rates. *The robust plans created using both models outperform the plans generated by the greedy algorithm, which that approximates plans generated by humans and the plans generated by the Deterministic EBO Model.* The robust plans had longer expected time until the plan fails, and higher expected achieved value than the plans generated by other methods.

- We discussed how a human planner would interact with the Robust EBO Models. *We applied principles from Human Machine Collaborative Decision Making to make suggestions on how to implement the robust EBO models in the planning process* while capitalizing on the strengths the humans planners bring to the planning process and allowing the human to gain trust in the solution generated by the computer-based planner. We presented a framework for human interaction with the robust EBO models, allowing the human to specify the problem data, review initial planning options, and select a final desired objective level.

6.2 Future Work

The contributions in this thesis only take a small step in successfully creating robust plans for EBO. Much research can still be done, both in solving the theater-level planning problem for UAVs and

in robust planning approaches and applications of EBO in general. We summarize some specific recommendations for future work here:

- *Incorporate clustering tasks into the objective function.* We presented several approaches in Section 2.5 for creating task assignments that are more sophisticated than just minimizing the total distance between the tasks and their assigned squadrons. These clustering approaches or other approaches like them should be incorporated into the EBO planning process to ensure that the best squadron-task assignments are made.

- *Test the performance of the Extended Robust Models.* We would like to determine how well the extended models (i.e. the Extended Chance-Constrained Model and the Delta Formulation) perform in creating robust plans. We also would like to assess a human's ability to identify the best performing protection-level/objective function levels using the extended models versus the Chance-Constrained EBO Model and the Bertsimas/Sim EBO Model.

- *Create a post-processor.* Because both robust models were susceptible to drastic variations in performance based on which of the multiple solutions that achieve the same objective function value that the solver settles upon, it would be beneficial to develop a post-processor that can take a solution from the robust models and equally distribute the tightness among available constraints by changing task-time assignments. It would be interesting to attempt to incorporate the post-processor into the objective function of the robust models or possibly the extended robust models.

- *Test left-hand-side uncertainty.* In our testing, we only considered uncertainty in the right-hand-side of the models. We did this so we could compare the performance of the Chance-Constrained EBO Model against the Bertsimas/Sim EBO Model, because chance-constrained programming is limited to right-hand-side uncertainty. It would be beneficial to examine how both models perform under uncertainty in other areas, possibly making Bertstimas/Sim the better choice of the robust models because it is designed to handle uncertainty in both the right and left-hand-sides.

- *Test real planning scenario data with the robust models.* All scenarios used in this research were made using contrived values. It would be interesting to determine how the plans generated by the robust EBO models perform on real mission scenarios. The models should also be

reviewed by military planning experts who can verify that the EBO Framework and resulting formulations accurately represent the theater-level planning problem.

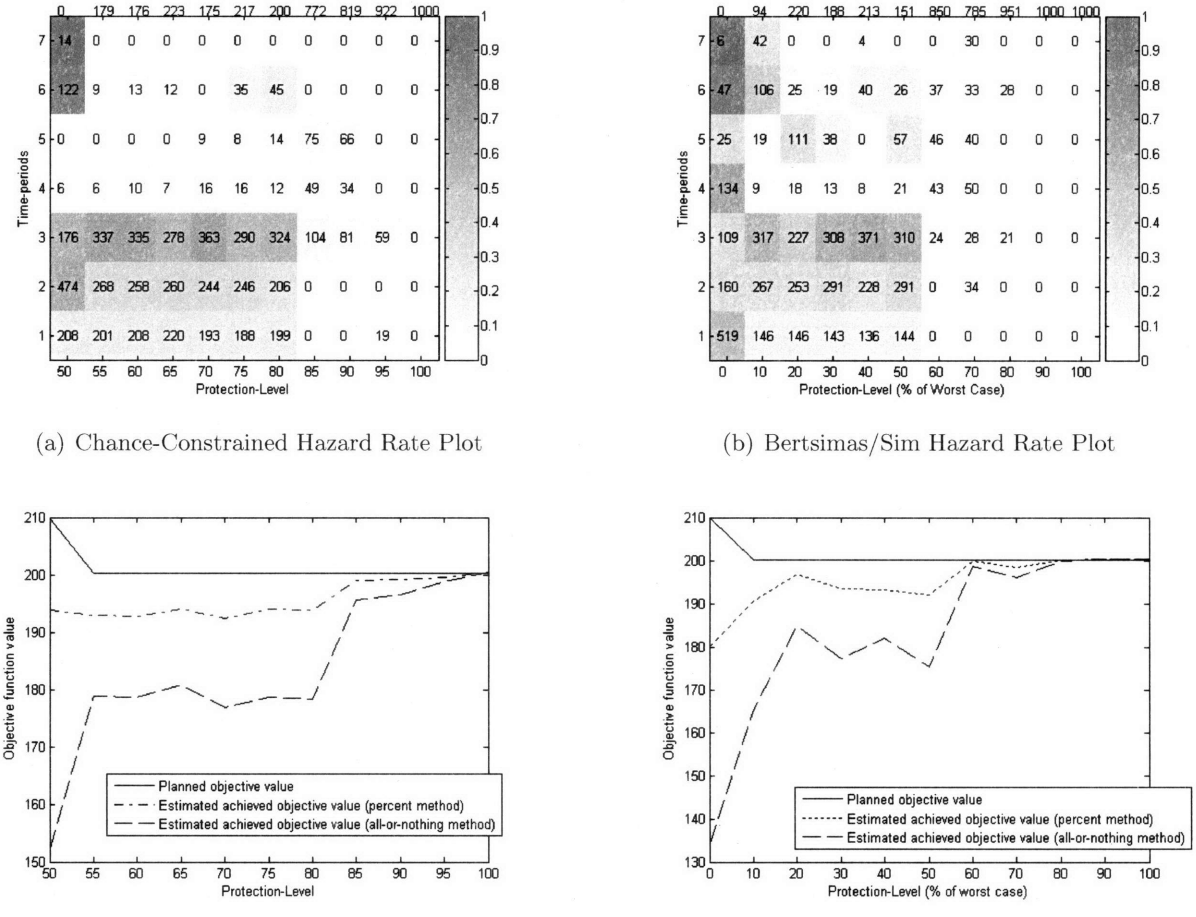
- *Test a model of the complete planning hierarchy, including squadron and vehicle-level planners.* In this research, we focused only on the theater-level planning problem. Solving this problem, in reality, is an iterative process that also involves solving the lower levels. Future research should model the interactions among all the levels of the planning hierarchy. Furture work should tie this thesis to the work done by Sakamoto[50].

- *Make a functional interface to test human interaction with the models.* The robust EBO models show potential for the human-in-the-loop planning process. In order to test human interaction with the models, we need to develop a working interface for the human planners. If the interface incorporates the simulations as suggested in Chapter 5, we need to make the simulations solver faster for practical implementation.

- *Further research into robust planning techniques and applications to EBO.* We have presented several approaches to applying robust planning to EBO. By no means are these the only or best approaches. The robust optimization techniques presented in this thesis have limitations and we made certain simplifying assumptions to develop a usable EBO Framework. Robust planning for EBO shows promise for making better plans. Research should continue to investigate other approaches and methods for robust planning and applications to EBO.

Appendix A

Normal Uncertainty Plans



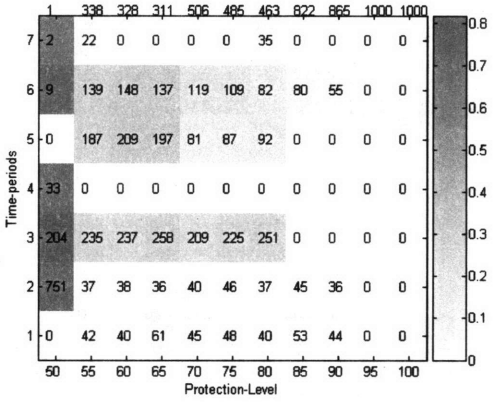
(a) Chance-Constrained Hazard Rate Plot

(b) Bertsimas/Sim Hazard Rate Plot

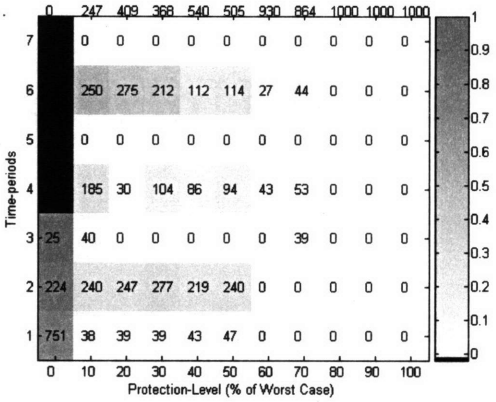
(c) Chance-Constrained Achieved Obj. Func.

(d) Bertsimas/Sim Achieved Obj. Func.

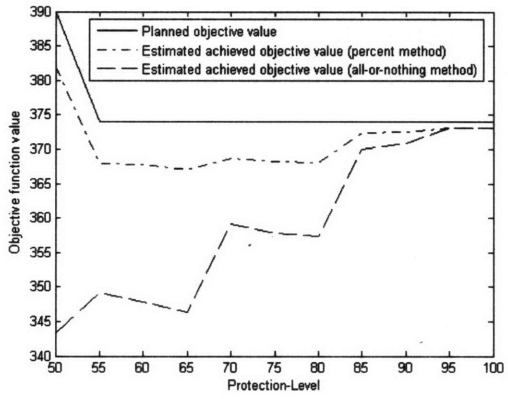
Figure A-1: Scenario 2 Performance with Normal Distribution



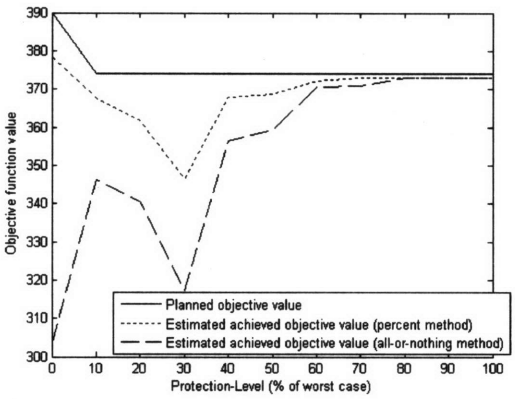
(a) Chance-Constrained Hazard Rate Plot



(b) Bertsimas/Sim Hazard Rate Plot

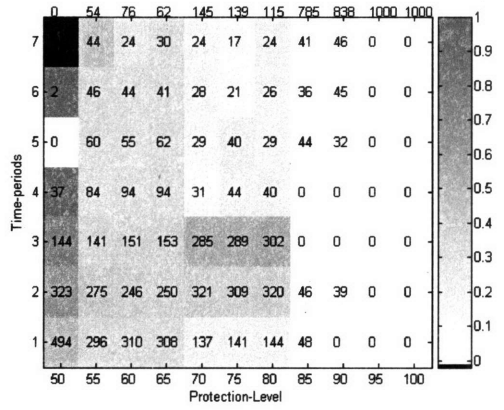


(c) Chance-Constrained Achieved Obj. Func.

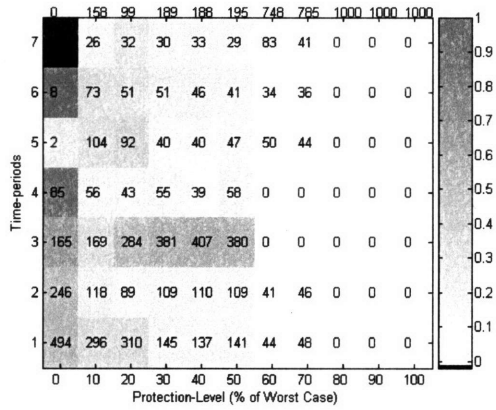


(d) Bertsimas/Sim Achieved Obj. Func.

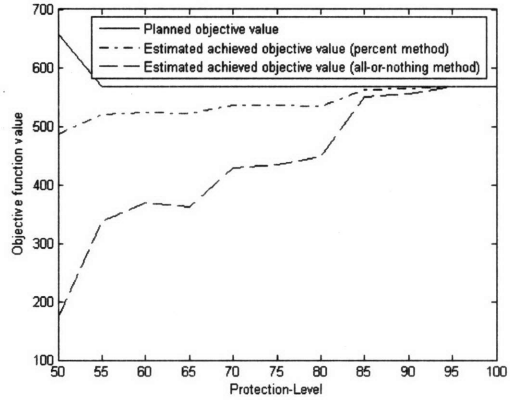
Figure A-2: Scenario 3 Performance with Normal Distribution



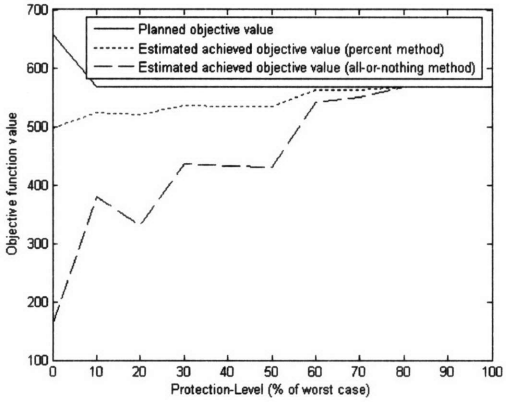
(a) Chance-Constrained Hazard Rate Plot



(b) Bertsimas/Sim Hazard Rate Plot



(c) Chance-Constrained Obj. Func.



(d) Bertsimas/Sim Achieved Obj. Func.

Figure A-3: Scenario 4 Performance with Normal Distribution

THIS PAGE INTENTIONALLY LEFT BLANK

Bibliography

[1] Air Force Doctrine Document 2-1, (Air Warfare). United States Air Force, January 2000.

[2] Air Force Doctrine Document 2, (Organization and Employment of Aerospoace Power). United States Air Force, February 2000.

[3] Air Force Doctrine Document 1, (Basic Doctrine). United States Air Force, November 2003.

[4] Air Force Doctrine Document 2-1.9 (draft), (Targeting). United States Air Force, October 2005.

[5] Air Force Doctrine Document 2-5.3, (Public Affairs). United States Air Force, June 2005.

[6] Air Force Doctrine Document 2 (draft), (Organization and Organization). United States Air Force, January 2006.

[7] C. Barnhart and L. Marla. Large-scale optimization: Models and algorithms for strategic, tactical, and real-time planning. Presentation for Independent Research and Development Project Mid-year Review, December 2005.

[8] T. Beagle. *Effects-Based Targeting-Another Empty Promise.* PhD thesis, School of Advanced Airpower Studies, Air University, Maxwell AFB, June 2000.

[9] A. Ben-Tal, L. El-Ghaoui, and A. Nemirovski. *Semidefinite programming and applications.* Kluwer Academic Publishers, Reading, Massachusetts.

[10] A. Ben-Tal and A. Nemirovski. Robust convex optimization. *Mathematical Operations Research*, (23):769–805, 1998.

[11] A. Ben-Tal and A. Nemirovski. Robust solutions of uncertain linear programs. *Operations Research Letters*, (25):1–13, 1999.

[12] A. Ben-Tal and A. Nemirovski. Robust solutions of linear programming problems contaminated with uncertain data. *Math Programming*, (88):411–424, 2000.

[13] D. Bertsimas and M. Sim. Robust discrete, dynamic optimization and network flows. Presentation at Massachusetts Institute of Technology, October 2002.

[14] D. Bertsimas and M. Sim. The price of robustness. Massachusetts Institute of Technology, Submitted to *Mathmatical Programming*, 2001.

[15] D. Bertsimas and M. Sim. Robust discrete optimization and network flows. Massachusetts Institute of Technology, Submitted to *Operations Research Letters*, 2002.

[16] D. Bertsimas and A. Thiele. A robust optimization apporach to supply chain management. Massachusetts Institute of Technology, 2003.

[17] L. Bertuccelli. Robust planning for heterogeneous UAVs in uncertain environments. Master's thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, June 2004.

[18] L. Bremmer, D. Koehler, V Liberman, and A Tversky. Overconfidence in problebility and frequency judgments: A critical examination. *Organizational Behavior and Human Decision Processes*, 65:212–219, 1996.

[19] S. Cambone. Unmanned aircraft systems roadmap, 2005-2030. Office of the Secretary of Defense, August 2005.

[20] R. Carter and M. Cahill. Regression models of search time for color-coded information displays. *Human Factors*, 21:293–302, 1979.

[21] A. Charnes and W. Cooper. Chance-constrained programming. *Management Science*, 6(1):73–79, October 1959.

[22] A. Charnes and W. Cooper. Deterministic equivalents for optimizing and satisficing under chance constraints. *Operations Research*, 11(1):18–39, 1963.

[23] A. Charnes, W. Cooper, and G. Thompson. Critical path analyses via chance constrained and stochastic programming. *Operations Research*, 12(3):460–470, 1964.

[24] A. Charnes and M. Kirby. Some special p-models in chance-constrained programming. *Management Science*, 14(3):183–195, November 1967.

[25] A. Charnes, M. Kirby, and W. Raike. Chance-constrained generalized networks. *Operations Research*, 14(6):1113–1120, 1966.

[26] A. Charnes and A. Stedry. A chance-constrained model for real-time control in research and development management. *Management Science*, 12(8), 1966.

[27] R. Christ. Review and analysis of color coding reasearch for visual displays. *Human Factors*, 17:542–570, 1975.

[28] P. Davis. Effects-based operations (EBO): A grand challenge for the analytical community. Technical report, RAND, 2001.

[29] R. Dawes, D. Faust, and P. Meehl. Clinical versus statistical judgment. *Science*, 243:1668–1673, 1989.

[30] W. Edwards. *Decision Making*, pages 1061–1104. Handbook of human factors. Wiley, New York, 1987.

[31] H. Einhorn and R. Hogarth. Confidence in judgement: Persistence of the illusion of validity. *Psychological Preview*, (85):395–416, 1978.

[32] L. El-Ghaoui. Robust solutions to least square problems to uncertain data matrices. *SIAM Journal of Matrix Anal. Appl.*, (18):1035–1064, 1997.

[33] L. El-Ghaoui, H. Lebret, and F. Oustry. Robust solutions to uncertain semidefinite programs. *SIAM Journal of Optimization*, (9):33–52, 1998.

[34] B. Fischhof and D MacGregor. Subjective confidence in forcasts. *Journal of Forecasting*, 1:155–172, 1982.

[35] D. Fisher and K. Tan. Visual displays: The highlighting paradox. *Human Factors*, 31:17–30, 1989.

[36] D. Griffin and A. Tversky. The weighting of evidence and the determinsants of confidence. *Cognitive Psychology*, 24:411–435, 1992.

[37] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*, chapter 13.2.1, pages 412–414. Springer-Verlag, New York, 2001.

[38] E. Johnson, R. Cavenaugh, R. Spooner, and M. Samet. Utilization of reliability estimates in bayesian inference. *IEE Transactions of reliability*, 22:176–183, 1973.

[39] D. Kahneman and A. Tversky. On the psychology of prediction. *Psychological Review*, 80:251–273, 1973.

[40] D. Kahneman and A. Tversky. Choice, values and frames. *American Psychologist*, 39:341–350, 1984.

[41] S. Kolitz and L. Forest. Robust HMCDM under uncertainty RHUU. Presentation Slides for Draper Laboratory Project, January 2006.

[42] R. Lathrop. Perceived variability. *Journal of Experimental Psychology*, 23:498–502, 1967.

[43] M. McCrabb. Explaining effects: A theory for an effects-based approach to planning, executing, and assessing operations. Technical report, Defense Technical Information Center, August 2001.

[44] J. Meyer, M. Taied, and I. Flascher. Correlation estimates as perceptual judgments. *Journal of Experimental Psychology*, 3(1):3–20, 1997.

[45] B. Miller and H. Wagner. Chance constrained programming with joint constraints. *Operations Research*, 13(6):930–945, 1964.

[46] D. Navon. Forest before the tress: The precedence of global features in visual processing. *Cognitive Psychology*, 9:353–383, 1964.

[47] D. O'Hare. *Aeronautical decision making: Metaphors, models, and methods*, chapter 5, pages 201–237. Principles and Practice of Aviation Psychology. Lawrence Erlabaum Associates, Publishers, Mahwah, New Jersey, 2003.

[48] S. Oskamp. Overconfidence in case-study judgments. *Journal of COnsulting Psychology*, 29:261–265, 1965.

[49] M. Richard and S. Kolitz. The adept framework for intelligent autonomy. Technical report, The Charles Stark Draper Laboratory, 2001.

[50] P. Sakamoto. Robust planning for UAV squadrons. Master's thesis, Massachusetts Institute of Technology, Operations Research Center, June 2001.

[51] A. Soyster. Convex programming with set inclusive constraints and applications to inexact linear programming. *Operations Research*, 21(5):1154–1157, 1973.

[52] E. Stone, J. Yates, and A. Parker. Effect of numerical and graphical displays on professed risk-taking behavior. *Journal of Experimental Psychology: Applied*, 3(4):243–256, 1997.

[53] P. Tsang and M. Vidulich, editors. *Principles and Practice of Aviation Psychology*. Lawerence Erlbaum Associates, Publishers, Mahwah, New Jersey, 2003.

[54] E. Tufte. *Envisioning information*. Graphics Press, Cheshir, CT, 1990.

[55] A. Tversky. Elimination by aspect: A theory of choice. *Psychological Review*, 79:281–299, 1972.

[56] A. Tversky and D. Kahneman. Judgment under uncertainty: Heuristics and biases. *Science*, 185:1124–1131, 1974.

[57] C. Varey, B. Mellers, and H. Birnbaum. Judgement of proportions. *Journal of Experimental Psychology: Human Perception and Performance*, 16(3):613–625, 1990.

[58] C. Wickens and A. Andre. Proximity compatibility and information display: Effects of color, space, and objectness of information integration. *Human Factors*, 32:61–77, 1990.

[59] C. Wickens and J. Hollands. *Engineering Psychology and Human Performance*. Pretence Hall, Upper Saddle River, New Jersey, third edition, 2000.

[60] P. Wright. The harassed decision maker: Time pressures, distractions, and the use of evidence. *Journal of Applied Psychology*, 59:555–561, 1974.

[61] W. Zhao. Multiple autonomous vehicle mission planning and management. Master's thesis, Massachusetts Institute of Technology, System Design and Management Program, June 1999.