3-1-2018

The Developmental Test Scheduling Problem

Joseph E. Schoenbeck

THE DEVELOPMENTAL TEST SCHEDULING PROBLEM

THESIS

Joseph E. Schoenbeck

AFIT-ENS-MS-18-M-160

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OFTECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENS-MS-18-M-160

THE DEVELOPMENTAL TEST SCHEDULING PROBLEM

THESIS

Presented to the Faculty

Department of Operational Sciences

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Operations Research

Joseph E. Schoenbeck, MBA

March 2018

AFIT-ENS-MS-18-M-160

THE DEVELOPMENTAL TEST SCHEDULING PROBLEM

THESIS

Joseph E. Schoenbeck, MBA

Committee Membership:

Dr. Jeffery D. Weir
Chair

Lt. Col. Jeremy D. Jordan

PhD
Reader

Dr. Stephen P. Chambal
Member

Abstract

Developmental testing of aircraft systems in the United States Air Force requires a complex set of resources for each test. The optimal scheduling of those resources is the job of the 412^{th} Test Wing at Edwards Air Force Base. With more than 20 different Combined Task Forces requesting resources for roughly 300 flying missions each week, manual scheduling is a difficult task. The current process takes a team of schedulers several days to get a workable result from which they can start tailoring the final schedule. While concepts and techniques can be taken from industry scheduling problems, the body of knowledge as it relates to developmental test scheduling is sparse. The contribution of this paper is to initially document the Developmental Test Scheduling Problem, define it in structured terms for which a solution methodology can be designed, and present an Integer Programming based solution. The design allows for a scheduler to tailor an initial answer to fit nuanced and timely objectives and constraints. For this prototype effort the problem is scoped to the "Iron" Resources while bearing in mind the extensibility of the approach to "Range" Resources. This study and prototype will demonstrate results that will create an initial schedule in several hours and serve as a good starting point for the final schedule.

Acknowledgements

I am thankful to many whose patient mentoring and teaching were instru-mental to this effort, and my development. Our project manager and developer make a great team. My Research Advisor, Dr. Jeffery Weir supported me throughout the entire AFIT experience, and specifi-cally in the crafting of this document. To my wife for her unwavering support and love on this unconventional journey that continues in partnership. This is dedicated to her.

Joseph E. Schoenbeck

Table of Contents

Page

List of Figures

List of Tables

I. Introduction and Background

1.1 Edwards Air Force Base

In September 1933 the Muroc Bombing and Gunnery Range was established by Colonel H.H. "Hap" Arnold who saw the area encompassing the Rogers Dry Lake Bed as a natural aerodrome that could be acquired at virtually no cost to the taxpayer. The 12x5 mile dry lake bed, made famous partly by the first landing of the Space Shuttle Columbia in 1981, is the centerpiece of the Mojave Desert land occupied by the base. The name was changed to Edwards Air Force Base in 1949 in honor of Captain Glen Edwards who was killed in a crash flying the YB-49.

> The 412^{th} Test Wing, located at Edwards, performs developmental testing of airframe, avionics, propulsion and electronic warfare systems of manned and unmanned aircraft for the Air Force, other U.S. military services and government agencies, and international partners. Current and recent systems tested by the wing include the B-2, F-22A, F-35, Airborne Laser, and Global Hawk. The wing's expertise in flying operations, maintenance and engineering ensures the successful test and evaluation of a fleet of more than 90 highly modified aircraft. Every single aircraft to enter the Air Fore Inventory - and a great many that failed to do so - has been put through its paces at Edwards.
>
> – Air Force Materiel Command

The 412^{th} Test Wing performs a unique mission in the U.S. Air Force. Developmental Test can be differentiated from other Test and Evaluation Centers operated by the 96^{th} Test Wing at Eglin and Holloman Air Force Bases, and from Operational Test

Centers around the country, by its diversity in aircraft, tests, and partners. Working with such a diverse group of systems and agencies requires flexibility and an equally diverse set of resources. Scheduling the Missions with those resources is the mission.

1.2 Motivation and Purpose

This paper defines the Developmental Test Scheduling Problem by focusing on a scheduling modernization effort for the 412^{th} Test Wing at Edwards Air Force Base. Scheduling Test Sorties is a process that takes intricate coordination across multiple squadrons and functional groups involving a myriad of players, hand-offs, and checkpoints. The wing relies on a strong network of relationships and communication channels to accomplish the overall mission of putting a test schedule together every week. These relationships and intense processes compensate for the lack of planning tools and structured data to put together the most effective schedule in the most efficient manner. The objective of this project is to define the problem in such a way that the solution can be designed and executed with Operations Research (OR) techniques.

The environment can be defined as centralized scheduling where a Combined Task Force (CTF) submits mission requests with one or more sorties requiring a myriad of resources. The CTF is a combination of military, government, and contractor personnel responsible for the testing of a particular aircraft system. They are sometimes encompassed by a squadron. The teams responsible for scheduling and conducting the tests evaluate the requests through a series of deconfliction meetings and processes to create a weekly schedule that runs through many iterations on its way to a final schedule. The Wing Commander will sign the final schedule for execution. The initial schedule can be unstable in this environment for several reasons:

- The inherent nature of developmental testing requires flexibility with priori-

ties and a fly-learn-fly methodology where the results of the current test can determine the nature of the next test. That flexibility is paid for in terms of scheduling instability, referred to as "thrash" or "churn."

- The shared, often scarce and complex resource environment contributes to the flexibility and fluidity of the schedule.
- There are a critical mass of resources not well established early enough in the process, and late changes to those resources cause a ripple effect to other resources, creating more thrash.
- The system designed to manage the process is an excellent scheduling execution platform but lacks the data elements and logic to be a robust pre-planning tool.

This confluence of destabilizing factors perpetuates a universal distrust of the initial schedule, thus daily activities are expected to finalize the executable schedule. The incompleteness of the planning elements along with the general distrust of the original plan can lead to scheduling behaviors which exacerbate the thrash.

The project charter as agreed to by the Air Force Institute of Technology (AFIT) Center for Operational Analysis (COA) and the Project Sponsor reads:

> For over 16 years, the Test Wing has relied on a highly labor-intensive scheduling process that is increasingly cumbersome, does not adequately account for the types of sorties being flown or significant constraint changes, and fails to take advantage of modern, automated optimization tools and methods to de-conflict the shared assets necessary for sortie execution. The current process requires schedulers to work each sortie into preliminary schedules a minimum of three times in the two weeks leading up to sortie execution. In practice, this number is even higher due to numerous changes in the set of constraints affecting a mission and the many downstream effects of any change. The process would benefit greatly by capitalizing on readily available systems engineering concepts and proven operations research practices in the scheduling and execution of its missions.

This statement was drafted by the Project Sponsor, Lt. Col Chris Keithley, who has spent more than a decade at Edwards in various capacities. At the time of this writing he is Commander, 416^{th} Flight Test Squadron (FLTS) and Director, Global Power Fighter CTF. He responsible for the F16 Fleet which both supports missions and flies the lead on missions in support of Air Force testing priorities. Lt. Col Keithley sees the need to analyze the process from an Operations Research mindset to put in place the analytical rigor to enhance the process and the outcomes.

1.2.1 Success Outcomes.

The motivation to initiate this project is to improve scheduling efficiency and transparency, while enhancing the effectiveness of the schedule produced.

1. Efficiency: Building a schedule quickly thus allowing time for schedulers at all levels to enhance the process and system through creative solutions, build relationships with the engineers and the broader test community, and work towards more effective missions.

2. Effectiveness: Measuring the schedule by the quality of the throughput. A quality mission is one that advances the objectives of the Program and Unit, and does not merely fill flying hours. This ultimately leads to more test programs being executed in a timely manner, and the war-fighters being faster to operational testing and capability.

3. Transparency: Beginning with an upfront plan that is understood and can be evaluated in totality early in the process. This also means understanding the results in the context of the priorities and established rules.

Efficiency can be roughly measured by the man-hours it takes to produce a schedule and the number of changes that schedule endures on its path to execution. The

squadrons optimize their tests on a longer time horizon and then submit that week's portion of their plan to the central scheduling process. The planning week process takes requests and turns them into a flying schedule. The inefficiency in the process is seen in rescheduling missions multiple times before they fly, and also in a high cancellation rate. Inefficiency can lead to sub-par effectiveness as time is dedicated to simply getting missions on the schedule rather than value added activities ensuring the success of those missions.

Effectiveness of the schedule can be nebulous to measure. "You know a good schedule when you see one" is a phrase that schedulers in all industries understand, and this environment is no exception. Those involved in the process generally believe their planning efforts produce an effective schedule. Although, nearly all concede they should get to that schedule with less rework. The simplest explanation of a good schedule is one where the most important missions are scheduled with the right resources in the time slots that give the highest probability for success. While pure throughput is one measure of effectiveness, not all missions advance the overall program at an equal rate. A good scoring mechanism is necessary to capture high quality missions, a thoughtful rule set is required to dole out the resources consistently, and a robust process is needed to ensure accurate and timely input and output. Every aspect of the problem from the data capture, to the mission requests, to the handling of unscheduled missions contributes to the overall perceived effectiveness. A robust and efficient process leads to fewer reschedules which leads to a more effective schedule in the form of a lower cancellation rate and high quality tests minimizing retesting.

Transparency in this context means the process is perceived to be fair and consistent. Some of the Air Force Program objectives contain classified and secret information which will remain opaque. There are political sensitivities in balancing the internal and external (contractor and government agencies) programs compet-

ing for scarce resources. Therefore, the priorities as input to an optimization effort need not be public or fully transparent in order for the scheduling process to be so. Transparency contributes to efficiency and effectiveness in that it provides a foundation for process trust amongst the participants, which leads to higher quality input perpetuating higher quality output.

Creating efficiency, effectiveness, and transparency can be summarized as:

1. A Robust data input with processes and data structures capable of capturing enough fidelity critical to an optimization program.

2. A trusted scoring system that allows for cognizant trade-offs in schedule building and encourages good behavior amongst the squadrons.

3. An optimized plan before the schedule is released to the base.

4. An enhanced process which can effectively leverage the results of the optimization.

The optimized plan is not effective without the other aspects of a solution, so a successful project will build an optimization ecosystem, not just an optimization model.

1.3 Background

In a developmental test environment, a "test" of a capability, system, configuration, or procedure is referred to as a Mission as it has a specific purpose to meet the objectives of the Program funded for research and development. Missions are made up of one or more Sorties, or a single aircraft playing its part in the Mission. The taxonomy of a Mission can be depicted as in Figure 1. The Programs are Air Force funded entities which the Combined Task Force Squadrons are responsible for carrying out, reporting on, and analyzing. These programs reflect the Air Force capability

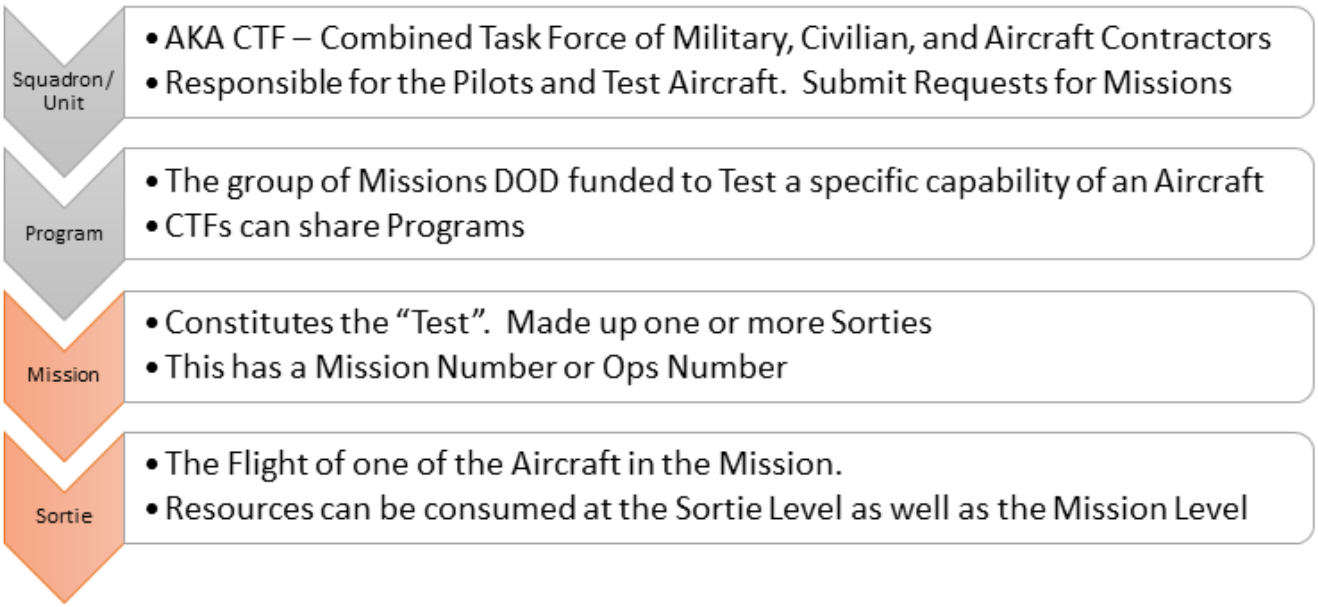

Figure 1: Mission Taxonomy

priorities. The completion of a program has a range of outcomes depending on the maturity of the system. Successful programs advance to an Operational Test Phase while others are canceled or stalled. The more tests the Wing can carry out, the sooner Programs can be completed, increasing the capability of the Air Force.

1.3.1 Functional Scheduling Departments.

The Long Range Scheduling Team is part of the Operations Groups and is tasked with understanding where the programs intersect with high level capabilities. They make recommendations that will avoid conflicts before they occur.

The Short Range Scheduling Team has the operational responsibility to systemically shepherd missions through the system, share timely information with the range and maintenance teams, and coordinate with outside agencies like TACC (Tankers) and other services for external range support or shared frequencies. The Long and Short Range teams work together as Central Scheduling to turn program initiatives

and flying requests into a tangible flying schedule.

The Maintenance Team is responsible for scheduling the F16 fleet to missions. They work closely with the Long Range and Short Range Teams to make the core of the central scheduling function.

1.4 Scope

The scope of the initial work is to demonstrate the ability to efficiently optimize a centralized schedule, based on criteria from the Testing Units and Operations Group, on a significant subset of the resources. Other projects have tackled squadron scheduling and while that context is important this project scope is to optimize central scheduling.

The project is designed in phases with the overall goal of a comprehensive planning solution. The solution encompasses a scheduling ecosystem that includes data input and management, a modeling engine, and process infrastructure. The centralized resources can be broken into roughly two groups; "Iron"" and "Range". The Iron refers to scheduling the support F16s, and their maintenance and configuration requirements. This includes scheduling the Tankers to ensure Missions can refuel in their flying windows. The Range encompasses the airspace to fly, the control room to monitor, and the telemetry resources to capture test data (including the TM frequencies). In short, the project is to build a comprehensive planning application which produces a high quality schedule as input to the scheduling execution system (CSE) and process. The full scope is large enough to encompass several phases of work. Figure 2 roughly depicts the prototype scope and thesis focus relative to the overall scheduling problem scope. There is still more modeling and analysis to be completed before a product is delivered to the Test Wing. This thesis represents a significant portion of the prototype and model design work towards the goal of a

finished product.

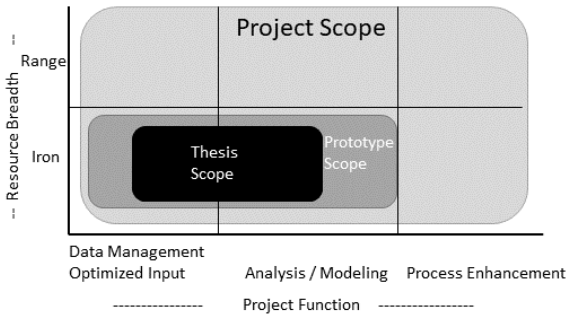

Figure 2: Thesis Scope

The prototype phase is designed to show all levels of the Test Wing that this problem can be solved. Having the Iron scope to build confidence externally has been critical to success. All development takes place with the understanding that the Range will come into scope as soon as the Iron Model is verified and validated to avoid short sighted or expedient decisions.

1.5 Conclusion to Chapter

The scope of this thesis is to analyze the problem data and structure, build and solve an appropriate model for schedule optimization, analyze the results coherently, and provide insights for successful completion of the overall project. The resource breadth of the thesis will encompass the "Iron" to demonstrate the modeling solution. The goal is to design an optimization ecosystem that is flexible enough to model the complexity, and robust enough to be scalable to the entire breadth of resources.

II. Problem Structure and Data Analysis

2.1 Introduction

While others have tackled the topic of Test and Evaluation [11], the literature relative to Test Scheduling is sparse. Given the unique nature of Developmental Testing it is difficult to find directly relevant material to research. Thus one contribution of this work is to define and structure the problem so that research can be done in a cohesive way for constant improvement. The search for relevant industry scheduling parallels will be aided by describing the problem in the most fundamental terms possible.

The inputs into the problem are mission requests. This batch of requests is akin to a batch of service orders that are fulfilled, or not, with defined centralized resources over a finite time horizon. The fulfillment of these requests in this problem is scheduling the test at the appropriate time, with the appropriate resource mix, to enable success. These orders, or complex tasks, compete for shared resources and require exact scheduling so all the resource needs are coordinated. The material resources are manned and prepped by the pilots, maintainers, and control room personnel. The schedule is defined in terms of time blocks in which the resources are coordinated to fulfill the request. The schedule is executed to some degree of success, with the results feeding back to the requesters as input to plan the next batch of orders.

2.2 Scheduling Constructs

With these building blocks in mind, the following scheduling constructs were researched to provide insight into the problem and creative ideas for the solution.

The resource constrained scheduling problem (RCSP) is scheduling a set of com-

plex tasks requiring resources. A good overview for comparison to other scheduling algorithms and approaches is found in Lawler's survey article [7]. The idea of a basic Gantt chart time horizon is part of the solution output and the overall design is influenced by critical path thinking. The phenomena in the Developmental Test Problem is that often high priority Missions (e.g. F35 Integration Testing) are resource intense and have the least amount of time flexibility. These "big rocks" can be thought of as being on the critical path of resource consumption. Proficiency flights, with more flexibility and limited resource consumption, can be used to fill the slack in the schedule. Terms like "holes" are used to denote the notion of slack.

A parallel machine scheduling approach was adapted in the formulation of Crash Test Optimization at Ford Motor Co, which shares many similarities to scheduling missions. The Ford scheduling work drew some inspiration from a seminal article on parallel machine scheduling and solution techniques [1]. The requests in the developmental test scheduling problem could be viewed as jobs being worked on by parallel non-uniform "machines" over a make-span. So the mission is effectively split between resources all working on the task at the same time. There are a finite number of machines of each type to perform a certain function. If the aircraft and other resources are viewed as machines, this has applicability. None of the constructs fit perfectly and that is to be expected when defining a problem of this nature.

2.3 Classifying the Developmental Test Scheduling Problem

Taking the fundamental description above, it is important to classify the problem in terms of classic scheduling language and notation. Most scheduling descriptions use the $\alpha \mid \beta \mid \gamma$ scheme to classify the problems [7]. α represents the machine environment, β captures the processing restrictions and constraints, and γ denotes the objective of the scheduling problem. This classification is taken from the Scheduling

book by Pinedo [9]. This system allows for new classifications to be slight variants of a more generalized classification that indicate appropriate solution algorithms.

2.3.1 The Machine Environment.

The machine environment for the Developmental Test Scheduling Problem is a specialized Job Shop where parallel "machines" are in work centers, but not any machine can work on any task. The best notation for this, in line with page 15 [9] is J_c as a combination of J_m and FJ_c. Jobs are not routed in this case, but there are offsets such that a work center can start ahead of another. A machine in each work center will work on, or support, one job at a time, and each job will use a machine in each work center at most one time.

The work centers in this case are the resource groups that include F16s, Tanker Refueling Support, Control Rooms, Communication Equipment, and Airspace. The individual machines are the Aircraft, available Tankers, specifically designed control rooms, numbered antennas, frequency bands, and flying areas at altitudes.

A fundamental shift from a classic scheduling problem is that the jobs, or missions, do not move between work centers. The work centers focus their effort on the mission when it is scheduled. That shifts the focus from how to move the job through the machines to how to coordinate the machines in a work center to work on the proper jobs in unison.

2.3.2 The Processing Environment.

The processing environment contains multiple entries to describe the problem. Here are the applicable entries in this case:

- $r_j \pm \sigma_j$: Each job has its own release date with some flexibility but is not released at the beginning of the make-span. Job j may not be available to start

until $r_j \pm \sigma_j$. The due date is not applicable at the job level as the mission has an inherent duration at which time all the machines it needs must be working. All jobs are due at the end of the week.

- *prec*: While not like a Project Scheduling Problem where there is flow to the completion of a single project, there are precedent relationships such that $C_1 < C_2$. In this case the first job may be completed independent of the second job being completed.

- $s_{j1,j2}$: There are processing times (Turn Times) on the F16s and the control rooms and equipment. These times are dependent on the previous job. For example, a flying mission needs 3 hours of turn time on an F16, while a ground mission does not require any turn time, with the restriction the next job must also be grounded.

- $fmls$: The previous example is that of a family of jobs (Air vs Ground). There are other examples such as F35Bs versus F35As requiring different types of resources for refueling.

- $brkdwn$: There are machine breakdowns in this environment that the initial formulation of the planning problem does not consider. Later implementations should react to breakdowns, and even plan for them in a stochastic sense.

- $M_{j,c}$: This notation implies that not any machine can work on Job j, rather only a subset are available. For example, a sortie needs a certain type of F16 in the fleet, and some missions require control rooms with more capability such that not any will do. The c subscript implies the further classification of the machines into the work centers.

2.3.3 The Objective.

The objective is to schedule as many weighted missions as possible. The make-span is the flying horizon (5 day week for initial implementation) but not all times are appropriate for all Missions. The objective $\sum_j w_j C_j$ is not applicable here as the completion time compared to a due date is not relevant. The job is either completed or not. A unit variable is used to describe a completed mission:

$$x_j = \begin{cases} 1 & C_j > 0 \\ 0 & otherwise \end{cases}$$

The objective can then be defined as:

Max $\sum_j w_j x_j$

S.T.

$(r_j - \sigma_j)x_j \leq C_j x_j \leq r_j + \sigma_j$

In this scenario, the job must be completed in a window of time or not at all. The mission in this case is generally canceled and resubmitted for another week. This represents another fundamental shift from a classic scheduling problem that attempts to minimize the make-span. The classic batch of orders are more central and the objective is to minimize the lateness or tardiness of that batch. The Developmental Test Scheduling Problem operates on the rhythm of a flying week and all the infrastructure for the broader mission supports that rhythm. The objective is to schedule as many quality missions as possible in the flying week that becomes a finite make-span.

The compact notation for the Developmental Test Scheduling Problem is :

$$J_c \mid r_j, prec, s_{j1,j2}, fmls, M_{j,c} \mid Max \sum_j w_j x_j$$

Given the fundamental departures from classic scheduling in the α and γ parameters, and the quite complex β parameter of the Developmental Test Scheduling Problem, it is difficult to find a directly applicable solution technique. In discussing the job shop environment, Pinedo states that only the two machine case can be found to solve reliably in polynomial time. Other special circumstances require all processing times to be 0 or 1 (page 184)[9].

At this point of complexity, the solution techniques in the literature turn to math programming formulations with branch and bound solutions, of which Integer Programming formulations are a subset.

2.4 Scheduling Solutions

This section looks at projects that are of similar form or environment to developmental test scheduling. The process of scheduling pilots to missions over a broader time horizon has been studied in previous AFIT theses. This work looked at the process of generating a sortie schedule to submit to central scheduling, thus the feeder to the central scheduling problem. In 1991 Lisa Hassel formulated a 0-1 Integer Program (IP) to solve the "TPS Scheduling Problem." The Test Pilot School (TPS) is a squadron making weekly requests to the Edwards central scheduling process in order to fulfill curriculum requirements. The TPS Problem is essentially assigning students and instructors to curriculum sorties requiring a specific type of aircraft over a time horizon. For the same reason an assignment problem is generally difficult to solve as an IP, the combinations in this problem became intractable as she modeled a problem of reasonable time duration. There were simply too many combinations of time slots, sorties, students, and instructors to solve. Hassel worked on a series of pre-processing techniques, using acute knowledge of the inputs, to cut the problem down, but concluded only subsets of the course could be modeled this way [5]. In 1992 Gary Foster

developed an iterative heuristic that was applied to a new landscape each week [3]. Twenty Five years later the "TPS Problem" is still solved with heuristic and rule based methods with a constant monitoring as the situation changes each week.

Other well documented heuristic scheduling approaches come from the General Employee Scheduling Problem. Several articles discuss applying Tabu Search to scheduling problems that have a Mixed Integer Programming (MIP) formulation [4]. Heuristic techniques have been applied to well structured problems, and successful implementations in projects are documented. Dowsland's nurse rostering implementation is a good example of applying Tabu Search [2].

The Ford Motor Company is engaged in a multi-year project to enhance the scheduling of crash tests. The Test Planning Scheduler Support System (TP3S) demonstrates success at the end of a scheduling project implementation. Like scheduling test missions, scheduling crash tests is costly and complicated with the results having impactful consequences to the design of the aircraft or car. The University of Michigan and the Ford team describe their core problem as a hybrid between bin-packing and the parallel machine scheduling problem. The bin-packing aspect is that each crash car aims to accommodate multiple tests with different criteria. If the bin (prototype car) is configured optimally within the rules, it will accommodate the most items (tests) in a single crash and the schedule will get more throughput. The parallel machine scheduling aspect reflects time sensitive tasks with given processing durations that need to be scheduled on a given set of machine resources to minimize a time-related additive criterion, such as the make-span [10]. The team employed an explicit enumeration methodology to defining sets for a Mixed Integer Program to keep it manageable in size. They took the added step of solving the problem with a column generation technique as some instances were still too large for the memory of the solution platform.

From a planning application standpoint, TP3S interacts directly with the engineers who dial in the tests they need, optimizing through "runs" the car configurations and the times at which they should be tested. The Development Test Scheduling Problem structure has a layer of separation between the engineers and the schedule in the form of squadron schedulers. However, the concept of getting detailed test requirements and configuration specifications through a robust user interface and optimizing that input to form a comprehensible schedule is applicable [10].

2.4.1 Integer Programming as a Generic Construct.

The utility of an Integer Program (IP) approach is that it is flexible enough to formulate in light of the constructs discussed without having to fit neatly into any one structure. Integer Programming Formulations and Solutions abound. The Air Force Research Laboratory and The Massachusetts Institute of Technology Center for Transportation and Logistics have teamed up on several military and commercial projects with IP Formulations. Reading about these is good validation for such an approach, and interesting to see the creative solution techniques that can be applied to these formulations [6]. Even a cursory literature review will reveal that real world (MIPs) and IPs are notoriously combinatorially intense and thus too large for basic techniques like Branch and Bound or Cutting Plane techniques. The work of Hassel and [5] and Foster [3] demonstrates how quickly a problem can explode, particularly if all the combinations of discrete time, missions, and resources are allowed to persist unchecked.

2.5 Current State Data Analysis

To further explore the problem structure, data profiling is necessary. Three types of data are helpful to this effort:

1. Operational Data to Produce a Schedule. This is sourcing, cleaning, and organizing the request and parameter data used to build a schedule, so that an optimization model can be built. This includes looking forward and backward at source data to support building and validating a schedule. The sources of the data are the Central Scheduling Enterprise System (CSE) and a Maintenance Data Warehouse maintained locally at Edwards Air Force Base. This is where a majority of the data has been sourced in order to produce a working prototype. Part of this work's contribution is to define, in a specific way, the data that must be gathered in order to make scheduling optimization possible.

2. Historical Profiles. This includes the number of sorties flown and the hours flown, and other resource consumption metrics. Given the "Iron" scope for the prototype the focus will be on the F16 data profiles.

3. Efficiency Measures. This involves crafting some high level metrics related to the process of generating a schedule. This is helpful for comparison purposes to determine the impact of an optimized schedule early in the process.

This paper is not a statistics focused effort, and thus this section seeks to gain an understanding of the problem rather than test the suitability of a probability distribution to the data. The discussion around the nature of the data is valuable for context when considering results, and will become important for follow on statistical efforts that become the basis for optimization model parameters, or robust inputs to the Integer Program.

2.5.1 F16 Profile Data.

Figure 3 is a contextual measurement of F16 flying hours. The nature of the tests changes rapidly as programs finish and new ones get approved, and while the data is not simply demand driven, there are some interesting observations:

Year	Role	Data Weeks	Avg #F16s Flown	Avg Sortie Count	Avg Flying Hours	Avg Sortie Count / F16	Avg Fly Hrs / F16	Avg Fly Hrs / Sortie
2014	SUPPORT	12	11.8	56.3	229.8	4.8	19.4	4.1
2014	TEST	12	5.5	14.3	42.1	2.6	7.7	2.9
2015	SUPPORT	50	10.3	53.7	200.6	5.2	19.4	3.7
2015	TEST	50	6.9	21.9	89.2	3.2	13.0	4.1
2016	SUPPORT	50	9.7	42.5	140.6	4.4	14.5	3.3
2016	TEST	50	7.0	20.7	78.1	3.0	11.2	3.8
2017	SUPPORT	25	11.0	47.7	145.4	4.3	13.2	3.0
2017	TEST	25	6.7	17.2	58.9	2.6	8.8	3.4
2014		12	17.3	70.6	271.9	4.1	15.7	3.9
2015		50	17.2	75.6	289.8	4.4	16.8	3.8
2016		50	16.7	63.1	218.7	3.8	13.1	3.5
2017		25	17.7	64.9	204.3	3.7	11.5	3.1

Figure 3: Annual F16 Usage since Late 2014

- The wing utilizes between 16 and 17 jets each week, which represents roughly 65% of the fleet.
- The average number of sorties for an active jet is between 3.7 and 4.1.
- The average duration of a sortie is between 3.1 and 3.9 hours, so the missions from 2014/2015 are approximately 48 minutes longer on average than those in 2017.
- The combination of fewer sorties per jet and shorter sorties has led to a roughly 30% decrease in total flying hours from 2015 to 2017. This is impactful as it precipitates the need for less phase maintenance (every 300 or 400 hours per jet) or general maintenance, and thus less maintenance staff.
- There are two main roles for an F16, Test or Support, meaning the jet is the focal point of what is being tested (Test) or in support of such (Support). Once a jet is in a particular role it can only fly sorties fitting that mission's purpose.

The next interesting delineation is the flying by Aircraft Model (A,B,C,D). Given the shift in 2016 to fewer flying hours, the latter two years are used to obtain a clean

profile of jet usage by model. The D models are two seat aircraft that can be used for

MDS	Weeks Flown	Pot Wks	% Wks Flown	First Week	Last Week	Total Sortie Count	Avg Sortie Count (Fly Week)	SD Srt Ct	Total Fly Hrs	Avg Fly Hrs (Fly Week)	SD Fly Hrs
F16D	697	1326	53%	2016 - 1	2017 - 26	3,034	4.4	2.7	9,718	13.9	13.8
F16C	432	858	50%	2016 - 1	2017 - 26	1,347	3.1	1.8	4,985	11.5	11.3
F16B	79	312	25%	2016 - 2	2017 - 26	249	3.2	1.9	1,028	13.0	9.8
F16A	99	312	32%	2016 - 2	2017 - 26	224	2.3	1.3	614	6.2	4.1
	1,307	2,808	47%			4,854	3.7	2.4	16,345	12.5	12.5

Figure 4: F16 Usage by Model for 78 weeks of 2016-17

training and chase missions. D models can be used sometimes when a C is called for, but the single seat C model cannot substitute for a D. Models A and B are almost exclusively used as Test aircraft, which fly less, and are often configured with special software needed for specific tests. As of this writing, there are no B models in the fleet. The percent of weeks flown is measured for the type of aircraft as a whole, so while 65% of the available aircraft are in use in any week, the aircraft comprising this data were not available in all weeks. Therefore, the overall model shows a much lower utilization over the course of the 78 weeks.

The utility of an Integer Program is the ability to run "what-if" analysis in determining the effectiveness of a schedule that has a different fleet composition. Strategically optimizing the characteristics of the fleet, which is very diverse, is a good extension of this work.

2.5.2 Probability Distributions for Sorties and Hours.

Ideally, subsequent models will attempt to predict the flying hours of a jet to inform the optimal hours to schedule. The secondary goal is to optimize the flow of jets into phase maintenance. Phase maintenance refers to the prescribed checks performed on a jet after so many hours of flying. This concept exists across all aircraft types and uses. The goal of maintenance, given limited staffing and for maximum utilization, is

to have one aircraft in maintenance and one ready for extended maintenance at any point in time. This means a crew is busy working at all times, but only one jet is out of service. This is tedious to manage manually and well suited for a scheduling model. The current process grants to maintenance the responsibility for defining the minimum and maximum hours a jet can fly in a week. Armed with probability distributions a model can be tuned to determine the same.

Maintenance and Operations influence a request to shape the demand towards desired jets to meet their objectives, thus the demand data is not pure.

Intuition indicates that the sortie counts, being discrete and low, follow a Poisson Distribution. Figure 5 is the result of checking that intuition using a λ that is equal to $\overline{x}$, as $\overline{x}$ is the maximum likelihood estimator of λ for the Poisson Distribution. There

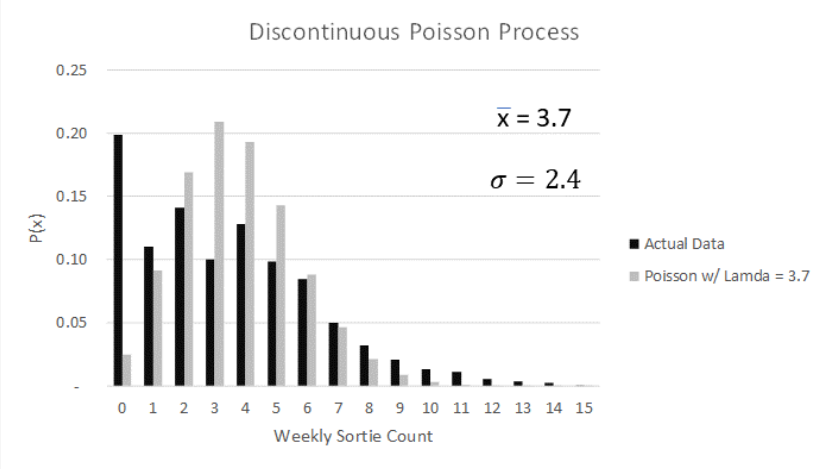

Figure 5: Fitting Sortie Counts to a Poisson Distribution

is a natural discontinuity that takes place between zero and one for sorties and hours. The difference between not flying and flying one hour is more significant than flying one sortie versus two sorties, and definitely one hour versus two hours. Since 35% of the available jets are not on the schedule any given week, zero is the highest frequency of sorties and hours, and can obscure how the data fits a distribution. Perhaps a two stage distribution could be explored that first has a binomial distribution capturing

the probability of being on the schedule, then a separate distribution that predicts the sorties.

The flying hours are a continuous random variable, but the sample data suffers from the same problem with zero hours dominating the frequency counts, thus making it difficult to fit a distribution. This can be diluted to some extent by grouping the data into buckets so that zero is grouped with one to four hours leaving buckets that follow an exponential pattern with a parameter λ that will be off at first due to the influence of zero, but then fit well. Figure 6 is the histogram of flying hours along with the empirical CDF and a fitted Exponential Distribution with $\lambda = .10$; again from 2016 to mid 2017. The aggregate distributions are helpful in understanding how jets expend hours, but might not be as useful for predicting the future flying hours of any particular jet. The diversity of the fleet would seem to indicate each jet needs its own distribution parameters.

2.5.3 Efficiency Measures.

The instability of requests can be seen in the number of missions that are canceled and added in a week. For the first 25 weeks in August, an analysis of the data shows that 2,596 missions were canceled and 2,516 were added. A good portion of these are the same physical test with a different identifier (OPS_NUM). Roughly 100 missions per week were impacted. Of the missions that persisted throughout the week, 2,045 changed their start time with 875 of those changing days.

Another measure of efficiency is the time taken to produce a schedule. The Maintenance Pro-Super spends 10-14 hours from Friday afternoon to Monday afternoon scheduling F16s to Missions. That plan is modified in "Tanker Wars" and subsequent compromises throughout the week. At Iron and Tanker Wars there are at least 30 people that spend at least two hours manipulating the F16 and Refuel portions of the

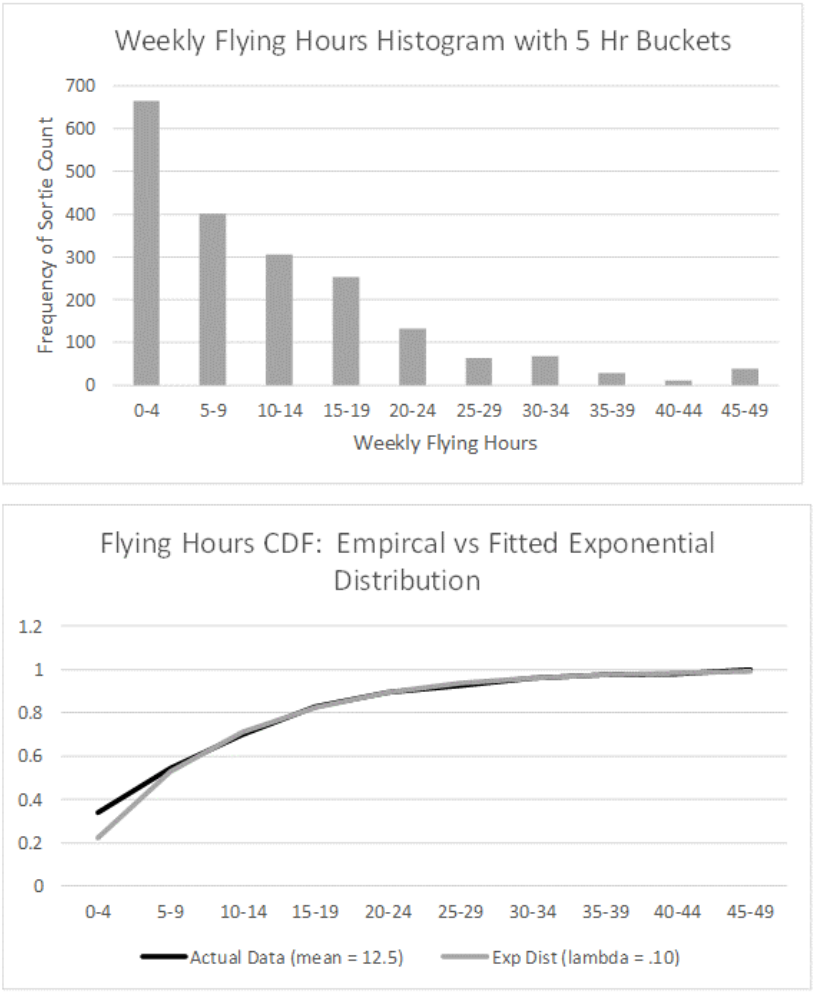

Figure 6: Flying Hours Histogram and Cumulative Distribution Function

Schedule. This is followed up by other meetings to further deconflict as the process unfolds throughout the week. That means a minimum of 70 man-hours of time that might be avoided by several hours from a small team and an optimization model.

A goal of this project is to create efficiency by getting robust input, and optimizing it before releasing a published schedule to minimize the changes and create more productive hours for the schedulers.

2.6 Specific Problem Parameters

The following attributes of the Developmental Test Scheduling Problem are pertinent to designing a workable solution.

2.6.1 Time and Volume.

Time Horizon.

The mission requests submitted represent a relatively short period of time. The relevant window is at most two weeks and more realistically one week with perhaps several missions that could be pulled forward from the succeeding week. This is in contrast to the more linear scheduling problem faced by Foster and Hassel in their effort to formulate an Integer Program to slot pilots and aircraft to a curriculum [3] [5].

The requested time periods are limited. Although some types of requests such as Proficiency Missions or Standby Missions can go anytime during the flying horizon, most requests have relatively tight boundaries around the desired start time. This has been a major area of collaboration with the teams to open up the time windows to have more scheduling freedom. In the current scheme, the squadron requests one point in time and the weekly process determines the range of possibilities to ultimately schedule the mission. In this process the time windows will always be limited. External factors like pilot availability, aircraft availability, engineering test plan dependencies, and human factors make an unconstrained schedule untenable for the requesters.

Time Input.

The squadrons currently provide only a desired start time to CSE which gets adjusted multiple times on the path to a finished schedule. A quick analysis of CSE

data, as seen in Figure 7, demonstrates how the missions are getting spread from what was requested. This elongation is not necessarily against the wishes of the squadron

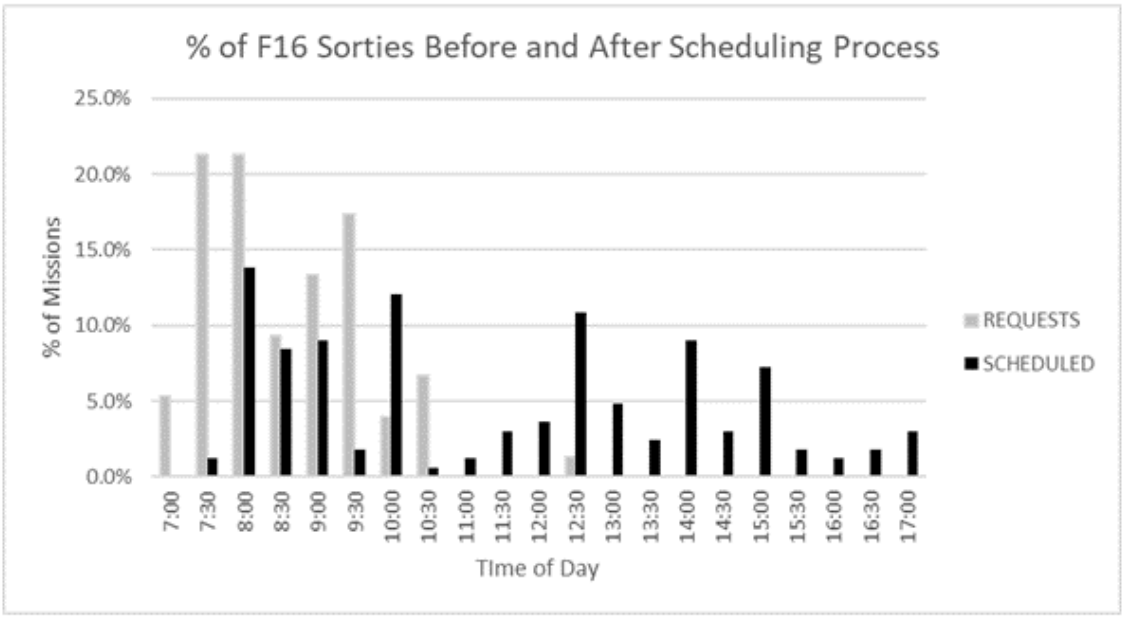

Figure 7: Request vs Scheduled Time Periods (Two Weeks in August 2017)

as experience tells them this will happen.

Volume of Requests.

The number of requests each week ranges between 250 and 400. The impact on variables these requests might generate depends on the interaction with the resources. One third of the missions require F16 support, need refueling, or be involved in some other relationship that requires special handling to determine the proper resource alignment. For two-thirds of the missions, a model must only ensure that aircraft assigned by the squadron are protected from flying two missions concurrently. There is no conflict for centralized resources for these missions, limiting the associated variables. This will change with the addition of the Range Resources by adding to the missions requiring special handling.

2.6.2 F16 Characteristics.

Scheduling the F16s to a mission is the most intense resource allocation task. The data to optimize the schedule is lacking in direct input from the squadrons and parameters from the Operations Group. Additional data elements must either be collected directly from the schedulers or parameter driven rules will have to make inferences. Figure 8 is a subset of the F16 attributes that a mission might specifically require, be indifferent to, or not desire. To optimize the scheduling of the F16s,

VIPER PILOT GOUGE

Tail #	Block	Engine Type	Inlet	OFP	FCR	CMDS	Link 16	IFF	Video/Data Recorder (current config)	JHMCS	High AOA	DAS	TSPI		ACE II Pod	ACE III Pod
													G-lite or GAINR (Installed)	ARDS Pod		
F-16D 87-0377	30	GE100	Big	SCU-7	APG-68V1	ALQ-213/ALE-47**	SADL	3A, 4	ENERTEC (Note 7)		H/B	TTC		x		
F-16D 87-0378	30	GE100	Big	SCU-7	APG-68V1	ALQ-213/ALE-47**	SADL	3A, 4	ENERTEC (Note 7)		H/B	TTC		x		
F-16D 86-0050	30	GE100	Big	SCU-7	APG-68V1	ALQ-213/ALE-47**	SADL	3A, 4	ENERTEC (Note 7)		H/B	TTC		x		
F-16D 87-0391	40	GE100	Big	M6.2+	APG-68V1	ALE-47	x	3A, 4, S	ENERTEC (Note 7)	x	H/B	TTC	GAINR-2 and GAINR-3	x		
F-16D 90-0797	40	GE100	Big	M6.2+	APG-68V1	ALE-47	x	3A, 4, S	ENERTEC (Note 7)	x	H/B	TTC	GAINR-2 and GAINR-3	x		
F-16C 86-0371	30	GE100	Big	SCU-7	APG-68V1	ALQ-213/ALE-47**	SADL	3A, 4	ENERTEC					x		
F-16C 85-1547	30	GE100	Small	SCU-7	APG-68V1	ALQ-213/ALE-47**	SADL	3A, 4	HEIM					x		
F-16C 85-1551	30	GE100	Small	SCU-7	APG-68V1	ALQ-213/ALE-47**	SADL	3A, 4	HEIM					x		
F-16C 85-1560	30	GE100	Small	SCU-7	APG-68V1	ALQ-213/ALE-47**	SADL	3A, 4	ENERTEC					x		
F-16D 86-0047	30	GE100	Big	SCU-7	APG-68V1	ALQ-213/ALE-47**	SADL	3A, 4	ENERTEC					x		
F-16D 87-0370	30	GE100	Big	SCU-7	APG-68V1	ALQ-213/ALE-47**	SADL	3A, 4	ENERTEC					x		
F-16D 85-1572	30	GE100	Small	SCU-7	APG-68V1	ALQ-213/ALE-47**	SADL	3A, 4	ENERTEC		H			x		
F-16D 89-2169	40	GE100	Big	M6.2+	APG-68V1	ALE-47	x	3A, 4, S	ENERTEC	x				x		
F-16D 91-0464	50	GE129	Big	M7.1+/FTR7	APG-68V5	ALE-47	x	"A" IFF, 3A, 4, S	2x8mm	x						
F-16D 87-0386	30	GE100	Big	SCU-7	APG-68V1	ALQ-213/ALE-47**	SADL	3A, 4	2x8mm		H			x		
F-16D 88-0174	40			M6.2+										x		
F-16D 89-2176	40			M6.2+										x		
F-16A 80-0584	15	PW220		M6.5.2	APG-66		x	3A, 4, S	HEIM (Note 8)	x		Aydin-Vector	G2/GCR	x		
F-16A 80-3666	15	PW220		S1.1 FTR4	APG-66		x	3A, 4	HEIM (Note 11)	x		Aydin-Vector	G2/GCR	x		
F-16C 83-1120	30	GE100		SCU-9 C-MoRE	APG-68(v)10		SADL	3A, 4	HEIM (Note 10)			TTC	GAINR-2	x		
F-16C 87-0352	40	GE100		M7.1+/FTR7B	APG-68(v)1	ALE-47	x	3A, 4, S (transponder only), S	HEIM (Note 8)	x		Aydin-Vector	GAINR-2	x		
F-16D 87-0392	40/50	GE129		RTT 2.3B Rev 3	APG-83	ALE-47	x	3A, 4, 5,S	HEIM (Note 8)	x		Aydin-Vector	GAINR-2	x		
F-16C 88-0445	42/50	PW220		M7.1+/FTR7 B	APG-68(v)5	ALE-47	x	3A, 4, 5, S	HEIM (Note 8)	x		Aydin-Vector	G-lite	x	x	
F-16C 88-0456	42/50	PW220		RTT 2.3B Rev 1	APG-83	ALE-47	x	3A, 4, S	HEIM (Note 8)	x		Aydin-Vector	GAINR-2	x		
F-16D 90-0835	50	GE129		M7.1 FTR 7	APG-68(v)5	ALE-47	x	3A, 4, 5, S	HEIM (Note 9)	x		TTC	GAINR-2 G-lite C2R****	x		
F-16D 90-0840	50	GE129		M7.1 FTR 7B	APG-68(v)5	ALE-47	x	3A, 4, 5, S	HEIM (Note 8)	x		TTC	G2/GCR	x	x	x
F-16C 91-0383	50	GE129		M7.1+/FTR7B	APG-68(v)5	ALE-47	x	3A, 4, 5, S	HEIM (Note 9)	x		TTC	G-lite C2R	x		x

Figure 8: F16 Characteristics by Tail

those that are suitable for any given mission must be identified and enumerated. Understanding the range of possibilities will be important for an optimization model.

2.6.3 Configurations.

Beyond these semi-permanent characteristics of a tail, each F16 sortie flies in a temporary configuration. The configuration desired must be discerned to some level of fidelity high enough to be useful and low enough to be practical. The configurations are changed on a regular basis, and while they cannot be changed intra-day, a subset of combinations can be changed between days. The task is to honor these rules while minimizing the changes in order to allow maintenance time for other important activities. There are more than 10,000 possible configurations if considering every specific combination on the various positions of an aircraft as shown in Figure 9.

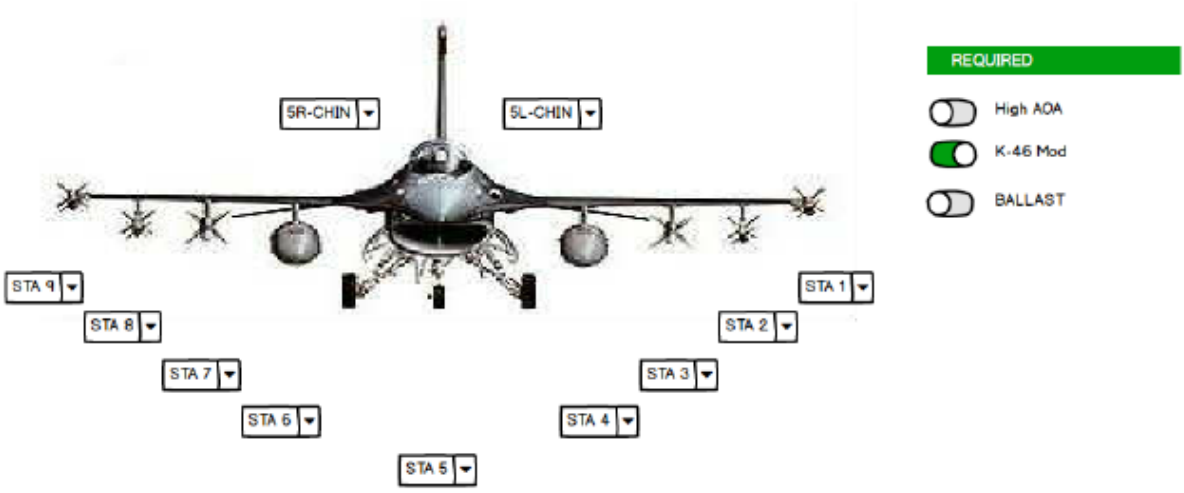

Figure 9: Positions on an F16 Representing the Combinations of Possible Configurations

2.6.4 Refueling.

Another significant Iron oriented resource attribute is refueling requests. This can be a complex resource negotiation between the Tanker Airlift Command Center (TACC) and the Test Wing. TACC wants to fly missions, but the 412^{th} Mission must fund the Tanker Mission. Only certain programs have the funds to request and

launch a Tanker. Once in the air, the Tanker is open to any mission for refueling subject to available fuel and time.

These resource attributes all have unique constraints and parameters that must be designed, and brought together in one solution approach.

2.6.5 Planning Time.

This is not a strategic planning problem where planners are comfortable running models for days before analyzing the answer. This is also not a flight line operations problem where the answer is needed in a matter of minutes or even seconds, requiring a quick solution. There will be formal or informal meetings between operations and maintenance to clarify priorities and "sign off" on the initial schedule. This is ahead of releasing the schedule to the Test Wing at large so the scheduling process can begin with more clarity and less churn than before. Having a model that takes several hours to solve is too long for a one day planning process as this would mean a one run approach, with not much time to change parameters and rerun if necessary.

2.7 Keeping Score

Defining an objective is a key element of optimizing any schedule. The scheduling process has implicit objectives that must be made explicit to formulate an optimization. The quality of a schedule can be a subjective measurement, but there are certain objective attributes:

- High priority missions on the schedule over lower priority missions. There is an unpublished, yet understood priority rank each week, at least as it concerns the top 2 spots. This can be expressed at the unit level or at the program level. Having an explicit scoring system that all understand is important to the transparency and thus quality of the schedule.

- High capacity utilization with high quality missions. One bottleneck resource tends to be the maintenance capacity for the F16s. They will have a capacity for some number of sorties during the week given the pre-sortie and post-sortie maintenance requirements. A schedule that utilizes that capacity with high quality missions is considered good.

- A good schedule will find the balance between throughput and time preferences. High priority missions generally get the first shot at preferred time windows. In a deconfliction meeting, the chair will simply ask the high priority team their latitude for change. The answers generally reflect a very sincere attempt to be a team player, but often the dependencies for that unit dictate they cannot move "left" or "right" very far to accommodate another mission. The explicit question in a modeling environment is what is the quantifiable trade-off, in terms of time slots, between moving a high priority mission to get a lower priority mission on the schedule? This latitude should be understood in order to avoid the back and forth that hurts the efficiency in the process.

- Making configuration or phase schedule changes costs time and money. This can be very difficult to quantify and is the source of intense but cordial debate in the Test Wing. Having a method to quantify the value of flying a mission that disrupts a phase schedule or requires an extra configuration change is important to weigh against the cost of making that change.

- A good schedule will acknowledge external commitments. If a mission schedules an external range managed by the Navy (e.g. Sea Test Range) there is acknowledgement of this fact. Likewise, if a Tanker has been called in and paid for, it is best to utilize it fully. While it is clear this cost matters, these dollars are hard to quantify and thus a monetary objective, or even portion of an objective

function is difficult.

2.8 Model Execution Considerations

Figure 10 is a high level view of the scheduling process.

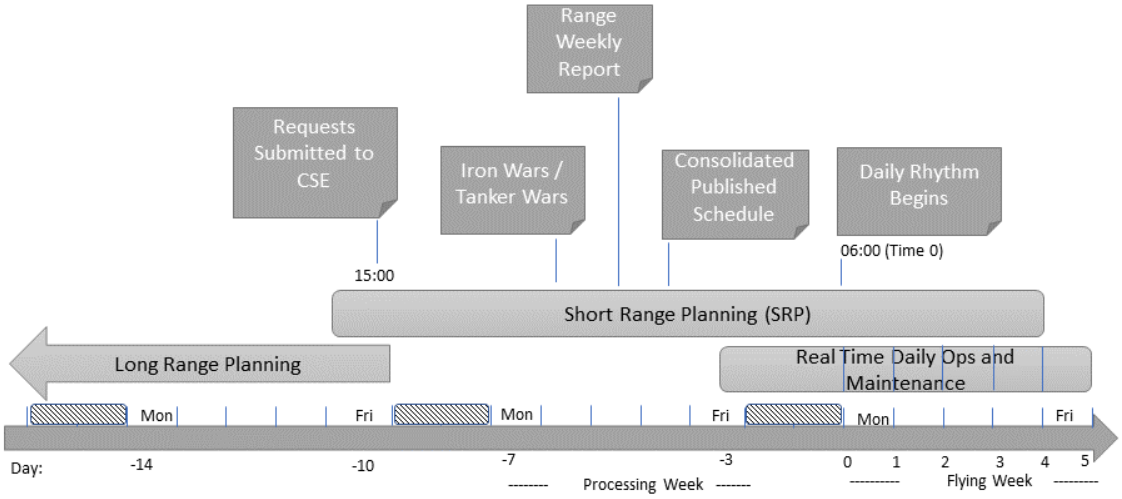

Figure 10: High Level Scheduling Process

2.8.1 Robustness.

Some missions are planned to be canceled if the preceding mission meets its goals. These are called "Backup" Missions. They do not have a designation in CSE, but are generally known throughout the scheduling community. To combat these potential cancellations and other high risk missions, the concept of a "standby" mission is used. The standby is ready to step in and use the resources released by the canceled backup mission. A scheduling approach should identify these backup missions and further identify appropriate standby missions to have built in robustness against a disruption caused by a canceled mission that can be predicted.

2.8.2 Tactical Execution Considerations.

There is considerable discussion about an execution mode model that would be employed in the event of a flying week disruption. This might be a jet breakdown precipitating a ripple effect of change to the schedule. Having the ability to solve for only the affected missions with a rule set that minimizes the changes is a requirement of the maintenance team. While this is a similar problem to creating an initial schedule, the key differences must be captured and clarified for model design purposes. The IP approach employed by the Koepke team is designed to aid planners in finding a feasible schedule with inevitable schedule changes in the Air Force Channel Route Network [6]. If a planning answer has been submitted, one challenge is to only update the missions directly impacted by the disruption, and know which missions should be "locked" down. Beyond using constraints to lock certain missions, the objective function bias should be to impact as few missions as possible with the reshuffling that will occur if a model is rerun. The second challenge is collecting the data from CSE, the execution system, in order to discern the current state of the missions.

2.9 Conclusion to Chapter

The task is to blend the research on scheduling problems and typical solution constructs with what is known about the Developmental Test Scheduling Problem structure to design the best solution. The formulation of an Integer Program in light of the specific problem parameters is a good fit. The IP is a platform to test formulations for each aspect of the problem, and research solution techniques that best fit the formulation. From the data analysis, the bounds for a good solution are roughly established. A quantifiable and transparent scoring mechanism representing an approach to navigate complex trade-offs is critical. The focus is on building a planning model to get a good initial schedule while acknowledging the need to leverage

the solution approach for a more tactical environment as a future project. Modeling the Developmental Test Scheduling Problem will be fraught with challenges and missteps, but worth the effort. "In a system of this magnitude, intuition and experience do not always yield the optimal use of resources" [8].

III. Methodology

3.1 Introduction

The methodology for the Developmental Test Scheduling Problem is to formulate an Integer Program Ecosystem. An ecosystem in the sense that significant preprocessing and post-processing will ensure the Integer Program only works on relevant decisions at the right level of fidelity. The solution is thus broken into three parts all working together. Figure 11 is a high level depiction of the process. This chapter will

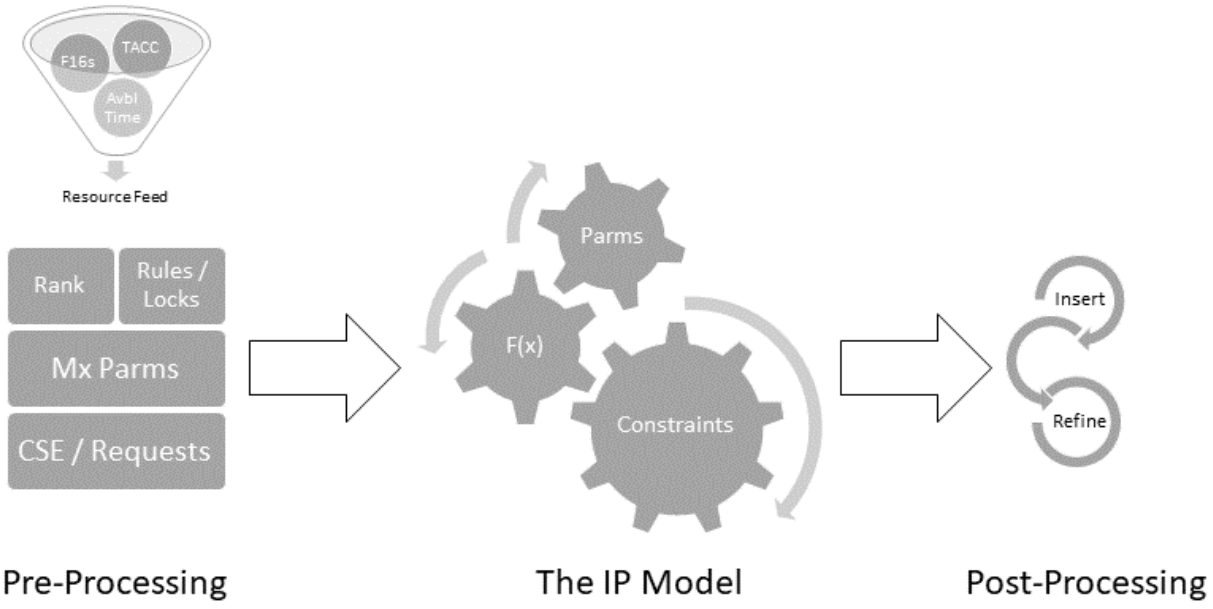

Figure 11: The Model Ecosystem

discuss the problem characteristics that make an Integer Program appropriate then layout a high level design and formulation for a solution approach that leverages data processing around the Integer Program.

3.2 An Integer Programming Approach

Chapter II laid the foundation for the applicability and challenges of an Integer Programming approach to scheduling problems in general, and detailed specific characteristics of the problem. An IP formulation is applicable here for several reasons:

- This is an operational planning model. This allows adequate time to solve if the size of the problem is managed well.
- An IP Formulation is relevant to many solution procedures. If the ultimate solution methodology is a dynamic program or heuristic solve, an IP formulation is a good base [4].
- The length of the planning horizon and the number of missions to schedule is manageable. With a one week horizon for detailed planning, and 300-400 missions in a batched environment, the problem suitable for an IP.
- The level of time fidelity combined with the time horizon makes for manageable time slots. The section on time windows will discuss the level of abstraction to the execution system in terms of time slots. The current design employs 30 minute time intervals across a 12 hour day representing 276 start periods across a planning week.
- Feasible solutions can be difficult to find in a complex environment. Finding a starting solution for a heuristic search procedure to start performing swaps or otherwise search the solution space is not trivial. A sequential heuristic is difficult to implement as it is hard to anchor on a feasible search space in which to optimize. The global nature of the IP algorithm makes it suited for the task. Figure 12 is a conceptual diagram of how a heuristic would start, which is also the manual process, versus an Integer Programming approach.

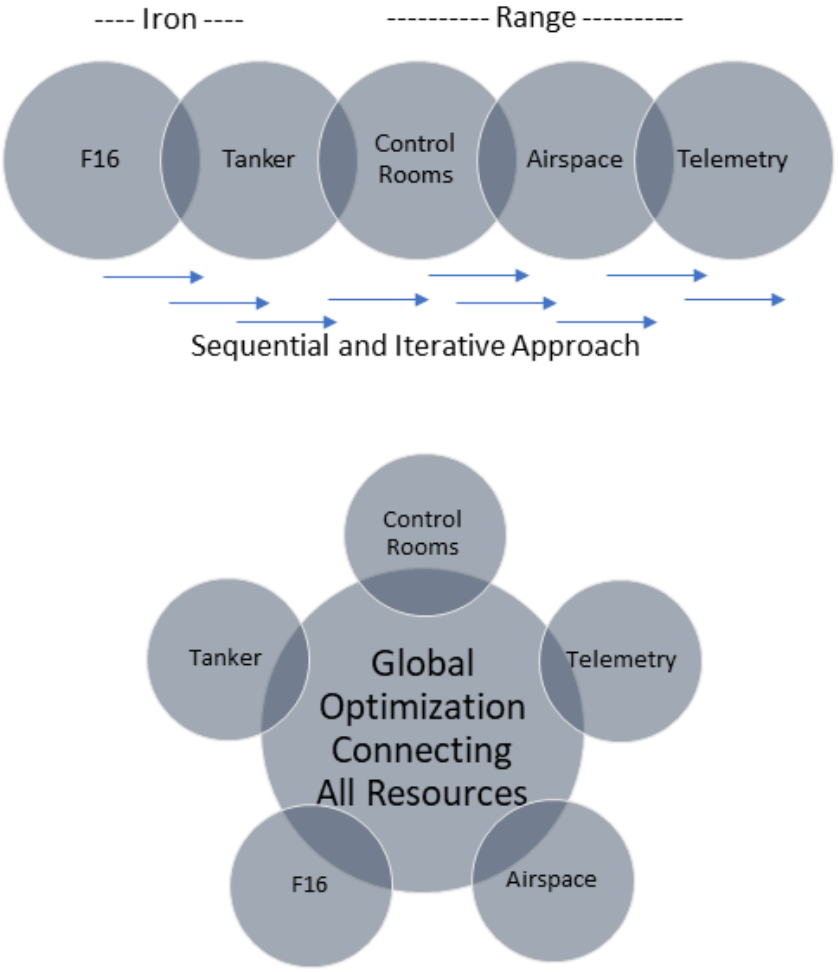

Figure 12: Global Optimization of an Integer Program

An IP is a flexible construct where the data elements and constraints can be expressed intuitively and then abstracted to balance fidelity with solution considerations. This can then be solved with traditional IP methods or other creative approaches.

3.3 Capturing Necessary Data Elements

The prototype work will prove or disprove that the investment of time, cost and effort required to properly capture this data will yield much greater returns of time, transparency and optimized results when generating the schedule. A robust user

interface will be required to acquire the necessary data elements in the most intuitive manner to the schedulers.

3.3.1 Time Windows.

Every time period adds multiple variables to the problem. Given the constraints on the problem those added time periods for a Mission might not yield the potential for a better answer. In some cases adding an extra time slot provides a key degree of freedom that yields a better answer. Discerning the difference takes trial and error.

In the Edwards instance of CSE, there is one field representing the desired start time of a mission. The date field is better represented by two fields for the day and time. Period 1 is presently defined as 06:00 with period 24 starting at 17:30. Indexes exist for periods outside the flying window to accommodate maintenance periods and even out of window flying. For a standard mission, the data presented to the model will not allow for a mission that starts before 06:00 or ends after 18:00. With this flexibility the requests can be shaped to be in line with squadron needs. In the case of TPS they are less concerned about the day they fly Training Missions, but cannot fly past 13:00 to prepare for the academic day that starts at 14:00. The process creates variables each day for morning periods that have the missions down by 13:00. In other cases, the day is constrained and the time of day is flexible. If a mission is scheduled which does not require any centralized resources, the pre-processing routines simply pass the model the requested time period with no other options.

3.3.2 F16 Characteristics.

In the current process, the scheduler requests a specific (F16D-0370) or generic (F16D-REQF) tail number. They use the remarks field to describe the aircraft characteristics needed for the mission. This project designs an interface, starting with

Figure 8, which will allow users to select aircraft characteristics to determine the specific tails that meet their requirements. This gives the model several possibilities upon which to optimize.

3.3.3 Configurations.

The configuration scheme is similar to F16 characteristics in that roughly 10,000 configurations are synthesized to a series of codes. The codes represent a family of configurations similar enough such that maintenance can easily switch between them. The squadron scheduler maps the configuration they require from Figure 9 into one of the codes. In the prototype phase of this project, the rule is that one configuration is requested per sortie. There is currently not a structured data field for the configurations. An analysis of the Remarks column for each F16 sortie determines the most likely configuration and codes it per the scheme developed. Part of the verification and validation testing will be to change these configurations and determine the impact on the answer.

3.4 Explicit Enumeration Methodology

Even with the advantageous problem characteristics for this operational problem, it would be too large to solve with implicit set creation. If 300 Missions are multiplied by 115 time periods and 29 unique tail numbers (not counting the non F16 tails) indiscriminately, roughly 1 million combinations, or variables, are created. Commercial and open source solver software accommodates explicit enumeration where the set members are defined for each set combination such that not all combinations of the primary sets are viable. This is critical in a discrete time period environment where not all periods are possible for each mission. In order to leverage this capability, tight control of the inputs is required. If the sets are represented by a series of matrices,

or a multi-dimensional cube, the vast majority of entries are zero and could never be one based on the constraints of the problem. In explicit enumeration the input is structured as a relational database where the problem is fed to the solver in rows, and the indexes of each row represent the possible combinations. The set creation becomes integral to the formulation. Figure 13 is an example of defining the possibilities for for a specific mission: This Mission has 1 sortie, but 29 Tail possibilities

MISSION	SORTIE	TAIL	DAY	PERIOD
325	1	F16D-0174	2	5
325	1	F16D-1464	2	5
325	1	F16D-2169	2	5
325	1	F16D-2176	2	5
325	1	F16D-0174	2	16
325	1	F16D-1464	2	16
325	1	F16D-2169	2	16
325	1	F16D-2176	2	16
325	1	F16D-1464	3	5
325	1	F16D-2169	3	5
325	1	F16D-2176	3	5
325	1	F16D-1464	3	16
325	1	F16D-2169	3	16
325	1	F16D-2176	3	16
325	1	F16D-0174	4	5
325	1	F16D-1464	4	5
325	1	F16D-2169	4	5
325	1	F16D-2176	4	5
325	1	F16D-0174	4	16
325	1	F16D-1464	4	16
325	1	F16D-2169	4	16
325	1	F16D-2176	4	16

Figure 13: Explicit Enumeration Data

and 115 Time periods or 3,335 entries. In an explicit enumeration environment there are 22 combinations representing only the possible tails and times as defined by the

request and predetermined parameters. This is less than 1% of the size an implicit formulation would produce.

The software compresses the problem even further by only considering the nonzero coefficients. When looking at the "A" matrix created of variables and constraints the combinations of constraints and variables would have 11,421 x 21,707 entries to consider or 248 MM possibilities. As it is, the problem is represented to the computer as 60,929 entries or .025% of the potential problem size.


```
"Problem": [
    {
      "Lower bound": -607.0,
      "Name": "tmp52i8_yjp.pyomo",
      "Number of constraints": 11421,
      "Number of nonzeros": 60929,
      "Number of objectives": 1,
      "Number of variables": 21707,
      "Sense": "minimize",
      "Upper bound": -607.0
    }
```


Figure 14: Compressing The Problem Size (from JSON output file in Pyomo)

Consider the following setting constraint:

$$\sum_{S}\sum_{T} x_{m,s,t,d,p} \;\leq\; xmdp_{m,d,p} * BigM \quad \forall\, m,d,p$$

An equivalent representation is below with $SCHED$ defined as the reduced Cartesian combination of the Sets M,S,T,D,P:

$$\sum_{SCHED} x_{m0,s0,t0,d0,p0} \;\leq\; xmdp_{m1,d1,p1} * BigM \quad \forall\, m1,d1,p1$$

$$where\, m0 = m1,\; d0 = d1,\; p0 = p1$$

The implementation of this constraint in a modeling language follows the latter syn-

tax.

```
#Pyomo
def Link_x_xmiss(model,m1,d1,p1):
        return (model.xmiss[m1,d1,p1] * 100) -
        sum(model.x[m,s,d,p,t] for
        (m,s,d,p,t) if
        m==m1 and d==d1 and p==p1) <= 0
model.Set_xmiss =
Constraint(model.Miss_Day_Per, rule = Link_x_xmiss)
```

```
!Lingo
@FOR(XMISS_DAY_PER(m1,d1,p1):[x_xmiss]
                @SUM(XSCHEDULE(m,s,t,d,p)|
        m #eq# m1 #and# d #eq# d1 #and# p #eq# p1:
                        x(m,s,t,d,p)) <=
                xmiss(m1,d1,p1) * BigM);
```

3.5 Solution Approaches

3.5.1 Generating Time Windows.

The two considerations for creating time periods include when to create a time slot and how many time slots to create. The time windows which ultimately get created for a mission will be a combination of the desires of the CTF (as expressed through a robust user interface) and the parameters defined by the Operations Group. This requires "bang for the buck" tuning of the model. The "bang" being the objective function value and throughput realized from more degrees of freedom, and the "buck"

being the solve time incurred by adding variables. The simple approach of putting a buffer of time periods and days around the desired time did not work well. As Figure 7 indicates, the requests are bunched in the morning and thus adding a time buffer around these missions did not alleviate the competition for resources in the morning periods. Even if an F16 flies for one period (half an hour) it must be in maintenance for 6 periods so moving another mission a couple periods to the right does not give it access to a jet released from a short duration mission.

An alternative approach is to create an offset so that each mission has a morning and afternoon possibility. This allows the model to put each mission into a wave during the day where there is a better probability that resources will be free. As discussed in the next chapter this yields better results, but still has shortfalls. There were cases where moving the mission later one or two periods would have allowed it to be scheduled, but just having the shift to the afternoon was inadequate. That leads to a hybrid approach of putting a small buffer around a morning and afternoon time period, which yields the best results, but adds more variables resulting in longer run times.

Moving missions to an afternoon takeoff would require consent. This is a function of the Long Range Planning Team. One possible approach is to capture at the squadron level parameters that indicate the bounds the unit is willing to accept. These would include times for early departure, late departure, and late landing. The code to create the offsets can honor those times and make adjustments to find a proper alternate time period. For example the afternoon time period would be $LateLand_u - Duration_m$ if the normal offset would throw the mission outside it's late landing time. The u in this case is unit level and the m mission level. The difference in Days and Periods from the preferred time are created as parameters on the appropriate composite sets which is detailed in Section 3.7.

3.5.2 Post Processing Approach.

The two basic outcomes from solving the Integer Program are that it solves to optimality within a specified gap to the Linear Program bound, or that it stops with the best answer at the time limit. In either case, there will be unscheduled missions to post-process.

With the right visualization tool it is easy to discern where there is unused capacity to potentially match with an unscheduled Mission. These are the time slots for a given F16 on a particular day where a "Go" is feasible, but not employed. Figure 15 is output, exported to MS Excel, of the F16 Missions for Tuesday of the flying week. The solid gray areas indicate the flying periods for a Mission using the Tail

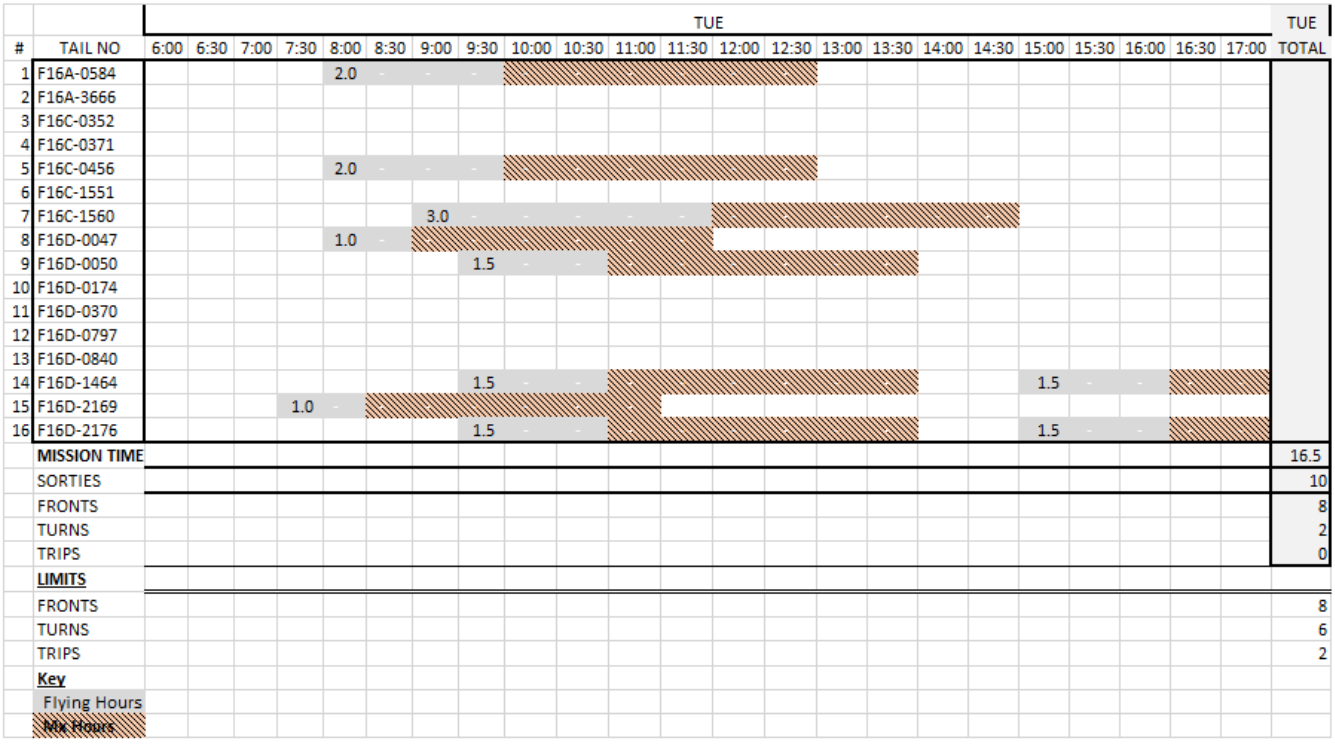

Figure 15: Example Day of F16 Missions Schedule

specified. The hatched areas represent the 3 hours of maintenance required before that jet can fly again on Tuesday. Tail F16D-0370 and others are not flying at all on

Tuesday. These times are not open however as the "Fronts" limit has been hit for the day. Fronts is the term used to indicate the number of aircraft that Maintenance will support for first flight, or "Go", on any particular day. The current rules at Edwards are such that it does not matter what time of day the F16 takes off. A Front is defined by the first time it flies. This leaves four "Turns" and two "Trips" as available capacity. A Trip (3rd Go) will not work for any of the jets unless Missions are moved to the 06:00 or 06:30 time slots. This tends to be highly undesirable and these time slots often go unused. Four Turns of unused capacity exist for appropriate missions to spread amongst 6 aircraft (F16A-0584, F16C-0456, F16C-1560, F16D-0047, F16D-0050, F16D-2169). This information allows the central scheduler to use their knowledge of the flying units and the current operating conditions to determine the next course of action. While building a schedule from scratch is difficult, for even a feasible answer, refining a schedule that is over 85% built is a plausible exercise for a scheduler and one they are very good at. Implementing very nuanced rules with knowledge of the players involved and this week's special circumstances is best left to the experts.

The next step is to identify, in a report format, the missions that can use the open jets. There will typically be zero to only a very few missions that will fit, and the question is whether or not they can move their requested time window to take advantage of this unused capacity. This post-processing exercise is an effective alternative to presenting all the time windows to the model. The Results Chapter will show an example of this post-processing approach.

3.6 Implementation Design

The implementation is an ecosystem of modules working together to be a functional Scheduling Support System for the Test Wing. The pieces of the ecosystem

represented in Figure 10 are:

1. A Microsoft Access Database. This serves as the user interface to the prototype model. The database and VBA code handle data storage, set creation, and post-processing.

2. A Pyomo MIP compiler. Pyomo was developed by Sandia Labs as a bolt-on to the popular open source programming platform Python. The closest commercial counterpart is AMPL. A developmental version of the model was written in Lingo. First using MS Excel for the toy problem, and then linking to MS Access for larger instances with more fidelity. Pyomo was the choice for implementation given its portability and open source architecture.

3. An IP Solver. The open source consortium COIN-OR is designing and coding solvers. The MIP version is called CBC which employs a branch and cut algorithm similar to commercial solvers such as CPLEX or Gurobi. Part of the future research is to compare the performance of the CBC Solver with a trial version of CPLEX.

4. A Windows Batch Script. In the Pyomo implementation reading the data from a file is more efficient than having Pyomo establish a connection to the database. The script is required to read the database and build a file with the appropriate headers and subsequent data for both Sets and Parameters. This written to a file that is subsequently read by the abstract Pyomo model looking for data.

5. A JavaScript Web Application for output. Viewing the schedule in a Gantt Chart format is conducive to quickly understanding how well the week was utilized and where the bottlenecks are. A rudimentary version of this functionality exists in Excel and provided a good tool for viewing answers while the web application was being developed.

A high level illustration of the prototype technology stack is shown in Figure 17.

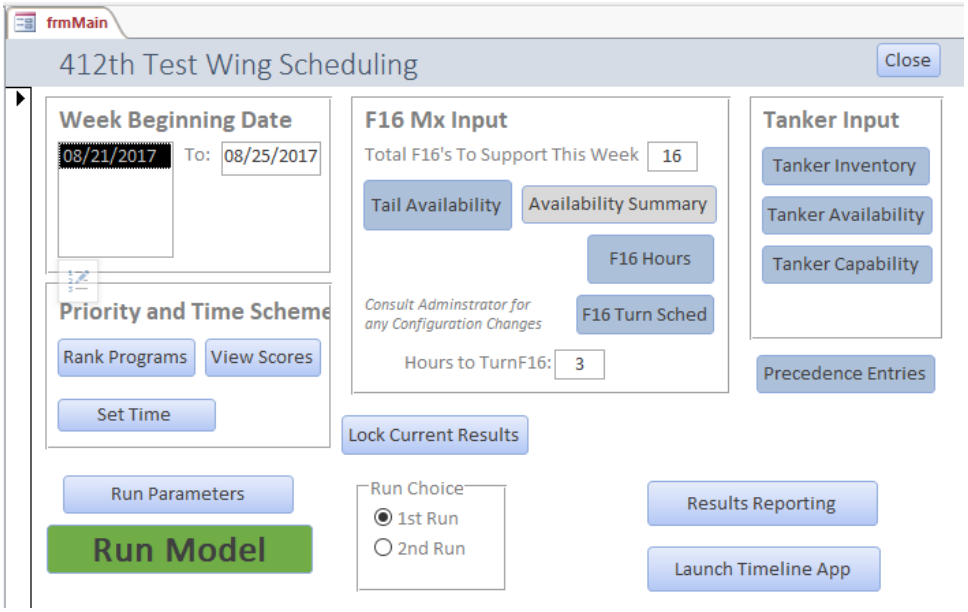

Figure 16: User Main Menu Screen for Prototype

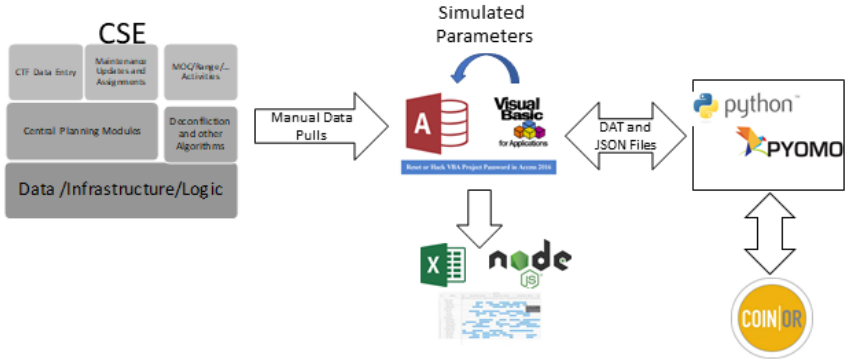

Figure 17: Prototype Implementation Technology Stack

3.6.1 Implementation of Another Planning Tool.

Capturing and clarifying the alternatives is the first step in determining how to implement a planning ecosystem. One approach is for CSE to remain the data capture point with a bolt-on planning tool. The alternative is to create a planning tool that captures the input from the squadron schedulers and submits the plan to CSE in the

form of requests. In the former approach, there will be either CSE modifications or off-line processes which capture the structured data. Figures 18 and 19 are schematics of the two approaches.

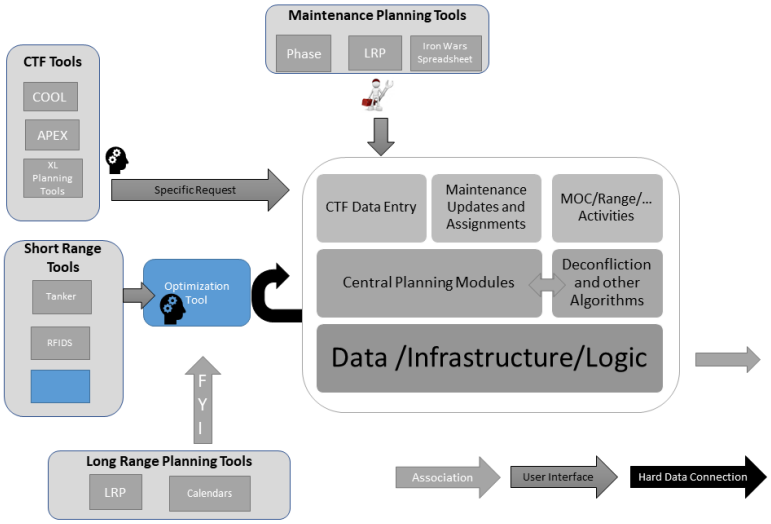

Figure 18: Planning Tool Bolted on to CSE

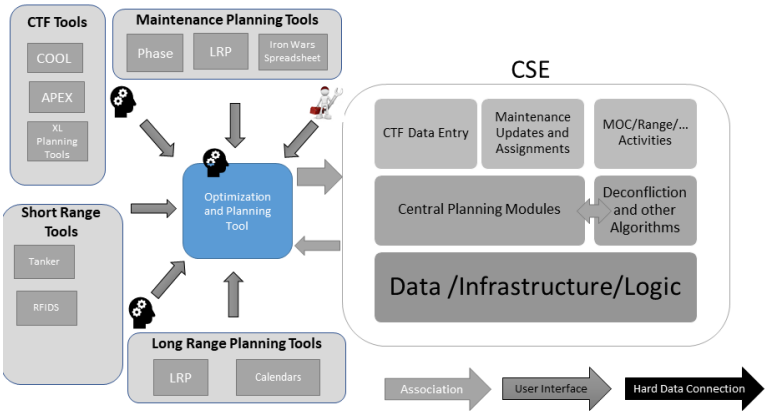

Figure 19: An External Planning Tool

In the end, a hybrid of these approaches may be implemented. With exposure to this solution methodology by the Edwards and CSE teams, an implementation strategy will emerge.

3.7 Iron Model IP Formulation

The IP Formulation, for clarity and development purposes, was broken into three sections. The core is described immediately below, followed by extensions to the model for Tanker refueling, and F16 Configurations that maintenance must implement.

3.7.1 Core Model Formulation.

The core formulation will set a base for the model, establishing the key indexes and decision variables upon which all the functionality will be built. The core model handles the physical mission and flying rules (e.g. A Mission can only go once, and a Tail can only fly with one Sortie at a time), as well as the maintenance rules. It also enforces prescribed precedence relationships, and any locking of Missions the user indicates prior to running the model.

The primary decision variable is $x_{m,s,t,d,p}$, upon which all other core variables will be linked. The x variable represents the starting period of a Mission, Sortie, and Tail combination with the index p. Flying, to include the take-off period is tracked with $v_{m,s,t,d,pv}$ that syncs with $x_{m,s,t,d,p}$. The periods tracked by v utilize the pv index to indicate additional periods of activity. The indexes are equal at the Mission start. While the Mission starts at p, the Sorties can start later as indicated by and offset parameter.

Core Sets.

Basic Functions for Explicitly Enumerating Derived Sets.

$$SORT(m,s) = \begin{cases} 1 & \text{If Sortie s is in Mission m} \\ 0 & \text{Otherwise} \end{cases}$$

Table 1: Iron Model Core Basic Sets

Set and Index	**Description**	**Example**
$M : m \in M$	All Missions m in Planning Horizon	Ops Number = 0325
$S : s \in S$	All Sorties s in Mission m	0325-1 or just 1
$T : t \in T$	All Tails t	F35A-0350
$D : d \in D$	All Planning Days d	1 through 5
$P : p, pv \in P$	All Planning Periods p in Day d	1 through 23
$F \subseteq T : f \in F$	F16 Tails f as Subset of Tails t	F16D-0074
$Q \subseteq M : q \in Q$	Precedent Missions q as Subset of m	8642
$L \subseteq M : l \in L$	Locked Mission (Must Go) for Planning Horizon	0325

$$GO(m,d,p) = \begin{cases} 1 & \text{If Mission m can start on Day d and Period p} \\ 0 & \text{Otherwise} \end{cases}$$

$$GODAY(m,d) = \begin{cases} 1 & \text{If Mission m can start on Day d} \\ 0 & \text{Otherwise} \end{cases}$$

$$GOPER(m,d) = \begin{cases} 1 & \text{If Mission m can start on Period P} \\ 0 & \text{Otherwise} \end{cases}$$

$$AC(m,s,t) = \begin{cases} 1 & \text{If Tail t is eligible to Fly with Sortie s on Mission m} \\ 0 & \text{Otherwise} \end{cases}$$

$$UP(t,d) = \begin{cases} 1 & \text{If Tail t can Go Day d} \\ 0 & \text{Otherwise} \end{cases}$$

$$SFLY(m,s,d,pv) = \begin{cases} 1 & \text{If Sortie m,s can fly on Day d and Period pv} \\ 0 & \text{Otherwise} \end{cases}$$

$$TFLY(t,d,pv) = \begin{cases} 1 & \text{If Tail t can Fly on Day d and Period pv} \\ 0 & \text{Otherwise} \end{cases}$$

Derived Sets.

Table 2: Iron Model Derived Tuple Sets

Set	Description	Definition
SCHED	Starting Schedule Set	$\{(m,s,t,d,p)\|SORT \cdot AC \cdot GO \cdot UP = 1\}$
VSCHED	Flying Schedule Set	$\{(m,s,t,d,pv)\|SORT \cdot AC \cdot SFLY \cdot TFLY = 1\}$
MSDP	Mission Sortie Day Period	$\{(m,s,d,p)\|SORT \cdot GO = 1\}$
MSP	Mission Sortie Period	$\{(m,s,p)\|SORT \cdot GOPER = 1\}$
MDP	Mission Day Period	$\{(m,d,p)\|GO = 1\}$
MD	Mission Day	$\{(m,d)\|GODAY = 1\}$
MSF	F16 Mission Sortie	$\{(m,s,f)\|SORT \cdot AC = 1\}$
MS	Mission Sortie	$\{(m,s)\|SORT = 1\}$
TDP	Tail Day Period for Flying	$\{(t,d,pv)\|TFLY = 1\}$
FDP	F16 Tail Day Period for Flying	$\{(f,d,pv)\|TFLY = 1\}$
FD	F16 Tail Up for the Day	$\{(f,d)\|UP = 1\}$
PREC	Precedent Relationship at Day Level	$\{(q1,q2) \subseteq Q \times Q\}$
LSCHED	Locked by User at SCHED level	$\{(l,s,t,d,p) \subseteq SCHED\}$
LDP	Locked Mission at Day/Period	$\{(l,d,p) \subseteq MDP\}$
LD	Locked Mission on a Day	$\{(l,d) \subseteq MD\}$
τ_m	Reduced Set of Times (d,p) For M	$\{(d,p)\|GO = 1\}$
$P_{m,d}$	Red. Set Periods for M on D	$\{(p)\|GOPER = 1\}$
$T_{m,s,d,p}$	Red. Set of Tails for m,s at d,p	$\{(t)\|SORT \cdot AC \cdot GO = 1\}$
F_d	Red. Set of F Flying on Day d	$\{(f)\|UP = 1\}$

Parameters and Constants.

Table 3: Iron Model Core Parameters

Parameter	Description	Type
$Priority_m$	Mission Priority 1 -5 w/ 5 best for Max Obj Fcn	INT
$SortCount_m$	Number of Sorties in Mission m	INT
$F16SortCount_m$	Number of F16 Sorties in Mission m	INT
$PrefDay_m$	Preferred Day for Mission m	INT
$PrefPd_m$	Preferred Period for Mission m	INT
$FxSort_s$	Pre-Calc Sortie Multiplier for Objective Function	REAL
$Dur_{m,s}$	Duration of Sortie Converted to Periods	INT
$Offset_{m,s}$	Offset from Mission Start of Sortie	INT
$TurnPers_{m,s}$	Turn Periods Rqd. Stored at MDS level	INT
$TypeAcft_{m,s}$	Type of Acft Requested	STRING
$DayDiff_{m,d}$	Calc Difference Between PrefDay and SchedDay	INT
$PerDiff_{m,d,s}$	Calc Difference Between PrefPd and SchedPd	INT
$MinPds_f$	Minimum Periods Jet Should Fly in a Week	INT
$MaxPds_f$	Maximum Periods Jet Should Fly in a Week	INT
$Fronts_d$	Limit on First GO's of any Day for F16s	INT
$Turns_d$	Limit on Second GO's of any F16/Day for F16s	INT
$Trips_d$	Limit on Third GO's of any F16/Day for F16s	INT
$DayNum_d$	Integer for the Indexed Day d	INT
$PerNum_p$	Integer for Indexed Period	INT
$MxWkF16Avbl$	Global F16s Available to Schedule for Planning Horizon	INT
$WGHT\_PRI$	Constant for Weighting the Mission Priority in Obj Fcn	REAL
$WGHT\_DAY$	Constant for Weighting the days from preferred in Obj Fcn	REAL
$WGHT\_PER$	Constant for Weighting the periods from preferred in Obj Fcn	REAL

Variable Declarations.

Table 4: Iron Model Variables

Variable	Description
$x_{m,s,t,d,p} \in \{0,1\}$	Primary Decision Variable. Msn/Sortie m,s with Tail t Starting at time d,p
$v_{m,s,t,d,pv} \in \{0,1\}$	Msn/Sortie m,s with Tail t Flying at time d,pv
$xmiss_{m,d,p} \in \{0,1\}$	Mission Scheduled at time d,p
$xsort_{m,s,d,p} \in \{0,1\}$	Sortie Scheduled
$xm_m \in \{0,1\}$	Mission was Scheduled
$xwt_f \in \{0,1\}$	F16 f was active during Week
$xfd_{f,d} \in \{0,1\}$	F16 f starting on Day d
$xfdp_{f,d,p} \in \{0,1\}$	F16 f Starting Day d Per p
$xfd2_{f,d} \in \{0,1\}$	F16 f Starting twice on Day d
$xfd3_{f,d} \in \{0,1\}$	F16 f Starting 3x on Day d
$xday_q \in \mathbb{Z}^+$	Captures the Go Day of Prec Mission q
$xif_f \in \mathbb{R}^+$	Guard against impossible Min Hrs Param
$xim_l \in \mathbb{R}^+$	Guard against improper locking

Core Model.

There are several variants of the Objective Function (Eq 1) that can be used depending on the user's preference. The Mission Level with preference given to a prescribed start time is shown.

In the core formulation equations 2 - 5 link x to $xsort$, $xmiss$, & xm, and ensure the Sorties are congruous to their Mission. Equations 6 and 7 require only one Mission starting and flying at a time, and 10 - 19 represent the maintenance rules. Equations 8 - 10 set the xfd variables needed in 11 - 13. Equation 14 sets xwt from the global constraint 17. The rules are rounded out with equations 16 and 17 to enforce the minimum and maximum hours (converted to periods in the model) the maintenance team has indicated is appropriate to fly each aircraft. To ensure the minimum hours does not cause an infeasibility if its taken out of service or does not

meet the requirements for any of the Sorties, $xinfminhrs$ will catch the discrepancy. The precedence rules are captured in equations 18 and 19 with 18 setting the integer $qday_q$ and 21 using the relationship between $q's$. Finally, equations 20 - 23 represent the user prescribed locking of Missions at various levels from Schedule to Mission.

$$MAX \quad \sum_{(m,d,p)\in MDP} xmiss_{m,d,p} \times ((Priority_m * WGHT\_PRI) - (DayDiff_{m,d} * WGHT\_DAY) - (PerDiff_{m,d,p} * WGHT\_PER)) \quad (1)$$

Where the Constants:

$$WGHT\_PRI >> WGHT\_DAY > WGHT\_PER$$

SUBJECT TO:

$$x_{m,s,t,d,p} \times (Duration_{m,s} + TurnPers_{m,s}) - \sum_{VSCHED} v_{m,s,t,d,pv} = 0 \quad \forall\,(m,s,t,d,p) \in SCHED \quad Where:$$

$$p + Offset_{m,s} \leq pv \leq (p + Offset_{m,s} + Duration_{m,s} + TurnPers_{m,s} - 1) \quad (2)$$

$$\sum_{T_{m,s,d,p}} x_{m,s,t,d,p} = xsort_{m,s,d,p} \,\forall\,(m,s,d,p) \in MSDP \quad (3)$$

$$xsort_{m,s,d,p} = xmiss_{m,d,p} \qquad \forall (m,s,d,p) \in MSDP \quad (4)$$

$$\sum_{\tau_m} xmiss_{m,d,p} = xm_m \qquad \forall\, m \quad (5)$$

$$\sum_{\tau_m} xmiss_{m,d,p} \leq 1 \qquad \forall\, m \qquad (6)$$

$$\sum_{MS} v_{m,s,t,d,p} \leq 1 \qquad \forall\, (t,d,p) \in TDP \qquad (7)$$

$$\sum_{MSP} x_{m,s,t,d,p} \leq 3 \times xfd_{f,d} \qquad \forall\, (f,d) \in FD \qquad (8)$$

$$\sum_{MSP} x_{m,s,f,d,p} \leq 1 + xfd2_{f,d} \qquad \forall\, (f,d) \in FD \qquad (9)$$

$$\sum_{MSP} x_{m,s,f,d,p} \leq 2 + xfd3_{f,d} \qquad \forall\, (f,d) \in FD \qquad (10)$$

$$\sum_{F_d} xfd_{f,d} \leq Fronts_d \qquad \forall\, d \qquad (11)$$

$$\sum_{F_d} xfd2_{f,d} \leq Turns_d \qquad \forall\, d \qquad (12)$$

$$\sum_{F_d} xfd3_{f,d} \leq Trips_d \qquad \forall\, d \qquad (13)$$

$$xfd_{f,d} \leq xwt_f \qquad \forall\, (f,d) \in FD \qquad (14)$$

$$\sum_{f} xwt_f \leq MxWkF16Avbl \qquad (15)$$

$$\sum_{MSDP} x_{m,s,f,d,p} \times Dur_{m,s} \leq MaxPds_f \qquad \forall\, f \qquad (16)$$

$$\sum_{MSDP} x_{m,s,f,d,p} \times Dur_{m,s} \geq MinPds_f - xif_f \quad \forall\, f \qquad (17)$$

$$\sum_{\tau_q} xmiss_{q,d,p} \times DayNum_d = qday_q \qquad \forall\, q \qquad (18)$$

$$qday_{q1} \leq qday_{q2} - 1 + (2 * (1 - xm_{q2})) \qquad \forall\, (q1,q2) \in PREC \qquad (19)$$

$$x_{l,s,t,d,p} \geq 1 - xim_l \qquad \forall\, (l,s,t,d,p) \in LSCHED \qquad (20)$$

$$xmiss_{l,d,p} \geq 1 - xim_l \qquad \forall\, (l,d,p) \in LDP \qquad (21)$$

$$\sum_{P_{l,d}} xmiss_{l,d,p} \geq 1 - xim_l \qquad \forall\, (l,d) \in LD \qquad (22)$$

$$xm_l \geq 1 - xim_l \qquad \forall\, l \qquad (23)$$

3.7.2 Refueling Extension Formulation.

As an extension to the Core Iron Model, Refuel Missions require refueling by Tankers, which are considered an "Iron" Resource. The schedulers define refueling as a Mission attribute at the planning stage, leaving Sortie level detail to real time execution. CSE captures refueling at the Sortie level. This formulation currently reflects the Mission level philosophy, but might be adapted upon live testing. The level of fidelity is for the Tanker and the Mission to "Hookup" to dispense a prescribed amount of fuel at a particular time. This time should be appropriate in its offset from both the Mission and the Tanker starting. Meaning, a refuel should not be scheduled shortly after a Mission takes off or simultaneous to the Tanker launching. The location of the hookup will not be prescribed here but will be reserved for a later implementation when airspace is included in the formulation.

Certain Missions which belong to Programs that are capable of funding a Tanker launch, known as Business Effort (BE) Missions. The Missions that want to refuel, but cannot afford a tanker launch are known as Tanker of Opportunity (ToO) Missions. The model must consider the fuel level of the Tanker as it will be dispensing fuel to Refuel Missions and consuming its own. The Tanker must land with a prescribed amount of reserve fuel, which is taken out of the total fuel availability to establish a base fuel level for each day.

Both the Tanker and the Refuel Mission must be flying for a hookup to occur. The index p, as before, indicates the starting period for the Tanker or Mission. There is not a pv index as the flying is handled implicitly by Offset and Duration Parameters on the Tanker and Mission. The index ph is the potential period when the Tanker and Refuel Mission can connect based on a bounded p.

Table 5: Tanker Sets

Set and Index	Description	Example
$K : k \in K$	Tanker Inventory which become Tanker Missions	KC10A-SB01
$R \subseteq M : r \in R$	Refueling Missions r as Subset of Missions m	0409

Sets and Indexes.

Refuel Functions for Explicitly Enumerating Derived Sets.

$$KGO(k,d,p) = \begin{cases} 1 & \text{If Tanker k can start on Day d, Period p} \\ 0 & \text{Otherwise} \end{cases}$$

$$KGODAY(k,d) = \begin{cases} 1 & \text{If Tanker k can start on Day d} \\ 0 & \text{Otherwise} \end{cases}$$

$$KGOPER(k,p) = \begin{cases} 1 & \text{If Tanker k can start in Period p} \\ 0 & \text{Otherwise} \end{cases}$$

$$RGO(r,d,p) = \begin{cases} 1 & \text{If Refuel Mission r can start on Day d, Period p} \\ 0 & \text{Otherwise} \end{cases}$$

$$RGODAY(r,d) = \begin{cases} 1 & \text{If Refuel Mission r can start on Day d} \\ 0 & \text{Otherwise} \end{cases}$$

$$RGOPER(k,p) = \begin{cases} 1 & \text{If Refuel Mission r can start in Period p} \\ 0 & \text{Otherwise} \end{cases}$$

Derived Sets.

Table 6: Tanker Derived Tuple Sets

Set	Description	Notation
KDP	Tanker Launch	$\{(k,d,p)\|KGO = 1\}$
KD	Tanker Launch	$\{(k,d)\|KGODAY = 1\}$
KH	Tanker Hookup	$\{(k,d,p)\|KGO = 1\}$
K_r	Tanker that can Refuel Mission r	$\{(k)\|KGO \cdot RGO = 1\}$
RDP	Refuel Mission Go	$\{(r,d,p)\|RGO = 1\}$
RD	Refuel Mission Go	$\{(r,d)\|RGODAY = 1\}$
RP	Refuel Mission Starting	$\{(r,p)\|RGOPER = 1\}$
RH	Refuel Hookup	$\{(r,d,p)\|RGO = 1\}$
R_k	Refuel Mission r can hit Tanker k	$\{(r)\|RGO \cdot KGO = 1\}$
R2KDP	Possible Hookup Times	$\{(r,k,d,ph)\|RGO \cdot KGO = 1\}$
$P_{k,d}$	Reduced Set Periods for Tanker Msn k on Day d	$\{(p)\|KGOPER = 1\}$

Parameters.

$$RTypeMsn_r = \begin{cases} 1 & \text{If the Mission is a BE capable of financing Tanker launch} \\ 0 & \text{If Mission is designated ToO} \end{cases}$$

Table 7: Tanker Parameters

Parameter	Description	Type
$KOffset_{k,d}$	Period Offset for Tanker to Dispense Fuel from Takeoff	INT
$KDur_{k,d}$	Preliminary Duration of Tanker that could be cut short by Fuel	INT
$KType_{k,d}$	Tanker Type for ensuring Proper Hookups	STRING
$KBurnRate_{k,d}$	KGAL burned every Period by Tanker Flying	INT
$KBaseFuel_{k,d}$	Calc KGAL available to include Reserves for Diversion	INT
$KMaxRef_{k,d}$	Maximum Number of Refuels Per Period for the Tanker	INT
$KEarlyPer_{k,d}$	Earliest Period from TACC Tanker is Available	INT
$KLatePer_{k,d}$	Latest Period Tanker is Available	INT
$RFuelReq_r$	Fuel Required for Mission r in KGAL	INT
$REarlyPer_r$	Early Offset from Mission Go to Receive Fuel	INT
$RLatePer_r$	Late Offset from Mission Go to Receive Fuel	INT
$RTypeMsn_r$	Denotes ability of Mission to Pay for Tanker Launch	BOOL

Table 8: Tanker Variables

Variable	Description
$kmiss_{k,d,p} \in \{0,1\}$	Tanker Go Variable. $rmiss$ handled by $xmiss_{r,d,p}$
$kper_{k,d} \in \mathbb{Z}^+$	Period the Tanker Launches
$xhookup_{r,k,d,ph} \in \{0,1\}$	Hookup between Mission r and Tanker k
$xhookBE_{k,d} \in \{0,1\}$	Set for Business Effort Missions that Finance Tankers
$xhookToO_{k,d} \in \{0,1\}$	Set for Tanker of Opportunity Missions
$xhookper_{k,d} \in \mathbb{Z}^+$	Period for Hookup
$bigk_{k,d} \in \mathbb{R}$	Used to relieve Check Fuel Constraint if no Tanker Launches
$kinftank_k \in \mathbb{Z}^+$	Guard against empty Tanker constraints

Variables.

Tanker Constraints.

Equations 24 and 25 set the integer variables $kper_{k,d}$ and $xhookper_{k,d}$ that will be used in equation 36 to control the Tanker fuel level. Equations 26 and 27 are the physical constraints on daily Tanker activity and the Mission Hookups. Equations 28 and 29 link the Hookup to the Tanker and Refuel Mission, and 30 - 32 manage the relationship between paying and opportunistic Missions. Equations 33 and 34 control the Tanker's fuel level as it both dispenses and burns fuel. The variable $bigk_{k,d}$ is set if a Tanker does not take off on any given day so that the Fuel Constraint (34) is

enforced only for an active Tanker.

$$\sum_{P_{k,d}} kmiss_{k,d,p} \times p = kper_{k,d} \qquad \forall\,(k,d) \in KD \tag{24}$$

$$xhookup_{r,k,d,ph} \times ph \leq xhookper_{k,d} \; \forall\,(r,k,d,ph) \in R2KDP \tag{25}$$

$$\sum_{P_{k,d}} kmiss_{k,d,p} + kinftank_k \leq 1 \qquad \forall\,(k,d) \in KD \tag{26}$$

$$\sum_{KH} xhookup_{r,k,d,ph} \leq 1 \qquad \forall\, r \tag{27}$$

$$\sum_{K_r} xhookup_{r,k,d,ph} \geq xmiss_{r,d,p} \quad \forall\,(r,d,p) \in RDP \quad Where:$$

$$(p + REarlyPer_r) \leq ph \leq (p + RLatePer_r \tag{28}$$

$$\sum_{R_k} xhookup_{r,k,d,ph} \geq kmiss_{k,d,p} \quad \forall\,(k,d,p) \in KDP \quad Where:$$

$$(p + KOffset_{k,d} - 1) \leq ph \leq (p + KDur_{k,d} - 1) \quad And$$

$$KEarlyPer_{k,d} \leq ph \leq KLatePer_{k,d} \tag{29}$$

$$\sum_{RH} xhookup_{r,k,d,ph} \times RTypeMsn_r \leq BigM \times xhookBE_{k,d} \quad \forall\,(k,d) \in KD \tag{30}$$

$$\sum_{RH} xhookup_{r,k,d,ph} \times (1 - RTypeMsn_r) \leq BigM \times xhookToO_{k,d}$$

$$\forall\,(k,d) \in KD \tag{31}$$

$$xhookBE_{k,d} \geq xhookToO_{k,d} \qquad \forall\,(k,d) \in KD \tag{32}$$

$$BigM \times \left(1 - \sum_{P_{k,d}} kmiss_{k,d,p}\right) = bigk_{k,d}\ \forall\,(k,d) \in KD \tag{33}$$

$$KBaseFuel_{k,d} \quad - \left[\left(\sum_{RH} xhookup_{r,k,d,ph} \times RFuelReq_r\right) + \left((xhookper_{k,d} - kper_{k,d}) \times KBurnRate_{k,d}\right)\right] + \quad bigk_{k,d} \geq 0 \quad \forall\,(k,d) \in KD \tag{34}$$

3.7.3 Configuration Extension Formulation.

Configurations represent the removable "accessories" on the aircraft to include external fuel tanks, functional PODS for communication or signature changes, and munitions. The modeling team in conjunction with maintenance represented every combination across the nine positions on the aircraft as 104 codes. The team then identified configurations that can change between flights and days.

The key is the pre-processing the sets to work with the constraints to express only the appropriate possibilities. The sets look at the demand for Configurations by the Sorties, and the F16s that are possible for those Sorties. The WKCFGLIM set is structured to force the variables to pick the most appropriate configuration to start the week. The DAYCFGLIM set expresses infeasible configuration combinations between days. In the present rules, there are no changes allowed between flights, or intra-day.

Table 9: Configuration Sets

Set and Index	Description	Example
$C : c \in C$	Configuration Codes for F16	F101_B0

Configuration Sets.

Configuration Functions.

$$CSORT(m, s, c) = \begin{cases} 1 & \text{If Sortie s requests Configuration c} \\ 0 & \text{Otherwise} \end{cases}$$

$$CGO(f, c) = \begin{cases} 1 & \text{If F16 f can be, and could be, in Configuration C} \\ 0 & \text{Otherwise} \end{cases}$$

$$CWK(c1, c2) = \begin{cases} 1 & \text{If c1 cannot coexist with c2 on a weekly basis} \\ 0 & \text{Otherwise} \end{cases}$$

$$CDAY(c1, c2) = \begin{cases} 1 & \text{If c1 cannot coexist with c2 between days} \\ 0 & \text{Otherwise} \end{cases}$$

Derived Sets.

Operating behind the scenes is the derived set $C \times C$. This is where the rules for configuration changes are held and these parameters are used to create the WKCFGLIM and DAYCFGLIM sets which are resident in the model.

Table 10: Configuration Derived Tuple Sets

Set	Description	Notation
F16SCHEDCFG	Append CONFIG to SCHED	$\{(m,s,t,d,p,c)\|SCHED \cdot CSORT = 1\}$
F16CFG	An F16 in a Starting Configuration	$\{(f,c)\|CGO = 1\}$
F16CFGDAY	An F16 in a Configuration on a Day	$\{(f,d,c)\|F16CFG \cdot D = 1\}$
WKCFGLIM	F16 must be in c1 or c2	$\{(f,c1,c2)\|CGO \cdot CWK = 1\}$
DAYCFGLIM	F16 must be in c1 or c2 daily	$\{(f,d,c1,c2)\|CGO \cdot CDAY = 1\}$

Configuration Variables.

Table 11: Configuration Variables

Variable	Description
$xwkcfg_{f,c} \in \{0,1\}$	Controls the Configuration Changes in a Week
$xdaycfg_{f,d,c} \in \{0,1\}$	Controls the Configuration Changes Intra-Day

Configuration Constraints.

Equations 35 and 36 work across the F16SCHEDCFG set to link x to a configuration. These force the cfg variables to be set, and equations 37 and 38 limit how they can be set. The constants 15 and 3 are the smaller versions of $BigM$ representing the maximum number of times an F16 could fly in a week and day respectively.

$$\sum_{MSDP} x_{m,s,f,d,p} \leq 15 \times xwkcfg_{f,c} \; \forall \, (f,c) \in F16CFG \tag{35}$$

$$\sum_{MSP} x_{m,s,f,d,p} \leq 3 \times xdaycfg_{f,d,c} \; \forall \, (f,d,c) \in F16CFGDAY \tag{36}$$

$$xwkcfg_{f,c1} + xwkcfg_{f,c2} \leq 1 \qquad \forall \, (f,c1,c2) \in WKCFGLIM \tag{37}$$

$$xdaycfg_{f,d,c1} + xdaycfg_{f,d,c2} \leq 1 \quad \forall \, (f,d,c1,c2) \in DAYCFGLIM \tag{38}$$

3.8 High Level Solution Steps

This section outlines the high level steps users of this solution will execute to create a schedule. The initial steps currently require some intervention by data experts. As the project progresses these steps will be automated and a scheduling tool will start to take shape.

1. Gather Input Data
    i) CSE Data Pull
    ii) User Input for Key Mission Attributes
2. Set Key Parameters
    i) Operations Group
        a) Validate Program Rankings are sound
        b) Time Window Rules by Ranking
        c) General Hours of Operation
        d) Tanker Availability
        e) Lock Missions as appropriate
    ii) Maintenance
        a) Turn Capacity
        b) Jet Availability by Day or Week
        c) Min and Max Hours Bounds on F16s
    iii) Run Parameters
        a) MIP Gap
        b) Run Time

3. Run Model

    i) Go get Coffee and Breakfast

    ii) Read Email

4. Post Processing

    i) Incorporate Results into Database

    ii) Find Open Capacity and Match Potential Missions/Sorties

    iii) Report on Possible Standby and "Flex" Missions

5. User Reviews Results through Reporting Features

    i) Return to Step 2 and Run Again as Necessary

6. User "Executes" Results

    i) Update CSE (manually at first)

    ii) Disseminate Schedule with Mix of CSE and Tool Reporting

3.9 Conclusion to Chapter

An Integer Programming Solution was implemented for The Developmental Test Scheduling Problem. The operational nature of the input and solve time horizons, as well as difficulty in finding a feasible solution across the competing resource groups, makes the IP approach well suited to the problem. As the resource scope expands, the variables and constraints will increase, impacting the complexity and potentially the run time. The realities of Test Scheduling, and the schedulers assessment, will be the ultimate arbiter of whether or not this approach works.

IV. Results

4.1 Introduction

The results have been verified and the validation phase has commenced with the Test Wing. A specific set of data, called the "Golden Problem," or more accurately the "Golden Instance" of the problem, was used for model development and testing. The Golden Problem results across sub-instances of parameter inputs are discussed for the purpose of showing how the solution works. Four additional weeks of data have also been processed validating the approach.

The Data Capture Section in Chapter III discusses gaps in the data for optimization and the schemes employed to overcome those gaps. Specific parameter input from the base schedulers is needed to codify the results.

4.2 Golden Problem Profile

The Golden Problem encompasses the flying week of 21 - 25 August 2017 with the inputs coming from the CSE data pull on the morning of 11 August. This ensured all the requests were submitted, but no scheduling activity had started. Data was subsequently pulled to show what was actually scheduled. There are 259 Mission requests totaling 361 Sorties, including 73 F16 Sorties and 21 Refuel Missions. The only consideration for Missions/Sorties that do not have a centrally scheduled F16 or do not need a refuel is that any given aircraft cannot fly concurrently and must have the prescribed downtime between flights. The squadrons will manage flying hours, phase maintenance, and configurations required for those decentralized aircraft.

The following tables highlight the significant elements of the Golden Problem.

Other parameters, which are not inherent to the data capture, are more dynamic in the Golden Problem and have to be specified for each sub-instance, or run, during

Table 12: Golden Problem Mission Summary

F16 Mission	Refuel Mission	Mission Count	Sortie Count
No	No	191	244
Yes	Yes	11	25
Yes	No	47	76
No	Yes	10	16
	TOTAL	259	361

Table 13: Golden Problem F16 Sortie Profile

Sorties Per Mission	F16 Sorties	Missions	Other AC Sorties
1	37	37	0
2	16	11	6
3	12	8	12
7	4	1	3
11	4	1	7
TOTAL	73	58	28

testing. For example, the initial Turn Pattern is shown in Table 16. This can be updated to show the impact of maintenance capacity on the answer.

4.3 Base Results

The base results will be considered runs where only the run time, gap, and variables generated are altered. Depending on the combination of run time allowed, the IP Optimality Gap, and the variables generated, the IP will consistently achieve between 54 and 58 F16 Sorties scheduled, and between 18 and 19 refuel Missions. This is with an effective maintenance capacity of 64 (not counting Trip Turns). This compares with the 56 sorties that were actually scheduled during the Golden Problem week. To set up the base results, Table 17 shows the pertinent instance parameters.

Figure 20 depicts the range of best answers obtained by balancing the variables generated and the run times, with a MIP Gap of less than 1%. This reflects a desire to run for a limited length of time to get the best answer possible. A good trade-

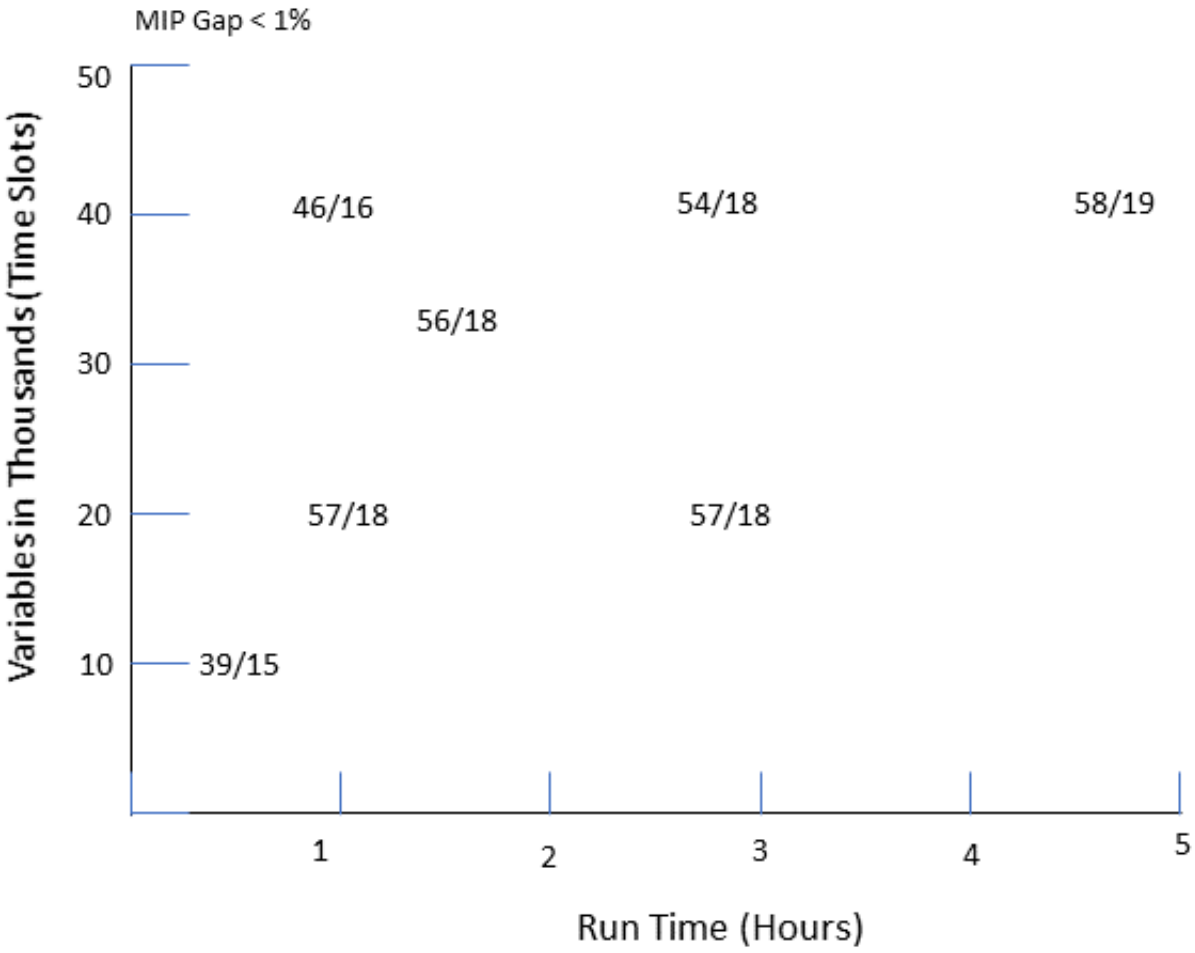

Figure 20: F16s/Refuels Scheduled varying Run Time and Variables

Table 14: Golden Problem Refuel Mission Profile

Lead Ship MDS	Mission Count	Sortie Count
F16C	1	2
F16D	1	1
F35A	8	20
F35B	3	4
F35C	2	2
KC30	6	6
TOTAL	21	41

Table 15: Golden Problem Special Mission Summary

Mission Type	Missions	Total Sorties	F16 Sorties
F16	58	101	73
Refuel	21	41	9
Precedent	3	3	3

off for this instance of the problem is to generate roughly 22,000 variables. This equates to one day available for higher priority missions and three days available for lower priority missions with two time periods available each day, 12 periods apart. Generating more time windows does not necessarily improve the answer with a fixed run time.

Figure 21 is a snippet of an answer in a Gantt Chart Format. The numbers in the cells are the durations in hours that each Sortie will fly.

Table 16: Starting Sortie Capacity for F16s

Day	Fronts	Turns	Trips	Capacity
Mon	6	4	0	10
Tue	8	6	2	16
Wed	8	6	0	14
Thu	8	6	2	16
Fri	8	4	0	12
Total	38	26	4	68

	PERIOD:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23				
													FRI												FRI	TOTAL		
	TAIL NO	6:00	6:30	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30	12:00	12:30	13:00	13:30	14:00	14:30	15:00	15:30	16:00	16:30	17:00	TOTAL	TIME		
1	F16A-3666																									12.0		
2	F16C-0352																									2.0		
3	F16C-0371			3.0																						3.0		
4	F16C-0445																									7.0		
5	F16C-1547			3.0																						3.0		
6	F16C-1551																									7.5		
7	F16C-1560			3.0																						3.0		
8	F16D-0050																									5.0		
9	F16D-0174																									6.0		
10	F16D-0370																									1.5		
11	F16D-0386																									5.5		
12	F16D-0391								1.5													1.5				17.0		
13	F16D-0835					2.0													3.0							5.0		
14	F16D-1464			3.0																						12.5		
15	F16D-2169					2.0													1.5							15.0		
16	F16D-2176				1.0																1.5					7.5		
	MISSION TIME																								26.0	112.5		
	SORTIES																								12	58		
	FRONTS																								8	38		
	TURNS																								4	20		
	TRIPS																								0	-	% of Cap	% of Total
	LIMITS																								Total Sorties =	58	85%	79%
	FRONTS																								8	38		
	TURNS																								4	26		
	TRIPS																								0	4		%Cap/Total
																									Total Capacity =	68		93%
																									Total F16 Sorties =	73		

Figure 21: 58 F16 Sortie Answer in Time Line Format

4.3.1 Finding the right MIP Gap.

The quality of the answers are a function of the MIP gap and the run time allowed. A high MIP gap might leave sorties unscheduled unnecessarily when a longer time was acceptable. A MIP Gap of 3%, with 22,000 variables and 50 minutes of allotted run time is a good mix for this instance of the problem where the IP runs to conclusion in about 45 minutes. The answer will generate 56 F16 Sorties and 17 Refuel Missions. Depending on how the model fits into the scheduling rhythm, this might be a good balance of the IP run parameters.

Table 17: Base Parameter Settings

Parameter	Setting
Period Offset Each Day:	12 Periods from Preferred Period
Mx Weekly Capacity:	68 Sorties per Pattern in Table 16
Fuel Requested:	5K lbs for each of 21 Refuel Missions
Configurations:	Defined at Sortie Level
F16 Configurations:	No Starting Configuration (Open)
F16s Available for Sortie	Simple Groups by F16 characteristics

4.4 Variants on the Base Results

4.4.1 Objective Function Variants.

This current objective function scores at the mission level, effectively penalizing multiple sortie count missions that are inherently more difficult to schedule. Planners might want to give some benefit to multiple sortie missions. There were two missions in the Golden Week that had seven and eleven sorties respectively that were described as "Show of Force Missions." A linear benefit in sorties is considered unfair as these missions consume more resources. A decreasing function like the square root of the sorties or the natural log of the sortie count is a good compromise. With the square root function a two F16 Sortie Mission would be worth 1.41 times the one Sortie Mission and four sorties is worth double. Implementing sortie level scoring is shown in equation 39:

$$\begin{aligned} Max: \quad & \Big(\sum_{\{m,d,p,s\}} (xmiss_{m,d,p} * Pri_m * FxSortie_s * WGHT\_PRI) \Big) \\ & - (DaysDiff_m * WGHT\_DAY) \\ & - (PerDiff_m * WGHT\_PER)) \end{aligned} \quad (39)$$

The results for this variant behave as expected with multiple Sortie Missions getting on the Schedule at the expense of the number of overall missions scheduled. This will be validated with the schedulers to determine what makes the most sense to them.

4.4.2 Two Pass Approach.

In this variant, the first pass is a tight window run for a short period of time similar to the 39/15 result from Figure 20. The second run opens the time windows and runs again for a short time with the scheduled Missions from the first pass locked down.

The results are mixed with the answers being no better than a one pass approach and often worse. The combined time is faster on average than the one pass approach which is a benefit to further explore. There is complexity in setting up two runs that might not be the worth the benefit in run time. The detailed results are not published as more instances of the problem need to be investigated to determine if this is a valid approach.

4.5 Post Processing

The post processing heuristic approach discussed in Chapter III is used in two instances. First where the MIP gap is high enough or run time short enough to miss sorties that could have been scheduled. The second is when the time periods for a mission do not line up with the available time periods, but the flying unit would agree to go outside their requested time windows.

Figures 21 and 22, translated into MS Excel for readability, show the latter situation and the results of the post processing approach. Figure 21 is the Possible Sortie Report with comments by the research team. Figure 22 is the answer that preceded the report.

Unschedule Sortie Profile									Open Turn Profile				Scheduler Comments	
Ops Num	MsnRank	Sortie	Sort Ct	Pref Day	Pref Pd	Duration	Config	Refuel	Tail	Day Per		Config	Works	Notes
468	1	8	11	5	3	6	F101_00	No	F16D-2169	1	5	F101_90	No	11 Sortie Mission with 4 F16 Sorties.
8700	3	1	3	3	5	6	F102_90	No	F16D-0392	4	15	F102_90	No	Two Sorties Competing for Same Tail
8700	3	2	3	3	5	6	F102_90	No	F16D-0392	4	15	F102_90	No	Must Go Together so will not work
8700	3	3	3	3	5	6	F102_90	No	F16D-1464	4	15	F102_90	No	
8867	5	1	1	3	10	3	F999_B0	No	F16D-0377	1	5	F999_90	Yes	If Mission will move this works
8867	5	1	1	3	10	3	F999_B0	No	F16D-0377	5	15	F999_B0	Yes	Ditto

Figure 22: Possible Sortie Report Generated from Post Processing Heuristic

Mission 8867 had one swing day from Wednesday to Thursday and Tuesday, but was not given the opportunity to be scheduled on Monday or Friday, where it would have been able to fly.

F16 GANTT CHART

F16 Flying Schedule using Requests from 21-25 AUG 2017

Periods / Hours

Unit of Time Represented

	PERIOD:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	
		MON																							MON
	TAIL NO	6:00	6:30	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30	12:00	12:30	13:00	13:30	14:00	14:30	15:00	15:30	16:00	16:30	17:00	TOTAL
1	F16A-0584					2.0																			
2	F16A-3666																								
3	F16C-0352																								
4	F16C-0371																								
5	F16D-0047								1.5									1.0							
6	F16D-0050				1.5																	1.5			
7	F16D-0174																								
8	F16D-0377					-	-	-	-	-	-	-	-	-	3.5										
9	F16D-0378				1.0												1.0								
10	F16D-0386																								
11	F16D-0392																								
12	F16D-0464																								
13	F16D-0840																								
14	F16D-1464																								
15	F16D-2169																	3.5							
16	F16D-2176																								
	MISSION TIME																								16.5
	SORTIES																								9
	FRONTS																								6
	TURNS																								3
	TRIPS																								0
	LIMITS																								
	FRONTS																								6
	TURNS																								4
	TRIPS																								0

Figure 23: Output of Run with Sortie that can move to another Day (8867)

4.6 Conclusion to Chapter

The results from running an Integer Programming based ecosystem are promising. The right mix of run time, variables generated, and optimality gap will continue to be explored to find the balance of inputs yielding good answers in an acceptable time frame for this environment.

V. Conclusions and Future Research

5.1 Introduction

The research into the Developmental Test Scheduling Problem can expand in all directions.

- The research should immediately extend into scheduling the Range Resources. The team has had success scheduling the Mission Control Rooms with a limited rule set.
- The research can explore the critical inputs. For example, the model can take responsibility for setting the minimum and maximum hours for each aircraft rather than accepting them as an input from Maintenance.
- The model could be used to aid the strategic decision makers in what-if analysis of having alternate or additional resources at their disposal to understand the cost/benefit trade-offs.
- There is work to determine the key areas of similarity and differentiation between a planning tool and an execution tool that captures the current state of a flying week and can make quick recommendations when a disturbance to the schedule is experienced.
- Varying solution procedures such as heuristics should be examined for speed and quality. Given the subjective nature of the input, optimality is to be balanced with speed-to-answer and the ability to implement the answer.
- To realize a successful implementation of this tool, process and organizational design work will be required.

5.2 Range Resources

The Range is the blanket term used to describe a set of resources that includes control rooms, telemetry frequencies and equipment, and airspace. While each resource has nuances and challenges, the Airspace will be the most challenging resource to deconflict at Edwards given the required inputs and current processes.

5.3 Optimization of the Critical Model Inputs

The minimum and maximum hours a jet can fly in a week is left to the maintenance team in the current implementation. Diving deeper into the statistical analysis presented in Chapter III, future research would develop a probability distribution for demanded flying hours on each individual jet. A function must be developed to solve for the appropriate range of flying hours in order to maintain an acceptable phase flow across the fleet. This includes some jets flying proportionally more hours, dictated by the demand, such that they may be in phase twice as often as other jets. Maintenance will be biased toward keeping the hours to phase within a tolerance from the line to ensure they do not have too many jets hitting phase, nor have zero jets for the crew to work on.

5.4 Strategic Decision Making

While the solution approach discussed focuses on an optimal resource allocation in an operational sense, strategically using the model to aid in optimizing local resource availability should be explored. "What-If" analysis looking back at flying weeks where a different mix of resources were made available could be valuable in justifying higher level resource allocation decisions for Air Force Materiel Command.

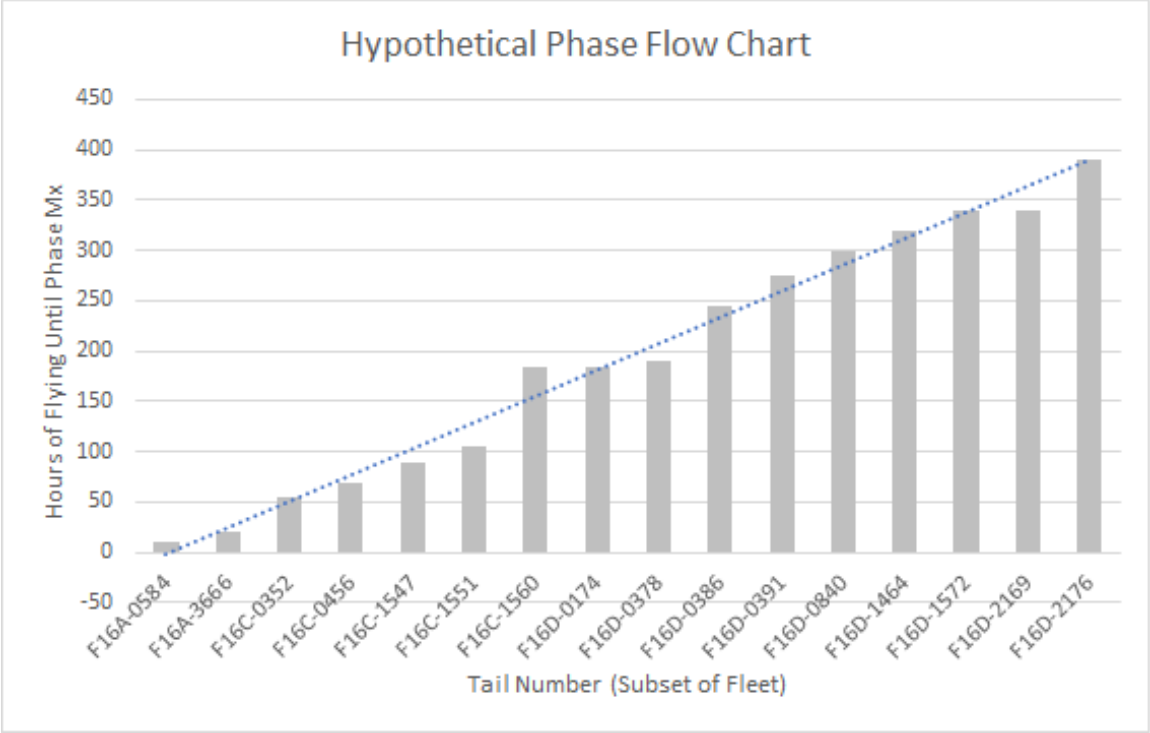

Figure 24: Phase Flow Hypothetical Example

5.5 A Tactical Implementation

The Koepke team shows that a more tactical model can be developed with Integer Programming in their Channel Route Mission Implementation where the expectations for solve times are relatively quick [6]. The Maintenance Team at Edwards has repeatedly asked for a Tactical level implementation of the tool with the ability reschedule after an in-week perturbation to the schedule like unscheduled maintenance. This is an excellent area for future research as the problem parameters and characteristics are inherently different than an operational level scheduling problem.

5.6 Developing other Solution Procedures

The IP formulation for the Developmental Test Scheduling Problem is a sound initial approach regardless of the solution procedure. The parallel development of

heuristic approaches with other IP algorithms will be prudent, and each approach lends insight to the other. The work on a heuristic solution led to the development of some of the processing techniques before and after the IP solve. The addition of Range Resources, particularly the Airspace, could be the impetus for trying different solution techniques.

There are not millions of variables in the Developmental Test Problem, but the Range considerations could add a significant number of variables. Given the run times already being experienced for runs with wider time possibilities, different formulation and solution approaches have to be explored. Dr. Jeffery Weir successfully used column generation on a large scale Mixed Integer Program used to aid airlines in pilot scheduling [12].

5.7 Conclusions

The characteristics of the Developmental Test Scheduling Problem involve a complex interaction between the resources, making the problem well suited for an Integer Programming Approach. The solution is a modeling ecosystem leveraging preprocessing and post-processing with an open source solver running a conventional branch and cut algorithm. This approach solves within acceptable tolerances and time limits. As the problem develops, there will be the need for even more creative solution procedures across the ecosystem. To that end the problem should develop given the strategic and vital role Developmental Test plays in the U.S. Air Force. This R&D function is a core competency that deserves the best tools possible.

The design and development of new and refined data inputs is a valuable contribution of this project to the central scheduling function of the Test Wing.

This initial work represents some thought leadership and structure to build upon for The Developmental Test Scheduling Problem.

Bibliography

1. Zhi-Long Chen and Warren B. Powell. Solving parallel machine scheduling problems by column generation. *Informs Journal on Computing*, 11(1):78–94, 1999.

2. Kathryn A. Dowsland. Nurse scheduling with tabu search and strategic oscillation. *European Journal of Operational Research*, 106(23):393–407, 1998.

3. Gary G. Foster. Automating the weekly flight scheduling process at the USAF Test Pilot School. Technical report, AFIT, 1992.

4. Fred Glover and Manuel Laguna. *Tabu Search*. Kluwer Academic Publishers, Norwell, MA.

5. Lisa Hassel. Investigation of a zero-one integer programming approach to automating the scheduling process at the USAF Test Pilot School. Technical report, AFIT, 1992.

6. Corbin G. Koepke, Andrew P. Armacost, Cynthia Barnhart, and Stephan E. Kolitz. An integer programming approach to support the US Air Force's Air Mobility Network. *Computers & Operations Research*, 35(6):1771–1788, 2008.

7. Eugene L. Lawler, Jan Karel Lenstra, Alexander H.G. Rinnooy Kan, and David B. Shmoys. Chapter 9 sequencing and scheduling: Algorithms and complexity. In *Logistics of Production and Inventory*, volume 4 of *Handbooks in Operations Research and Management Science*, pages 445 – 522. Elsevier, 1993.

8. Christopher A. Nielsen, Andrew P. Armacost, Cynthia Barnhart, and Stephan E. Kolitz. Network design formulations for scheduling US Air Force Channel Route Missions. *Mathematical and Computer Modeling*, 39(6-8):925–943, 2004.

9. Michael L. Pinedo. *Scheduling: Theory, Algorithms, and Systems*. Springer International Publishing, AG Switzerland.

10. Daniel Reich, Yuhui Shi, Marina Epelman, Amy Cohn, Ellen Barnes, Kirk Arthurs, and Erica Klampfl. Scheduling crash tests at Ford Motor Company. *Interfaces*, 46(5):409–423, 2016.

11. Michael O. Said. Theory and practice of integrated test for Navy Programs. Technical report, Assistant Secretary of the Navy (Research, Development and Acquisition) Wash DC, 2009.

12. Jeffery D. Weir. A three phase approach to solving the bidline problem with an emphasis on mitigating pilot fatigue through circadian rhythm rule enforcement. Technical report, Georgia Institute of Technology, 2002.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
22-03-2018	Master's Thesis	Sept 2016 — Mar 2018

4. TITLE AND SUBTITLE

THE DEVELOPMENTAL TEST SCHEDULING PROBLEM

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Joseph E. Schoenbeck, Joseph E., Civilian

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENS-MS-18-M-160

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Department of Operational Sciences
2950 Hobson Way
WPAFB OH 45433-7765
DSN 271-0690, COMM 937-255-3636
Email: Joseph.Schoenbeck@afit.edu

10. SPONSOR/MONITOR'S ACRONYM(S)

AFIT COA

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Developmental testing of aircraft systems in the United States Air Force requires a complex set of resources for each test. The optimal scheduling of those resources is the job of the 412^{th} Test Wing at Edwards Air Force Base. With more than 20 different Combined Task Forces requesting resources for roughly 300 flying missions each week, manual scheduling is a difficult task. The current process takes a team of schedulers several days to get a workable result from which they can start tailoring the final schedule. While concepts and techniques can be taken from industry scheduling problems, the body of knowledge as it relates to developmental test scheduling is sparse. The contribution of this paper is to initially document the Developmental Test Scheduling Problem, define it in structured terms for which a solution methodology can be designed, and then present an Integer Programming based solution. The design allows for a scheduler to tailor an initial answer to fit nuanced and timely objectives and constraints. We will demonstrate results that create an initial schedule in several hours and serve as a good starting point for the final schedule.

15. SUBJECT TERMS

Integer Programming, Scheduling

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Jeffery D. Weir, AFIT/ENS
U	U	U	U	90	19b. TELEPHONE NUMBER *(include area code)* (937)255-3636 x4523; Jeffery.Weir@afit.edu