12-2022

# Optimal Scheduling Of Aircraft Test and Evaluation Fleets to Balance Availability for Testing and Training

Sarah E. Hoops

**OPTIMAL SCHEDULING OF AIRCRAFT TEST AND EVALUATION FLEETS TO BALANCE AVAILABILITY FOR TESTING AND TRAINING**

THESIS

Sarah E. Hoops, Civilian, USAF

AFIT-ENS-MS-22-D-024

**DEPARTMENT OF THE AIR FORCE**
**AIR UNIVERSITY**

# AIR FORCE INSTITUTE OF TECHNOLOGY

**Wright-Patterson Air Force Base, Ohio**

**DISTRIBUTION STATEMENT A.**

AFIT-ENS-MS-22-D-024

# OPTIMAL SCHEDULING OF AIRCRAFT TEST AND EVALUATION FLEETS TO BALANCE AVAILABILITY FOR TESTING AND TRAINING

THESIS

Presented to the Faculty

Department of Aeronautics and Astronautics

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

Sarah E. Hoops, BS

Civilian, USAF

December 2022

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

AFIT-ENS-MS-22-D-024

# OPTIMAL SCHEDULING OF AIRCRAFT TEST AND EVALUATION FLEETS TO BALANCE AVAILABILITY FOR TESTING AND TRAINING

Sarah E. Hoops, BS

Civilian, USAF

Committee Membership:

Dr. Brian J. Lunday
Chair

Lt Col Jesse G. Wales
Reader

AFIT-ENS-MS-22-D-024


## Abstract

The 96th Test Wing at Eglin Air Force Base manually schedules a fleet of approximately 26 aircraft to conduct a range of missions over a one-to-two-year planning period. This study automates the scheduling process, in a manner that optimizes multiple planning goals related to aircraft availability for training and provides the 96th Test Wing with a software tool for the implementation that can be used by operational analysts within the command. We formulate the scheduling problem as a multi-objective, nonlinear, binary integer math program that seeks to maximize both the lowest percent of time any aircraft is available for training and the lowest percent of aircraft available for training for any week. Applying the Weighted Sum Method for multi-objective optimization, a conversion of nonlinear operations yields a binary integer program that is directly solvable via a commercial solver. An examination of the multi-objective nature of the problem identified a lack of tension between the objectives, so empirical testing affixes equal weights to the well-scaled objective function components. The 96th Test Wing directed the use of Excel as a modeling environment and What'sBest! by Lindo Systems, Inc. for their analysts to use. Subsequent testing examined for increasing time horizons the ability of the model and solver combination to develop optimal or near-optimal schedules for increasing levels of mission densities. The required computational effort was not predictable by mission density level, but increasing levels did yield instances not solvable to optimality within four hours for each time horizon. Optimal solutions can be readily identified within a few seconds for the current fleet and current or higher mission densities for 8-, 26-, and 52-week schedules, using weekly granularity for mission scheduling. In contrast, 78-week schedules can only be solved for a lower mission density than currently addressed by the 96th Test Wing within a four-hour time limit. However, the 96th Test Wing's current process is conducted manually over a period of hours, and the fragility of a such a long-term schedule to evolving mission requirements makes its merit subjective. In aggregate, this study shows an integer programming approach is feasible and has utility to the 96th Test Wing organization for their scheduling problem.

*This research is dedicated to my family, friends, and especially my husband and puppies for being the support system that I needed to complete this endeavor.*

**Acknowledgments**

I would like to express my sincere appreciation to my faculty advisor, Dr. Lunday, for his guidance and support throughout the course of this thesis effort. The insight and experience was certainly appreciated. I would, also, like to thank my coworkers in A9A of Head Quarters Air Force Material Command, for their encouragement, support, and guidance.

Sarah E. Hoops

## Table of Contents

## List of Figures

**List of Tables**

OPTIMAL SCHEDULING OF AIRCRAFT TEST AND EVALUATION FLEETS TO BALANCE AVAILABILITY FOR TESTING AND TRAINING

# I. Introduction

## 1.1 Motivation and Background

The Air Force conducts Developmental Test and Evaluation (DT&E), Operational Test and Evaluation (OT&E), and Integrated Developmental Test/Operational Test (Integrated DT/OT). Actions within DT&E evaluate design methodology, validate models, identify systemic problems, predict the operational performance of the integrated systems effectiveness and suitability, quantify manufacturing quality measures, and quantify contract technical performance [1]. In aggregate, DT&E seeks to minimize design risk. In contrast, OT&E determines system effectiveness and suitability after systems are procured and fielded, while simultaneously evaluating military utility. OT&E determines the effectiveness and suitability of weapons, equipment, or munitions for use in combat by typical military users, and it yields results to validate or modify the systems to improve their utility. In comparison, Integrated Developmental Test/Operational Test (Integrated DT/OT) is defined as

> An efficient approach to T&E, executed with the deliberate intent and planning to use specific test events and activities for both developmental test and operational test analysis and reporting, when there are clear cost and/or schedule advantages. The high cost or lack of sufficient test articles may provide an overall benefit for DT&E and OT&E teams to share test

resources and data. IDT/OT usually ends with a phase of dedicated OT&E. AFOTEC always considers doing IDT/OT for all programs. The restriction for contractor involvement in USC, Title 10 applies only to dedicated OT&E [1].

As a part of the ecosphere of US Air Force test and evaluation activities, the 96th Test Wing at Eglin AFB conducts a range of DT&E, OT&E, and IDT/OT [2]. Eglin AFB is a central hub for T&E due to its favorable climate and numerous available ranges. As a result, many testing units are co-located at Eglin AFB. The activities of a test wing include nonnuclear munitions testing and munitions ground testing; munitions seeker performance evaluation; aircraft stores integration; command, control, and information testing; computer network attack; aircraft systems and electronic countermeasure evaluation; munitions guidance systems; aircraft navigation and guidance systems; radar target signatures; unmanned aerial vehicles; and base intrusion and interdiction systems.

The 96th Test Wing has 32 squadrons and divisions, and their test and evaluation activities support the largest of procurement programs in the US Air Force. The 96th Test Wing has two operational test partners: the Air Force Operational Test and Evaluation Center (AFOTEC) and the 53rd Test Wing, both having an organizational presence at Eglin AFB. AFOTEC conducts initial operational tests for newly purchased devices. Once that item is in the USAF inventory, responsibility for further testing lies with the appropriate Major Command (MAJCOM). For example, F-15C Eagles and F-15E Strike Eagles have been flying for over 30 years, and Air Combat Command (ACC) has the responsibility to oversee their maintenance and upgrades. Supporting ACC, the 53rd Test Wing conducts OT&E for these aircraft.

The 96th Test Wing's partners are important because, in combination with the wing's DT&E focus, their combined efforts yield a notable amount of IDT/OT. In doing so, they share devices and assets. In practice, sometimes Operational Test (OT) pilots will fly an IDT/OT mission, and sometimes the Developmental Test (DT) pilots will fly it. Thus, it is necessary to coordinate the use of aircraft, pilots, ranges, sensors, and related assets in a manner that effectively supports the test activities while making efficient use of resources.

Relevant to the complexity of the coordination necessary to schedule testing activities at Eglin AFB, there is a natural competition for resources among three different squadrons:  the 40th Flight Test Squadron, the 85th Test and Evaluation Squadron, and the Operational Flight Program Combined Test Force (OFPCTF) squadron [3]. The 40th Flight Test Squadron is a DT&E squadron assigned to the 96th Test Wing, whereas the 85th Test and Evaluation Squadron is assigned to the 53rd Test Wing. The 40th and 85th squadrons have the primary missions of flying DT&E and OT&E missions, primarily focused on variants of the F-15 Eagle and the F-16 Falcon [4]. The OFPCTF squadron conducts both DT&E and OT&E, and it is comprised of elements from and supports both the 96th and 53rd Test Wings. The OFPCTF squadron conducts testing related to the operating systems on the respective aircraft and their variants.

The 96th Test Wing schedules the activities of these three squadrons, both at Eglin AFB and in support of off-site testing activities, and that scheduling problem is the focus of this research. These three squadrons have a combined fleet of 26 aircraft of various types. There are several demands for aircraft beyond test and evaluation activities. Other demands on the fleet include modifying and de-modifying aircraft for specific testing activities, maintaining the aircraft, and supporting exercises and testing at other ranges (i.e., away from Eglin AFB). Moreover, readiness is always an important consideration for the 96th Test Wing; they want their pilots (and aircraft) to train routinely and remain proficient in fundamentals related to air combat. Such dedicated training should consider the fleet of pilots and aircraft as well as the amount of available ranges for training at Eglin AFB. There is a natural tension between test and evaluation mission execution and training to maintain readiness, so it is important to balance the mission demands within a schedule of activities.

This research directly supports the 96th Test Wing and their efforts to schedule a fleet of different aircraft to conduct a range of missions over an extended time horizon. The 96th Test Wing currently develops a schedule of aircraft from the three squadrons to conduct the aforementioned activities over a schedule to meet test and evaluation; maintenance; and other exercise-related demands over a 6- to 18-month horizon, all while maximizing the availability of aircraft for training.

From 2015 through 2021, the 96th Test Wing develops and/or refines this schedule manually using Microsoft Excel, and they refer to the Excel notebook and its schedule as the *Iron Flow* tool because it directs the flow (i.e., schedules) and the iron (i.e., aircraft) to their various missions and activities. Figure 1 depicts a screen capture of this tool and a representative schedule for non-specific aircraft (i.e., “Frogbats” and “Ratcats” in lieu of actual US Air Force aircraft) and a variety of letter-coded missions other than training. Table 1 characterizes the missions the 96th Test Wing schedules with the Iron Flow tool, as of 2021. Blank entries indicate an aircraft is available to conduct testing and training missions.

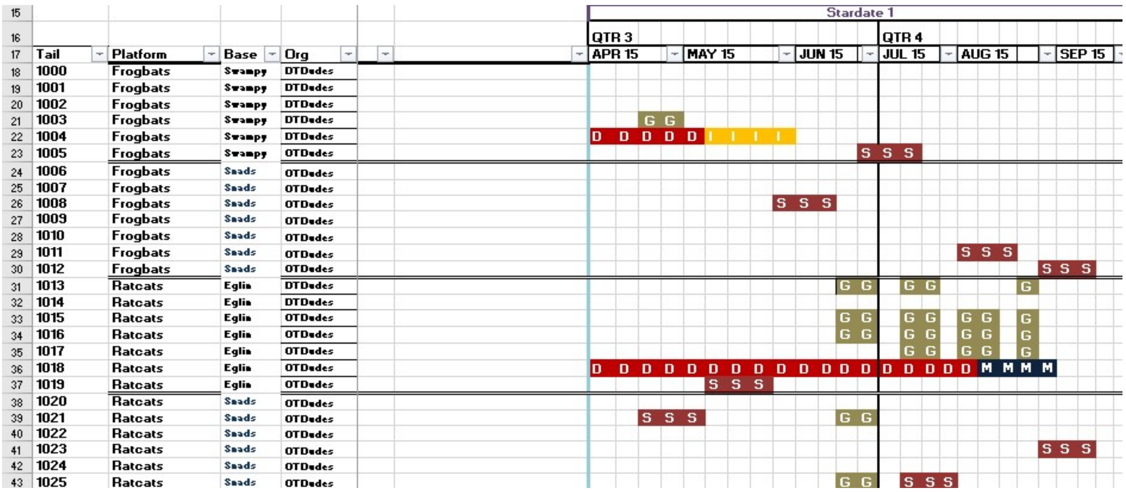


**Figure 1. Screenshot of the Iron Flow Tool to Schedule Aircraft**

**Table 1. Missions Scheduled via the Iron Flow Tool**

| Letter Code | Mission | Description |
|---|---|---|
| I | Instrumentation Modification / Demodification | Test flights that can include a tiny high-speed camera, shock vibration strain gauge, thermometer, and data recorder. Typically lasts for 4 weeks. |
| O | Instrumentation Other | Miscellaneous instrumentation, deconfliction, and upgrades to instrumentation |
| S | Phase/Scheduled Major Maintenance | Local activity that includes stripping the plane but not as much as Program Depot Maintenance. Typically lasts 3 weeks; conducted after a certain number of flying hours |
| G | Off Station/Non-Supporting Status | Temporary duty off station to support a test or exercise. Typically lasts 3 to 6 weeks. |
| T | TCTO (Time Critical Tech Orders) | Maintenance including product recalls and must be completed as soon as possible. Not scheduled in advanced. A notice comes out, and individual aircraft are grounded until the TCTO is completed. |
| R | Radar Upgrades | Self-explanatory. Typically lasts 12 to 20 weeks |
| D | Programmed Depot Maintenance | Take aircraft to Warner Robins AFB and strip it completely down to inspect and conduct maintenance. As an example, this lasts about 17 weeks, every 6 years for F-15s. |
| A | ADCP-II Mod (Advanced Computer Processor) | Major modifications and internal computer upgrades to replace graphics processor. Typically lasts 2 to 19 weeks. |
| E | EPAWSS Mod (Eagle Passive Active Warning Survivability System) | Major modified receiver targeted at F-15 variant radars. Typically lasts 26-30 weeks. |
| M | Other Modification | Typically lasts 4 to 7 weeks. |

The Iron Flow tool depicted in Figure 1 illustrates the variety of demands, scheduled out for different aircraft and aircraft types to conducts over an extended duration of weeks. Within the Iron Flow tool are embedded Microsoft Visual Basic macros to calculate the number of available aircraft in each week and update the graphic depiction of the schedule. However, the actual schedule developed with the Iron Flow tool are created manually, without any direct attempt at optimizing any scheduling objectives other than satisfying demands. Another shortcoming to the Iron Flow tool is that, whenever information or assumptions change, the schedule must be manually edited. The 96th Test Wing did not articulate the exact duration to create or edit a schedule because it varies based on the nature of the changes, but they acknowledged that doing so is a slow, (cognitive) labor intensive process that requires notable subject matter expertise. Moreover, maintaining this tool requires a non-trivial effort as well.

Reinforcing the nature of the scheduling challenge, personnel from the 96th Test Wing informed the author of additional procedural changes for developing and refining schedules. Of recent, the Iron Flow tool has fell out of use, and the 96th Test Wing has recently embraced a coarser solution technique involving the manual adjustment of colored blocks around on PowerPoint slides to schedule aircraft against demands.

Due to the obvious disadvantages of using PowerPoint for this scheduling problem, the senior analyst from the 96th Test Wing wants to leverage an Excel-based tool similar to the Iron Flow tool, yet via an automated process that yields optimal or near-optimal outcomes with respect to readiness-related objectives. Such is the intended outcome of this research.

The $96^{th}$ Test Wing's Operations Group seeks an automated solution process that can be iteratively and repeatedly applied to effectively and efficiently solve the aircraft scheduling problem, saving personnel effort in the process. The elements of this process should include:

1. The use of parametric data in a format familiar to the $96^{th}$ Test Wing
2. An appropriate mathematical programming model
3. Either an exact solution method or heuristic approach, as appropriate to ensure computational tractability
4. Implementation of these elements in a software platform usable by the operations group.

### 1.2 Problem Statement

This research seeks to model and efficiently solve the $96^{th}$ Test Wing's aircraft scheduling problem, attaining a solution that maximizes aircraft availability for testing and training while meeting testing, maintenance, and other selected demands for combined fleet of aircraft from the $40^{th}$ Flight Test Squadron, the $85^{th}$ Test and Evaluation Squadron, and the OFPCTF squadron over a 6- to 18-month time horizon [5].

### 1.3 Research Questions

The following research questions are sequentially answered to address the problem statement:

1. Can the $96^{th}$ Test Wing's aircraft scheduling problem be modeled via a compact mathematical program that addresses nuanced, complicating formulation aspects

(e.g., nonlinearities, binary or integer-restricted decision variables), with the goal of optimizing multiple desired outcomes?

2. When using the proposed model in combination with both a modeling environment and commercial solver directed by the 96$^{th}$ Test Wing, what are the limits of computational tractability for the proposed mathematical programming formulation, in terms of density of mission demands and duration of a schedule?
3. Does the combination of the proposed model and demonstrated performance on realistically sized instances portend a practical tool for use by the 96$^{th}$ Test Wing that can be utilized within Microsoft Excel and identify optimal or near-optimal schedules without resorting to either manual or heuristically solution methods?

### 1.4 Organization of the Thesis

The remainder of the thesis is organized as follows. Chapter II reviews literature related to optimal scheduling and aircraft scheduling. Chapter III introduces the mathematical programming formulation for the underlying problem to address Research Question 1. Chapter IV conducts validation testing for the model and empirical testing to address Question 2. Chapter V concludes by addressing Research Question 3 as it summarizes the major outcomes of the research, ultimately identifying selected recommendations for both the research sponsor's implementation of this work and future research to extend it.

# II. Literature Review

This chapter reviews relevant aspects of the technical literature from two threads of research. Section 2.1 reviews fundamental mathematical programming constructs necessary to formulate the mathematical programming formulation set forth in Chapter III. Section 2.2 reviews technical research related to scheduling problems, in general, whereas Section 2.3 reviews relevant work that addresses aircraft scheduling applications.

## 2.1 Selected Fundamental Concepts from Mathematical Programming

Important for this research are the concepts of multi-objective optimization, integer programming, binary integer programming, and the branch-and-bound algorithm as a method to solve integer and binary-integer programs.

### Multi-Objective Optimization

Many real-world optimization problems have multiple conflicting objectives and would benefit from examining decisions within a multi-objective optimization framework, including the problem examined herein. Recall that the 96$^{th}$ Test Wing has more than one objective to optimize, as will be discussed in detail in Chapter III. As such, a multi-objective optimization framework will be appropriate to consider.

Multi-objective optimization problems seek to find decision variable values that address more than one objective function, and those objectives are typically in tension [6]. Such a formulation is conventionally represented for a linear program having $n$ objective functions as follows [7].

$$
\begin{array}{ll}
min & (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \dots f_n(\boldsymbol{x})) \\
s.t. & \boldsymbol{Ax} \geq \boldsymbol{b} \\
 & \boldsymbol{x} \geq \boldsymbol{0}
\end{array}
$$

A "good decision" within multi-objective optimization is a non-dominated solution; one may not improve one objective function value without worsening the value(s) or one or more other objective functions. Such a solution is known also known as a *Pareto optimal* solution, and the set of such non-dominated solutions is known as the Pareto front, efficient front, or Pareto frontier.

There are multiple techniques to explore a Pareto front, the most prevalent within the literature being the Weighted Sum Method and the $\varepsilon$-constraint Method. The Weighted Sum Method constructs a single objective function by adding the respective objective function values, each with a scaled weight and typically with the weights scaled such that their sum equals one. Such an approach for a linear program yields the following representation.

$$
\begin{array}{ll}
min & w_1 f_1(\boldsymbol{x}) + w_2 f_2(\boldsymbol{x}) \dots + w_n f_n(\boldsymbol{x}) \\
s.t. & \boldsymbol{Ax} \geq \boldsymbol{b} \\
 & \boldsymbol{x} \geq \boldsymbol{0}
\end{array}
$$

In contrast, the $\varepsilon$-constraint Method optimizes one objective function while bounding any other objective functions, iteratively adjusting those bounds. For a given set of bounds and while optimizing the first objective function, the $\varepsilon$-constraint Method formulation for the aforementioned linear program would be represented as follows.

$$
\begin{array}{ll}
min & f_1(\boldsymbol{x}) \\
s.t. & f_i(\boldsymbol{x}) \leq \varepsilon_i, \quad i = 2, \dots, n \\
 & \boldsymbol{Ax} \geq \boldsymbol{b} \\
 & \boldsymbol{x} \geq \boldsymbol{0}
\end{array}
$$

The $\varepsilon$-constraint Method is well-suited for problems in which most objective functions – preferably $(n-1)$ objective functions – can only take on integer values because there is only a finite number of readily discernible $\varepsilon$-values to consider for the iterative exploration of the Pareto front. Alternatively, the Weighted Sum Method is versatile, but one must account for the scale of the relative objective function values. If a problem is not well scaled, the set of non-dominated solutions may not be identified across a uniformly distributed set of weights. In either case, the existence of multiple objectives does not require that they be in tension; a first step when examining a multi-objective optimization problem is to ascertain whether tradeoffs exist between the different objectives.

This research will leverage the Weighted Sum Method for multi-objective optimization. As will be evident in Chapter III, the preferred criterion for using the $\varepsilon$-constraint Method is not present in the proposed mathematical programming formulation.

**Integer Programming (IP)**

Integer programming (IP) is a subset of mathematical programming, related to linear programming (LP) with the additional constraint that at least a subset of decision variables must be integer-valued [6]. If all decision variables are integers, an integer program is said to be pure [8]; otherwise, it is known as a mixed-integer linear program. Scheduling problems are a class of integer programs because schedules naturally entail decisions that relate to integer-valued numbers, e.g., one cannot schedule half of an aircraft to conduct a mission, but one can schedule two aircraft to do so.

**Binary Integer Programming (BIP)**

A binary integer programs is a special case of an IP in which decision variables are equal to either 1 or 0, typically representing a “yes” or “no” decision. Within the context of aircraft scheduling, a BIP is a logical framework to consider; a value of 1 may represent whether an aircraft is scheduled to conduct (or begin) a given mission in a given week.

Teixeira, *et al*. [9] solved production scheduling problems by utilizing a binary integer programming formulation. The first decision variable defined is $a_{icfr}$, which represents the selected production orders for the scheduling cycle.

$$a_{icfr} = \begin{cases} 1 & order\ i\ is\ poured\ in\ load\ c\ of\ furnace\ f\ reactor\ r \\ 0 & otherwise \end{cases}$$

The second decision variable defined is $b_{l,c,f,r}$ which represents which metal alloy will be in each furnace in each reactor.

$$b_{lcfr} = \begin{cases} 1 & alloy\ l\ is\ cast\ in\ load\ c\ of\ furnace\ f\ reactor\ r \\ 0 & otherwise \end{cases}$$

This work provides a BIP formulation example in which all decision variables are binary-restricted.

**The Branch-and-Bound Algorithm**

Integer programs are commonly solved using the branch-and-bound algorithm [6]. This algorithmic procedure begins by relaxing all integer restrictions on decision variables and solving the corresponding math program having a continuous feasible region. For an IP, the relaxation yields an LP; for a minimization problem, and the value of the optimal objective function for the LP provides a lower bound on the optimal objective function value for the IP.

For this initial solution, which is identified as occurring at the *root node* in a *branch-and-bound tree*, if the LP relaxation yields an integer-valued solution, the algorithm terminates with an optimal solution to the original problem. Otherwise, the optimal solution to the LP relaxation yielded decimal-valued decision variable values. In this case, the algorithm *branches* by creating two (or more) subordinate nodes in the tree from a node for which the optimal solution to the LP relaxation did not yield an IP-feasible solution. Each of these *child nodes* corresponds to the previous IP formulation in the *parent node*, except that the IP formulation in each of those child nodes is augmented with a constraint that *bounds* one (or more) of the decision variables that was decimal-valued in the optimal solution to the parent node's LP relaxation. The bounding constraints prevent a decimal-valued solution for at least one of the decision variables from being feasible.

This procedure continues to grow the branch-and-bound tree while leveraging user-determined procedures to both select the next parent node (i.e., branch) and identify augmenting constraints (i.e., bounds). If a node other than the root node yields an optimal solution to its LP relaxation that is integer-feasible, no further branching occurs from that node; it is *fathomed*. Fathoming also occurs when the value of a node's LP-optimal objective function value is worse (or not $\epsilon$ better, for a user defined $\epsilon > 0$) than the objective function value for the *incumbent solution*, i.e., the best IP-feasible objective function value found so far.

Relevant to this research, the branch-and-bound algorithm is guaranteed to identify an optimal solution to an integer program [10], so any IP or BIP formulation for the 96th Test Wing's scheduling problem can be solved optimally. However, it may not find an optimal solution efficiently (e.g., see [11] and [12]). Practical implementation of this solution method may terminate it when it exceeds a certain run time. So long as the branch-and-bound tree upon termination has found an incumbent, IP-feasible solution, one can compute both an absolute and a relative optimality gap by comparing the IP-feasible objective function value, for which the solution is feasible to the original problem, with the best LP-optimal lower bound at an active node, which indicates a best IP-feasible objective function value that *might* be attainable. Of importance, a relative optimality gap of 5% merely indicates that an IP-feasible solution's objective function value is no more than 5% worse than the best LP-optimal objective function value upon termination of the algorithm. It is quite possible that the incumbent IP-feasible solution is optimal, and a continuation of the branch-and-bound algorithm might prove it by eventually fathoming all active branches without finding a better IP-feasible solution. Of note, a user-defined (absolute or relative) optimality gap may also serve as a termination criterion for the branch-and-bound algorithm.

This research will seek optimal scheduling solutions, but it will also impose a time constraint on a branch-and-bound solution procedure. If the solution procedure fails to identify an optimal solution within the allotted time, testing will report the relative optimality gap of the incumbent solution if an IP-feasible solution was identified. Otherwise, testing will indicate that no feasible solution was identified within the allotted computational time.

## 2.2 Scheduling

Scheduling is widely used by manufacturing and service industries, taking on different resources and tasks. At its core, scheduling deals with the allocation of limited resources to tasks over time, with the goal of optimizing one or more objectives [13]. This research seeks to identify a schedule of aircraft to missions that considers two objectives to maximize. First, the 96$^{th}$ Test Wing wants all aircraft and assigned pilots to be highly trained. Across the scheduling time horizon, each aircraft has a percentage of time available for training, and the wing seeks to maximize the smallest such percentage over all aircraft. Second, the 96$^{th}$ Test Wing wants an efficient use of training ranges, without large outliers. Across the aircraft, every week within the schedule has a percentage of aircraft available for training, and the wing seeks to maximize the minimum such value over all weeks in the schedule.

### Optimal Job Scheduling

The optimal job shop scheduling problem is the classical foundation for scheduling problems. Elmaghraby and Park [14] describe these scheduling problems, in which $n$ jobs are assigned to be completed on $m$ identical machines, and wherein each job has a set-up time, duration, and due date. Classical objectives include minimizing the make span of the set of jobs, minimizing the number of late jobs, minimizing the maximum lateness of any job, and maximizing the utilization of machines, given set-up times may depend upon the previous job completed on a given machine. In this framework, scheduling is the same as sequencing because a sequence defines the order for the set of jobs.

The optimal job shop scheduling problem is related to the 96th Test Wing's aircraft scheduling problem. The aircraft are like machines and the missions an aircraft performs are like jobs. However, the 96$^{th}$ Test Wing's scheduling problem differs from the aforementioned job shop scheduling problem in several key aspects. First, the wing's problem considers discrete intervals of time (i.e., weeks) in which tasks may (or must) begin, and, for the sake of developing a simple model, missions are assumed to require integer increments of weeks to complete. Second, the aircraft are not homogenous. Given the mission-driven nature of the 96$^{th}$ Test Wing's operations, we find it appropriate to embrace the alternative cognitive framework for scheduling: to schedule aircraft to accomplish time-affixed missions rather than assign missions to aircraft.

**Time and Resource-Constrained Scheduling**

Time-Constrained Scheduling (TCS) refers to a type of scheduling problem that is constrained by time, whereas Resource-Constrained Scheduling (RCS) refers to a type of scheduling problem that is constrained by the limited number of resources available [15]. The 96$^{th}$ Test Wing's scheduling problem entails a combination of TCS and RCS. As an application of TCS, any model must consider assignments that align aircraft to missions *and* the time in which they begin. Moreover, the 96$^{th}$ Test Wing likely cannot, e.g., schedule all aircraft maintenance to occur during the same week within a time horizon; the organization has naturally occurring resource limitations on some missions, resulting from maintenance, equipment, personnel, and other capacities. Therefore, as application of RCS, any model for the 96$^{th}$ Test Wing's scheduling problem must consider period-specific capacities that may limit the number of aircraft-to-mission assignments during any given period.

## 2.3 Aircraft Mission Scheduling

Desaulniers, *et al*. [16] discussed a daily aircraft routing and scheduling problem. The authors' research goal is to maximize profits from the assignment of flight paths to aircraft within a fleet, such that all flight legs are covered. Each aircraft type has various performance metric requirements related to originating airport, departure times, and expected profit. They examined two alternative math programming formulations to solve the problem: one based on set partitioning of the possible routes into flight paths and another based on a multicommodity network flow problem [16]. Among the various solution methods examined were a direct application of the branch-and-bound algorithm. Ultimately, the authors examined a routing and scheduling problem that was more complex than the 96th Test Wing's scheduling-only problem, but their exploration of the branch-and-bound algorithm indicates it should be the first approach considered to solve any integer programs within this research.

In fact, there is a plethora of aircraft scheduling literature that examines specialized topics. Tsai *et al*. [17] examined the problem of assigning aircraft to a time table for flights, and they leveraged genetic algorithms to solve the problem having binary decision variables in an effort to develop high quality solutions quickly. Samà *et al*. [18] studied the problem of both scheduling and, as necessary, rerouting aircraft to manage their assignments to busy airport terminals. Formulating a nonlinear-integer program, the authors examined various metaheuristic performances. Huo *et al*. [19] studied the assignment of arriving aircraft to terminals at the Paris Charles de Gaulle airport under conditions of uncertainty and with the desired outcome of a robust flight arrival schedule.

Ultimately, most of the literature related to aircraft scheduling differs from the problem-of-interest for the 96th Test Wing. Most authors examine assigning aircraft to routes or terminals, all of which have spatial or temporal aspects, as well as interdependencies that are not present in the problem considered herein. The 96th Test Wing's problem is much closer in nature to a hybrid TCS/RCS job shop scheduling problem. However, it is also worth reviewing related work conducted in support of US Air Force test wings. Two recent theses completed at the Air Force Institute of Technology considered related aircraft scheduling problems. Schoenbeck [20] examined a developmental test scheduling problem in support of the 412th Test Wing at Edwards AFB, and Macias [21] addressed a scheduling problem in support of the 57th Wing at Nellis AFB.

In his research, Schoenbeck [20] sought to schedule test aircraft to maximize the number of missions flown over a weekly period, decomposed into hours of each day. Similar to the research considered herein, he sought to replace a manual scheduling process that takes weeks with an automated process that, preferably, attains an optimal solution to a sponsor-informed objective function. The various demands on the aircraft include specific missions and sorties supporting each mission. Constraints on the problem were informed by both a limited number of aircraft, specific demands for types of aircraft (e.g., F-16s), and aircraft capabilities. Extended models considered aspects such as refueling and specific aircraft configurations. The scope of the problem addressed "more than 20 different Combined Task Forces requesting resources for roughly 300 flying missions each week" [20]. Figure 2 shows an example for defining mission-specific possibilities. The example mission shown has 29 tails, 1 sortie, and 115 time periods, which is equivalent to 3,335 entries. This combinatorial complexity motivates the need for identifying to utilize a powerful (i.e. commercial) solver to address integer programming formulations herein. To wit, Schoenbeck utilized the open-source solver Coin-OR Branch and Cut (CBC) to address one instance that sought to schedule 58 aircraft and involving approximately 40,000 binary decision variables in the formulation; the solver required approximately five hours to identify a feasible solution satisfying a 1% relative optimality gap termination criterion. Schoenbeck's research differs from the 96th Test Wing's problem in several ways. Foremost, Schoenbeck's model is granular at the sortie level; we instead schedule aircraft (i.e., 'tails') directly to missions. Second, Schoenbeck's study examined a scheduling specific to hours within each day; herein, the model considers weekly missions, and it seeks to do so over a longer time horizon (i.e., months rather than days). Third, because of the hourly focus of Schoenbeck's model, the previous work accounts for pre-flight preparation and post-flight maintenance. The 96th Test Wing's problem sets routine maintenance considerations aside by limiting the duration of flights each day and assuming maintenance can support post-flight operations within the remaining workdays, be they regular or adjusted hours.

| 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 2. Possibilities for Specific Missions [20]**

In his scheduling research, Macias [21] sought to automate a mission scheduling problem using an integer programming approach. Macias decided against using heuristics due to the degree of complexity and the instance sizes for the underlying problem. The goal of Macias' thesis was to save time with an automated scheduling approach as well as improve range scheduling efficiency. Macias formulated a problem to maximize the number of daily missions scheduled, subject to the following constraints: 1) Range resources are not used more than available, 2) Missions start within time frame, 3) All missions scheduled are active for the entire duration. Macias' tool produced daily mission schedules for a scheduling horizon of 1 month. The tool Macias built to schedule each day is created by user inputs, from each organization and in the form of Microsoft Excel workbooks. The integer programming formulation considered 24-hour days, partitioned into 15-minute increments. Macias modeled the problem by integrating the data in Microsoft Excel and, using Microsoft Visual Basic, passed the math programming formulation to MATLAB and invoked an MILP solver native to MATLAB. The run time limit was set at 4 minutes to find the best feasible schedule for a month's worth of missions. The majority of the daily schedules gave optimal solutions prior to the 4-minute time limit. Macias' research differs from the current study in several ways. Macias captured data from user inputs; we expect to input general criteria and let the tool develop the schedule itself, subject to mission constraints. Macias model is granular at the contents of the sortie (e.g., enemy or friendly air participants and ordnance involved in flight). Additionally, missions in his work can use more than one facility/range area.

### 2.4 Summary

This chapter reviewed past relevant research regarding aircraft mission scheduling. Based upon a review of literature related to math programming, integer and binary-integer programming, the branch-and-bound algorithm, multi-objective optimization, scheduling, job shop scheduling, aircraft scheduling, and USAF test aircraft scheduling, it is apparent that the nature of the 96$^{th}$ Test Wing's scheduling problem requires the development of a customized math programming model, which will be a binary integer program. Based upon precedence in the literature and lessons learned from previous AFIT thesis, it is also relevant to first consider the use of the branch-and-bound algorithm to solve the customized math program, and to do so via a commercial solver.

# III. Model Formulation and Solution Methodology

The 96th Test Wing's scheduling problem is solved by formulating an integer program and solving it via a commercial solver. If the solver identifies an optimal solution or a feasible solution within the time limit imposed upon the solver, a schedule is produced. The schedule determines when all missions, maintenance, temporary duty missions (TDYs), and testing/training will occur. In this chapter, we discuss the assumptions of the formulation, formulation of problem, as well as solution methodology.

## 3.1 Model Assumptions

Modeling assumptions attempt to simplify a complex process. Assumptions for this work include standardizing the mission length of each mission. Missions do not typically last the exact number of weeks each time the mission is performed. Moreover, we assumed that each maintenance would occur after an approximate number of weeks and not after a certain number of flight hours. This modeling assumption was adopted due to the intricacy of tracking flight hours within a scheduling model, given actual flight hours may differ from planned hours. All assumptions were approved by sponsor.

## 3.2 Model Formulation

We formulated this model to maximize two objectives: $\alpha$, the lowest percent of weeks that a tail (i.e., aircraft) is available for training; and $\beta$, the lowest percent of tails that any week has available for training. The mathematical program uses sets of missions, aircraft tails, and weeks. To formulate our model, we have defined a list of sets, parameters, and decision variables.

**Sets:**

- $M: \{I, O, D, \ldots, G, Other\}$ is the set of missions, indexed by $m$
  - $M_1: M \setminus \{O, D, G, Other\}$ is the subset of missions that are tail-specific, and for which the timing is generally flexible, but determined by a scheduled set of aggregate requirements (e.g., because the upgrades like radar may be performed at a fixed rate of "5 tails/week over 4 specific weeks when a contractor is available")
  - $M_2: \{O, D\}$ is the subset of missions that are tail-specific, but for which the start time may vary
  - $M_3: \{G_1 - 1\ week, G_2 - 2\ weeks, G_3 - 3\ weeks\}$ is the subset of missions that can be accomplished by <u>any</u> tail
  - $M_4: \{Other\}$ is the subset of missions where tails are not tasked out and are available for training
- $T: \{1,2, \ldots, |T|\}$ are the aircraft tails, indexed by $t$.
- $W: \{1,2, \ldots, |W|\}$ are the weeks, indexed by $w$, over which aircraft are scheduled to missions

**Parameters:**

- $\omega_\alpha$: a non-negative parameter in the weighted sum objective function, indicating the relative importance on aircraft-specific availability for training over weeks
- $\omega_\beta$: a non-negative parameter in the weighted sum objective function, indicating the relative importance on week-specific availability for training over aircraft
- $d_m$: the number of weeks to perform mission $m \in M$
- $a_{tm}$: a binary parameter equal to 1 if tail $t$ is capable of performing mission $m$, 0 otherwise
- $b_{tm}$: a binary parameter equal to 1 if tail $t$ must perform mission $m \in \{M_1 \cup M_2\}$ once over the time horizon, 0 otherwise
- $u_{mw}$: the maximum number of tails that can perform mission $m \in \{M_1 \cup M_2\}$ in week $w$, a parameter informed by, e.g., maintenance capacity
- $n_{mw}$: the number of tails required to perform mission $m \in M_3$ in week $w \in W$, a parameter informed by projected mission and exercise demands
- $\delta_m$: parameter indicates the number of weeks before and after the scheduled maintenance mission $m \in M_2$ can be performed
- $W_{tm}$: a binary parameter equal to 1 if tail $t$ must perform mission $m \in M_2$, 0 otherwise

**Decision Variables:**

- $x_{tmw}$: a binary decision variable equal to 1 if tail $t \in T$ begins mission $m \in M$ during week $w \in W$, 0 otherwise
- $\alpha$: the minimum aircraft-specific availability for training, over all weeks
- $\beta$: the minimum week-specific availability for training, over all aircraft

Given the framework above, we set forth the following formulation:

$$\text{maximize} \quad (\alpha, \beta)$$

$$\text{subject to:} \quad \alpha = \min_{t \in T, m \in M_4} \left\{ \frac{1}{|W|} \sum_{w \in W} x_{tmw} \right\} \tag{1}$$

$$\beta = \min_{w \in W, m \in M_4} \left\{ \frac{1}{|T|} \sum_{t \in T} x_{tmw} \right\} \tag{2}$$

$$\sum_{w \in W} x_{tmw} = b_{tm} \text{ , } \forall\ t \in T, m \in M_1 \tag{3}$$

$$\sum_{\widehat{w}=\max\{1, w-\delta_m\}}^{w+\delta_m} x_{tm\widehat{w}} \geq 1 \text{ , } \forall\ t \in T, m \in M_2, w \in W_{tm} \tag{4}$$

$$\sum_{\widehat{w}=\max\{1, w-\delta_m\}}^{w+\delta_m} x_{tm\widehat{w}} \geq 0 \text{ , } \forall\ t \in T, m \in M_2, w \in W \backslash W_{tm} \tag{5}$$

$$\sum_{w \in W} x_{tmw} = |W_{tm}| \text{ , } \forall\ t \in T, m \in M_2 \tag{6}$$

$$\sum_{t \in T} \sum_{\widehat{w}=\max\{1, (w+1-d_m)\}}^{w} x_{tm\widehat{w}} \leq u_{mw} \text{ , } \forall\ m \in \{M_1 \cup M_2\}, w \in W \tag{7}$$

$$\sum_{t \in T} x_{tmw} = n_{mw} \text{ , } \forall\ m \in M_3, w \in W \tag{8}$$

$$\sum_{m \in M} \sum_{\widehat{w}=\max\{1, (w+1-d_m)\}}^{w} x_{tm\widehat{w}} = 1 \text{ , } \forall\ t \in T, w \in W \tag{9}$$

$$x_{tmw} \leq a_{tm} \text{ , } \forall\ t \in T, m \in M, w \in W \tag{10}$$

$$x_{tmw} \in \{0,1\}, \forall\ t \in T, m \in M, w \in W \tag{11}$$

This formulation is a multi-objective, nonlinear, binary integer math programming formulation, with the nonlinearities manifest in Constraints (1) and (2). The objective function maximizes a multi-objective formulation comprised of two objective functions: the lowest percent of weeks available and the lowest percent of tails available to compel fairness across tail-specific training availability over the time horizon and week-specific training availability of tails. Constraints (1) and (2) calculate the respective objective function terms. Constraint (3) ensures each tail $t$ conducts $b_{tm}$ missions of type $m \in M_1$ during the time horizon, if required. Constraint (4) ensures mission type ($M_2$) starts the correct number of times within a customer-specified time window (i.e., up to -$\delta_m$ weeks beforehand), imposed for weeks $w \in W_{tm}$ when that mission $m \in M_2$ must be completed. Constraint (5) adopts an identical form to Constraint (4) for the remaining weeks, and it can be omitted from most modeling environments (e.g., GAMS, AMPL, Python/Pyomo) as a redundant constraint set; we present it here and retain it as an artifact of implementing this model in Excel, an aspect of the solution method that will be discussed in Section 3.4. Constraint (6) ensures mission type ($M_2$) is scheduled the correct number of times throughout the model's duration. Constraint (7) prevents the scheduling of missions of type $M_1$ and type $M_2$ from exceeding the mission capacity (e.g., due to available maintenance equipment or personnel) for any week. Constraint (8) ensures, for all $M_3$ type missions starting in a given week, the correct number of aircraft tails are assigned. Constraint (9) ensures each tail is assigned to a mission each week by examining whether a tail is assigned to start a mission in week $w$ or was previously assigned to start a mission $m$ that requires $d_m$ weeks, and the tail is still busy conducting that mission. If the tail is not performing a tail-specific mission ($m \in M_1$), maintenance ($m \in M_2$), or a TDY ($m \in M_3$), it will be available for testing/training (i.e., $m \in M_4$). Constraint (10) restrict assignment of tails to missions which an aircraft can perform, and Constraint (11) imposes binary restrictions on the decision variables.

Applying the Weighted Sum Method, the objective function in (1) is replaced by the following objective function for a fixed set of weights $(\omega_\alpha, \omega_\beta)$.

$$\omega_\alpha \alpha + \omega_\beta \beta$$

The weighted sum includes the decision variables $\alpha$ and $\beta$. This technique for multi-objective optimization allows the customer to place more importance on maximizing the minimum training availability of the tails, considering their respective schedules over all weeks ($\alpha$); or of week-specific training availability of aircraft, considering the week-specific schedules for all tails ($\beta$).

To enable the identification of an optimal solution by a commercial solver, Constraints (1) and (2) are linearized by replacing them with the following constraints.

$$\frac{1}{|W|}\sum_{w \in W} x_{tmw} \geq \alpha, \forall\, t \in T, m \in M_4$$

$$\frac{1}{|T|}\sum_{t \in T} x_{tmw} \geq \beta, \forall\, m \in M_4, w \in W$$

The modification to Constraint (1) ensures that the sum over the weeks is greater than or equal to $\alpha$, and the modification to Constraint (2) does likewise by ensuring the sum over tails is greater than or equal to $\beta$. These representations are equivalent when each constraint is binding for at least one combination of indices over which each constraint is applied (i.e., at optimality).

### 3.3 Solution Methodology

With these minor transformations, the resulting formulation is a binary integer program and can readily be solved via a commercial solver, subject to the practical limits on computational tractability as instances sizes grow.

Extended coordination with the 96th Test Group's leadership indicated a desire for the model to be implemented using Microsoft Excel to be more accessible to GS-1515 analysts who may not have a computer programming background. Such a modeling environment compelled certain modeling and solution methodology decisions for testing herein. With regard to modeling, implementation retained and imposed Constraint (5) because the combined imposition of Constraints (4) and (5) can be accomplished automatically within a spreadsheet modeling environment, whereas imposing Constraint (4) while omitting Constraint (5) would induce extensive, manual effort that undermines the point of speeding up the modeling and solution identification process.

With regard to solution methodology, it was obvious that Microsoft Excel's build-in Solver cannot address realistic sizes of the formulation. It is limited to 200 decision variables [22]. Rather than resort to a heuristic solution method, the 96$^{th}$ Test Group evaluated alternative commercial solvers that can be leveraged from Microsoft Excel via an add-in. In late 2021, they selected *What'sBest!* by LINDO Systems Inc. [23]. What'sBest! can address large optimization instances of integer programs via the branch-and-bound solution method, and it enhances that method with both customized and proprietary heuristics and range reduction techniques that that render it more capable than open-source software.

# IV. Analysis & Results

In this chapter, an analysis of the model's results is examined. Section 4.1 describes the computational testing environment. Section 4.2 validates the model for an illustrative instance. Section 4.2 explores the multi-objective nature of the formulation for the underlying instance. Section 4.3 examines a synthetic instance of 8-week duration having mission densities that are relatively comparable to an example instance provided by the 96$^{th}$ Test Wing. Subsequent testing in Section 4.4 considers an increasing durations of synthetic instances and, for each such duration, an increasing density of specified missions.

## 4.1 Testing Environment

All instances of the problem were solved from Microsoft Excel as a modeling environment and invoking What'sBest! version 17.0.1.2. A time limit of four hours (14,440 seconds) was imposed to What'sBest!. The justification for this time limit is the expected work schedule within the 96$^{th}$ Test Group; we seek to identify scheduling solutions within a reasonable amount of time, we consider "reasonable" to be approximately half of a workday. The relative optimality gap for What'sBest! was set at 0.01 (i.e., a 1% optimality gap), meaning we are satisfied with a solution that is assuredly within 1% of the true optimal objective function value. No absolute optimality gap was imposed. All instances were solved on a computer having an Intel Core i7-10510U CPU 2.30 GHz processer with 16 GB of random-access memory (RAM).

### 4.2 Validation Instance Introduction

The validation instance contains a schedule with 9 missions, utilizing 3 tails, over 8 weeks. The 9 missions are broken down into 4 mission sets as shown in Table 2.

**Sets for Validation Instance:**

- $M$={1,2,…,9}
- $T$={1,2,3}
- $W$={1,2,…,8}

**Table 2. Mission Details**

| Mission Type | |
|---|---|
| M1 - Tail specific and timing is flexible | |
| I | Mission I |
| O | Mission O |
| D | Mission D |
| M2 - Tail specific but start time may vary | |
| MA1 | Maintenance 1 Week |
| MA2 | Maintenance 2 Weeks |
| M3 - TDYs that can be accomplished by all tails | |
| G1 | Mission G - 1 Week |
| G2 | Mission G - 2 Weeks |
| G3 | Mission G - 3 Weeks |
| M4 - Aircraft is available for Training | |
| A | Training/Available |

For each of the 9 missions, there were 3 tails able to be scheduled across a time horizon of 8 weeks. If a mission is scheduled to start maintenance MA2 in Week 3 with Tail 1000, there would be a 1 in that position and not immediately after, even though the maintenance takes more than one week to complete. Figure 3 shows such a feasible assignment of three tails to accomplish maintenance mission MA2 (i.e., two-week-long maintenance). Tail 1000 starts mission MA2 in Week 3 and conducts it over Weeks 3 and 4, even though the latter week's activity is not shown explicitly in the decision variable values; it solely indicates when the mission *starts*, it does not indicate the *duration* of the mission.

| | Maintenance 2 Weeks | Week 1 | Week 2 | Week 3 | Week 4 | Week 5 | Week 6 | Week 7 | Week 8 |
|---|---|---|---|---|---|---|---|---|---|
| MA2 | Tail 1000 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| | Tail 1001 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| | Tail 1002 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |

**Figure 3. Example Schedule for a Two-week-long Mission**

### 4.2.1 Examining the Multi-objective Nature of the Problem

Of interest is whether the objective functions $\alpha$ and $\beta$ are in tension within the multi-objective framework, and the validation instances is examined to assess this characteristic. Leveraging the Weighted Sum Method, the validation instance was solved for $(w_{\alpha,}w_{\beta}) \in \{(0.01,0.99),(0.5,0.5),(0.99,0.01)\}$. Whereas $(w_{\alpha,}w_{\beta}) \in (0.01,0.99)$ and $(w_{\alpha,}w_{\beta}) \in (0.99,0.01)$ considered a preemptive priority on $\alpha$ and $\beta$, respectively, $(w_{\alpha,}w_{\beta}) \in (0.5,0.5)$ weighted these relatively well-scaled objective functions equally. We did not consider $(w_{\alpha,}w_{\beta}) \in (1,0)$ or $(w_{\alpha,}w_{\beta}) \in (0,1)$ because it is important to retain the multi-objective optimization framework when assessing the relative nature of the two objectives; to do so would ignore one objective and not discriminate among any alternative optima that may exist. Moreover, it is known that the problem is well-scaled, not because $\alpha \in [0,1]$ and $\beta \in [0,1]$, but because we expect the missions can be relatively well-balanced across both tails and weeks, so the optimal $\alpha$- and $\beta$-values should be close.

Table 3 provides the optimal objective function values $(\alpha^*, \beta^*)$ for the validation instances for each of these objective function weights. For each of the various weights used $(\omega_{\alpha,}\omega_{\beta}) \in \{(0.5,0.5),(0.99,0.01),(0.01,0.99)\}$, the solver identified an optimal schedule. Each of the validation instances were solved in less than 1 second.

**Table 3. Optimal Objective Function Values for Validation Instance for Varying Objective Function Weights**

| $(\boldsymbol{\omega_{\alpha,}\omega_{\beta}})$ | $\boldsymbol{\alpha^*}$ | $\boldsymbol{\beta^*}$ |
|---|---|---|
| $(\mathbf{0.01, 0.99})$ | 0.375 | 0.333 |
| $(\mathbf{0.5, 0.5})$ | 0.375 | 0.333 |
| $(\mathbf{0.99, 0.01})$ | 0.375 | 0.333 |

As is visible in Table 3, the values for values $(\alpha^*, \beta^*)$ do not vary with the different objective function weights $(\omega_{\alpha,}\omega_{\beta})$. The interpretation of the value for $\alpha^*$, indicates the lowest number of aircraft training is 37.5% of the time or 3 out of the 8 weeks. Similarly for $\beta^*$, indicating the lowest number of weeks training 33.3% of the time, which is 1 out of 3 tails. Based on this testing, the objective functions are not in tension. As such, we affix $(\omega_{\alpha,}\omega_{\beta}) = (0.5{,}0.5)$ for subsequent testing with Chapter 4.

However, it is worth noting that the actual aircraft schedules are not identical. Figures 4-6 respectively illustrate the schedule for the three instances corresponding to the objective function weights $(\omega_{\alpha,}\omega_{\beta}) = (0.01{,}0.99)$,$(0.5{,}0.5)$, and $(0.99{,}0.01)$. Of note, this graphic user output is not a function of What'sBest!. It is part of a graphic output designed to convert the optimal (or feasible) scheduling solution into a schedule similar to what analysts in the 96th Test Wing have utilized for years.

| | June 15th | | | | July 15th | | | |
|---|---|---|---|---|---|---|---|---|
| | Week 1 | Week 2 | Week 3 | Week 4 | Week 5 | Week 6 | Week 7 | Week 8 |
| Tail 1000 | A | O | MA1 | MA1 | MA1 | MA1 | A | A |
| Tail 1001 | I | MA1 | A | MA1 | A | MA1 | A | MA1 |
| Tail 1002 | D | A | MA1 | A | MA1 | A | MA1 | MA1 |

**Figure 4. Validation Instance with $\boldsymbol{\omega_{\alpha,}\omega_{\beta}}) = (\mathbf{0.01, 0.99})$**

| | June 15th | | | | July 15th | | | |
|---|---|---|---|---|---|---|---|---|
| | Week 1 | Week 2 | Week 3 | Week 4 | Week 5 | Week 6 | Week 7 | Week 8 |
| Tail 1000 | A | A | MA1 | MA1 | MA1 | MA1 | O | A |
| Tail 1001 | I | MA1 | A | MA1 | A | MA1 | A | MA1 |
| Tail 1002 | D | MA1 | A | A | MA1 | A | MA1 | MA1 |

**Figure 5. Validation Instance with $(\omega_{\alpha,}\omega_{\beta}) = (0.5, 0.5)$**

| | June 15th | | | | July 15th | | | |
|---|---|---|---|---|---|---|---|---|
| | Week 1 | Week 2 | Week 3 | Week 4 | Week 5 | Week 6 | Week 7 | Week 8 |
| Tail 1000 | A | A | MA1 | MA1 | MA1 | MA1 | O | A |
| Tail 1001 | I | MA1 | A | MA1 | A | MA1 | A | MA1 |
| Tail 1002 | D | A | MA1 | A | MA1 | A | MA1 | MA1 |

**Figure 6. Validation Instance with $(\omega_{\alpha,}\omega_{\beta}) = (0.99, 0.01)$**

Each of these schedules yields the same $\alpha$- and $\beta$-values, indicating the existence of alternative optima for each of them. Any of these schedules would be optimal for any of these objective function weights.

### 4.3 Realistic Sized Instance Testing

The 8-week validation instance was transitioned into the realistic-sized testing instance by adding the appropriate number of aircraft and missions to imitate the real-world schedule. The validation instance consisted of 3 tails and 9 missions. The realistic number of tails associated with the Eglin AFB scheduling problem includes 26 tails and 21 missions as seen in Figure 7.

**Sets for Full Instance:**

- $M=\{1,2,\ldots,21\}$
- $T=\{1,2,\ldots,26\}$
- $W=\{1,2,\ldots,78\}$

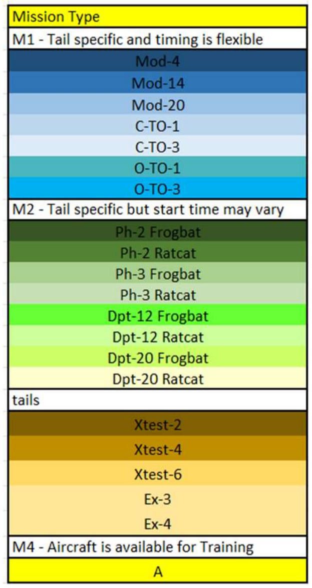


**Figure 7. Realistic Sized Instance Mission Breakdown**

### 4.3.1 Iron Flow's 8-Week Instance

The Iron Flow tool had scheduled 26 tails over an 8-week duration manually. The 8-week duration has 208 possible scheduling tail-weeks. During that 8-week duration, Iron Flow had 3 weeks of Instro Mod/Demod, which falls under Mission Type $M_1$. Type $M_1$ missions are tail-specific, but the time during which they are executed is generally flexible. $M_1$ represents 1.4% of the total number of missions in the example Iron Flow instance. The Iron Flow instance had 6 weeks of Scheduled Major Maintenance and 13 weeks of Phase Depot Maintenance, which both fall under Mission Type $M_2$. Type $M_2$ missions are tail-specific but start times may vary within some tolerance. $M_2$ represents 9.1% of the total number of missions in the Iron Flow instance. It also had 2 weeks of TDY missions, which fall under Mission Type $M_3$. Type $M_3$ missions are not tail-specific; the mission can be accomplished by any tail unless otherwise specified by setting $a_{\{tm\}} = 0$ to prevent such an assignment (e.g., if a TDY requires a certain type of aircraft or an aircraft with specific capabilities). $M_3$ missions represent 0.96% of the total number of missions in the example Iron Flow instance. Collectively, these specified missions leave 88.54% of tail-weeks for mission type $M_4$. $M_4$ type missions indicate the tails are not tasked out and they are available for training.

### 4.3.2 8-Week Instance Using Iron Flow's Mission Load

Testing considered a synthetic, 8-week instance having 2 missions of Type $M_1$ (i.e., Tail 1009 had to complete "Mod 14", and Tail 1025 had to complete "Mod 4"); 1 mission of Type $M_2$ (i.e., each of the 26 tails of the "Ratcat" airframe needed to perform maintenance in or close to Week 7); and 1 mission of Type $M_3$ (i.e., a two-week TDY beginning in Week 2). Relative to the example Iron Flow instance, these synthetic demands were relatively comparable. The synthetic instances contains 0.95% as Mission type $M_1$, 12.5% as mission type $M_2$, 0.48% as mission type $M_3$, and the remaining 86.07% is mission type $M_4$.

This instance entailed a combined workload of $M_1 \cup M_2 \cup M_3$ missions, comprising 13.93% of available tail-weeks. Based on this aggregate workload, the best possible values of $\alpha$ and $\beta$ would be 0.86. Table 4 reports the results from identifying the optimal solution to the 8-Week instance using What'sBest!, which it found in less than 1 second.

**Table 4. Results of all duration instance input**

| Decision Variables | Constraints | Iterations | Solution Time (Seconds) | $\alpha$ | $\beta$ |
|---|---|---|---|---|---|
| 4370 | 4895 | 214 | <1 | 0.75 | 0.81 |

In this case $\alpha$ was 0.11 units away from best possible value for $\alpha$, and $\beta$ was 0.06 units aways from the best possible value for $\beta$. However, some *post hoc* analysis indicates that the best attainable $\alpha$-value was indeed 0.75 because each tail's schedule is based on one-week increments, and the highest possible fraction of an 8-week schedule that is less than 0.86 is 0.75. Thus, this schedule is optimal for the minimum percentage of tail-specific training weeks. With regard to $\beta$, the attainable values are increments of $(1/26)$, the highest of which that does not exceed 0.86 is 0.85, followed by 0.81. The optimal solution did not attain a value of $\beta = 0.85$ because the maintenance window for all tails was identical, and the solution spread out those assignments, subject to the tolerance window for scheduling. Thus, this result is also verifiably optimal.

Most important is that the What'sBest! solver found an optimal solution to an 8-week instance of the underlying problem, with the right number of tails, mission types, and tail-weeks or mission types in 1 second. Should the 96th Test Wing seek to solve short-range scheduling problem instances, this model and solver combination works very well.

**4.4 Sensitivity Analysis**

Sensitivity analysis is applied to examine the tractability of scheduling instances over increasing time durations and increased (tail-week) workload. Relative to the validation instance examined in Section 4.2 and the 8-week instance examined in Section 4.3, testing within Section 4.4 will consider realistically sized instances to examine how the limits of computational tractability as both the time horizon and the density of specified missions increases. For time horizons of 8-, 26-, 52-, and 78-weeks, Section 4.4.1 defines comparable mission density levels across all such time horizons. The time horizons were decided due to the 8-week instance correlating to 2 months, which was a small enough model to validate. The 26-week instance correlates to half-a-year; the 52-week model correlates to a year; and the 78-week instance correlates to a year-and-a-half. Subsequent discussion presents and analyzes the results of examining increasing levels of mission density for each of these time horizons, and Section 4.4.2 summarizes the aggregate results.

### 4.4.1 Testing Increasing Mission Densities for Four Schedule Durations

To enable an equitable comparison of test results across instances having different durations, it is important to generate synthetic instances having relatively similar if not identical mission densities. For time horizons of 8-, 26-, 52-, and 78-weeks, Table 5 displays the percentage of mission density for mission types $M_1$, $M_2$, and $M_3$ and assigns common labels for the levels. The percentages shown indicate the total number of $M_1 \cup M_2 \cup M_3$ missions scheduled and, for simplicity, assumes $|M_1| = |M_2| = |M_3|$.

**Table 5. Mission Density Levels and $M_1 \cup M_2 \cup M_3$ Mission Densities**

| Level | 8-Week | 26-Week | 52-Week | 78-Week |
|---|---|---|---|---|
| 1 | 1.44% | 1.32% | 1.32% | 1.32% |
| 2 | 2.88% | 2.67% | 2.67% | 2.67% |
| 3 | 4.32% | 4.44% | 4.44% | 4.44% |
| 4 | 5.76% | 5.76% | 5.76% | 5.76% |
| 5 | 7.20% | 7.11% | * | 7.11% |
| 6 | 8.64% | 8.43% | | 8.43% |
| 7 | 10.11% | 10.20% | | * |
| 8 | 11.55% | 11.55% | | |
| 9 | 12.99% | * | | |
| 10 | 14.43% | | | |
| 11 | 15.87% | | | |
| 12 | 17.31% | | | |
| 13 | 18.75% | | | |

* Not tested due to tractability issues encountered; see subsequent discussion

The levels are roughly comparable. For time horizon-specific testing of the efficacy of our model and solver combination, the mission density level is incrementally increased until the solver cannot identify an optimal solution within a time limit of 4 hours (14,400 seconds). It is important to note each new level of mission density was run from a blank schedule (i.e., a cold start).

**8-week Instance Results**

The 8-week instance results are shown in Table 6. The mission density levels are increased by increments of 0.0144 each time, ensuring each level is 0.0144 larger than the previous. What'sBest! successfully finds an optimal solution – technically, a solution within 1% of optimal – for the first 12 levels, running out of time only on Level 13, which has a mission density of 18.75%. For Level 13, the solver exhibits a slow convergence and identifies a feasible solution to the problem with an optimality gap of 1.21%. As mentioned in Section 2.1, it is possible that this solution is indeed optimal, but we can only be assured that it is close to optimal.

**Table 6. Results of 8-week Instance with Increased Load**

| Level | Solution Time (Seconds) | Iterations | Objective Function Value | Solution Status | $\alpha$ | $\beta$ |
|---:|---:|---:|---:|---|---|---|
| 1 | 0 | 0 | 0.9183 | Optimal | 0.86 | 0.96 |
| 2 | 0 | 14 | 0.9183 | Optimal | 0.86 | 0.96 |
| 3 | 1 | 55,140 | 0.8365 | Optimal | 0.75 | 0.92 |
| 4 | 0 | 4,325 | 0.7440 | Optimal | 0.63 | 0.92 |
| 5 | 0 | 11,575 | 0.7440 | Optimal | 0.63 | 0.92 |
| 6 | 0 | 3,012 | 0.7440 | Optimal | 0.63 | 0.92 |
| 7 | 11 | 317,328 | 0.7548 | Optimal | 0.63 | 0.88 |
| 8 | 1 | 35,105 | 0.7548 | Optimal | 0.63 | 0.88 |
| 9 | 712 | 18,151,482 | 0.7548 | Optimal | 0.63 | 0.88 |
| 10 | 100 | 2,322,676 | 0.7356 | Optimal | 0.63 | 0.85 |
| 11 | 15 | 441,051 | 0.7356 | Optimal | 0.63 | 0.85 |
| 12 | 12 | 401,072 | 0.7356 | Optimal | 0.63 | 0.85 |

| Level | Solution Time (Seconds) | Iterations | Objective Function Value | Solution Status | $\alpha$ | $\beta$ |
|---|---|---|---|---|---|---|
| 13 | 14,400 | 123,821,054 | 0.7163 | Feasible | 0.63 | 0.81 |

An initial assumption would be, as the mission density increases, the number of iterations required by the branch-and-bound algorithm would also increase, but that is not the case. The number of iterations are not predictable. Iterations in What'sBest! indicate the number of branch-and-bound iterations used to solve an instance of the model. There is no identified correlation between the increase in mission density of the model and the increase in the number of iterations. While the solution time does not appear to be scalable to the number of iterations, it does increase as the number of iterations increase. At Level 12 we have a solution time of 12 seconds, at Level 13 the solution time abruptly goes to 4 hours. As expected, $\alpha$ and $\beta$ follow the trend of either decreasing or staying the same. This trend holds true even when the model reaches its time limit. The values of $\alpha$ and $\beta$ at Level 13 follow the same pattern as the $\alpha$ and $\beta$ values for optimal solutions. At Level 13, $\beta$ decreases, but $\beta$ is expected to degrade a bit due to the increased mission load, emulating the trend shown at previous levels. The trend of $\beta$ degrading by 0.03 or 0.04 leads us to believe the feasible solution at Level 13 may have been optimal, but that can only be ascertained for certain if the algorithm is allowed to run until the branch-and-bound tree's branches are all fathomed.

**26-week Instance**

The 26-week instance results are shown below in Table 7. The mission density levels are increased by increments of about 0.01 each time, ensuring each level is 0.01 larger than the previous. What'sBest! successfully finds an optimal solution for the first 7 levels, timing out on Level 8 with a mission density of 11.55%. The model reaches its run time limit of 4 hours at Level 8, identifying a feasible solution to the problem with an optimality gap of 5.28%.

**Table 7. Results of 26-week Instance with Increased Load**

| Level | Solution Time (Seconds) | Iterations | Objective Function Value | Solution Status | $\alpha$ | $\beta$ |
|---|---|---|---|---|---|---|
| 1 | 0 | 113 | 0.9231 | Optimal | 0.88 | 0.96 |
| 2 | 0 | 361 | 0.9038 | Optimal | 0.88 | 0.96 |
| 3 | 0 | 1637 | 0.9038 | Optimal | 0.88 | 0.92 |
| 4 | 1 | 27,333 | 0.8654 | Optimal | 0.84 | 0.88 |
| 5 | 14 | 230,522 | 0.8462 | Optimal | 0.84 | 0.84 |
| 6 | 1 | 8,445 | 0.8462 | Optimal | 0.84 | 0.84 |
| 7 | 2 | 18,131 | 0.8259 | Optimal | 0.84 | 0.80 |
| 8 | 14,400 | 143,628,986 | 0.7692 | Feasible | 0.81 | 0.73 |

There is still no identifiable correlation between the increase mission density of the model and the increasing number of branch-and-bound iterations, making the number of iterations unpredictable. Whereas the solution time does not appear to be scalable to the number of iterations, the number of iterations does increase when the solution time increases. This in turn means we also cannot predict the solution time. At Level 7 we have a solution time of 2 seconds, and at Level 8 the solution time abruptly goes to 4 hours. As expected, $\alpha$ and $\beta$ follow the trend of either decreasing or staying the same as mission density increases, an expected outcome. This trend holds, even when the model reaches its time limit. The values of $\alpha$ and $\beta$ at Level 8 follow the same pattern as the $\alpha$- and $\beta$- values for optimal solutions. That is, $\alpha$ and $\beta$ are lower at Level 8 than Level 7, but they are expected to degrade due to the increased mission load, as occurs at previous levels. The trend of $\alpha$ degrading by 0.03 or 0.04 maintains at Level 8, but $\beta$ has a trend of decreasing by 0.04, although at Level 8 it decreases by 0.07.

**52-week Instance**

The 52-week Instance results are shown below in Table 8. The levels are increased by increments of about 0.01 each time, ensuring each level is 0.01 bigger than the previous. What'sBest! identifies an optimal solution for Levels 1-3 but terminates due to the time limit on Level 4, identifying a feasible solution and a corresponding 3.33% relative optimality gap.

**Table 8. Results of 52 Week Instance with Increased Load**

| Level | Solution Time (Seconds) | Iterations | Objective Function Value | Solution Status | $\alpha$ | $\beta$ |
|---|---|---|---|---|---|---|
| 1 | 1 | 906 | 0.9327 | Optimal | 0.94 | 0.92 |
| 2 | 7 | 101,454 | 0.9135 | Optimal | 0.94 | 0.88 |
| 3 | 2 | 8,325 | 0.8846 | Optimal | 0.92 | 0.84 |
| 4 | 14,400 | 67,715,755 | 0.8365 | Feasible | 0.90 | 0.77 |

As with previous schedule durations, the number of branch-and-bound iterations required by What'sBest! to solve increasing mission density levels are not predictable. There remains no identified correlation between the increase mission density of the model and the increasing number of iterations, making the number of iterations unpredictable. As before, the solution time is roughly correlated to the number of iterations, but not predictable by mission density level. At Level 3 we have a solution time of 2 seconds, which goes up abruptly to 4 hours at Level 4. As expected, $\alpha$ and $\beta$ follow the trend of either decreasing or staying the same, even when the model reaches its time limit for Level 4. The values of $\alpha$ and $\beta$ at Level 8 follow the same pattern as the $\alpha$-

and $\beta$-values for optimal solutions. At Level 4 both $\alpha$ and $\beta$ gets worse, but they are expected to degrade a bit due to the increased load and follow the trend shown at previous levels. The trend of $\alpha$ degrading by 0.02 maintains at Level 4, but $\beta$ has a trend of decreasing by 0.04 but at Level 4 it decreases by 0.07.

**78-week Instance**

The 78-week instance results are shown below in Table 9. As with the 26- and 52-week instances, the levels are increased by increments of about 0.01 each time, ensuring each level is 0.01 bigger than the previous. What'sBest! finds an optimal solution for Levels 1-5, terminating due to the time limit for Level 6. For the Level 6 instance with an 8.43% mission density, the solver identified a feasible solution and a relative optimality gap of 1.11%.

**Table 9. Results of 78-week Instance with Increased Load**

| Level | Solution Time (Seconds) | Iterations | Objective Function Value | Solution Status | $\alpha$ | $\beta$ |
|---|---|---|---|---|---|---|
| 1 | 2 | 1,823 | 0.9423 | Optimal | 0.96 | 0.92 |
| 2 | 4 | 11,765 | 0.8974 | Optimal | 0.94 | 0.84 |
| 3 | 128 | 862,405 | 0.8529 | Optimal | 0.94 | 0.77 |
| 4 | 101 | 628,706 | 0.8077 | Optimal | 0.92 | 0.69 |
| 5 | 101 | 280,011 | 0.7179 | Optimal | 0.92 | 0.62 |
| 6 | 14,400 | 34521833 | 0.7179 | Feasible | 0.90 | 0.54 |

Both the number of branch-and-bound iterations and the required solution time remain unpredictable. At Level 5 we have a solution time of 101 seconds, at Level 6 the solution time abruptly goes to 4 hours. Both $\alpha$ and $\beta$ continue to follow the trend of either decreasing or staying the same as the level increases, for all optimal solutions. At Level 6 both $\alpha$ and $\beta$ gets worse, but they are expected to degrade a bit due to the increased load and follow the trend shown at previous levels. At level 6 both $\alpha$ and $\beta$

keep their trend: $\alpha$ continues to degrade by 0.02 and $\beta$ continues to decrease by 0.07 or 0.08.

### 4.4.2 Aggregate Results

From the testing results in Section 4.4.1, the model and solver combination encounter practical computational tractability limitations for lower levels of mission densities as the time horizon for the schedule increases, with a notable exception. A natural assumption would be that if the model hits the time limit at a certain level, it will hit the time limit at every increasing level. Our 8-week instance reaches the time limit at Level 13, the 26-week instance reaches it at Level 8, the 52-week instance reaches it at Level 4, and 78-week instance reaches it at Level 6. Our 78-week model does not follow the natural assumption of reaching the time limit before Level 4. This result is unexpected and requires further scrutiny.

More specifically, it is important not to make premature conclusions, so this section will analyze increasing levels of mission density for the 52- and 78-week schedules. Testing will examine increasing mission density levels to ascertain whether What'sBest! can address greater levels of mission density, doing so until the next notable level is identified for which What'sBest! cannot find an optimal solution within 4 hours. Table 10 reports the solution times in seconds for increasing levels of mission density for each of the time horizons considered. When What'sBest! does not identify an optimal (i.e., within 1% of optimal) solution, the identification of a feasible solution is indicated with a number of asterisks (*) and with the relative optimality gap upon termination indicate in the footnote of the table, whereas a failure to identify a feasible solution after four hours is identified with a dagger (†).

**Table 10. Aggregate Run Times (seconds)**

| | Level | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Weeks** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** |
| 8 | 0 | 0 | 1 | 0 | 0 | 0 | 11 | 1 | 712 | 100 | 15 | 12 | * |
| 26 | 0 | 0 | 0 | 1 | 14 | 1 | 2 | ** | -- | -- | -- | -- | -- |
| 52 | 1 | 7 | 2 | *** | **** | 2 | 3 | **** | -- | -- | -- | -- | -- |
| 78 | 2 | 4 | 128 | 101 | 101 | ***** | 37 | † | -- | -- | -- | -- | -- |

* 1.21% optimality gap upon termination at 14,400 seconds
** 5.28% optimality gap upon termination at 14,400 seconds
*** 3.33% optimality gap upon termination at 14,400 seconds
**** 2.12% optimality gap upon termination at 14,400 seconds
***** 1.11% optimality gap upon termination at 14,400 seconds
† No feasible solution identified upon termination at 14,400 seconds

For the 52-week model, we hit the time limit first at Level 4 which gives us a feasible solution with an optimality gap of 3.33%. At Level 5 it also reaches the time limit and gives us a feasible solution with an optimality gap of 2.12%. At Levels 6 and 7, it does not hit the time limit; in fact, the run time only lasts a couple of seconds, and What'sBest! identifies an optimal solution for both instances. At Level 8, the model and solver combination once again hit the time limit and identify a feasible solution, this time with an optimality gap of 2.12%. There is no discernible pattern to these results, once again exhibiting that the number of iterations is not predictable by mission level. Similarly, with the 78-week model, Levels 1 through 5 produce low solution times with optimal solutions. The first level to reach the time limit is Level 6 which produces a feasible solution with an optimality gap of 1.11%. Level 7 produces a fast solution time as well as an optimal solution. Finally, Level 8 was unable to find even a feasible solution during the time limit.

The results show that the levels at which the model and solver combination terminate prematurely due to time limits are not deterministic. If we reach a time limit at a certain level, it does not mean higher levels of mission density will reach the limit as well. Because solution time with an increasing mission load is not predictable, the ability to find an optimal solution is also difficult to predict.

**Aggregate $\alpha$- and $\beta$-values**

On a final note, it is worth examining any trends in the optimal $\alpha$ and $\beta$ values for the comparable mission density levels across different schedule lengths. Table 11 reports these values for Levels 1-4, for which What'sBest! identified optimal solutions.

**Table 11. Aggregate $\alpha$- and $\beta$-values**

| | **8-Week** | | **26-Week** | | **52-Week** | | **78-Week** | |
|---|---|---|---|---|---|---|---|---|
| **Level** | $\alpha$ | $\beta$ | $\alpha$ | $\beta$ | $\alpha$ | $\beta$ | $\alpha$ | $\beta$ |
| 1 | 0.86 | 0.96 | 0.88 | 0.96 | 0.94 | 0.92 | 0.96 | 0.92 |
| 2 | 0.86 | 0.96 | 0.88 | 0.96 | 0.94 | 0.88 | 0.94 | 0.84 |
| 3 | 0.75 | 0.92 | 0.88 | 0.92 | 0.92 | 0.84 | 0.94 | 0.77 |
| 4 | 0.63 | 0.92 | 0.84 | 0.88 | 0.90 | 0.77 | 0.92 | 0.69 |

Of course, as the level of mission density increases for a given duration of a schedule, both $\alpha$ and $\beta$ stay the same or decrease. The optimal values of $\alpha$ increase as the model increases in duration. This occurs because the granularity of $\alpha$ values is higher; it can occur in $(1/78)$ increments for the longest scheduled considered, rather than $(1/8)$ increments for the shortest duration schedule. In practical terms, What'sBest! can "spread out" the demands better across the aircraft.

Conversely, the values of $\beta$ stay the same or decrease as the duration of the schedule increases for comparable mission densities. This result is counterintuitive. Although $\beta$-values will occur in increments of $(1/26)$ for instances having 26 aircraft to schedule, the increase in the time horizon for a given mission density is accompanied by a proportional increase in the number of weeks in which to schedule the missions. A *post* *hoc* examination of the optimal solutions identified the reason for this unexpected behavior. Although Type $M_1$ missions can be spread out as necessary to maximize $\beta$, that is not the case for Type $M_2$ and $M_3$ missions. The periodic nature of $M_2$ missions can induce surges in aircraft demand (i.e., drops in training availability) that can be compounded with concurrent TDY missions. The significance of this result is that period maintenance demands should be spread across the calendar to reduce the potential for simultaneous or near-simultaneous demands.

### 4.5 Summary

In this chapter, performance of the combination of the math programming model and the What'sBest! solver was analyzed by several measures. Through testing the multi-objective nature of the problem by adjusting the weights associated with $\alpha$ and $\beta$, we were able to show there is no tension between the objective functions in a representative example of the underlying problem. When increasing the time horizon as well as the mission density, it is clear the run time is not deterministic. This in turn makes it difficult to predict solution time. The increase in mission density for a given duration also caused both $\alpha$ and $\beta$ to typically stay the same or decrease, as expected. The optimal values of $\alpha$ increase as the model increases in duration whereas the values of $\beta$ stay the same or decrease. We determined the decrease in $\beta$ was due to the non-randomization of maintenance missions, forcing all tails to perform maintenance in a small number of weeks.

# V. Conclusions and Recommendations

This chapter summarizes the research within the context of the research questions and makes recommendations for future studies on mission scheduling.

## 5.1 Conclusions

A mathematical programming approach was successful for scheduling Eglin AFB aircraft while finding optimal solutions within acceptable tolerances and time limits. Due to the priorities of this problem, the weighted sum method was utilized to maximize a multi-objective formulation comprised of two objective functions: the lowest percent of weeks available and the lowest percent of tails available to compel fairness across tail-specific training availability over the time horizon and week-specific training availability of tails. The 52-week instance was able to find a nearly optimal (i.e., 1.1% relative optimality gap) schedule within approximately 1 hour for a mission density comparable to an example schedule provided by the 96$^{th}$ Test Wing. Overall, this model has the equivalent utility as Iron Flow, while both decreasing the number of personnel hours needed to create a schedule and producing an optimal schedule automatically.

A more specific examination is warranted. Section 1.3 stipulated three research questions to address the problem statement in support of the 96$^{th}$ Test Wing. This research addressed them as follows.

**Research Question 1: Can the 96th Test Wing's aircraft scheduling problem be modeled via a compact mathematical program that addresses nuanced, complicating formulation aspects (e.g., nonlinearities, binary or integer-restricted decision variables), with the goal of optimizing multiple desired outcomes?**

The 96$^{th}$ Test Wing aircraft scheduling problem was modeled via a compact mathematical program by utilizing a binary integer formulation. The solution method was able to address the complex formulation aspects with a goal of optimizing multiple desired outcomes. Section 3.2 displays the model's multi-objective, nonlinear, binary integer formulation as well as the reformulation into a binary integer problem, directly solvable using a solver. The objective function optimizes the multiple desired goals of the sponsor; it maximizes the minimum percentage of time that any tail is training and the minimum percentage of tails training for any week.

**Research Question 2: When using the proposed model in combination with both a modeling environment and commercial solver directed by the 96th Test Wing, what are the limits of computational tractability for the proposed mathematical programming formulation, in terms of density of mission demands and duration of a schedule?**

In accordance with direction from the 96$^{th}$ Test Wing, computational testing used the modeling environment Microsoft Excel and the commercial solver What'sBest! by Lindo Systems, Inc. The What'sBest! add-in solver for Excel can address large optimization instances of integer programs via the branch-and-bound solution method, and it enhances that method with both customized and proprietary heuristics and range reduction techniques that render it more capable than open-source software. Chapter IV analyzed different mission density loads and durations to determine the limits of computational tractability. The 52-week instance has the largest schedule duration for which What'sBest! can identify near-optimal instances having a mission density comparable to an 11.4% mission density in an example schedule provided by the 96$^{th}$

Test Wing using their Iron Flow tool. The limit of mission density for the 52-week schedule is 15.75%. The 78-week instance handles a much lower mission density, making it less useful for the 96th Test Wing. If a 78-week model is necessary, the 4-hour time limit may be removed in order to handle a larger amount of mission density.

**Research Question 3: Does the combination of the proposed model and demonstrated performance on realistically sized instances portend a practical tool for use by the 96th Test Wing that can be utilized within Microsoft Excel and identify optimal or near-optimal schedules without resorting to either manual or heuristically solution methods?**

The proposed model and demonstrated performance on realistically sized instances portend a practical tool for the use of the 96th Test Wing. The combination of model, modeling environment, and solver produce a working tool that identifies optimal or near-optimal solutions without the need to resort to either a heuristic or a manual approach.

## 5.2 Recommendations for Future Research

Further studies with Eglin's aircraft scheduling might investigate ways to reduce the solution time for difficult to solve instances. Possible ways to do this are to fix the mission type $M_2$, also known as maintenance-specific missions, to prevent surges in maintenance workload. This measure would reduce the largest mission load implemented in this research. It would also reduce the number of constraints and decision variables that slow down the model.

Additionally, the computational challenges indicated for larger instances motivate the future examination of a heuristic scheduling approach. Scheduling one tail at-a-time would be computationally efficient, although such a greedy approach could yield suboptimal solutions and it may unnecessarily favor whichever tail(s) are scheduled first. However, it should be fast and may not actually require a mathematical program, given sound scheduling rules to distribute missions across tails and weeks.

Another worthy exploration is a different model that considers an existing schedule and develops refinements to it based upon changes to the anticipated missions in the future. In doing so, the existing schedule would provide a warm start to a solver, and an additional objective would be to minimize the number of changes from the existing schedule.

## Bibliography


1. AF Operational Test and Evaluation Center, "A Brief History and Heritage of AFOTEC", 2020. [Online]. Available: https://www.eglin.af.mil.
2. Eglin Air Force Base, "96th Test Wing", 2022. [Online]. Available: https://www.eglin.af.mil.
3. Eglin Air Force Base, "F-15 Unit Tests New Game-Changing Processor", 2016. [Online]. Available: https://www.eglin.af.mil.
4. Skytrailer, "85th Test & Evaluation Squadron", 2022. [Online]. Available: https://skytrailer.nl/eagle-squadrons/85-test-evaluation-squadron/.
5. Hutto, Greg T. Wing Operations Analyst for 46 Test Wing USAF, Eglin Air Force Base. Personal interview. 10 October 2021.
6. Winston, Wayne W. and Goldberg, Jeffrey B., *Operations Research: Applications and Algorithms, Fourth Edition*. Curt Hinrichs, 2004.
7. Ehrgott, Matthias. "Multicriteria Optimization (2nd Edition)." *Springer International Publishing,* 2005.
8. *Integer Programming.* Massachusetts Institute of Technology. [Online]. Available: http://web.mit.edu/15.053/www/AMP-Chapter-09.pdf.
9. Teixeira, Rodolfo F, Faria Fernandes, Flavio C, and Pereira, Néocles A. "Binary Integer Programming Formulations for Scheduling in Market-Driven Foundries." *Elsevier: Computers and Industrial Engineering,* 59, 425-435, 2010.
10. Christofides, Nicos, Alvares-Valdes, R, Tamaritm J.M. "Project scheduling with resource constraints: A branch and bound approach." European journal of Operations Research. 29, 262-273, 1987

11. Hassel, Lisa M. “Investigation of a zero-one integer programming approach to automating the scheduling process at the USAF Test Pilot School." Master’s thesis, Air Force Institute of Technology, 1992.

12. Foster, Gary G. “Automating the weekly flight scheduling process at the USAF Test Pilot School.” Master’s thesis, Air Force Institute of Technology, 1992

13. Pinedo, Michael L. “*Scheduling: Theory, Algorithms, and Systems*.” Springer International Publishing, 2016.

14. Elmaghraby, Salah E. and Park, Sung H. “Scheduling Jobs on a Number of Identical Machines.” *AIIE Transactions*, 6:1, 1-13, 1974.

15. Cavalcante, Victor F, Cardonha, Carlos H, Herrmann, Ricardo G, “A resource constrainted project scheduling problem with bounded multitasking.” *IFAC Proceedings*, 46, 433-437, 2013.

16. Desaulniers, Guy, Desrosiers, Jacques, Dumas, Yvan, Solomon, Marius, and Soumis, Fracois. “Daily Aircraft Routing and Scheduling.” *Institute for Operations Research and Management Science (INFORMS).* 43(6), 841-855, 1997.

17. Tsai, Ming-Wen, Hong, Tzung-Pei, and Lin, Woo-Tsong. “A Two-Dimensional Genetic Algorithm and Its Application to Aircraft Scheduling Problem.” *Mathematical Problems in Engineering*, 2015.

18. Samà, Marcella, D’Ariano, Andrea, Corman, Francesco, and Pacciarelli, Dario. “Metaheuristics for efficient aircraft scheduling and re-routing at busy terminal control areas.” *Roma Tre University Department of Engineering*, 2017.

19. Huo, Ying, Delahaye, Daniel, Sbihi, Mohammed. “A probabilistic model-based optimization for aircraft scheduling in terminal area under uncertainty.” *Transportation Research Part C*, 2021.

20. Schoenbeck, Joseph E. “Developmental Test Scheduling Problem.” Master’s thesis, Air Force Institute of Technology, March 2018.

21. Macias, Miguel J. “Scheduling Tool for the Nevada Test and Training Range.” Master’s thesis, Air Force Institute of Technology, March 2018.

22. Jablonsky, Josef. “MS Excel based software support tools for decision problems with multiple criteria.” *Elsevier*, 12, 251-258, 2014.

23. LINDO Systems, Inc. “What’sBest! User’s Manual.” [Online]. Available: https://www.lindo.com.

| REPORT DOCUMENTATION PAGE | | | Form Approved OMB No. 0704-0188 |
|---|---|---|---|



| 1. REPORT DATE *(DD-MM-YYYY)* | 2. REPORT TYPE | 3. DATES COVERED *(From – To)* |
|---|---|---|
| 22-12-2022 | Master's Thesis | June 2021 - December 2022 |

| TITLE AND SUBTITLE | |
|---|---|
| Optimal Scheduling of Aircraft Test and Evaluation Fleets to Balance Availability for Testing and Training | 5a. CONTRACT NUMBER |
| | 5b. GRANT NUMBER |
| | 5c. PROGRAM ELEMENT NUMBER |
| **6. AUTHOR(S)** | 5d. PROJECT NUMBER FYDJON |
| Hoops, Sarah E., GS-11 | 5e. TASK NUMBER |
| | 5f. WORK UNIT NUMBER |

| 7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) | 8. PERFORMING ORGANIZATION REPORT NUMBER |
|---|---|
| Air Force Institute of Technology<br>Graduate School of Engineering and Management (AFIT/EN)<br>2950 Hobson Way, Building 640<br>WPAFB OH 45433-7765 | AFIT-ENS-MS-22-D-024 |

| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) | 10. SPONSOR/MONITOR'S ACRONYM(S) |
|---|---|
| 96th Test Wing<br>Amber Romeo<br>101 West D Ave<br>Eglin AFB, FL 32542<br>(937)-872-0747 amber.romeo.3@us.af.mil | 96th TW<br>**11. SPONSOR/MONITOR'S REPORT NUMBER(S)** |

**12. DISTRIBUTION/AVAILABILITY STATEMENT**
**DISTRUBTION STATEMENT A.** APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

**13. SUPPLEMENTARY NOTES**


**14. ABSTRACT**
The 96th Test Wing at Eglin Air Force Base manually schedules a fleet of approximately 26 aircraft to conduct a range of missions over a one-to-two year planning period. This study automates the scheduling process, does so in a manner that optimizes multiple planning goals related to aircraft availability for training, and provides the 96th Test Wing with a software tool for the implementation that can be used by operational analysts within the command. We formulate the scheduling problem as a multi-objective, nonlinear, binary integer math program that seeks to maximize both the lowest percent of time any aircraft is available for training and the lowest percent of aircraft available for training for any week. Applying the Weighted Sum Method for multi-objective optimization, a conversion of nonlinear operations yields a binary integer program that is directly solvable a via commercial solver. An examination of the multi-objective nature of the problem identified a lack of tension between the objectives, so empirical testing affixes equal weights to the well-scaled objective function components. The 96th Test Wing directed the use of Excel as a modeling environment and What'sBest! by Lindo Systems, Inc. for their analysts to use. Subsequent testing examined for increasing time horizons the ability of the model and solver combination to develop optimal or near-optimal schedules for increasing levels of mission densities. The required computational effort was not predictable by mission density level, but increasing levels did yield instances not solvable to optimality within four hours for each time horizon. Optimal solutions can be readily identified within a few seconds for the current fleet and current or higher mission densities for 8-, 26-, and 52-week schedules, using weekly granularity for mission scheduling. In contrast, 78-week schedules can only be solved for a lower mission density than currently addressed by the 96th Test Wing within a four-hour time limit. However, the 96th Test Wing's current process is conducted manually over a period of hours, and the fragility of a such a long-term schedule to evolving mission requirements makes its merit subjective. In aggregate, this study shows an integer programming approach is feasible and has utility to the 96th Test Wing organization for their scheduling problem.

**15. SUBJECT TERMS**
Optimization, Scheduling, Integer Programing

| 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. B. Lunday, AFIT/ENS |
| U | U | U | UU or SAR | 65 | 19b. TELEPHONE NUMBER *(Include area code)* (937) 255-6565 x4624; Brian.Lunday@afit.edu |

**Standard Form 298 (Rev. 8-98)**
Prescribed by ANSI Std. Z39-18