AFIT/GOR/ENS/91M-7

# INVESTIGATION OF A ZERO-ONE INTEGER PROGRAMMING APPROACH TO AUTOMATING THE SCHEDULING PROCESS AT THE USAF TEST PILOT SCHOOL

THESIS

Lisa M. Hassel, Captain, USAF

AFIT/GOR/ENS/91M-7

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No 0704-0188*



| 1. AGENCY USE ONLY (*Leave blank*) | 2. REPORT DATE | 3. REPORT TYPE AND DATES COVERED |
|---|---|---|
| | March 1991 | Master's Thesis |

4. TITLE AND SUBTITLE
INVESTIGATION OF A ZERO-ONE INTEGER PROGRAMMING APPROACH TO AUTOMATING THE SCHEDULING PROCESS AT THE USAF TEST PILOT SCHOOL

5. FUNDING NUMBERS

6. AUTHOR(S)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology, WPAFB OH 45433-6583

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GOR/ENS/91M-7

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

10. SPONSORING MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
Approved for publid release; distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

This study investigated models that could be used to determine a feasible schedule for the TPS training program and form the basis for an automated scheduling system. The TPS problem is formulated as a 0-1 integer linear program and is solved using preprocessing techniques and the Branch-and-Bound algorithm. The method produced feasible solutions for small problems that equate to portions of the TPS schedule but cannot efficiently solve large problems like the full TPS problem. More investigation should be conducted to alter the method developed in the study so it could be used for large problems. A promising approach might be to develop a specialized Branch-and-Bound algorithm that uses the preprocessing methods discussed in the study.

14. SUBJECT TERMS
resource-constrained scheduling, scheduling, integer linear programming

15. NUMBER OF PAGES
68

16. PRICE CODE

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| Unclassified | Unclassified | Unclassified | UL |

AFIT/GOR/ENS/91M-7

# INVESTIGATION OF A ZERO-ONE INTEGER PROGRAMMING APPROACH TO AUTOMATING THE SCHEDULING PROCESS AT THE USAF TEST PILOT SCHOOL

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Operations Research

Lisa M. Hassel, B.S.

Captain, USAF

March, 1991

THESIS APPROVAL

STUDENT: Captain Lisa Hassel CLASS: GOR-91M

THESIS TITLE: Investigation of a Zero-One Programming Approach to Automating the Scheduling Process at the USAF Test Pilot School

DEFENSE DATE: 22 February 1991

| COMMITTEE: | NAME/DEPARTMENT | SIGNATURE |
|---|---|---|
| Advisor | Dr James Chrissis (AFIT/ENS) | James W. Chrissis |
| Reader | Captain John Borsi (AFIT/ENS) | John J. Borsi |

## Preface

The United States Air Force Test Pilot School (TPS) seeks to automate their scheduling process. Current methods used to build the TPS schedule are manual and time-consuming. Also, after the initial schedule is developed, interruptions in the schedule occur that require the TPS to spend many hours trying to reschedule events. The purpose of this research was to investigate models that could be used to determine a feasible schedule for the TPS training program and form the basis for an automated scheduling system.

The primary model investigated involves binary (0-1) integer linear programming and the proposed solution method employs heuristic and standard techniques. It is the conclusion of this research that the method developed here could produce a feasible schedule for small problems that are representative of a portion of the TPS scheduling problem. However, the method cannot efficiently solve large problems like the full TPS problem.

This thesis recommends that more investigation be done to determine a way to allow the method developed here to efficiently solve the large scheduling problem. The method could then be used in an automated scheduling system for the TPS. Such a system would reduce the amount of time required for initial schedule development and rescheduling as well as track program progress and resource utilization.

The idea for this research came from Major Daniel Isbell, an

The idea for this research came from Major Daniel Isbell, an instructor of test management at the TPS. Major Isbell has displayed a general concern for improving the TPS scheduling process and has spent considerable time researching a solution to the problem. I would like to express my appreciation to him for his ingenuity, support, and enthusiasm. I would also like to thank my faculty advisor, Dr. James Chrissis, for his help and patience, and faculty member Captain John Borsi, for his enthusiasm, keen insight, and many helpful comments. Finally, I would especially like to thank Tom, for somehow always being there.

## Table of Contents

# List of Tables

## Abstract

This study investigated models that could be used to determine a feasible schedule for the TPS training program and form the basis for an automated scheduling system. Current methods for developing the TPS schedule are mostly manual and equate to mentally scheduling 1,000 events over 46 weeks using 9 different resources, while considering resource and activity constraints. This process is time-consuming, and makes rescheduling a nightmare.

A literature search revealed that the TPS scheduling problem belongs to the class of resource-constrained scheduling (RCS) problems. Among the methods available for modeling RCS problems is integer linear programming. This study formulates the TPS problem as a binary (0-1) integer linear program (BIP) and employs a solution method consisting of heuristics and the Branch-and-Bound technique.

The method developed in this study can produce a feasible schedule for small problems that equate to **portions** of the TPS schedule. However, it cannot efficiently solve large problems such as the **full** TPS problem because of the computational complexity of the 0-1 formulation. An efficient solution of the full TPS problem using the 0-1 formulation would at least require the development of a specialized Branch-and-Bound algorithm. Alternatively, a new model could be developed that exploits the network structure of the TPS problem and thus would solve the problem more efficiently. However, more work would still need to

be done to find ways to allcw the network model to handle the TPS resource constraints.

It is recommended that more investigation be conducted to alter the method developed in this study so it could be used to solve the TPS scheduling problem. The method could then form the basis for an automated TPS scheduling system that could save the TPS considerable time, resources, and frustration.

Investigation of a Zero-One Integer Programming Approach to Automating the Scheduling Process at the USAF Test Pilot School

# 1. Introduction

## 1.1 Background

The United States Air Force conducts flight testing to ensure that only operationally effective and suitable systems are delivered to the Air Force (7:1,6). In support of flight test operations, the Test Pilot School (TPS), located at Edwards AFB, trains technically competent flight test pilots, navigators, and engineers.

To successfully manage the TPS training program and meet all of the training objectives, the TPS develops a detailed training schedule before the start of each class. The term 'class' refers to a specific group of students who train together in the 46-week TPS program. Development of the training schedule is a complex, mostly manual, time-intensive task that can take weeks to complete initially, and many hours to revise when changes are needed. For this reason, the TPS desires an automated scheduling system that would aid the development of the initial schedule and subsequent revised schedules (12).

The current TPS schedule development process starts about five to six months before the entry of each class. At that time, the schedulers meet in a room equipped with what the TPS refers to as its "magnetic anomaly board", a wide magnetic board that shows a rough outline of the entire forty-six week curriculum at once. With a list of available resources on hand, the schedulers try to validate the sequencing of the schedule and match resources to needs. The end product of this meeting is a rough, weekly schedule, recorded manually. The specific daily schedules are then developed a few weeks in advance of their intended execution. The schedules must frequently be revised, however, due to interruptions such as flight physicals and other medical appointments, illness, annual standardization and evaluation inspections, and instructor training. Using current methods, it can take as long as sixteen hours to develop a new, feasible schedule (13).

Scheduling in general can be a very complex process because of its combinatorial nature (24:65). For each TPS class, for example, the scheduler must consider 500 events over 46 weeks using 18 different resources. The TPS scheduling process is additionally complicated because it has 1) specific precedence requirements (e.g., flight test techniques must precede each flying session); 2) priorities (e.g., academics should receive a higher priority in scheduling than flying because the academics are less flexible); and 3) various resource constraints. These resource constraints include a limited number of academic and flight instructors, each of whom can perform up to eleven different tasks -- thus special care must be taken not to over-task (i.e., expect the instructors to perform two distinct tasks at the same time) the resources.

As a step toward automating the TPS scheduling process, planners used spreadsheets such a QUATTRO PRO and TIMELINE to construct charts and graphs. The school has also surveyed commercially-available project management software that provide managers with traditional management aids such as bar charts, Gantt charts, and Program Evaluation Review Technique (PERT)/Critical Path Method (CPM). However, none of this software can adequately handle the TPS resource constraints. That is, even packages that allow for resource-constrained scheduling do not allow resources to be defined with multiple capabilities and, as a result, can over-task resources (12). As an illustration of how this can happen, consider the resources and their associated capabilities in Table 1.

TABLE 1

Illustration of Resource Conflicts

| Pilot Resource | Aircraft Capabilities A | B | C | D |
|---|---|---|---|---|
| Joe | X | X | | |
| Pete | | X | X | |
| Jim | X | | | X |

X --> the pilot can instruct in that aircraft

Table 1 indicates, for example, that Joe can instruct in aircraft A and B. In the available commercial packages, resource assignments are made by matching the capability directly to the need, not accounting for situations where two different tasks require the same resource, but for different purposes. For example, if task 1 requires experience in aircraft B and task 2 requires experience in aircraft C at the same time, then Pete could be erroneuously assigned to both tasks 1 and 2.

## 1.2 TPS Curriculum

Each TPS class consists of twenty-five students, including fifteen pilots, two or three navigators, and seven or eight engineers. Classes last for forty-six weeks, starting in both January and July and ending in December and June, respectively. Thus there is a six-month overlap period between the classes wherein the two classes share resources. During this overlap period, the older class is referred to as the "A" class and the newer class is the "B" class (25).

The TPS curriculum is divided into four distinct phases and each phase consists of an integrated academic and flying program. These phases include the following

I. Performance Phase

II. Flying Qualities Phase

III. Systems Phase

IV. Test Management Phase

Ideally, the four phases occur sequentially and within each phase occurs (in order) 1) academic theory; 2) flight test techniques (FTT); 3) flying; and 4) final reports.

## 1.3 Research Objective

The primary objective of this research is to develop a model that can be used to determine a feasible schedule for the TPS training program. Feasibility (i.e., elimination of resource conflicts) should be the model's objective rather than optimality (i.e., early graduation) in that a schedule that will allow the students to graduate on time is more desirable than one that provides for early graduation. The model developed should form the basis for an automated scheduling system that can be used for initial schedule development and rescheduling. To accomplish this objective, this research

1) Formulates the TPS scheduling problem as a mathematical programming model;
2) Selects an appropriate algorithm for solving the formulated problem;
3) Demonstrates the approach for a sample problem representative of the TPS schedule; and
4) Discusses the application of this process to the full TPS problem.

## 1.4 Overview

The thesis effort is described in detail in the remaining chapters. An overview of the literature that contributed to and motivated the model formulation is provided in Chapter 2. The primary model formulation and the solution algorithm investigated to solve the problem are described in Chapter 3. Chapter 3 also discusses sample problems solved using this formulation and solution algorithm. Finally, conclusions and recommendations for continuing the analysis of the TPS scheduling problem are presented in Chapter 4.

# 2. Literature Review

This chapter presents a formulation of the problem, as well as an overview of the literature that contributes to the formulation. This first section describes the resource-constrained scheduling problem in general. The second section discusses the techniques available to solve this type of scheduing problem. The chapter concludes with a simplified formulation of the TPS problem.

## 2.1 The Resource-Constrained Scheduling (RCS) Problem

Scheduling is "the allocation of resources over time to perform a collection of tasks." Thus vital elements in scheduling models include resources and tasks and, as a result, classification of the scheduling problem depends upon the configuration of the resources and the behavior of the tasks (1:26). The TPS scheduling problem can thus be regarded as one that possesses

- multiple resource types;
- single stage tasks;
- resources available in unit amounts;
- priority and precedence among the tasks;
- dynamic possibilities (new tasks can occur).

The TPS problem can be further described as:

- non-preemptive (tasks cannot be stopped once started);
- deterministic (task durations are certain and known).

In general, the TPS scheduling problem belongs to the class of RCS problems, scheduling situations where there are fixed levels of resources available. However, the TPS problem is slightly more complex than general RCS problems presented in the literature (20:1205; 23:94; 15:48) because most authors discuss problems in which the exact resources required by a task are known in advance. In the TPS problem, several resources could be used for a task and thus the problem becomes one of assignment as well as scheduling.

A common objective in solving this type of problem is to minimize project duration or "makespan", the length of time required to complete all tasks. This is achieved primarily by specific sequencing of the activities (1:299). Other objectives may exist, such as trying to determine least-cost schedules or optimal amounts of resources to procure (5:298); however, none of these latter objectives is applicable to the TPS scheduling problem. Hence, a model of the TPS problem must seek to minimize total project duration or rather, ensure that all tasks (TPS curriculum objectives) are completed by the designated graduation date.

In the literature, project scheduling is sometimes discussed as job shop scheduling, flow shop scheduling, or assembly line balancing (ALB) (1:6). However, this thesis refrains from describing the TPS schedule as a job shop, flow shop, or ALB type of schedule and does not use any of the associated terminology.

## 2.2 Solution Approaches for the RCS Problem

Solving the RCS problem amounts to answering two kinds of questions (1:5)

Sequencing: When will each task be performed?

Allocation: Which resources will be allocated to perform each task?

Among the earliest and simplest techniques used to schedule projects are bar charts, Gantt charts, milestone charts, and line balancing techniques. While useful for most simple project management functions, these techniques are not sufficient for solving the RCS problem because (3: 8-11,13; 1:53-56)

1) They become unmanageable and unreadable for large projects which may be comprised of many hundreds of interrelated activities;
2) They cannot depict the interdependencies of activities;
3) Their static scales make it difficult to reflect the dynamic nature of changing plans; and
4) They cannot adequately differentiate between critical and noncritical problem areas.

A more advanced method for solving the RCS problem involves formulating a model of the problem and then deriving a solution from this model using a specific algorithm, where the algorithm is preferably applied on a computer using any one of a number of commercially available software packages. A key issue in selecting an algorithm is its computational complexity, which can determine how practical the algorithm is (15:4). The next two subsections describe available modeling alternatives and solution algorithms for the RCS problem, and the third subsection discusses the computational complexity of the algorithms.

2.2.1 Model Formulation. Two general ways to model or formulate the RCS problem involve representation of the project as a network or as a linear program. A network model widely used in project management is the Program Evaluation Review Technique (PERT) or the Critical Path Method (CPM). PERT and CPM are very similar. The only difference between them is that in CPM, activity completion times are assumed to single, deterministic values, whereas in PERT, completion times are more uncertain, expressed in terms of lower and upper bounds (22:271). Much commercial software implementing PERT/CPM is available to efficiently solve the RCS/TPS problem. However, PERT/CPM use only activity time information, without consideration of resource requirements or availabilities (5:298). In fact, in the general area of networks, increasing attention has been given in recent years to the problem of resource allocation associated with project scheduling (22:296). Given the difficulty in defining the resource constraints in network models, this thesis will focus on linear programming as a method of modeling the TPS problem and will not discuss network approaches.

Relative to network formulations, linear programming uses a mathematical model to describe the problem in which all of the functions in the model are required to be linear. Linear programming involves the planning of activities (represented as variables in the formulation) to obtain an optimal result, i.e., a result that reaches a specified goal best among all feasible alternatives (11:24). The goal is expressed in an objective function and feasibility is defined by constraints.

A special type of linear program that can be used to model the RCS problem is the integer linear program (IP), in which all of the variables are restricted to integer values. A variant of the IP is the binary integer linear program (BIP), in which all variables are restricted to binary (or 0-1) values. As an illustration of how to represent a project as a BIP, consider the simple problem of scheduling four tasks over four days, where

1) the objective is to minimize total project duration;
2) each task is completed only once;
3) each task can be completed in one day;
4) no resource constraints are considered;
5) tasks 3 and 1 must precede tasks 1 and 2, respectively; and
6) task 4 is the terminal activity.

Although this problem is small enough that it can be solved by inspection (schedule tasks 3, 1, 2, and 4 on days one through four, respectively), the BIP model for illustrative purposes is

Let $x_{jt}$ = 1, if task j is completed during time period t
0, otherwise,

Minimize $1x_{41} + 2x_{42} + 3x_{43} + 4x_{44}$

Subject to
1) $x_{11} + x_{12} + x_{13} + x_{14} = 1$
2) $x_{21} + x_{22} + x_{23} + x_{24} = 1$
3) $x_{31} + x_{32} + x_{33} + x_{34} = 1$
4) $x_{41} + x_{42} + x_{43} + x_{44} = 1$
5) $1x_{31} + 2x_{32} + 3x_{33} + 4x_{34} + 1 <= 1x_{11} + 2x_{12} + 3x_{13} + 4x_{14}$
6) $1x_{11} + 2x_{12} + 3x_{13} + 4x_{14} + 1 <= 1x_{21} + 2x_{22} + 3x_{23} + 4x_{24}$
7) $x_{jt} = 0$ or 1

The first four constraints ensure each task is completed only once. The fifth and sixth constraints represent the required precedence relationships.

2.2.2 Solution Algorithms. The general procedure for solving LP problems is the simplex method, "an algebraic procedure where each iteration involves solving a system of equations to obtain a new trial solution for an optimality test" (11:53). It is a very efficient algorithm for solving large problems. A detailed description of the algorithm is available in almost any elementary management science text.

Integer programming (IP) problems can be more difficult to solve than LP problems because the amount of computational work can grow exponentially as the number of variables increases and, in contrast to LP problems, the number of variables becomes more important than the number of constraints. In fact, "developing IP algorithms continues to an active area of research." A common technique used in IP algorithms is a technique called LP-relaxation, where the original problem is considered except that the integer restriction is deleted. The solution obtained is then rounded to the nearest integral value(s). However, pitfalls associated with this approach are that the rounded solution may not be optimal or even feasible (11:486-489). Certain IP algorithms are often used to avoid these pitfalls, including (10:455-456,473) cutting plane algorithms: Iterative approaches that ignore the integrality constraints but which add constraints to the problem formulation which exclude non-integer extreme points.

implicit enumeration and branch-and-bound algorithms: These algorithms conduct an exhaustive search of all possible integer solutions but in such a way that it is often not necessary to explicitly consider every integer solution to ensure that optimality has been found.

Each of the implicit enumeration procedures begins the process by solving only part of the original problem through relaxation of certain of the constraints or temporarily ignoring certain imposed restrictions. As fewer and fewer of the restrictions are ignored, a tree of partial solutions is generated. The methods differ in how the tree of partial solutions is generated and in the manner in which inferior solutions are recognized and discarded (21:855). Patterson compares these three approaches and concluded that the Branch-and-Bound technique produced a solution in the minimum amount of computation time. In fact, the basic IP algorithm used by commercial software is a Branch-and-Bound algorithm with LP-relaxation. A summary of the Branch-and-Bound algorithm can be found in most management science textbooks.

2.2.3 Computational Complexity. A necessary consideration when formulating and solving the RCS problem is the computational complexity of the procedure.

> Practical experience makes it clear that some computational problems are easier to solve than others. For some scheduling problems, algorithms have been known for decades that are capable of solving instances of thousands of jobs whereas for other problems, the best algorithms strain to cope with only a handful of jobs (15:4).

As defined and discussed by Lawler, an area known as complexity theory provides a mathematical framework in which computational problems can be classified as hard or easy. According to this theory, a computational problem is viewed as a function **f** that maps each input **x** in some domain to an output **f(x)** in some given range. Suppose that **n** is some measure of the size of the input problem. Then, letting **T(n)** represent an upper bound on the number of steps the algorithm takes on any input **x**, a problem is classified as "easy" if **T(n)** is bounded by a polynomial function of **n**. Typically, **P** is used to refer this class of so-called easy problems and **NP**-complete is a term used to refer to problems for which a polynomial-time algorithm is unlikely to exist (i.e., "hard" problems) (15:5-6).

The general integer programming problem is included in the class of NP-complete problems, and thus is computationally intractable for large size problems. As the number of variables in the problem increases linearly, the amount of computational time needed for solution can increase exponentially" (2).

2.2.3.1 Problem Size. For BIP problems, a procedure known as preprocessing can be useful in reducing the number of variables and constraints, thus enhancing the chances for an efficient solution. "Given a formulation, preprocessing refers to elementary operations that can be performed to improve or simplify the formulation by tightening bounds on variables, fixing values, and so on." Preprocessing can be thought of as a phase between model formulation and solution whose main purpose is to prepare a formulation quickly and automatically for a more sophistocated algorithm (19:17-18,456).

One of the most recent breakthroughs in IP methods that can be used in preprocessing is an algorithmic approach presented by Johnson, Kostreva, and Suhl. According to one technique

> if setting a variable $\mathbf{x}_j$ to zero (or one) causes a constraint to become obviously infeasible, then $\mathbf{x}_j$ can be permanently fixed to one (or zero). For example, the constraint
>
> $$3\mathbf{x}_{12} - 4\mathbf{x}_{18} + 8\mathbf{x}_{23} <= 2$$
>
> cannot be satisfied in 0-1 values if $\mathbf{x}_{23} = 1$. Hence, $\mathbf{x}_{23}$ can be fixed to zero.

According to another similar technique, known as Spielberg's probing, a variable $\mathbf{x}_j$ is set either to zero or one. If the problem is infeasible, then the original variable can be fixed in the other direction (14:803).

2.2.3.2 Efficiency of the Algorithm. If the problem size is small enough (under a few hundered variables in the BIP formulation), then optimal procedures such as the Branch-and-Bound algorithm (as used by STORM and most other commercial IP solvers) can solve the problem in a reasonable amount of time. However, when the problem size is larger, these procedures can become computationally intractable.

The complexity of the optimal approaches to solving IP problems has motivated many authors to seek polynomial time methods, known generally as heuristics, which may generate suboptimal or infeasible solutions. Among the heuristics reported for scheduling problems, those that are applicable to the RCS problem include (1:286; 20:1203-1207)

> critical path based methods: The critical path is utilized to prioritize the operation of jobs;
>
> priority rule methods: Activities are scheduled first based upon some aspect of those activities. Factors that can be used to assign priority include (16:17)
>
> > number of successor tasks;
> > number of predecessor tasks;
> > task execution time;
> > earliest execution time;
> > input resource requirement; and
> > output resource requirement; and
>
> combination optimal/suboptimal methods: Heuristics are embedded in optimization methods. For example, heuristics have been used successfully in the search phase of the Branch-and-Bound technique.

In general, is is not possible to tell in advance which heuristic will produce the best results. It is often most useful to use a computer to experiment with several different heuristics to select the best one (22: 323).

## 3. Model Formulation and Solution Methodology

The RCS problem is recognized as frustrating in that (6:285-286)

1) It is fairly easy to state and to visualize; and
2) The substantial literature on the subject contains any number of sophistocated and clever optimum-seeking schemes, yet are not always computationally practical.

This chapter presents a BIP formulation of the TPS problem and outlines a solution procedure that employes preprocessing (heuristics) and the Branch-and-Bound technique.

A BIP formulation of the TPS problem was chosen over a network formulation because

1) In the literature, the most common formulation of the RCS problem was as an LP (17:560; 20:1206; 21:856). A relatively simple and viable way to represent the TPS problem is as a BIP. BIPs are "useful in formulating the problem in a rigorous manner and they also supply theoretical insight to develop other approaches..." (20:1206).

2) It is easier to formulate the resource constraints in a BIP model rather than in a network model (2; 22:296).

Since BIP formulations can become computationally intractable due to their size, preprocessing is applied to reduce the size of the problem. The solution algorithm employed is the Branch-and-Bound technique because many (smaller) IP problems have been solved efficiently using this algorithm (Patterson, 866) and it is the method most commonly used in most commercial software to solve IP problems. To demonstrate this solution method, sample problems representative of portions of the TPS schedule are solved in the last section.

### 3.1 Model Formulation

The objective function and constraints represented in this section are patterned after an RCS problem formulation used by Pritsker, Watters, and Wolfe (23:94). In particular, the definition of the decision variables and the first two constraint types are similar to representations used in the Pritsker formulation.

The objective function and constraints described in the following sections represent a simplified formulation of the TPS problem in that

1) the overlap between classes is ignored, modeling for now only one class;
2) priority to academic versus flying activities is not explicitly given;
3) the limit of 15 class hours and 50 sorties per week is relaxed;
4) rescheduling (the chance that new activities will be introduced) is not directly addressed;
5) resource-leveling is ignored, whereas the desired solution should prevent the over- or under-utilization of the students and instructors; and
6) flying requires only one instructor pilot resource per period, whereas in the real TPS schedule, several flights can be conducted at the same time, requiring several instructor pilots per period.

It is assumed that this simplified version of the TPS schedule can still represent the essence of the TPS resource constraints so the key hurdle to overcome is finding a feasible schedule. These factors were only excluded here to help keep the model simple in an initial demonstration. To include these factors in a later model, more constraints would need to be added to the model and would not increase the number of variables.

3.1.1 Objective Function. As discussed in the literature review, the objective in solving the TPS problem is to minimize total project duration (or rather, meet the minimim graduation date). This is reflected in the objective function defined below where the terminal activity represents the last event required for completion before graduation.

The problem consists of **n** activities that need to be scheduled during **H** time periods, consuming **r** resources. Let

- **j** = an activity to be scheduled (**j** = 1,...,**n**);
- **t** = a time period (**t** = 1,...,**H**);
- **k** = a resource (**k** = 1,...,**r**);
- $\mathbf{m}_j$ = time required to complete activity **j**; and
- $\mathbf{a}_k$ = amount available of resource **k**.

**Define:**

$\mathbf{x}_{jkt}$ = 1, if activity j **completes** in time period t using resource k
0, otherwise.

**Minimize:**

$\Sigma_{t=1}^{H} \Sigma_{k=1}^{r} tx_{jkt}$ , where **n** is the terminal activity and task n can use resource k.

3.1.2 <u>Constraints</u>. A special constraint imposed on the problem is that $\mathbf{x}_{jkt} = 0$ whenever resource **k** does not possess the capability to perform activity **j**. An important constraint, one that allows the model to set resources busy so they will not be over-tasked, is the fourth constraint listed below. In summary, the constraints are

1) Precedence relationships -- where activity **i** must precede activity **j**

$$\Sigma_{t=1}^{H}\Sigma_{k=1}^{r} \mathbf{tx}_{ikt} + \mathbf{m}_j <= \Sigma_{t=1}^{H}\Sigma_{k=1}^{r} \mathbf{tx}_{jkt}$$

2) Each activity is completed only once

$$\Sigma_{t=1}^{H}\Sigma_{k=1}^{r} \mathbf{x}_{jkt} = 1, \quad 1<=\mathbf{j}<=\mathbf{n}$$

3) Restricted activity completion times

$$\mathbf{x}_{jkt} = 0 \quad \text{if } \mathbf{m}_j > \mathbf{t}$$

4) a. Resource limits -- where not more than the available amount ($a_k$) of each resource can be used during each time period **t**

$$\Sigma_{j=1}^{n} \mathbf{x}_{jkt} <= \mathbf{a}_k \quad \text{for } \mathbf{k}=(1,r),\ \mathbf{t}=(1,\mathbf{H})$$

b. For $\mathbf{m}_j > 1$ (and when activities **i** and **j** can both use resource **k**),

$$\mathbf{x}_{jkt} + \Sigma_{u=t-mj+1}^{t} \mathbf{x}_{iku} <= 1 \quad \text{for } i=(1,n_1),\ \mathbf{j}=(1,n_2),\ \mathbf{t}=(m_j,H), \text{ and } \mathbf{k}=(1,r)$$

NOTE: $\mathbf{n}_1$ and $\mathbf{n}_2$ represent the applicable subsets of **n** activities for activities i and j, respectively

5) Restricted flying times -- where flying is generally only permitted in the morning hours

$\mathbf{x}_{jkt} = 0$ for t=1-7,11-17,21-27,... and j is a flying activity

6) Restricted times for academics -- where academics are only permitted in the afternoon hours

$x_{jkt} = 0$ for t=8-10,18-20,28-30,... and j is an academic activity

A word of caution is necessary in regard to the latter two constraints in this formulation. That is, because activities are restricted to particular segments of the day, a solution is possible where an activity is viewed as completed when actually it could not have started yet. For example, an academic activity that requires 2 time units could not complete in time period 8 because it is restricted from being scheduled in time period 7. To remedy this situation, a seventh constraint set should be added to the model. This constraint set would involve a summation over a subset of times, defined by the activity durations and their designation as flying or academic activities. The constraint set should not allow activities to be scheduled over multiple days. Instead, the model will schedule all activities so that they are completed in a single day.

## 3.2 Solution Methodology

The following sections estimate the actual size of the problem, then present methods for reducing the size of the problem and solving the reduced problem.

### 3.2.1 The Size of the TPS Problem.

3.2.1.1 Decision Variables. For each TPS class, there are roughly 1,000 activities that need to be scheduled using 9 resources over 46 weeks. A typical TPS duty day is shown in Table 2.

TABLE 2

Typical TPS Duty Day

| | | |
|---|---|---|
| 0600-0930: | Flying period #1 | (for students) |
| 0930-1300: | Flying period #2 | (for students) |
| 1300-1630: | Flying period #3 | (for instructors) |
| 1330-1430: | Academic period #1 | |
| 1430-1530: | Academic period #2 | |
| 1530-1630: | Academic period #3 | |

Adapted from (Isbell, 20 Sep 90)

The day starts at 0600 and ends at 1630. Note that flying period #3 can overlap with the academic periods and is used to provide instructors with the training they need to retain their qualifications as instructors. Restrictions on the schedule set a limit of 15 class hours and 50 sorties per five-day week. A sample weekly schedule for an "A" class (flying activities only) is provided at Appendix A.

Since each day consists of approximately 10 hours, there are approximately 50 hours available each week and thus 50 x 46 = 2300 hours available overall. Letting **t** be in hour time units, **t** ranges from 1 to **H** = 2300. Thus for the model described in Chapter 2, there can be as many as (n x r x H) = (1,000 x 9 x 2,300) = 20,700,00 decision variables. The extreme size of this problem will be addressed in section 3.2.2.

3.2.1.2 Constraints. There is also the potential for quite a large number of constraints due to

1) Precedence relationships -- there can be as many as $\binom{1000}{2}$ = 1000!/2!998! = 499,500 precedence constraints;

2) Each activity is completed only once -- there will be exactly **n** = 1000 constraints; and

3) Resource limits -- there will be **r** x **t** = 9 x 2,300 = 20,700 constraints, plus well over six million constraints to ensure resources are set busy. This latter estimate is obtained from estimates that in the TPS problem,

> 10% of activities have $m_j > 5$ = (100 activities) x (9 resources) x (2,300-5) time periods, which accounts for 100x9x2,295 = 2,063,700 constraints;
>
> 10% of activities have $m_j > 4$ = (100 activities) x (9 resources) x (2,300-4) time periods, which accounts for 100x9x2,296 = 2,066,400 constraints;
>
> 10% of activities have $m_j > 3$ = (100 activities) x 9 resources) x (2,300-3) time periods, which accounts for 100x9x2,297 = 2,067,300 constraints;
>
> 10% of activities have $m_j > 2$ = (100 activities) x 9 resources) x (2,300-2) time periods, which accounts for 100x9x2,298 = 2,068,200 constraints; and
>
> 20% of activities have $m_j > 1$ = (200 activities) x 9 resources) x (2,300-1) time periods, which accounts for 200x9x2,299 = 4,138,200 constraints.

Note that the remaining 40% of the tasks have $m_j = 1$.

3.2.2 Preprocessing. Using any available technique, this problem would be highly unmanageable and computationally intractable unless it is reduced in size. As discussed in the literature review, a certain amount of preprocessing can be done to reduce the size of the problem. To reduce the number of decision variables in the TPS problem, there are basically five rules or heuristics (inherent in the nature of TPS problem) that could be applied. These rules are summarized in Appendix B and their potential reductions are discussed below:

1) Some activities are repetitive and occur at the same time each week. This holds true for at least three events -- operations meetings and safety meetings among flying events, and testing among academic events. These three types of events include 3x46=138 total events where each requires one hour for completion. So $\mathbf{x}_{jkt} = 0$ for a total of n x (2,300-t) = 138 x (2,300-138) = 298,256 decision variables. This means that the problem can initially be reduced by 298,256 decision variables. This rule also reduces the number of available time periods to 2,162 and the total number of events that need to be scheduled to 362. It should be noted that an alternate approach here would be to eliminate the repetitive events from the problem entirely, thus reducing the initial problem size to (1000-138) = 862 events and (2300-138) = 2162 time periods, so that the initial number of decision variables would be (862 events x 9 resources x 2162 time periods) = 16,772,796, rather than 20,700,000. However, the only

difference between these two ways to handle repetitive events is that the latter method is simpler in terms of data management.

2) As stated as a constraint in section 3.1.2, each resource is applicable to at most only about half of all activities. This provides a second reduction of the total number of decision variables by one-half, incurring a reduction of 10,350,000 decision variables.

3) Two other previously-mentioned constraints (that academic activities should occur in the morning and flying activities in the afternoon) can also be applied. Using 10,350,000 as the total number of yet unrestricted variables, this provides a third reduction of 10,350,000x(0.5) = 5,175,000 decision variables.

4) A fourth reduction can be made by applying the condition that $\mathbf{x}_{jkt}=0$ if $\mathbf{m}_j>\mathbf{t}$. It can be assumed that (a) approximately 10% of all activities have $\mathbf{m}_j>= 5$; (b) another 10% have $\mathbf{m}_j>4$; (c) another 10% have $\mathbf{m}_j>3$; (d) another 10% have $\mathbf{m}_j>2$; and finally (e) yet another 20% have $\mathbf{m}_j>1$. Then using (1,000-138) = 862 as the total number of events to schedule and 5 as the total number of applicable resources, we can reduce number of decision variables by approximately (a) 0.10x862x5x5 = 2,155; (b) 0.10x862x5x4 = 1,724; (c) 0.10x862x5x3 = 1,293 (d) 0.10x862x5x2 = 862; and (e) 0.20x862x5x1 = 862. This provides a total reduction of 6,896 decision variables.

5) As a final reduction, events can be designated to occur within specific months. Since there are 862 activities and 2,162 hours that are still not scheduled (or used), there remains approximately (862 activities/12 months) = 72 activities per month that can occur among (2,162 hours/12 months) = 180 hours. As reflected in the flying/academic activity constraints in section 3.1.2, the number of flying and academic activities and hours available for each occur in a ratio of 2:1. Thus, of the 180 hours available each month, 120 are flying-designated hours and the remaining 60 are academic hours. Also, of the 72 activities, 48 are flying and 24 are academic. Over the year, the total number of hours available for flying activities is (2,162 x 2/3) = 1,441 hours and the number of hours for academic activities is (2,162 x 1/3) = 720 hours. Thus, by specifying activities in a month period, $x_{jkt}$ can be set to zero for

flying activities: (1,441 - 120) = 1,321 hours and 48 activities; thus (48 activities x 1,321 hours x 5 resources) = 317,040 decision variables;

academic activities: (720 - 60) = 660 hours and 24 activities; thus (24 activities x 660 hours x 5 resources) = 79,200 decision variables.

The total reduction is 396,240 decision variables.

Table 3 provides a summary of the variable reduction rules available.

TABLE 3

Decision Variable Reduction Rules

| Rule | Reduction (# of decision variables) |
|---|---|
| 1. Repetitive events | 298,356 |
| 2. Resource Applicability | 10,350,000 |
| 3. Flying/academic time restrictions | 5,175,000 |
| 4. $x_{jkt} = 0$ for $m_j > t$ | 6,896 |
| 5. Designate activities within specific months | 396,240 per month |

Application of the first four reduction rules reduces the problem from 20,700,000 to approximately 4,869,748 decision variables. Then, if specific events are pre-set for all twelve months, the problem could be reduced to (4,869,748 - 396,240 x 12) = 114,868 decision variables. This is still an extremely large problem. However, the estimates made in the reduction techniques were conservative. It is hoped that in a real application, the reductions will turn out to be much greater. However, it may happen that no techniques can be found to reduce the size of the problem. Then the problem will remain computationally intractable. These ideas are discussed further in later sample problems.

In reducing the number of decision variables, certain constraints will be eliminated. However, if a large number of constraints remains (not estimated here due to the difficulty in tracking over 20 million decision variables), there are techniques that can be used to reduce the number of constraints. Such techniques include those derived by Johnson (14:803) and discussed in the literature review. By employing Johnson's ideas to the TPS problem, some variables could be set to one, rendering some constraints redundant. This idea is illustrated in section 3.3, where it is applied to a sample problem.

3.2.3 Application of the Branch-and-Bound Technique. Once the original TPS problem is reduced to a manageable size (under a few hundred decision variables), the Branch-and-Bound technique could be applied to obtain a solution using any one of a number of available commercial software packages. In this thesis, the STORM software (8) was used. STORM, like most commercial packages, uses an LP-relaxation in the bounding phase and a depth-first search in the branching phase of the Branch-and-Bound technique. In a depth-first search, if the current subproblem (parent node) is not fathomed, then the next subproblem considered is one created from the parent node (called a child node). A depth-first search offers three principal advantages (2; 19:358)

1) The LP-relaxation for a child is obtained from the LP-relaxation of its parent by the addition of a simple lower- or

upper-bound constraint. Hence, given the optimal solution for the father node, the optimal solution for the new subproblem (the child node) can easily be found by using a dual simplex algorithm without a basis reinversion or a transfer of the data;

2) Experience seems to indicate that feasible solutions are more likely to be found deep in the tree rather than at nodes near the root (original problem); and

3) A smaller set of solutions o subproblems needs to be retained (stored in the computer) for further consideration.

At each successive level of the search tree, STORM fixes one or more variables at integral values and finds a new solution.

## 3.3 Sample Problems and Solutions

This section discusses two sample problems that illustrate how the proposed model formulation and solution methodology can be used to solve problems representative of portions of the TPS scheduling problem.

### 3.3.1 Small Sample Problem.

This small sample problem is helpful in demonstrating how the proposed BIP formulation described in section 3.1 could be used to schedule events similar to the needs of the TPS. Because the problem is of a manageable size (72 decision variables), preprocessing is not necessary to reduce the size of the problem. The problem description is provided in Table 9 of Appendix C. The associated model formulation is also provided in Appendix C.

The given problem is so simple that a solution could be found manually. For example, a feasible schedule is shown in Table 4

TABLE 4

Feasible Schedule for the Small Sample Problem

| Time period t | Activity j | Resource k |
|---|---|---|
| 1 | 1/2 | 1/2 |
| 2 | 2 | 2 |
| 3 | 3/4 | 1/2 |
| 4 | 4 | 2 |
| 5 | 5/6 | 1/2 |
| 6 | 6 | 2 |

This means, for example, that activity 5 is scheduled to start and end in time period 5 using resource 1 and activity 6 is scheduled to start in time period 5 and end in time period 6 using resource 2.

However, the STORM software provided the following automated solution

$$x_{111},\ x_{222},\ x_{323},\ x_{413},\ x_{512},\ \text{and}\ x_{625} = 1,$$

which provides the schedule in Table 5.

TABLE 5

Automated Solution for the Small Sample Problem

| Time period t | Activity j | Resource k |
|---|---|---|
| 1 | 1/2 | 1/2 |
| 2 | 2/5 | 2/1 |
| 3 | 3/4 | 2/1 |
| 4 | 6 | 2 |
| 5 | 6 | 2 |
| 6 | | |

The problem input to STORM is shown in Appendix D. As shown in this file, only 33 of the constraints listed were actually included as input. This subset of constraints was used because STORM limits the problem size to 40 constraints and after three trials, it was discovered that this subset of constraints was able to produce a feasible schedule.

The automated solution differs from the manual solution but is also a feasible schedule. However, the automated solution completes the project before the targeted 6th period and, as stated previously, an early completion is not desirable for the TPS schedule. The automated solution allows for an earlier completion time because

1) all time periods are not being used (e.g., period 6 in the automated solution has no activity); and

2) several activities occur at once (e.g., activities 1 and 2 and 2 and 5 occur simultaneously).

These two deficiencies suggest that a few more constraints should be added to the model. The constraints are discussed in section 4.1.

3.3.2 Large Sample Problem. In this sample problem, preprocessing is used to reduce the problem size so that it can be easily solved using the STORM software (or any other IP solver). The problem description is provided in Table 10 of Appendix D. The specific numbers used in the problem were chosen from a sample manually-devised schedule shown in Table 6.

TABLE 6

Large Sample Problem Schedule

| | | DAY 1 Act | DAY 1 Res | DAY 2 Act | DAY 2 Res |
|---|---|---|---|---|---|
| 0600-0900 | Fly period #1 | 2 | 1 | 6 | 3 |
| 0900-1300 | Fly period #2 | 4 | 1 | 8 | 1 |
| 1300-1400 | Academic #1 | 1 | 3 | 5 | 1 |
| 1400-1500 | Academic #2 | 3 | 2 | 7 | 3 |
| 1500-1600 | Academic #3 | 3 | 2 | 9 | 1 |

| | | DAY 3 Act | DAY 3 Res | DAY 4 Act | DAY 4 Res |
|---|---|---|---|---|---|
| 0600-0900 | Fly period #1 | 14 | 2 | 18 | 3 |
| 0900-1300 | Fly period #2 | 16 | 2 | 20 | 2 |
| 1300-1400 | Academic #1 | 11 | 1 | 15 | 3 |
| 1400-1500 | Academic #2 | 11 | 1 | 17 | 2 |
| 1500-1600 | Academic #3 | 13 | 3 | 19 | 1 |

Act = activity
Res = resource

In this example, activities 10 and 12 are special instructor flying training periods that, for now, must be pre-set in the model because they need to be scheduled for Flying Period #3 (recall Table 2) and the current model does not allow for flying to be scheduled in the afternoon. A later modification to the model's fifth and sixth constraints could allow for this special flying period. So the model will really only be scheduling 18 activities.

In this problem, there are potentially 18x3x40 = 2,160 decision variables. To reduce this number, several decision variable reduction rules were applied. To record which variables remained nonzero, the MATLAB software (MATLAB Manual) was used. The reductions are discussed below and a sample portion of the MATLAB files are at Appendix E. The list of nonzero decision variables remaining is provided at Appendix F. Given activities j, resources k, and time periods t:

1) Resource applicability:

$x_{jkt}$ = 0 for all t and for

| | j |
|---|---|
| k=1: | 1,3,6,7,13,15,16,17,18,20 |
| k=2: | 2,6,8,9,18,19 |
| k=3: | 4,5,11,14,16,20 |

This rule reduces the number of decision variables by 880.

2) Restricted flying/academic times: Note that the "morning hours" consist of t = 1-7, 11-17, 21-27, and 31-37, and the "afternoon hours" consist of t = 8-10, 18-20, 28-30, and 38-40.

$x_{jkt}$ = 0 for the morning hours, all k, and
j = 1,3,5,7,9,11,13,15,17,19

$x_{jkt}$ = 0 for the afternoon hours, all k, and
j = 2,4,6,8,14,16,18,20

Disregarding redundant reductions with the first rule (as tracked in MATLAB), this second rule reduces the number of decision variables by 716.

3) $x_{jkt} = 0$ if $m_j > t$:

$x_{jk1}$ = 0 for all k and j = 2,3,4,6,8,11,14,16,18,20

$x_{jk2}$ = 0 for all k and j = 2,4,6,8,14,16,18,20

$x_{jk3} = 0$ for all k and j = 4,8,16,20

This rule reduces the number of decision variables by 18.

4) Similar to the fifth reduction rule, assume certain activities completed in hours 1-20 and 21-40: Restrict activities 1-9 to the hours 1-20 and activities 11 and 13-20 to the hours 21-40. Thus

$x_{jkt} = 0$ for

| | | j |
|---|---|---|
| t = 1-20 and | k=1: | 11,14,19 |
| | k=2: | 11,13,14,15,16,17,20 |
| | k=3: | 13,15,17,18,19 |

| | | j |
|---|---|---|
| t = 21-40 and | k=1: | 2,4,5,8,9 |
| | k=2: | 1,3,4,5,7 |
| | k=3: | 1,2,3,6,7,8,9 |

This rule reduces the number of decision variables by 282.

5) Also similar to the fifth reduction rule, assume activities 13 and 20 will not be completed before t=30 and t=37, respectively. Then

$x_{13,k,t} = 0$ for t = 1-29 and k=2,3

$x_{20,k,t} = 0$ for t = 1-36 and k=2

6) Set certain events: Let $x_{1,3,8}$, $x_{2,1,3}$, $x_{3,2,10}$, and $x_{4,2,7}$ = 1. This is equivalent to setting the first day of the schedule in Table 8. Then

$x_{1,k,t} = 0$ for t = 1-7, 9-40, and k=2,3

$x_{2,k,t} = 0$ for t = 1-2, 4-40, and k=1,3

$x_{3,k,t} = 0$ for t = 1-9, 11-40, and k=2,3

$x_{4,k,t} = 0$ for t = 1-6, 8-40, and k=1,2

This rule reduces the number of decision variables by 10, 22, 10, and 22 variables, respectively, for an average of 16.

A list of the nonzero decision variables remaining is in Appendix F. In total, these rules reduce the problem to about 183 decision variables. Table 7 compares these reductions to the reductions projected for the full TPS problem.

Table 7

Decision Variable Reduction Comparison

| | | TPS Problem | Sample Problem |
|---|---|---|---|
| | Original # of variables | 20,700,000 | 2,160 |
| | Rule | Projected # reduced (%) | Actual # reduced (%) |
| 1. | Repetitive events | 298,356 ( 1.44) | N/A |
| 2. | Resource applicability | 10,350,000 (50) | 880 (40.74) |
| 3. | Flying/academic time restrictions | 5,175,000 (25) | 716 (33.15) |
| 4. | $x_{jkt} = 0$ for $t_j > t$ | 6,156 (0.03) | 18 ( 0.83) |
| 5. | Designate activities to specific times (months) | 396,240 (1.91) **per month** | 315 (1.46) |

As shown in Table 7, the actual reductions in this sample problem appear to be consistent with the reductions projected for the full TPS problem. Even though the problem is now smaller, it is not quite yet small enough (the target is under a few hundred decision variables) to solve efficiently. Also, a large number of constraints remain. Reduction rules can be applied to reduce the number of constraints (and, in the process, reduce the number of decision variables). The approximate number of constraints, by constraint type, are

Precedence -- 18 constraints (activity 11 precedes activity 17 and all activities precede activity 19);

Each activity is completed only once -- 18 constraints; and

Resource Limits --

Only one type of each resource is available per each time period -- 3 x 40 = 120 constraints,

Resources must be set busy -- approximately 10 x 2 x 40 = 800 constraints.

This represents a total of 956 constraints. The key constraints to consider here are the first part of the resource constraints. Of these, all but 35 (listed in Appendix G) can be disregarded because they are of the form $x_{j,k,t} <= 1$, which is redundant with the restriction that $x_{j,k,t} = 0$ or 1 To reduce this number further, the Johnson method of preprocessing where variables are set to 0 or 1 can be employed. A summary of how this method can be applied to the sample problem is provided in Appendix G. The variable settings made were selected using the schedule in Table 8 as a guideline.

A word of caution must be mentioned here. The reduction methods described in Appendix G rely upon prior knowledge of feasible settings (from Table 6). In the real TPS scheduling process, known feasible settings are not available (otherwise the scheduling process would be trivial), but could be known approximately from previously-executed schedules or expert knowledge. However, if any setting (say variable $x_{123} = 1$) produces an infeasible solution, then the problem should be re-run with the opposite setting ($x_{123} = 0$). For cases where the number of feasible solutions is not as large, this is an efficient approach. A better way to apply the reduction technique would be as a special branching routine in the Branch-and-Bound algorithm itself. This idea will be discussed in more detail in Chapter 4.

At the end of the process the following variables have been set to one (recall that $x_{1,3,8}$, $x_{2,1,3}$, $x_{3,2,10}$, and $x_{4,1,4} = 1$ as a result of setting the first day):

$x_{5,1,8}$ $x_{8,1,17}$ $x_{9,1,20}$ $x_{11,1,29}$ $x_{13,3,20}$ $x_{14,2,23}$

$x_{15,3,38}$ $x_{16,2,27}$ $x_{17,2,39}$ $x_{20,2,37}$

Also, all of the first set and all but four of the second set of resource constraints have been eliminated.

Now only 46 decision variables and 11 constraints remain. The resulting problem is shown in Appendix D. Solved in STORM, the automated solution provided $x_{6,3,3}$, $x_{7,2,18}$, $x_{18,3,21}$, $x_{19,3,28} = 1$, resulting in the schedule in Table 8.

TABLE 8

Automated Solution for the Large Sample Problem

| t | j | k | t | j | k |
|---|---|---|---|---|---|
| 1 | 2/6 | 1/3 | 21 | 14/18 | 2/3 |
| 2 | 2/6 | 1/3 | 22 | 14 | 2 |
| 3 | 2/6 | 1/3 | 23 | 14 | 2 |
| 4 | 4 | 2 | 24 | 16 | 2 |
| 5 | 4 | 2 | 25 | 16 | 2 |
| 6 | 4 | 2 | 26 | 16 | 2 |
| 7 | 4 | 2 | 27 | 16 | 2 |
| 8 | 1/5 | 2/1 | 28 | 11 | 1 |
| 9 | 3 | 2 | 29 | 11 | 1 |
| 10 | 3 | 2 | 30 | 13 | 3 |
| 11 | | | 31 | | |
| 12 | | | 32 | | |
| 13 | | | 33 | | |
| 14 | 8 | 1 | 34 | 20 | 3 |
| 15 | 8 | 1 | 35 | 20 | 3 |
| 16 | 8 | 1 | 36 | 20 | 3 |
| 17 | 8 | 1 | 37 | 20 | 3 |
| 18 | 7 | 2 | 38 | 15 | 3 |
| 19 | | | 39 | 17 | 2 |
| 20 | 9 | 1 | 40 | 19 | 1 |

The solution in Table 8 is feasible, which means that the variables that were pre-set are allowed to stand. However, as with the small sample problem, the solution here indicates that a few more constraints should be added to the model to ensure that

1) all time periods are being used; and

2) simultaneous activities do not over-task the students. For example, activities 2 and 6 should possibly not occur simultaneously if both events involve the **same** class of students. However, a provision should be made that some activities can overlap -- such as would occur when a third flying period is added to occur in the afternoon.

3.4 Application to the Full TPS Problem. In the large sample problem, the Johnson reduction technique of pre-setting variables was used to reduce the problem from 183 decision variables to 46. As mentioned earlier, such feasible settings are not always known in the real TPS scheduling situation. At best, previously-executed schedules could be used as a guideline. An advantage of the TPS schedule is that a number of different feasible solutions exists (Isbell, 90) so that the chance of selecting feasible settings is increased. However, more often than not, the process would have to be repeated. It may not be practical, then, to include this procedure explicitly (as was done for the large sample problem) as a standard method for solving the TPS problem.

Without the Johnson reduction technique, the large sample problem (which equates to about one week of the TPS schedule) would remain at 183 decision variables -- too large to be solved efficiently using the Branch-and-Bound algorithm. This means that less than one week of the TPS schedule could be solved efficiently using the methods investigated in this study.

However, as discussed in the literature search, success has been found in applying combination optimal/suboptimal methods where heuristics are embedded within optimization methods. Although not specifically applicable to the TPS problem, Mazzola (17:569) solved scheduling problems of up to 2,500 0-1 decision variables using an altered Branch-and-Bound algorithm.

So a possible solution approach using the BIP model formulation of the TPS problem could be a specialized Branch-and-Bound algorithm where the branching phase uses the ideas of the Johnson reduction technique (i.e., pre-setting certain variables). This would reduce the amount of branching that would be done, especially since many alternate feasible solutions do exist (12). Another feature that could be added to the algorithm would be to stop as soon as **any** feasible solution is found -- again, because the objective in solving the problem is one of feasiblity and not optimality.

Another alternative for solving the TPS problem would be to formulate the problem as a network, for which algorithms exist to efficiently very large problems (of the size of at least one month of the TPS problem) (2). But, as mentioned in the literature review, further study would be required to find ways to adequately represent the resource constraints.

# 4. Conclusions and Recommendations

## 4.1 Conclusions

The current TPS scheduling process is manual and time-consuming. The objective of this thesis was to investigate algorithms or methods that could be used as the basis for an automated scheduling system. The primary method investigated formulates the problem as a BIP and solves the problem using the Branch-and-Bound technique. Application of this method to sample problems demonstrated that the method can produce a feasible schedule for small problems (under 100 variables) that equate to **portions** of the TPS schedule. However, it cannot efficiently solve large problems such as the **full** TPS problem.

Although more work needs to be done to develop an efficient algorithm to solve the TPS scheduling problem, much has been learned about the nature of the problem that should be used in future investigations. In general,

1) like most RCS problems, the size of the TPS problem is formidable and needs to be reduced before **any** solution algorithm could be applied;

2) **feasibility** rather than optimality is the key objective in solving the TPS problem. That is, the objective is to find a schedule that will meet the expected graduation date;

3) the basic resource constraints that should be included in the TPS problem formulation are provided in the BIP model

formulation described in this study; however, as discussed in section 3.1.2, one significant constraint set that should be added to further restrict activity completion times.

4) heuristics should be applied to reduce the problem size and for use in a specialized Branch-and-Bound algorithm (to enhance solution efficiency).

Specifically, the ideas of the Johnson reduction technique could be used in the branching phase of the Branch-and-Bound technique. Another feature that could be added is to stop the search process whenever any feasible solution is found.

The need for a specialized Branch-and-Bound algorithm stems from the fact that Branch-and-Bound algorithms based on linear relaxation, cutting planes, and Lagrangean relaxation of the constraints are reasonably effective for problems with only up to three resources and 25 jobs.

More specifically, to use the BIP model formulation investigated in this study for the TPS problem, then the model should be modified (or expanded) to

1) give priority to academic versus flying activities;

2) allow concurrent scheduling of the A and B classes;

3) address re-scheduling (i.e., the introduction of new events);

4) impose resource leveling so that neither the students nor the instructors are over- or under-utilized; and

5) ensure that all time periods are being used.

Although not addressed in the study, the BIP model formulation could be simplified by using periods (as in Table 2) rather than hours as the units of time. In this schema, the large sample problem, for example, would consist of only 25 time periods of 1.5 hours each (rather than 40 hours). The problem would then consist of (nxrxH) = (20x3x25)=1,500 decision variables rather than 2,400 decision variables. However, this is still a very large problem and the reduction rules would have less of an effect in that less time periods are available for manipulation.

## 4.2 Recommendations

Further study should be conducted to find a more efficient solution algorithm for the TPS scheduling problem. To use the BIP model formulation would require at least a specialized Branch-and-Bound solution algorithm, described in section 4.1. Alternatively, the problem could be formulated as a network but further study would be required to find ways to adequately represent the resource constraints.

The development of an efficient solution algorithm for the TPS problem would enable the development of an automated TPS scheduling system. An automated system could in turn reduce the scheduling process from weeks to days or hours. It could also be used to track resource usage and program progress. These features would spare the TPS and the Air Force time, money, and much frustration.

## Appendix A

### Sample TPS Weekly Schedule

| DAY | PERIOD | COURSE | INSTRUCTOR | REMARKS |
|---|---|---|---|---|
| MONDAY 17 SEP | 1330 | PROPULSION 6 | GALLUPS | MAJ STOFFERAHN |
| | 1430 | PROPULSION 7 | GALLUPS | MAJ STOFFERAHN |
| | 1530 | NT-33 HUD DEMO FTT | CALSPAN | |
| TUESDAY 18 SEP | 1330 | ASTTA PREP | | |
| | 1430 | ASTTA PREP | | |
| | 1530 | PROPULSION 9 | GALLUPS | MAJ STOFFERAHN |
| WEDNESDAY 19 SEP | | ALL FLY | | |
| THURSDAY 20 SEP | 1330 | QUAL FLT ORAL (F18) | MAJ SHELLEY | ALL |
| | 1430 | PROPULSION 10 | GALLUPS | MAJ STOFFERAHN |
| | 1530 | PROPULSION 11 | GALLUPS | MAJ STOFFERAHN |
| FRIDAY 21 SEP | 1330 | PROPULSION FTT | LTC LEWIS | |
| | 1430 | OPS MEETING | LTC LUTZ | AUDITORIUM |
| | 1600 | SAFETY MEETING | | O'CLUB |

FLYING:

A-37 SPIN (3)
T-38 SPIN CHASE (3)
A-7 DEPARTURES (2)
A-7 QUAL FAM (2)
A-7 IP CHASE (4)
C-141 QUAL (4)
TEST MGMT PROJECT (20)
T-38 TGT (W/TACAN) (8)
NT-33 HUD DEMO (8)
ASTTA (DAY) (8)
ASTTA (NIGHT) (1)

Adapted from (Isbell, 20 Sep 90)

## Appendix B

### Decision Variable Reduction Rule Summary

The reduction rules available to reduce the number of decision variables in the BIP model formulation of the TPS problem are described below. As illustrated in the large sample problem, MATLAB (or any other matrix software) can be used to record the decision variables that are set to zero as a result of applying these reduction rules.

1) Repetitive events. Determine which activities occur each week (or each time period). Then set the appropriate decision variables to one and zero. For example, if activity 5 (j=5) occurs in the last time period each week (say, t=50, 100, 150,...) and can use resource 1 (k=1), then set $x_{5,1,p} = 1$, where p=50n, n = 1,2,3,...,46. Also set $x_{5,k,q} = 0$ for all k where k does not equal 1 and q = 1-49, 51-99, 101-149,...,2252-2300.

2) Resource applicability. If resource k cannot be used in activity j, then set $x_{j,k,t} = 0$ for the specific j and k, and all t such that t = (1, 2300). For example if activity 3 cannot use resources 1 and 4, then set $x_{3,1,t}$ and $x_{3,4,t} = 0$ for t = (1, 2300).

3) Flying/academic time restrictions. Of the remaining decision variables, set $x_{j,k,t} = 0$ for all k and

j even (thus j is a flying activity) and t = 1-7, 11-17, 21-27,...,2221-2297;

j odd (thus j is an academic activity) and t = 8-10, 18-

20, 28-30,...,2298-2300.

4) $\underline{x_{j,k,t} = 0 \text{ for } m_t > t}$. If the projected completion time ($m_t$) of activity j exceeds one time period, set $x_{j,k,t} = 0$ for j, all k, and t, where $t < m_t$. For example, assume task 8 requires 2 time units for completion. Then x5,k,t = 0 for all k and t = 1,2.

5) <u>Designate activities to specific months</u>. If activity j could be designated to occur only in a certain month, then set $x_{j,k,t} = 0$ for all values of k and all values of t that occur in all the other months. For example, assume activity 9 will only occur in the the first month. Then set $x_{9,k,t} = 0$ for all k and t, where t >= 193 (because values of t = (1, 192) constitute the first month).

## Appendix C

### Small Sample Problem Description and Model

TABLE 9

Small Sample Problem Description

Given: 6 activities ($n = 6$), terminal activity is activity 6
6 time periods ($H = 6$), and
2 resources ($r = 2$)

Resource applicability (Resource $k$ can be used by activity $j$):

| | $j$ |
|---|---|
| $k=1$: | 1,3,4,5 |
| $k=2$: | 2,3,4,6 |

Activity completion times (activity durations) $m_j$:

$m_1,\ m_3,\ m_5 = 1$
$m_2,\ m_4,\ m_6 = 2$

Precedence requirements: Activity $i$ precedes activity $j$ ($i \rightarrow j$)

1 --> 3
1,2,3,4,5 -- > 6

Note that since activities 1 and 2 cannot consume resources 2 and 1, respectively, $x_{1,2,t}$ and $x_{2,1,t}$ are both zero. Also, since activities 2, 4, and 6 exceed one time period, $x_{2,k,1}$, $x_{4,k,1}$, and $x_{6,k,1}$ are all zero.

MODEL:

Minimize $2\mathbf{x}_{6,2,2} + 3\mathbf{x}_{6,2,3} + 4\mathbf{x}_{6,2,4} + 5\mathbf{x}_{6,2,5} + 6\mathbf{x}_{6,2,6}$

Subject to

(1) Precedence Constraints

1: $1x_{111}+2x_{112}+3x_{113}+4x_{114}+5x_{115}+6x_{116}-1x_{311}-2x_{312}-3x_{313}-4x_{314}-5x_{315}-6x_{316}-1x_{321}-2x_{322}-3x_{323}-4x_{324}-5x_{325}-6x_{326} <= -1$

2: $1x_{111}+2x_{112}+3x_{113}+4x_{114}+5x_{115}+6x_{116}-2x_{622}-3x_{623}-4x_{624}-5x_{625}-6x_{626} <= -2$

3: $2x_{222}+3x_{223}+4x_{224}+5x_{225}+6x_{226}-2x_{622}-3x_{623}-4x_{624}-5x_{625}-6x_{626} <= -2$

4: $1x_{311}+2x_{312}+3x_{313}+4x_{314}+5x_{315}+6x_{316}+1x_{321}+2x_{322}+3x_{323}+4x_{324}+5x_{325}+6x_{326}-2x_{622}-3x_{623}-4x_{624}-5x_{625}-6x_{626} <= -2$

5: $2x_{412}+3x_{413}+4x_{414}+5x_{415}+6x_{416}+2x_{422}+3x_{423}+4x_{424}+5x_{425}+6x_{426}-2x_{622}-3x_{623}-4x_{624}-5x_{625}-6x_{626} <= -2$

6: $1x_{511}+2x_{512}+3x_{513}+4x_{514}+5x_{515}+6x_{516}-2x_{622}-3x_{623}-4x_{624}-5x_{625}-6x_{626} <= -2$

(2) Each activity is completed only once

7: $x_{111}+x_{112}+x_{113}+x_{114}+x_{115}+x_{116} = 1$
8: $x_{222}+x_{223}+x_{224}+x_{225}+x_{226} = 1$
9: $x_{311}+x_{312}+x_{313}+x_{314}+x_{315}+x_{316}+x_{321}+x_{322}+x_{323}+x_{324}+x_{325}+x_{326} = 1$
10: $x_{412}+x_{413}+x_{414}+x_{415}+x_{416}+x_{422}+x_{423}+x_{424}+x_{425}+x_{426} = 1$
11: $x_{511}+x_{512}+x_{513}+x_{514}+x_{515}+x_{516} = 1$
12: $x_{622}+x_{623}+x_{624}+x_{625}+x_{626} = 1$

(3) Resource limits

13: for k=1, t=1: $x_{111}+x_{311}+x_{511} <= 1$
14: t=2: $x_{112}+x_{312}+x_{412}+x_{512} <= 1$
15: t=3: $x_{113}+x_{313}+x_{413}+x_{513} <= 1$
16: t=4: $x_{114}+x_{314}+x_{414}+x_{514} <= 1$
17: t=5: $x_{115}+x_{315}+x_{415}+x_{515} <= 1$
18: t=6: $x_{116}+x_{316}+x_{416}+x_{516} <= 1$
19: for k=2, t=1: $x_{321} <= 1$ (redundant since all variables are bounded by 1)

20: t=2: $x_{222}+x_{322}+x_{422}+x_{622} <= 1$

21: t=3: $x_{223}+x_{323}+x_{423}+x_{623} <= 1$

22: t=4: $x_{224}+x_{324}+x_{424}+x_{624} <= 1$

23: t=5: $x_{225}+x_{325}+x_{425}+x_{625} <= 1$

24: t=6: $x_{226}+x_{326}+x_{426}+x_{626} <= 1$

(Since $m_2$, $m_4$, and $m_6$ =2):

for j=2, k=2, and i = 3,4,6

25: t=2: $x_{222}+x_{321}+x_{322} <= 1$

$x_{222}+x_{422} <= 1$

$x_{222}+x_{622} <= 1$

26: t=3: $x_{223}+x_{322}+x_{323} <= 1$

$x_{223}+x_{422}+x_{423} <= 1$

$x_{223}+x_{622}+x_{623} <= 1$

27: t=4: $x_{224}+x_{323}+x_{324} <= 1$

$x_{224}+x_{423}+x_{424} <= 1$

$x_{224}+x_{623}+x_{624} <= 1$

28: t=5: $x_{225}+x_{324}+x_{324} <= 1$

$x_{225}+x_{424}+x_{425} <= 1$

$x_{225}+x_{624}+x_{625} <= 1$

29: t=6: $x_{226}+x_{325}+x_{326} <= 1$

$x_{226}+x_{425}+x_{426} <= 1$

$x_{226}+x_{625}+x_{626} <= 1$

for j=4, k=1, and

30: t=2: $x_{412}+x_{111}+x_{112} <= 1$

$x_{412}+x_{311}+x_{312} <= 1$

$x_{412}+x_{511}+x_{512} <= 1$

31: t=3: $x_{413}+x_{112}+x_{113} <= 1$

$x_{413}+x_{312}+x_{313} <= 1$

$x_{413}+x_{512}+x_{513} <= 1$

32: t=4: $x_{414}+x_{113}+x_{114} <= 1$

$x_{414}+x_{313}+x_{314} <= 1$

$x_{414}+x_{513}+x_{514} <= 1$

33: t=5: $x_{415}+x_{114}+x_{115} <= 1$

$x_{415}+x_{314}+x_{315} <= 1$

$x_{415}+x_{514}+x_{515} <= 1$

34: t=6: $x_{416}+x_{115}+x_{116} <= 1$

$x_{416}+x_{315}+x_{316} <= 1$

$x_{416}+x_{515}+x_{516} <= 1$

for j=4, k=2, and

35: t=2: $x_{422}+x_{222} <= 1$

$x_{422}+x_{321}+x_{322} <= 1$

$x_{422}+x_{622} <= 1$

36: t=3: $x_{423}+x_{222}+x_{223} \leq 1$

$x_{423}+x_{322}+x_{323} \leq 1$

$x_{423}+x_{622}+x_{623} \leq 1$

37: t=4: $x_{424}+x_{223}+x_{224} \leq 1$

$x_{424}+x_{323}+x_{324} \leq 1$

$x_{424}+x_{623}+x_{624} \leq 1$

38: t=5: $x_{425}+x_{224}+x_{225} \leq 1$

$x_{425}+x_{324}+x_{325} \leq 1$

$x_{425}+x_{624}+x_{625} \leq 1$

39: t=6: $x_{426}+x_{225}+x_{226} \leq 1$

$x_{426}+x_{325}+x_{326} \leq 1$

$x_{426}+x_{625}+x_{626} \leq 1$

for j=6, k=2, and

40: t=2: $x_{622}+x_{221}+x_{222} \leq 1$

$x_{622}+x_{321}+x_{322} \leq 1$

$x_{622}+x_{421}+x_{422} \leq 1$

41: t=3: $x_{623}+x_{222}+x_{223} \leq 1$

$x_{623}+x_{322}+x_{323} \leq 1$

$x_{623}+x_{422}+x_{423} \leq 1$

42: t=4: $x_{624}+x_{223}+x_{224} \leq 1$

$x_{624}+x_{323}+x_{324} \leq 1$

$x_{624}+x_{423}+x_{424} \leq 1$

43: t=5: $x_{625}+x_{224}+x_{225} \leq 1$

$x_{625}+x_{324}+x_{325} \leq 1$

$x_{625}+x_{424}+x_{425} \leq 1$

44: t=6: $x_{626}+x_{225}+x_{226} \leq 1$

$x_{626}+x_{325}+x_{326} \leq 1$

$x_{626}+x_{425}+x_{426} \leq 1$

Because this problem is so small, the constraint that restricts flying and academic activities to the morning and afternoon, respectively, has been excluded.

## Appendix D

## Large Sample Problem Description and Model

TABLE 10

Large Sample Problem Description

Given: 20 activities (n=20), where j=19 is the terminal activity,

40 time periods (H=40) in hours, and 3 resources

NOTE: even-numbered activities represent flying activities an odd-numbered activities represent academic activities

Resource Applicability:

| | j |
|---|---|
| k=1: | 2,4,5,8,9,11,14,19 |
| k=2: | 1,2,4,5,7,10,11,12,13,14,15,16,17,20 |
| k=3: | 1,2,3,6,7,8,13,15,17,18,19 |

Activity completion times

$m_1, m_5, m_7, m_9, m_{13}, m_{15}, m_{17}, m_{19} = 1$
$m_3, m_{11} = 2$
$m_2, m_6, m_{14}, m_{18} = 3$
$m_4, m_8, m_{16}, m_{20} = 4$

MODEL:

Minimize $30x_{19,1,30}+38x_{19,1,38}+39x_{19,1,39}+40x_{19,1,40}+$
$28x_{19,3,28}+29x_{19,3,29}+30x_{19,3,20}+38x_{19,3,38}+39x_{19,3,39}$

Subject to

(1) Precedence constraints

1: $3x_{6,3,3}+4x_{6,3,4}+5x_{6,3,5}+6x_{6,3,6}+7x_{6,3,7}+11x_{6,3,11}+12x_{6,3,12}$

$$+13x_{6,3,13}+14x_{6,3,14}+15x_{6,3,15}+16x_{6,3,16}+17x_{6,3,17}-30x_{19,1,30}$$
$$-38x_{19,1,38}-39x_{19,1,39}-40x_{19,1,40}-28x_{19,3,28}-29x_{19,3,29}$$
$$-30x_{19,3,30}-38x_{19,3,38}-39x_{19,3,39} <= -1$$

2: $$18x_{7,2,18}+19x_{7,2,19}+20x_{7,2,20}+18x_{7,3,18}+19x_{7,3,19}+20x_{7,3,20}$$
$$-30x_{19,1,30}-38x_{19,1,38}-39x_{19,1,39}-40x_{19,1,40}-28x_{19,3,28}-29x_{19,3,29}$$
$$-30x_{19,3,30}-38x_{19,3,38}-39x_{19,3,39} <= -1$$

3: $$21x_{18,3,21}+22x_{18,3,22}+23x_{18,3,23}+24x_{18,3,24}+25x_{18,3,25}+26x_{18,3,26}$$
$$+27x_{18,3,27}+31x_{18,3,31}+32x_{18,3,32}+33x_{18,3,33}+34x_{18,3,34}+35x_{18,3,35}$$
$$+36x_{18,3,36}+37x_{18,3,37}-30x_{19,1,30}-38x_{19,1,38}-39x_{19,1,39}-40x_{19,1,40}$$
$$-28x_{19,3,28}-29x_{19,3,29}-30x_{19,3,30}-38x_{19,3,38}-39x_{19,3,39} <= -1$$

(2) Each activity is completed only once

4: $$x_{6,3,3}+x_{6,3,4}+x_{6,3,5}+x_{6,3,6}+x_{6,3,7}+x_{6,3,11}+x_{6,3,12}$$
$$+x_{6,3,13}+x_{6,3,14}+x_{6,3,15}+x_{6,3,16}+x_{6,3,17} = 1$$

5: $$x_{7,2,18}+x_{7,2,19}+x_{7,2,20}+x_{7,3,18}+x_{7,3,19}+x_{7,3,20} = 1$$

6: $$x_{18,3,21}+x_{18,3,22}+x_{18,3,23}+x_{18,3,24}+x_{18,3,25}+x_{18,3,26}+x_{18,3,27}$$
$$+31x_{18,3,31}+32x_{18,3,32}+33x_{18,3,33}+34x_{18,3,34}+35x_{18,3,35} = 1$$

7: $$x_{19,1,30}+x_{19,1,38}+x_{19,1,39}+x_{19,1,40}+x_{19,3,28}+x_{19,3,29}+x_{19,3,30}$$
$$+x_{19,3,38}+x_{19,3,39} = 1$$

(3) Resource limits

8: $x_{18,3,21}+x_{7,3,19}+x_{7,3,20} <= 1$

9: $x_{18,3,22}+x_{7,3,20} <= 1$

10: $x_{18,3,31}+x_{19,3,30} <= 1$

11: $x_{18,3,32}+x_{19,3,30} <= 1$

## Appendix E

## MATLAB Tracking for Large Sample Problem

MATLAB (MATLAB citation) was used to record which decision variables were eliminated (i.e., set to zero) during the preprocessing phase in solving the large sample problem. To represent all 2,160 decision variables in the problem, three 720-element matrices were generated where each matrix represented one of the three resources. That is, the first matrix represented all $x_{j1t}$ variables, the second represented all $x_{j2t}$ variables, and the third matrix represented all $x_{j3t}$ variables, where j = (1,20) and t = (1,40) in each case. Initially, all elements in the matrices were one -- implying that all decision variables were active. Then, as a reduction rule was applied, MATLAB code was used to zero out the appropriate elements of each matrix. After applying all reduction rules in this manner, the ones that remained represented all of the nonzero (i.e., active) decision variables in the problem.

A sample portion of the code used to track decision variables for the third resource is provided below:

```
<>
c=ones(20,40); generates a 20x40 matrix (but recall that 80
               elements will be eliminated because
               activities 10 and 12 are pre-set)

<>
for t=1:40, c(10,t) = 0;
for t=1:40, c(12,t) = 0;  set x10,3,t and x12,3,t = 0 to
                           represent the fact that activities
```

10 and 12 are pre-set because they are special cases

<>

for t=1:40, c(4,t) = 0; sets $x_{4,3,t} = 0$ because resource 3 cannot be used for activity 4

<>

for t=1:40, c(5,t) = 0; sets $x_{5,3,t} = 0$ because resource 3 cannot be used for activity 5

<>

for t=1:7, c(1,t) 0; sets $x_{1,3,t} = 0$ for t=(1,7) because activity 1 is an academic activity and thus restricted to the afternoon

<>

for t=8:10, c(2,t) = 0; sets $x_{13,3,t} = 0$ for t=(8,10) because activity 2 is a flying activity and thus restricted to the morning

<>

c(2,1)=0; sets $x_{2,3,1} = 0$ since activity 2 requires two time periods to complete and thus cannot be completed during the first period

<>

c(8,3)=0; sets $x_{16,3,t} = 0$ for t=(1,3) since activity 8 requires four time periods to complete and thus cannot be completed during the first three time periods

<>

for t=1:20, c(13,t) = 0; sets $x_{13,3,t} = 0$ for t= (1,20) since activity 3 was designated to not start until after t=10

<>

for t=1:7, c(1,t) = 0;
for t=9:40, c(1,t) = 0; set $x_{1,3,t} = 0$ for t=(1-7) and (9-40) since activity 3 is designated for week 8

As an illustration, consider the small sample problem of section 3.3.1 that consisted of six activities, six time periods, and two resource. Consider a decision variable of the problem. For example, consider

```
      1 1 1 1 1 1
C =   0 0 0 0 0 0
      1 1 1 1 1 1
      1 1 1 1 1 1
      1 1 1 1 1 1
      0 0 0 0 0 0
```

```
<>
for t=(1:3), c(1,t)=0
for t=(1:3), c(3,t)=0
for t=(1:3), c(5,t)=0
```

```
      0 0 0 1 1 1
C =   0 0 0 0 0 0
      0 0 0 1 1 1
      1 1 1 1 1 1
      0 0 0 1 1 1
      0 0 0 0 0 0
```

```
<>
for t=(4:6), c(2,t)=0
for t=(4:6), c(4,t)=0
for t=(4:6), c(6,t)=0
```

```
      0 0 0 1 1 1
C =   0 0 0 0 0 0
      0 0 0 1 1 1
      1 1 1 0 0 0
      0 0 0 1 1 1
      0 0 0 0 0 0
```

```
<>
c(2,1)=0, c(4,1)=0, c(6,1)=0
```

```
      0 0 0 1 1 1
C =   0 0 0 0 0 0
      0 0 0 1 1 1
      0 1 1 0 0 0
      0 0 0 1 1 1
      0 0 0 0 0 0
```

Thus the remaining decision variables include:

$$x_{1,1,4},\ x_{1,1,5},\ x_{1,1,6},\ x_{3,1,4},\ x_{3,1,5},\ x_{3,1,6},\ x_{4,1,2},\ x_{4,1,3},\ x_{5,1,4},\ x_{5,1,5},\ x_{5,1,6}.$$

# Appendix F

## Nonzero Decision Variables After Variable Reduction

After applying the variable reduction rules (described in 3.2.2) during preprocessing, the following decision variables remain:

(NOTE: $x_{1,3,8}$, $x_{2,1,3}$, $x_{3,2,10}$, and $x_{4,1,4}$ have all been set to one)

$x_{5,1,8}$ $x_{5,1,9}$ $x_{5,1,18}$ $x_{5,1,19}$ $x_{5,1,20}$ $x_{5,2,8}$ $x_{5,2,9}$ $x_{5,2,10}$ $x_{5,2,18}$
$x_{5,2,19}$ $x_{5,2,20}$

$x_{6,3,3}$ $x_{6,3,4}$ $x_{6,3,5}$ $x_{6,3,6}$ $x_{6,3,7}$ $x_{6,3,11}$ $x_{6,3,12}$ $x_{6,3,13}$ $x_{6,3,14}$
$x_{6,3,15}$ $x_{6,3,16}$ $x_{6,3,17}$

$x_{7,2,8}$ $x_{7,2,9}$ $x_{7,2,10}$ $x_{7,2,18}$ $x_{7,2,19}$ $x_{7,2,20}$ $x_{7,3,18}$ $x_{7,3,19}$ $x_{7,3,20}$

$x_{8,1,4}$ $x_{8,1,5}$ $x_{8,1,6}$ $x_{8,1,7}$ $x_{8,1,11}$ $x_{8,1,12}$ $x_{8,1,13}$ $x_{8,1,14}$ $x_{8,1,15}$
$x_{8,1,16}$ $x_{8,1,7}$ $x_{8,3,4}$ $x_{8,3,5}$ $x_{8,3,6}$ $x_{8,3,7}$ $x_{8,3,11}$ $x_{8,3,12}$
$x_{8,3,13}$ $x_{8,3,14}$ $x_{8,3,15}$ $x_{8,3,16}$ $x_{8,3,17}$

$x_{9,1,8}$ $x_{9,1,9}$ $x_{9,1,10}$ $x_{9,1,18}$ $x_{9,1,19}$ $x_{9,1,20}$ $x_{9,3,8}$ $x_{9,3,10}$ $x_{9,3,18}$
$x_{9,3,19}$ $x_{9,3,20}$

$x_{11,1,28}$ $x_{11,1,29}$ $x_{11,1,30}$ $x_{11,1,38}$ $x_{11,1,39}$ $x_{11,1,40}$ $x_{11,2,27}$ $x_{11,2,28}$
$x_{11,2,29}$ $x_{11,2,38}$ $x_{11,2,39}$ $x_{11,2,40}$

$x_{13,2,38}$ $x_{13,2,39}$ $x_{13,2,40}$ $x_{13,3,30}$ $x_{13,3,38}$ $x_{13,3,39}$ $x_{13,3,40}$

$x_{14,1,21}$ $x_{14,1,22}$ $x_{14,1,23}$ $x_{14,1,24}$ $x_{14,1,25}$ $x_{14,..,26}$ $x_{14,1,27}$ $x_{14,1,31}$
$x_{14,1,32}$ $x_{14,1,33}$ $x_{14,1,34}$ $x_{14,1,35}$ $x_{14,1,36}$ $x_{14,1,37}$ $x_{14,1,40}$
$x_{14,2,21}$ $x_{14,2,22}$ $x_{14,2,23}$ $x_{14,2,24}$ $x_{14,2,25}$ $x_{14,2,26}$ $x_{14,2,27}$

$x_{14,2,31}$ $x_{14,2,32}$ $x_{14,2,33}$ $x_{14,2,34}$ $x_{14,2,35}$ $x_{14,2,36}$ $x_{14,2,37}$

$x_{15,2,28}$ $x_{15,2,29}$ $x_{15,2,30}$ $x_{15,2,38}$ $x_{15,2,39}$ $x_{15,2,40}$ $x_{15,3,28}$ $x_{15,3,29}$

$x_{15,3,30}$ $x_{15,3,38}$ $x_{15,3,39}$ $x_{15,3,40}$

$x_{16,2,21}$ $x_{16,2,22}$ $x_{16,2,23}$ $x_{16,2,24}$ $x_{16,2,25}$ $x_{16,2,26}$ $x_{16,2,27}$ $x_{16,2,31}$

$x_{16,2,32}$ $x_{16,2,33}$ $x_{16,2,34}$ $x_{16,2,35}$ $x_{16,2,36}$

$x_{17,2,28}$ $x_{17,2,29}$ $x_{17,2,30}$ $x_{17,2,38}$ $x_{17,2,39}$ $x_{17,2,40}$ $x_{17,3,28}$ $x_{17,3,29}$

$x_{17,3,30}$ $x_{17,3,38}$ $x_{17,3,39}$ $x_{17,3,40}$

$x_{18,3,21}$ $x_{18,3,22}$ $x_{18,3,23}$ $x_{18,3,24}$ $x_{18,3,25}$ $x_{18,3,26}$ $x_{18,3,27}$ $x_{18,3,31}$

$x_{18,3,32}$ $x_{18,3,33}$ $x_{18,3,34}$ $x_{18,3,35}$ $x_{18,3,36}$ $x_{18,3,37}$

$x_{19,1,28}$ $x_{19,1,29}$ $x_{19,1,30}$ $x_{19,1,38}$ $x_{19,1,39}$ $x_{19,1,40}$ $x_{19,3,28}$ $x_{19,3,29}$

$x_{19,3,30}$ $x_{19,3,38}$ $x_{19,3,39}$

$x_{20,3,37}$

## Appendix G

### Type IV Resource Constraints for Large Sample Problem

Most of the first set of type 4 resource constraints are eliminated because the number of decision variables has been reduced. Among the constraints that are still active, certain variables in them can be set to zero during preprocessing (as discussed in 3.2.2, and ultimately, all of the constraints can be eliminated. The constraints are:

1) $x_{5,1,8}+x_{9,1,8}$ <= 1
2) $x_{5,1,9}+x_{8,1,9}+x_{9,1,9}$ < 1
3) $x_{5,1,10}+x_{9,1,10}$ <= 1
4) $x_{5,1,18}+x_{9,1,18}$ <= 1
5) $x_{5,1,19}+x_{9,1,19}$ <= 1
6) $x_{5,1,20}+x_{9,1,20}$ <= 1
7) $x_{11,1,28}+x_{19,1,28}$ <1
8) $x_{11,1,29}+x_{19,1,29}$ <= 1
9) $x_{11,1,30}+x_{19,1,30}$ <= 1
10) $x_{11,1,38}+x_{19,1,38}$ <= 1
11) $x_{11,1,39}+x_{19,1,39}$ <= 1
12) $x_{11,1,40}+x_{19,1,40}$ <= 1
13) $x_{1,2,8}+x_{7,2,8}$ <= 1
14) $x_{5,2,8}+x_{7,2,8}$ <= 1
15) $x_{5,2,9}+x_{7,2,9}$ <= 1
16) $x_{5,2,10}+x_{7,2,10}$ <= 1
17) $x_{14,2,21}+x_{16,2,21}$ <= 1
18) $x_{14,2,22}+x_{16,2,22}$ <= 1
19) $x_{14,2,22}+x_{16,2,23}$ <= 1
20) $x_{14,2,24}+x_{16,2,24}$ <= 1
21) $x_{14,2,25}+x_{16,2,25}$ <= 1
22) $x_{14,2,26}+x_{16,2,26}$ <= 1
23) $x_{14,2,27}+x_{16,2,27}$ <= 1
24) $x_{15,2,28}+x_{17,2,28}$ <= 1
25) $x_{13,2,29}+x_{15,2,29}+x_{17,2,29}$ <= 1
26) $x_{15,2,30}+x_{17,2,30}$ <= 1
27) $x_{16,2,37}+x_{20,2,37}$ <= 1
28) $x_{13,2,38}+x_{15,2,38}+x_{17,2,38}$ <= 1
29) $x_{13,2,39}+x_{15,2,39}+x_{17,2,39}$ <= 1
30) $x_{13,2,40}+x_{15,2,40}+x_{17,2,40}$ <= 1
31) $x_{15,3,28}+x_{19,3,28}$ <= 1
32) $x_{15,3,29}+x_{19,3,29}$ <= 1
33) $x_{13,3,30}+x_{19,3,30}$ <= 1
34) $x_{13,3,38}+x_{15,3,38}+x_{19,3,38}$ <= 1
35) $x_{13,3,39}+x_{15,3,39}+x_{19,3,39}$ <= 1

In constraint 1 above, set $x_{5,1,8}=1$. This forces $x_{9,1,8}$ and $x_{5,k,t}$ to be zero, thus eliminating constraints 1, 3-6, and 13-16. Similarly, the following settings eliminate the indicated constraints:

| Settings | Constraints Eliminated |
|---|---|
| $x_{8,1,17}=1$ : | 2 |
| $x_{11,1,29}=1$: | 7-12 |
| $x_{14,2,23}=1$: | 7-22 |
| $x_{16,2,27}=1$: | 23 |
| $x_{20,2,37}=1$: (by default) | |
| $x_{13,3,30}=1$ and $x_{15,3,38}=1$: | 24-26, 28-35. |

Thus, constraints 1-35 have been eliminated.

Only four constraints from the second set of resource constraints are still active. These constraints include

36) $x_{18,3,21}+x_{7,3,19}+x_{7,3,20} <= 1$
37) $x_{18,3,22}+x_{7,3,20} <= 1$
38) $x_{18,3,31}+x_{19,3,30} <= 1$
39) $x_{18,3,32}+x_{19,3,30} <= 1$

## Bibliography


1. Baker, Bruce N. and Rene L. Eris. An Introduction to PERT-CPM. New York: Richard D. Irwin, Inc., 1964.

2. Borsi, Captain John, Instructor of Management Science, The Air Force Institute of Technology, personal interview, Wright-Patterson AFB, OH, Feb 91.

3. Burman, P.J. Precedence Networks for Project Planning and Control. London: McGraw-Hill Book Company, 1972.

4. Crowder, H., E.L. Johnson, and M. Padberg. "Solving Large-Scale Zero-One Linear Programming Problems," Operations Research, 31: 803-834, 1983.

5. Davis, Edward W. "Project Scheduling Under Resource Constraints--Historical Review and Categorization of Procedures," AIIE Transactions, 5,4: 297-313 (December 1973).

6. Davis, E.W. and G.E. Heidorn. "An Algorithm for Optimal Project Scheduling Under Multiple Resource Constraints," Management Science, 17: 803-816 (August 1971).

7. Department of the Air Force. Test and Evaluation. AFR 80-14. Washington, HQ USAF: 3 November 1986.

8. Emmons, Hamilton et al. STORM Personal Version 2.0: Quantitative Modeling for Decision Support. Oakland, CA: Holden-Day, Inc., 1989.

9. French, Simon. Sequencing and Scheduling: An Introduction to the Mathematics of the Job Shop. London: Ellis Harwood Ltd, 1982.

10. Hartley, Ronald V. Operations Research: A Managerial Emphasis. Pacific Palisades, CA: Goodyear Publishing Company, Inc., 1976.

11. Hillier, Frederick S. and Gerald J. Lieberman. Introduction to Mathematical Programming. New York: McGraw-Hill Publishing Company, 1990.

12. Isbell, Major Daniel, Instructor, Test Management. Telephone Interviews. USAF Test Pilot School, Edwards AFB, CA, 9 August and 7 September 1990.

13. Isbell, Major Daniel, Instructor, Test Management. Personal Interview. USAF Test Pilot School, Edwards AFB, CA, 20 September 1990.

14. Johnson, E.L., M.M. Kostreva, and U.H. Suhl. "Solving 0-1 Integer Programming Problems Arising From Large-Scale Planning Models," Operations Research, 17: 803-816 (1985).

15. Lawler, E.L. et al. Sequencing and Scheduling: Algorithms and Complexity. Report BS-R89xx. Amsterdam, The Netherlands: Centre for Mathematics and Computer Science, 1985.

16. Loomis, Mary E. Resource-Constrained Scheduling of Tasks with Precedence Relationships. IBM Research Report No. RJ1746. Yorktown Heights, NY: 30 March 1976.

17. Mazzola, Joseph B. and Alan W. Neebe. "Resource-Constrained Assignment Scheduling," Operations Research, 34: 560-571 (July-August 1986).

18. Moler, Cleve. MATLAB Users' Guide. Technical Report CS81-1. University of New Mexico, Albuquerque, NM: August 1988.

19. Nemhauser, George L. an Lawrence A. Welsey. Integer and Combinatorial Optimization. New York: John Wiley & Sons, Inc., 1988.

20. Norbis, Mario I. and Macgregor Smith. "Two Level Heuristic for the Resource Constrained Scheduling Problem," International Journal for Production Research, 24: 1203-1218 (1986).

21. Patterson, James H. "A Comparison of Exact Approaches for Solving the Multiple Constrained Resource, Project Scheduling Problem," Management Science, 30: 854-867 (July 1984).

22. Phillips, Don T. and Alberto Garcia-Diaz. Fundamentals of Network Analysis. Englewoood Cliffs, NJ: Prentice-Hall, Inc., 1981.

23. Pritsker, A.A.B., L.J. Watters, and P.M. Wolfe. "Multiproject Scheduling with Limited Resources: A Zero-One Programming Approach," Management Science, 16,1: 93-108 (September 1969).

24. Plebani, Louis J., Jr. "A Heuristic for Multiple Resource Constrained Scheduling," Production and Inventory Management: 65-80 (First Quarter, 1981).

25. Sasscer, Major Mark, Instructor. Telephone Interview. USAF Test Pilot School, Edwards AFB, CA, 7 September 1990.

## Vita

Captain Lisa M. Hassel was born on [redacted]n Berea, Ohio. She graduated from Brunswick High School in Ohio in 1982 and received an Air Force ROTC scholarship to attend the University of Akron in Akron, Ohio. In 1986, she received the degree of Bachelor of Science in Mathematics and a commission in the USAF. Her first assignment was as a requirements analysis officer for the Deputy Chief of Staff for Materiel Management in the Headquarters, Air Force Logistics Command at WPAFB, Ohio. She entered the Air Force Institute of Technology in August of 1989.

[redacted]