3-2000

A Greedy Multiple-knapsack Heuristic for Solving Air Mobility Command's Intratheater Airlift Problem

Nicholas J. Zeisler

A GREEDY MULTIPLE-KNAPSACK HEURISTIC FOR SOLVING AIR MOBILITY COMMAND'S INTRATHEATER AIRLIFT PROBLEM

THESIS

Nicholas J. Zeisler, 1Lt, USAF

AFIT/GOR/ENS/00M-21

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 2000	Master's Thesis

1. **TITLE AND SUBTITLE**

A GREEDY MULTIPLE-KNAPSACK HEURISTIC FOR SOLVING AIR MOBILITY COMMAND'S INTRATHEATER AIRLIFT PROBLEM

* **FUNDING NUMBERS**

***AUTHOR(S)**

Nicholas J. Zeisler, First Lieutenant, USAF

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 P Street, Building 640
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GOR/ENS/00M-21

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

HQ AMC/XPY
402 Scott Drive, Unit 3L3
Scott AFB, IL 62225-5307
DSN: 576-5954

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

12. **SUPPLEMENTARY NOTES**

Dr. James T. Moore, AFIT/ENS (James.Moore@afit.af.mil) (937) 255-6565

12a. DISTRIBUTION / AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

ABSTRACT (*Maximum 200 Words*)

This research develops a methodology using a greedy heuristic to solve Air Mobility Command's intratheater airlift scenario as a multiple knapsack problem. The objective is to maximize throughput in a theater given a vehicle mixture and assignment scheme. The model allows for a heterogeneous, user defined vehicle mix in a theater consisting of up to five bed down locations and up to seven forward operating locations. First, we preprocess routes, eliminating the large number of unattractive route choices in the problem. Then using a greedy heuristic, we select routes and assign them to aircraft located at any or all of the bed down locations. The model is tested by measuring the utilization rate of the vehicles as well as the maximum throughput of the scenario and the equality of distribution to the receiving bases.

14. SUBJECT TERMS
Intratheater Airlift, Heuristics, Knapsack Problem, Greedy Heuristics

15. NUMBER OF PAGES 58

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	UNC

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government.

AFIT/GOR/ENS/00M-21

A GREEDY MULTIPLE-KNAPSACK HEURISTIC FOR SOLVING AIR MOBILITY COMMAND'S INTRATHEATER AIRLIFT PROBLEM

THESIS

Presented to the Faculty

Department of Operational Sciences

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Operations Research

Nicholas J. Zeisler, B.S.

First Lieutenant, USAF

March 2000

AFIT/GOR/ENS/00M-21

A GREEDY MULTIPLE-KNAPSACK HEURISTIC FOR SOLVING AIR MOBILITY COMMAND'S INTRATHEATER AIRLIFT PROBLEM

Nicholas J. Zeisler, B.S.
First Lieutenant, USAF

Approved:

James T. Moore — 6 Mar 00

James T. Moore, Ph.D. (Advisor) — date
Associate Professor of Operations Research
Department of Operational Sciences
Air Force Institute of Technology

Raymond R. Hill — 6 Mar 00

Raymond R. Hill, Major, USAF (Reader) — date
Assistant Professor of Operations Research
Department of Operational Sciences
Air Force Institute of Technology

Acknowledgements

I would like to thank, first of all, all members of my thesis committee, past and present. Also, I thank all the instructors at AFIT who have been there to help this past year. Thank you as well to all my peers, my fellow AFIT students for your inspiration, your motivation, and your professional and personal support this past year and a half. Of course, I could not have done this without the support of my family and friends, and to them I am eternally grateful.

I dedicate this thesis and the past year and a half of work to Cody. Eighteen months is a long time, and it has been a lot of work. But some things are forever.

Table of Contents

List of Figures

List of Tables

Abstract

This research develops a methodology using a greedy heuristic to solve Air Mobility Command's intratheater airlift scenario as a multiple knapsack problem. The objective is to maximize throughput in a theater given a vehicle mixture and assignment scheme. The model allows for a heterogeneous, user defined vehicle mix in a theater consisting of up to five bed down locations and up to seven forward operating locations. First, we preprocess routes, eliminating the large number of unattractive route choices in the problem. Then using a greedy heuristic, we select routes and assign them to aircraft located at any or all of the bed down locations. The model is tested by measuring the utilization rate of the vehicles as well as the maximum throughput of the scenario and the equality of distribution to the receiving bases.

A GREEDY MULTIPLE-KNAPSACK HEURISTIC FOR SOLVING AIR MOBILITY COMMAND'S INTRATHEATER AIRLIFT PROBLEM

I. Background and Statement of the Problem

Background

A key Air Mobility Command (AMC) task is the effective assignment of intratheater airlift. The US Air Force cannot execute its combat mission effectively without the coordination of air traffic both *to* the location and *within* the location. Once the materials needed to support a commander's plan begin arriving in theater, the movement of supplies to exactly where they are needed begins.

A wise and prudent strategy for moving assets within a theater is therefore necessary. Accurate and meaningful measurements of strategy effectiveness are also needed. More specifically, given a mixture of aircraft sent to the theater, how much cargo can be moved within some specified period of time?

One of the most pressing political and financial questions posed is how many of which types of vehicles to procure. To this end, an accurate estimation of the capabilities of certain mixtures of vehicles is necessary to fill a part of the much bigger picture of procurement.

Problem Statement

During day-to-day operations, AMC schedules cargo shipments around the globe. In this sense, airlift is routine. However, when contingency operations are executed in a theater, questions arise as to the limitations of AMC's intra-theater abilities. Specifically, commanders want to know the maximum amount of cargo that can be moved from the aircraft bed down locations to the forward operating locations (FOLs), given some mix of cargo aircraft deployed in the theater.

Analysts need a quick running tool to find good answers to commander's questions regarding intratheater airlift capabilities. This tool must be simple to use, find solutions quickly, but offer a relatively realistic view of theater capabilities. We develop such a tool in Excel for Windows® using Visual Basic for Applications (VBA) to extend Excel's capabilities.

Vehicle and Theater Details

Our problem is to maximize the total amount of throughput based on an elective mix of vehicles located within a specific theater. This theater is defined by characteristics such as location and number of bed down locations, location and number of FOLs, the costs of transporting cargo between them, and which routes are available between all locations.

Common Vehicles

AMC has a wide variety of vehicles for intratheater airlift. The C-17 Globemaster II can fly directly to the FOLs or deliver cargo to the bed-down locations for later distribution. The C-141B Starlifter has a limited role in intratheater airlift although AMC

often employs it in medical evacuation. Finally, the C-130 Hercules has undergone several modifications in its 45 year history. The latest upgrade, the C-130J, has superior range and speed compared to the earlier models and is held out as the latest in tactical airlift technology (AMC Website).

Common Theater Attributes

Unlike AMC's regular cargo shipments, intratheater airlift deals specifically with distributing cargo from bed down locations to FOLs within a limited radius. A bed down location, also known as the point of debarkation (POD), is a stationing point from which to orchestrate theater supply distribution. The first move for cargo is from a stateside location to one of these PODs. Once it arrives in theater, the cargo is distributed among the FOLs. Two important assumptions are that all cargo is delivered and there are no ceilings on the amount of cargo any of the FOLs can receive.

Although we wish to solve the problem for any theater, it is instructive to recognize some of the characteristics of a typical theater. Two theaters specifically are available for observation, those of Southwest Asia and the Korean peninsula. In the Southwest Asia theater, there are generally five bed down locations and five FOLs. The usual distance from a bed down location to an FOL is about 200 miles. In Korea, the locations generally number the same but are normally 100 miles apart.

Since theaters can differ greatly, our Excel model must accommodate theater specifics and estimate maximum throughput for a large variety of bed down location and FOL combinations in these various theaters.

Flying Hours and Crew Numbers

Regulations closely control the total amount of flying time allowed for a crew or a plane in times of both peace and war. These times include some time on the ground. The clock starts when the aircraft initially pushes back and taxis. The clock does not stop until the plane has come to a complete stop in the chalks at its destination. For this reason, fliers developed the concept of *block speed*, which takes into account this extra time on the ground. As an automotive analogy, consider the following: when travelling between two distant cities, if a motorist stops for lunch, stops to refuel the car a couple times, and makes a few rest stops, he does not actually achieve the 70 miles per hour (mph) he is driving while on the interstate. Indeed, it may take him six or seven hours to go 350 miles, rather than the five hours it would intuitively take if he was really doing 70 mph the whole time. If it takes him 6.5 hours to travel the 350 miles, his average speed (or *block speed*) is actually about 54 mph.

Similarly, the block speed of a plane is affected by the number of times it stops en route from its origin to its destination. Since these on-ground times are included as flying time, we determine an aircraft's total distance allowable during a time period in terms of its block speed.

Furthermore, the total amount of flying allowed in a period of time is normally dictated in terms of hours. For modeling purposes, we translate this time period (using block speed) into approximate distance. We assume that a vehicle's maximum allowed mileage for a day is decreased by 100 miles for each FOL after the first FOL it visits. For example, if a vehicle can fly 3000 miles in one day, its allowable flight distance is calculated as only 2800 if it must stop at three FOLs during a route.

We do not consider limits on crews for the plane to fly. This is a realistic assumption considering redundancy of crews and length of the crew duty day.

Delivery Assumptions

We attempt to deliver a somewhat even amount of cargo to each FOL over the scope of the entire model. From day-to-day and from flight-to-flight, the amount of cargo needed at each FOL varies. Furthermore, we do not have any minimum delivery requirements for any of the FOLs. In the face of this uncertainty, an even distribution assumption is reasonable.

We further assume that aircraft deliver an even amount of cargo to each visited FOL. For example, if an aircraft visits four FOLs, each FOL receives ¼ of the aircraft's total cargo load. Also, we will consider the amount of cargo carried on each tour to be the same for all vehicles of a particular type. This standardizes the use of *units* of cargo.

An important assumption is the optimal use of theater vehicles. This means no vehicle down time and no halt in operations. Given this model's focus on daily delivery planning, this is reasonable.

A further assumption about delivery is that an aircraft will deliver a percentage of its maximum cargo load depending on the number of FOLs it visits in a route, according to Table 1. One reason for this assumption is that it is not necessarily practical to deliver a full 100% load to a single FOL, and as the number of FOLs in the tour increases, there is more cargo needed to go around. Further reasons for making this assumption will become clear in the discussion of the heuristic in Chapter 3.

Table 1: Percent Take-Off Load by Number of FOLs in Tour

Number of FOLs in Tour	Percent of Full Load Carried
1	85%
2	90%
3	95%
4	100%

Scope

The focus of this research is to determine the maximum amount of cargo deliverable in a defined theater. We want the characteristics of the theater user-defined. Further, we wish to apply the model to any existing or notional theater.

To meet the above goals, we let the problem be defined by user inputs. The user specifies locations for bedding down the vehicles as well as the specific mixes of vehicles stationed at each location. The solution to the problem are the missions flown by the aircraft based on demands and capabilities pre-specified by the user. An example problem is defined in Figure 1.

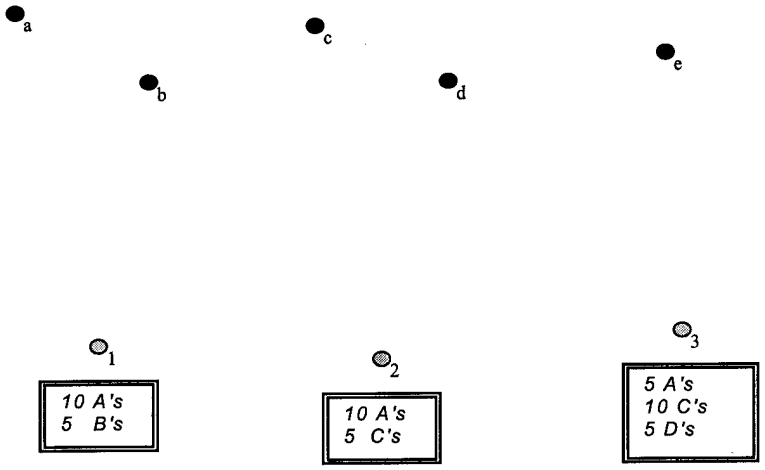

Figure 1: Sample Theater with Vehicle Assignments

Locations 1, 2, and 3 are the bed down locations, and locations a, b, c, d, and e are the FOLs. This example has 4 types of vehicles, A, B, C, and D. The boxes below each bed down location show specific vehicle allocations. Not shown, but crucial to the problem, are the distances between the bed down locations and FOLs. The task is to assign delivery routes to vehicles to maximize throughput for the scenario while balancing deliveries among the FOLs.

This is a routing and assignment problem and can be viewed as a multiple knapsack problem. Knapsack problems have wide applicability in many industries. Heuristic solution methods run quickly and are quite effective on problems such as the knapsack problem. We discuss this further in Chapter 2.

In Chapter 3, we spell out our methodology and the specific applications of knapsack heuristics in this problem. We also discuss the formulation and measurements used. In Chapter 4, we review our results, and draw our conclusions. Chapter 5 provides for suggestions for further study and model improvement.

II. Literature Review

The Knapsack Problem

Avid outdoorsmen and campers often face this situation: a weekend away in the mountains, a knapsack, a collection of supplies to take, and a limited amount of space in the knapsack. Our mountaineer wants to know which items to take and which items to leave behind. From a flashlight to a camera, each article has a value based on its utility, but each article also takes up kanpsack space. Certain cookware might be nice for breakfast in the morning, but are quite bulky, while the tiny compass is invaluable. The right balance of "bang for the space" is important to make his weekend enjoyable. This problem (the classical *knapsack problem*) is formulated mathematically as follows:

$$\text{maximize } \sum_{i=1}^{i=n} p_i x_i \qquad \textbf{(1)}$$

$$\text{subject to: } \sum_{i=1}^{i=n} c_i x_i \leq s \qquad \textbf{(2)}$$

$$x_i \in \{0,1\},\ i = 1, 2, ..., n. \qquad \textbf{(3)}$$

Here, p_i represents the benefit (or profit) gained by including item i in the knapsack, c_i represents the cost (weight or volume) of the item, and s represents the total capacity of the knapsack. When the value of x_i is 1, the item is included on the trip, and when x_i is 0, it is not. There are n total items being considered here.

This is a classic problem in operations research. A natural extension allows multiple copies of items in the knapsack. This adaptation is called the *bounded* knapsack problem. In the bounded knapsack problem, equation (3) is replaced by:

$$0 \le x_i \le b_i,\ i = 1, 2, ..., n, \qquad \textbf{(4)}$$

and we also add:

$$x_i \in \square,\ x_i \ge 0,\ \ i = 1, 2, ..., n. \qquad \textbf{(5)}$$

Here, b_i represents the bound (highest number available) for resource i. Notice now that x_i can take on non-negative integer values rather than just binary, 0 or 1, values.

This formulation can be further modified to describe an *unbounded* knapsack problem when some (or several) $b_i = +\infty$.

When there are multiple campers or multiple knapsacks, the bounded multiple knapsack problem can be formulated as:

$$\text{maximize } \sum_{i=1}^{i=n} \sum_{j=1}^{j=m} p_{ij} x_{ij} \qquad \textbf{(6)}$$

$$\text{subject to: } \sum_{i=1}^{i=n} c_{ij} x_{ij} \le s_j\ ,\ j = 1, 2, ..., m, \qquad \textbf{(7)}$$

$$\sum_{j=1}^{j=m} x_{ij} \le b_i\ ,\ i = 1, 2, ..., n, \qquad \textbf{(8)}$$

$$x_{ij} \in \square,\ x_{ij} \ge 0. \qquad \textbf{(9)}$$

Now with the variables doubly subscripted, the i,j combination refers to placing item i in knapsack j. There are still n types of items, but they are now distributed among m knapsacks, each with capacity, s_j. If (8) has a $b_i = \infty$ then the formulation is called the *(un)bounded multi-knapsack problem* (Winston). A more thorough review of the

knapsack problem and its relationship to problems such as the generalized assignment problem (GAP) and scheduling problems is found in Martello and Toth (1990), Pirkul (1987), and Pinedo (1995).

Heuristic Approaches

There are many ways to solve (6) through (9) to optimality. Specifically, specialists such as Bellman, with his dynamic programming method and Gomory's (1966) cutting-plane algorithm have contributed significantly to the field with successful results dating back to the 1950s and 1960s. Additionally, Kolesar advocated branch-and-bound algorithms in the late sixties which became the main focus of study through most of the next decade as well. Unfortunately, this problem can become very difficult to solve for large-scale instances. This led to many near-optimal algorithms such as Dantzig's (1957) relaxation approach to providing an upper bound. The multi-knapsack problem is described as NP-hard (Kan, et al.), so many realistic forms of the problem require an unwieldy amount of computing time to solve (Martello and Toth, 1990).

Heuristics, while not solving to optimality, seek reasonable solutions in reasonable computing time. One of the earliest heuristics was the Senju and Toyoda greedy heuristic, which uses an "effective gradient" approach to solve a multi-dimensional knapsack problem. (Senju and Toyoda, 1968) This is a knapsack problem wherein we are constrained by more than one limitation. For example, our hiker might have a weight limit as well as a size limit for his knapsack. What Toyoda (1975) describes as the "primal effective gradient" method selects items, based on their relative benefit to the overall solution, filling the knapsack, always maintaining a feasible

solution, until the knapsack is full. While this is applied in a multi-dimensional knapsack heuristic, the concept has merit even in the example such as ours, where we have multiple single-dimensional knapsacks. This approach begins with empty knapsacks and adds items until any more items would cause an infeasible solution.

Greedy Heuristic for the Knapsack Problem

Generally, greedy heuristics are the simplest methods for solving optimization problems. The basic concept is to choose the most immediately attractive move from the current position, and continuously move in an improving direction. In this sense, it is a myopic approach, but sometimes can yield good results. In fact, there are some specific types of problems wherein a greedy algorithm will guarantee an optimal solution (Martello and Toth, 1990).

In a bounded multiple knapsack problem, we can use a constructive algorithm treating each knapsack constraint independently. Our "bang for the buck" ratio is the ratio:

$$r_{ij} = \frac{p_{ij}}{c_{ij}}. \qquad \textbf{(10)}$$

These ratios are ordered such that:

$$r_{1j} \geq r_{2j} \geq \ldots \geq r_{nj}, j = 1, 2, \ldots, m \qquad \textbf{(11)}$$

We begin with a solution where $x_{ij} = 0$ for all i and j. Then we increase the values of the x_{ij} in decreasing order according to equation (11). We increase x_{ij} until its bound or the knapsack bound is reached and then continue on to $x_{i+1,j}$. We continue in this fashion until we run out of room in each knapsack or run out of knapsacks.

With this understanding of the (un)bounded multiple knapsack problem, we move on to our specific application which includes an additional constraint for the even distribution of jobs and the precise heuristic scheme for the problem at hand.

III. Methodology

Solution Overview

The user of this model seeks a maximal throughput given a certain vehicle mix for a specific theater. The model makes use of vehicle capacities and capabilities and assigns routes to the vehicles to maximize throughput. The problem is a multiple knapsack problem in which the model finds a solution by solving the route assignment for each vehicle at each location. The knapsack is the route capacity of each vehicle.

Input

First, the user inputs the theater definition. The user can specify up to five bed down locations and up to seven FOLs, making these choices according to their corresponding International Civil Aviation Organization (ICAO) listing. This is a four-character designator for every airfield in the world.

Once the theater is defined, the user assigns vehicles to bed down locations. Given the large diversity of vehicles, we generalize vehicles requiring the user to define the vehicles simply in terms of maximum allowed flight distance and maximum allowed cargo load. This allows for changes in the capacities of current vehicles and also makes the model dynamic enough to allow for the use of other vehicles.

The user may specify the maximum FOLs to visit in one tour. The default is that no pilot will stop at more than four FOLs in one tour. However, due to weather or other

factors, we may wish to limit this even further. A limit of four FOLs per tour means we only consider tours with four, three, two, or one (out-and-backs) stop.

We also provide the option to even out distribution among FOLs. This departure from a traditional multiple knapsack problem adds the further restrictions to even out the amount of cargo delivered to each FOL. Non-even cargo distribution is useful to derive an upper-bound for throughput in a specific scenario. However, the practicality of this option is limited as it is normally not beneficial to allow great disparity between the amounts of cargo delivered to each FOL.

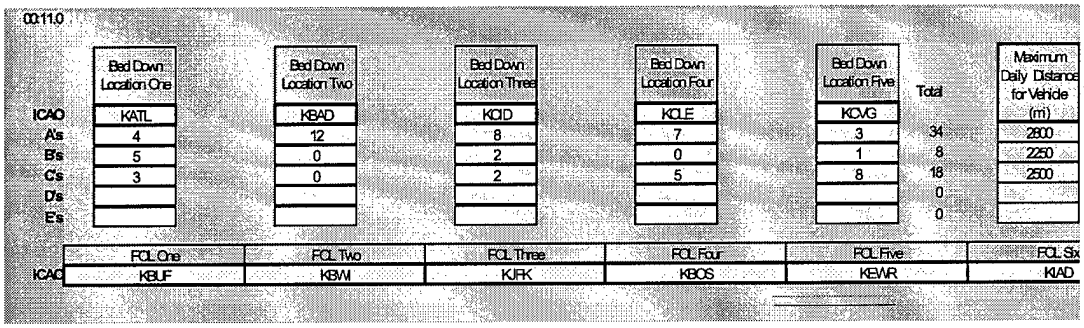

Figure 2: Screen Capture of Input

Once the theater and vehicle mix is input, we are ready to calculate routes and find a solution.

Output

Model output is summarized as throughput, aircraft utilization, and FOL distribution parity. Specific output for each vehicle are the routes it is assigned to fly.

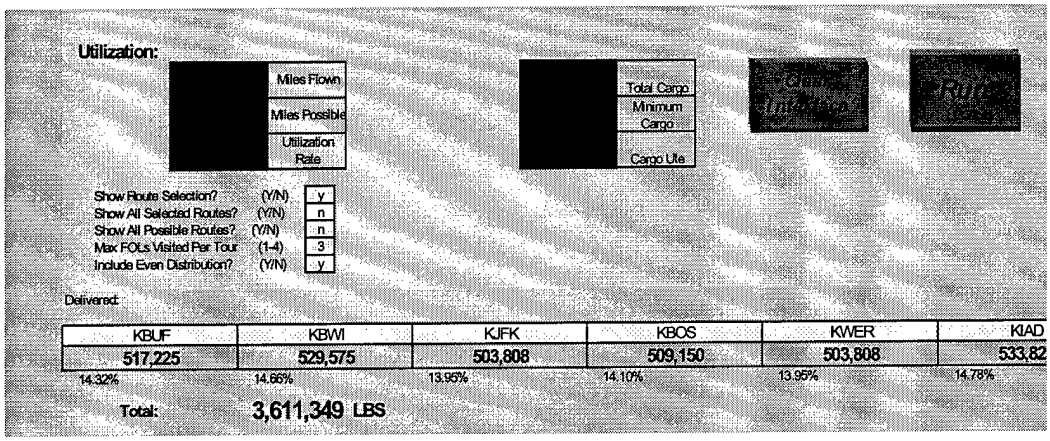

Figure 3: Screen Capture of Output

Model Procedure

The model matches airplanes assigned to bed down locations to routes to maximize throughput. Each vehicle has a distance capacity, each route assigned to the vehicle uses up vehicle flying miles. The throughput is the primary benefit of the route.

Two steps tie the input to the output and help to solve the problem. The first is to develop routes. This procedure constructs possible routes for our vehicles by preprocessing. The second step implements a constructive greedy heuristic to solve each knapsack problem assigning routes to vehicles.

Route Development

Considering all possible routes yields an unmanageable number of routes. Wise preprocessing reduces the considered routes to a manageable number.

Variable Definitions

Part of the user input is the capabilities and quantities of each vehicle type. A vehicle's limitations can be used to remove routes from consideration when the route cannot be handled by the vehicle.

Preprocessing of Routes

Given the largest scenario, a dozen vehicles stationed at each of the five possible bed down locations, servicing all seven possible FOLs, yields a myriad of routes to consider. One airplane at one of the bed down locations has 13,699 possible routes to consider (see Table 2). When we look at a dozen vehicles at each of five bed down locations, the entire problem becomes choosing routes from a total of $13{,}699 \times 12 \times 5 = 821{,}940$. Preprocessing reduces this number.

As a default, no pilot stops more than four times on one route. Therefore, routes with more than four stops are removed.

Next, we consider those routes that are simply mirror images, or reversals, of each other. For example, route 4-5-6 ≡ 6-5-4, and 4-5 ≡ 5-4. We can make this statement because we are assuming that we have symmetry in our distance matrix and that external conditions have little impact on the time it takes to fly a route. Our reasoning in making this assertion is that working in theater, the distances are normally short enough that considerations like winds, which might make a significant difference on longer routes, have little affect on our calculations.

Table 2: Number of Possible Routes; 1 Vehicle, 1 Bed Down Location, 7 FOLs

Number of FOLs Visited:	Possible Route Combinations:
1	$7 = 7$
2	$7 \times 6 = 42$
3	$7 \times 6 \times 5 = 210$
4	$7 \times 6 \times 5 \times 4 = 840$
5	$7 \times 6 \times 5 \times 4 \times 3 = 2520$
6	$7 \times 6 \times 5 \times 4 \times 3 \times 2 = 5040$
7	$7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1 = 5040$
Total	13,699

Further, routes that are merely permutations of each other can be replaced with the shortest of all permutations. Table 3 gives such an example. Each route delivers the same amount of cargo to the same FOLs. Therefore, it is redundant to consider all of them when selecting routes in our heuristic.

Table 3: Permutation of Three FOLs

Order of FOL Visit	Mirror Image
A-B-C	C-B-A
B-C-A	A-C-B
C-A-B	B-A-C

Table 4 shows the results of these reductions. The problem now chooses between $98 \times 12 \times 5 = 5880$ possible routes. These selections do not account for infeasibilities based on the vehicle constraints specific to the problem.

Table 4: Number of Possible Routes; 1 Vehicle, 1 Bed Down Location, 4 FOLs

Number of FOLs Visited:	Possible Route Combinations:
1	$7 = 7$
2	$\binom{7}{2} = 21$
3	$\binom{7}{3} = 35$
4	$\binom{7}{4} = 35$
Total	98

We further narrow the number of routes by accounting for repetition in terms of vehicle type. At each bed down location, we allow up to a dozen vehicles. In reality, there might only be two or three different types of vehicles stationed there. In this sense, we only need to consider vehicle types.

We assume that there may be a total of five different types of vehicles at each of the bed down locations. We can have *any number* of total vehicles stationed anywhere as long as they fall into these five categories. This further reduction yields a possible total of $98 \times 5 \times 5 = 2450$ routes from which to choose. Any solution we find can easily be

extrapolated by using a two-tier view of the routes. First, we generate the routes available to each vehicle type at each bed down location. Then we multiply the routes assigned to one vehicle "type" by the number of that type of vehicle at the bed down location in question.

Benefit Definition

The solution heuristic uses the benefit-to-cost ratio for each route where the cost of each route is its length. The route benefit, however, is not as easily defined. The goal of the model is to maximize total throughput. However, any vehicles of the same type provides the same amount of benefit to total throughput. This could lead to a situation where the routes selected favor those FOLs nearest a bed down location. Therefore, we add the condition that each FOL receive a similar amount of cargo as all others FOLs.

To achieve this, we use a weighting scheme based on the amount delivered to encourage routes that visit neglected FOLs. For each route, we define an FOL benefit vector indicating the amount delivered to each FOL. Figure 4 shows the distribution of 75 units of cargo to FOL 3, FOL 4, and FOL 5.

0	0	25	25	25	0	0

Figure 4: Sample FOL Benefit Vector

We transform this route benefit vector into one value via a dot product with a weight vector.

To equalize cargo delivery between the FOLs, it makes sense to account for how much has been delivered so far in the weight vector used to determine the quality of a route. If more has been delivered to certain FOLs, we want to de-value routes that visit

those FOLs. To start, we want the weight vector to be equal to a unit vector shown in Figure 5:

1	1	1	1	1	1	1

Figure 5: Beginning Weight Vector

When we begin, the needs of all FOLs are weighted equally. However, we want the weight vector to shrink in size as its corresponding FOL receives more cargo, thereby reducing the benefit of certain routes.

We assign a formula to each element of the weight vector that starts at 1 and decreases as the amount delivered to an FOL is increased. We use the formula:

$$w_{FOL} = \frac{1}{e^{d_{FOL}}}, \qquad \textbf{(12)}$$

where

$$d_{FOL} = \frac{totaldelivered_{FOL}}{100{,}000}, FOL = 1, 2, ..., 7. \qquad \textbf{(13)}$$

The 100,000 constant in equation (13) is used to keep the denominator of equation (12) from growing too quickly. This is based on a normal cargo load of the vehicles for this model being in the range of about 30,000 pounds. In different circumstances, it may be appropriate for us to use a different constant, keeping it at about $^1/_3$ of a total cargo load for the average vehicle.

This weighting vector ensures nearly uniform distribution to all FOLs in the model. The dot product of this vector with each route's individual FOL benefit vector is the assigned weighted benefit for each tour. This benefit is divided by the length of the tour to determine the benefit-to-cost ratio.

Greedy Heuristic Application

In Section 2.2.1, we discussed the use of the greedy heuristic for selecting elements in a knapsack. In our model, the knapsacks are the vehicles assigned to the theater. The limiting factor for our vehicles is the total miles they are allowed to fly per day. We use up these miles by choosing routes but also enjoy a benefit for taking cargo to FOLs. We maximize the benefits received while not exceeding each vehicle's limited resources.

Since we have attributed a single benefit to each route (rather than one for each FOL for each route), the greedy heuristic has application here, where we begin with no routes chosen and then add routes until a plane has used up its allowable miles for the day. We need not completely exhaust the total miles allowed to one vehicle, however, before moving on to the next.

Here we see another reason for the assumption that we made in Chapter 1 regarding take-off loads for our vehicles. Due to the greedy nature of our heuristic, we naturally select out-and-back routes more often. This is because the heuristic would always naturally choose an out-and-back due to its shorter length in spite of any imbalance in terms of cargo delivered to any FOL so far. The benefit of travelling to more than one FOL does not outweigh the cost. When the cargo delivered so far to the FOLs is relatively even, there is no incentive to travel farther than the shortest route, and when there is a significant difference, the heuristic will tend to deliver to the FOL that is lacking cargo. The shortest route to that FOL is naturally an out-and-back. However, delivering slightly less to an out-and-back FOL is a way to alleviate this problem. Therefore, we make the assumptions in Table 1.

Each vehicle is solved separately, although not necessarily sequentially. In fact, the vehicles are effectively solved simultaneously. This simultaneous process facilitates the bookkeeping associated with the benefit calculations as we update the deliveries to each FOL.

Finally, we report the results in a user-friendly format showing the total throughput, which is a reflection of his vehicle mix, and the performance measures of the heuristic.

The entire code is published in Appendix B. Figure 6 shows a flow-chart of the procedure logic.

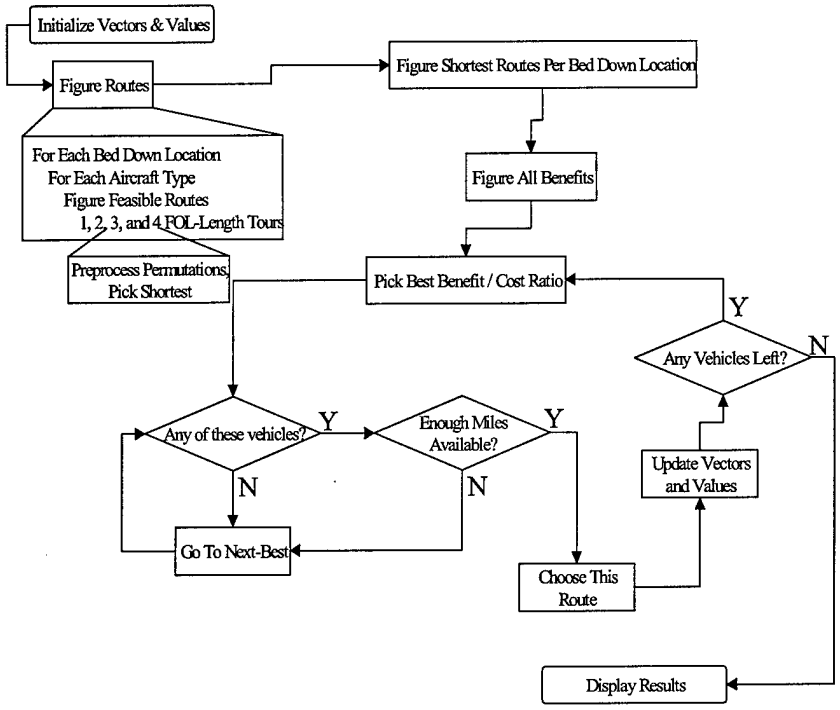

Figure 6: Greedy Heuristic Flowchart

IV. Results and Performance

Performance Measures

The performance of this model can be measured in many ways. Four of them are: throughput, homogeneity of delivery, mileage usage and cargo usage. Each tells a slightly different story and measures a different aspect of the solution.

Throughput

The user is looking to maximize throughput. Total throughput measures the quality of the choice for vehicle assignments. Having the right vehicles in the right places leads to a highly-attractive throughput value. Throughput, therefore, is interpreted as a measure of the amounts and types of vehicles and their placement at the bed down locations.

Departure from Even Delivery

Departure from even delivery refers to the difference between total cargo delivered to any of the FOLs. We want to ensure each FOL is kept "in the loop" and not neglected in terms of cargo. Ideally, each FOL receives the same percentage of the total cargo delivered within the theater. We wish to have a low percentage deviation from its fair share of cargo delivered for each FOL.

Mileage Utilization

Given a type of vehicle and the total number of that type assigned to the bed down location, we know how many miles could possibly be flown throughout the scenario. This is simply the product of the number of this type of vehicle and the total number of miles available per day for that type. The total miles for the scenario is the sum of all miles over each vehicle type.

We define the performance measure of mileage utilization as the percentage of total miles available that were actually flown. This is similar to figuring the percentage of the total space left in our knapsacks once we have filled them.

This is one measure of the effectiveness of our heuristic. Clearly, the closer to 100 percent we are, the more resources we were able to effectively use.

Cargo Utilization

Similar to mileage utilization, cargo utilization is a percentage of the cargo that can be delivered. This number is based on a "least delivered" baseline scenario wherein each vehicle assigned to the theater makes only one out-and-back trip and drops off its entire load at the one FOL it visits. Cargo utilization is the percentage increase over the baseline figures.

Results

Runs and Design

We set up experimental runs with two scenarios: a large-scale and a small-scale scenario. The large-scale scenario has all five bed down locations and all seven FOLs.

Twelve of a random combination of three types of vehicles are assigned to each of the bed down locations.

We remove two of the bed down locations and two of the FOLs to form the small scenario, and again randomly assign a combination of three types of vehicles, twelve total, to each of the bed down locations.

The three vehicle types are determined by their maximum daily range and their maximum cargo load as shown in Table 5.

Table 5: Vehicle Type Characteristics

Vehicle Type	Maximum Daily Range	Maximum Cargo Load
1	2,800	32,500
2	2,250	30,000
3	2,500	31,000

Twenty runs are executed with random vehicle assignments for each of the two scenarios. The maximum number of FOLs visited in one route is set at three by convention and we run the entire experiment within the heuristic once with the evening considerations and once without to produce a bound on a lesser-constrained problem. Then we also run the same lesser-constrained scenarios through an LP to create an optimal upper bound.

The upper bounds must be considered carefully in these scenarios. The first consideration is the even distribution of cargo. This condition is waived to find an upper bound, but it is not practical in real-world situations. Furthermore, in the LP optimal

solution, we allow for some rather creative uses of our vehicles. First, we allow fractional routes. Secondly, we also allow for some infeasibilities in terms of maximum mileage. When solving the LP, we allow ourselves to group all like vehicles at a bed down location into one vehicle, with one constraint for maximum mileage. This may lead to infeasible results in scenarios such as the following: Consider three vehicles of the same type stationed at one bed down location. Each has a 2000 mile limit. Together, there is a limit of 6000 miles. However, a solution that chooses to travel a 3000 mile route twice would be infeasible for our more constrained problem.

Results

Table 6 displays the averages of the runs. A comprehensive list of results can be found in Appendix A. Note that although there is a higher throughput on the lesser-constrained scenarios, the departure from even is unacceptable. In most cases, all the cargo is delivered to only two FOLs.

Table 6: Experimental Results

Run Size	Throughput (lbs)	Departure from Even
Small	1,415,904	1.3%
Small (Loose)	1,555,861	24%
Small (LP Opt)	2,026,622	24%
Big	3,601,503	0.3%
Big (Loose)	5,226,104	19.47%
Big (LP Opt)	5,923,229	19.22%

The throughput for the small scenario is 70% of the optimal for the lesser-constrained problem, and for the big scenario, the figure is 61%.

Clearly, however, the solutions for both the heuristic and LP lesser-constrained problems are not acceptable by virtue of the graphs in Figure 7, which show the distribution of cargo for all scenarios.

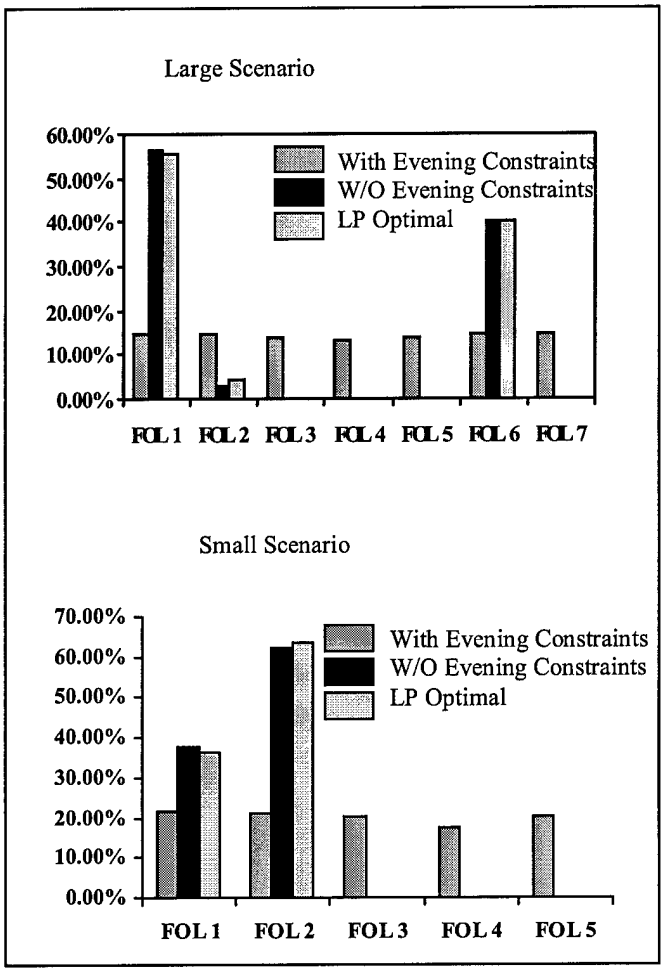

Figure 7: Distributions to FOLs for Scenarios

The mileage and cargo utilization for our heuristic runs are shown in Table 7. The results suggest that although cargo utilization increases when there are no evening constraints, the mileage utilization is slightly lower.

Table 7: Mileage and Cargo Utilization

Run Size	Mileage Utilization	Cargo Utilization
Big (w/ Evening)	85.13%	190.11%
Big (w/o Evening)	84.07%	275.86%
Small (w/ Evening)	82.75%	124.51%
Small (w/o Evening)	76.13%	136.86%

It is intuitive that without the evening constraints, the heuristic would ultimately deliver more cargo. This is simply due to its greedy nature. However, the higher mileage utilization when not using the constraints is a result of forcing the heuristic to take longer routes in order to evenly distribute the cargo.

Conclusions

As a quick-look tool, our heuristic offers a fast answer to the question of maximum throughput. Although not necessarily optimal, we observe results that keep our crews and aircraft gainfully employed for over 80% of their day and deliver a near even amount of cargo to our FOLs.

The heuristic employed gives a sufficient estimate of the throughput and mileage utilization for the aircraft in the scenarios. Its employment of an evening function in

terms of a weighting vector ensure that each FOL receives nearly identical amounts of cargo throughout the run of the model.

The easy-to-use format of Excel spreadsheets and already-existing ICAO and distance calculation information makes it very user-friendly. Its use of VBA as a platform aids in upgrade potential. This program can readily be changed to accommodate many new variables and conditions.

Unfortunately, there is not yet a way to deliver different amounts to FOLs on the same trip. Also the assumption that we made in Table 1 could be improved upon, and it may be possible to address the block speed in a more elegant fashion.

V. Further Research

Recommendations

Although our model allows for much flexibility and breadth in the scenarios we choose, there are still some areas where future research might make the model more accurately reflect the real-world scenarios in which AMC finds itself.

One of these suggestions is to allow more freedom in exactly how the cargo is delivered. Now, we deliver the same amount of cargo to each of the FOLs visited in a route. An improvement here could allow for heterogeneous delivery within one trip.

Another useful improvement would be to allow a more flexible use of the block-speed in the model. Although the distances within a theater are small, the differences in length of day as an effect of block speed can be dramatic. In our model, we build the block-speed into the preprocessing of routes in a stringent fashion. A variable treatment of block-speed, and for that matter the distances themselves (given weather and winds), might be another fruitful area for further research.

Real-world scenarios have randomness. The introduction of variability and randomness is an interesting and useful avenue of research. Clearly, ground can be gained here in using this as a part of a perhaps bigger stochastic model.

The final avenue suggested is to expand the model to not only assign routes but to allocate aircraft to bed down locations. In other words, answer the question: "*How* should we mix our vehicles in this theater?"

Appendix A: Run Results

Small Loose

Run	Throughpu	Mileage Ut	Cargo Ute	FOL 1	FOL 2	FOL 3	FOL 4	FOL 5	Dev
1	1464550	73.35%	129.43%	34.24%	65.76%	0.00%	0.00%	0.00%	0.24
2	1526175	80.37%	136.94%	38.71%	61.29%	0.00%	0.00%	0.00%	0.24
3	1453925	71.33%	127.87%	32.62%	67.38%	0.00%	0.00%	0.00%	0.24
4	1638800	75.19%	141.64%	40.15%	59.85%	0.00%	0.00%	0.00%	0.24
5	1524050	76.08%	134.87%	36.98%	63.02%	0.00%	0.00%	0.00%	0.24
6	1431400	74.39%	128.03%	33.49%	66.51%	0.00%	0.00%	0.00%	0.24
7	1587800	76.31%	138.79%	39.88%	60.12%	0.00%	0.00%	0.00%	0.24
8	1575475	75.53%	138.02%	38.31%	61.69%	0.00%	0.00%	0.00%	0.24
9	1583975	76.96%	138.88%	39.98%	60.02%	0.00%	0.00%	0.00%	0.24
10	1513425	77.72%	135.01%	37.41%	62.59%	0.00%	0.00%	0.00%	0.24
11	1566550	79.27%	139.25%	38.63%	61.37%	0.00%	0.00%	0.00%	0.24
12	1631575	75.66%	141.38%	39.39%	60.61%	0.00%	0.00%	0.00%	0.24
13	1523200	77.94%	135.82%	37.17%	62.83%	0.00%	0.00%	0.00%	0.24
14	1618400	78.85%	142.34%	40.65%	59.35%	0.00%	0.00%	0.00%	0.24
15	1612025	78.60%	141.78%	40.97%	59.03%	0.00%	0.00%	0.00%	0.24
16	1568250	75.77%	137.57%	38.16%	61.84%	0.00%	0.00%	0.00%	0.24
17	1491750	75.61%	132.42%	36.30%	63.70%	0.00%	0.00%	0.00%	0.24
18	1622650	72.90%	139.34%	38.71%	61.29%	0.00%	0.00%	0.00%	0.24
19	1544025	74.35%	135.32%	36.66%	63.34%	0.00%	0.00%	0.00%	0.24
20	1639225	76.44%	142.54%	40.45%	59.55%	0.00%	0.00%	0.00%	0.24
	1555861	76.13%	136.86%	37.94%	62.06%	0.00%	0.00%	0.00%	0.24

Small Tight

Run	Throughpu	Mileage Ut	Cargo Ute	FOL 1	FOL 2	FOL 3	FOL 4	FOL 5	Dev
1	1311625	78.93%	115.92%	20.67%	20.38%	20.47%	18.02%	20.47%	0.007937
2	1216950	78.34%	109.19%	20.12%	21.44%	20.56%	17.32%	20.56%	0.010712
3	1379525	80.41%	121.33%	21.35%	21.63%	19.67%	17.68%	19.67%	0.011907
4	1539625	84.16%	133.07%	21.53%	21.20%	19.79%	17.69%	19.79%	0.010925
5	1344350	80.60%	118.97%	20.26%	21.91%	19.97%	17.89%	19.97%	0.008691
6	1355250	83.61%	121.22%	21.73%	21.51%	19.71%	17.34%	19.71%	0.012979
7	1383350	80.20%	120.92%	21.97%	21.38%	19.62%	17.42%	19.62%	0.013398
8	1528850	86.87%	133.93%	21.54%	21.24%	20.59%	16.04%	20.59%	0.015841
9	1353850	80.02%	118.71%	22.45%	19.90%	19.93%	17.80%	19.93%	0.009781
10	1310100	81.56%	116.87%	20.92%	20.44%	20.32%	18.00%	20.32%	0.007983
11	1386575	83.79%	123.25%	20.90%	21.12%	19.46%	19.06%	19.46%	0.00809
12	1453700	80.53%	125.97%	20.73%	20.82%	19.86%	18.74%	19.86%	0.006176
13	1372800	83.31%	122.41%	21.67%	21.52%	21.49%	13.84%	21.49%	0.024646
14	1496500	86.19%	131.62%	22.15%	21.44%	21.00%	14.40%	21.00%	0.022406
15	1433400	84.11%	126.07%	25.05%	18.68%	18.87%	18.53%	18.87%	0.020216
16	1469550	84.42%	128.91%	20.53%	22.01%	20.50%	16.46%	20.50%	0.014177
17	1313675	80.67%	116.62%	20.87%	20.38%	20.38%	17.99%	20.38%	0.008049
18	1603950	85.40%	137.74%	22.39%	20.67%	19.08%	18.79%	19.08%	0.012231
19	1445250	82.72%	126.67%	20.64%	20.79%	20.80%	16.97%	20.80%	0.012129
20	1619200	89.17%	140.80%	22.18%	21.76%	19.58%	16.90%	19.58%	0.015753
	1415904	82.75%	124.51%	21.48%	21.01%	20.08%	17.34%	20.08%	0.012701

Small LP Optimal

Run	Thruput	FOL 1	FOL 2	FOL 3	FOL 4	FOL 5	Dev
1	1956056	34.14%	65.86%	0.00%	0.00%	0.00%	24.00%
2	1869976	38.48%	61.52%	0.00%	0.00%	0.00%	24.00%
3	2020597	32.68%	67.32%	0.00%	0.00%	0.00%	24.00%
4	2152203	36.43%	63.57%	0.00%	0.00%	0.00%	24.00%
5	1983169	35.91%	64.09%	0.00%	0.00%	0.00%	24.00%
6	1911997	35.58%	64.42%	0.00%	0.00%	0.00%	24.00%
7	2032599	38.70%	61.30%	0.00%	0.00%	0.00%	24.00%
8	2088737	36.83%	63.17%	0.00%	0.00%	0.00%	24.00%
9	2013237	39.07%	60.93%	0.00%	0.00%	0.00%	24.00%
10	1929892	37.42%	62.58%	0.00%	0.00%	0.00%	24.00%
11	1953706	34.67%	65.33%	0.00%	0.00%	0.00%	24.00%
12	2129298	34.02%	65.98%	0.00%	0.00%	0.00%	24.00%
13	1967465	36.71%	63.29%	0.00%	0.00%	0.00%	24.00%
14	2048334	38.28%	61.72%	0.00%	0.00%	0.00%	24.00%
15	2023734	39.23%	60.77%	0.00%	0.00%	0.00%	24.00%
16	2060928	36.36%	63.64%	0.00%	0.00%	0.00%	24.00%
17	1947686	37.21%	62.79%	0.00%	0.00%	0.00%	24.00%
18	2197790	34.89%	65.11%	0.00%	0.00%	0.00%	24.00%
19	2071005	34.87%	65.13%	0.00%	0.00%	0.00%	24.00%
20	2174042	36.98%	63.02%	0.00%	0.00%	0.00%	24.00%
	2026622	36.42%	63.58%	0.00%	0.00%	0.00%	24.00%

Big Loose

Run	Throughpu	Mileage Ut	Cargo Ute	FOL 1	FOL 2	FOL 3	FOL 4	FOL 5	FOL 6
1	5602100	85.34%	292.54%	55.84%	3.03%	0.00%	0.00%	0.00%	41.13%
2	5197325	87.56%	276.45%	52.09%	3.16%	0.00%	0.00%	0.00%	44.75%
3	5395350	85.63%	285.39%	61.39%	3.04%	0.00%	0.00%	0.00%	35.56%
4	5077575	83.46%	270.08%	60.13%	3.41%	0.00%	0.00%	0.00%	36.46%
5	5602200	85.17%	293.39%	55.80%	3.05%	0.00%	0.00%	0.00%	41.15%
6	5730250	89.66%	300.01%	56.66%	2.87%	0.00%	0.00%	0.00%	40.47%
7	5415725	85.97%	281.63%	51.23%	3.24%	0.00%	0.00%	0.00%	45.53%
8	5332950	83.86%	281.05%	53.04%	3.24%	0.00%	0.00%	0.00%	43.72%
9	5167550	85.77%	273.05%	55.74%	3.33%	0.00%	0.00%	0.00%	40.93%
10	5153500	82.57%	270.88%	57.99%	3.41%	0.00%	0.00%	0.00%	38.60%
11	5291225	87.66%	281.00%	55.90%	3.10%	0.00%	0.00%	0.00%	40.99%
12	4897850	81.07%	262.62%	53.54%	3.39%	0.00%	0.00%	0.00%	43.08%
13	5387250	84.08%	282.05%	54.83%	3.26%	0.00%	0.00%	0.00%	41.91%
14	5118875	83.00%	271.06%	54.76%	3.39%	0.00%	0.00%	0.00%	41.84%
15	4756400	79.21%	253.27%	57.99%	3.66%	0.00%	0.00%	0.00%	38.35%
16	4858525	83.24%	258.91%	59.20%	3.53%	0.00%	0.00%	0.00%	37.27%
17	5254150	84.46%	277.70%	58.68%	3.25%	0.00%	0.00%	0.00%	38.07%
18	5076875	81.01%	269.33%	60.21%	3.43%	0.00%	0.00%	0.00%	36.36%
19	5003475	80.55%	263.34%	57.08%	3.51%	0.00%	0.00%	0.00%	39.41%
20	5203650	82.06%	273.44%	58.95%	3.37%	0.00%	0.00%	0.00%	37.68%
	5226140	84.07%	275.86%	56.55%	3.28%	0.00%	0.00%	0.00%	40.16%

Big Tight

Run	Throughpu	Mileage Ut	Cargo Ute	FOL 1	FOL 2	FOL 3	FOL 4	FOL 5	FOL 6
1	3821000	86.10%	199.53%	14.40%	14.37%	14.13%	13.61%	14.13%	14.95%
2	3570850	86.32%	189.94%	14.87%	14.62%	14.23%	12.87%	14.23%	14.84%
3	3708200	87.30%	196.15%	14.77%	14.67%	14.16%	13.09%	14.16%	14.73%
4	3422625	83.34%	182.05%	14.22%	14.53%	14.27%	14.01%	14.24%	14.60%
5	3831650	86.40%	200.66%	14.29%	14.76%	14.09%	13.52%	14.09%	14.82%
6	3852125	88.07%	201.68%	14.95%	14.51%	14.45%	12.80%	14.45%	14.57%
7	3746425	85.03%	194.82%	14.41%	14.61%	14.13%	13.84%	14.13%	14.57%
8	3671775	84.64%	193.51%	14.65%	14.83%	14.04%	13.33%	14.04%	14.83%
9	3606325	85.65%	190.56%	14.85%	14.56%	14.27%	13.35%	14.27%	14.53%
10	3487450	82.48%	183.31%	14.05%	14.73%	14.16%	13.83%	14.16%	14.61%
11	3648525	87.28%	193.76%	14.70%	14.68%	14.45%	12.59%	14.45%	14.67%
12	3510775	86.35%	188.25%	14.36%	14.74%	14.39%	12.90%	14.39%	14.77%
13	3662300	84.83%	191.74%	14.11%	14.45%	14.23%	14.05%	14.23%	14.45%
14	3496225	83.34%	185.13%	14.61%	14.74%	13.96%	13.83%	13.96%	14.70%
15	3355775	82.69%	178.69%	14.37%	14.39%	14.14%	13.46%	14.14%	15.05%
16	3366100	84.18%	179.38%	14.29%	14.59%	14.16%	13.41%	14.16%	14.62%
17	3591575	85.28%	189.83%	14.40%	14.51%	14.34%	13.57%	14.28%	14.64%
18	3470700	83.59%	184.12%	14.20%	14.75%	14.07%	13.85%	14.01%	14.77%
19	3598300	85.41%	189.38%	14.30%	14.47%	14.10%	14.13%	14.04%	14.55%
20	3611350	84.23%	189.77%	14.32%	14.66%	13.95%	14.10%	13.95%	14.78%
	3601503	85.13%	190.11%	14.46%	14.61%	14.18%	13.51%	14.17%	14.70%

Big LP Optimal

Run	Thruput	FOL 1	FOL 2	FOL 3	FOL 4	FOL 5	FOL 6	FOL 7	Dev
1	6235015	55.72%	3.84%	0.00%	0.00%	0.00%	40.45%	0.00%	19.31%
2	5763408	53.14%	3.68%	0.00%	0.00%	0.00%	43.18%	0.00%	19.36%
3	6126879	58.65%	3.91%	0.00%	0.00%	0.00%	37.44%	0.00%	19.29%
4	5971749	55.56%	3.93%	0.00%	0.00%	0.00%	40.51%	0.00%	19.29%
5	6233974	55.50%	3.91%	0.00%	0.00%	0.00%	40.58%	0.00%	19.29%
6	6210653	57.05%	3.39%	0.00%	0.00%	0.00%	39.56%	0.00%	19.44%
7	6029721	51.86%	4.44%	0.00%	0.00%	0.00%	43.69%	0.00%	19.14%
8	5989292	53.76%	4.23%	0.00%	0.00%	0.00%	42.01%	0.00%	19.20%
9	5822569	54.13%	4.29%	0.00%	0.00%	0.00%	41.58%	0.00%	19.18%
10	5893025	55.85%	4.55%	0.00%	0.00%	0.00%	39.61%	0.00%	19.11%
11	5868115	55.39%	3.58%	0.00%	0.00%	0.00%	41.03%	0.00%	19.38%
12	5673602	54.06%	3.86%	0.00%	0.00%	0.00%	42.08%	0.00%	19.30%
13	6033695	55.13%	4.44%	0.00%	0.00%	0.00%	40.42%	0.00%	19.14%
14	5815295	54.32%	4.45%	0.00%	0.00%	0.00%	41.23%	0.00%	19.14%
15	5630048	56.05%	4.64%	0.00%	0.00%	0.00%	39.31%	0.00%	19.08%
16	5580796	57.13%	4.43%	0.00%	0.00%	0.00%	38.44%	0.00%	19.14%
17	5911743	57.75%	4.14%	0.00%	0.00%	0.00%	38.11%	0.00%	19.22%
18	5849518	58.16%	4.48%	0.00%	0.00%	0.00%	37.35%	0.00%	19.13%
19	5845448	54.53%	4.58%	0.00%	0.00%	0.00%	40.89%	0.00%	19.10%
20	5980038	56.45%	4.48%	0.00%	0.00%	0.00%	39.06%	0.00%	19.13%
	5923229	55.51%	4.16%	0.00%	0.00%	0.00%	40.33%	0.00%	19.22%

Appendix B: VBA Code

```
'Visual Basic Code for
'AMC Intratheater Airlift Model
'1Lt N.J. Zeisler, AFIT 2000
'
'This code performs a heuristic to search for a maximal routing assignment
'of vehicles within a user-specified theater.
'
'Variable Definitions:
Dim maxMiles, maxTons, total, shortLength, totalmilesflown As Double
Dim Route, veh, shortest, numSelected As Integer
Dim vehmatrix(10, 10), vehmatrixa(10, 10), milesleftmatrix(10, 10), selectedRoutes(1000, 2) As Variant
Dim routelist(2500, 20) As Variant
Dim GlobalVector(10), weight(7), shortVector(5), delivered(7) As Variant
Dim even As Boolean
'This function performs a dot-product on two passed vectors.
Function doproduct(v1, v2)
total = 0
For i = 1 To 7
   total = (total + ((v1(i)) * (v2(i))))
Next i
   doproduct = total
End Function
Function findShortest(BDL)
'This function selects the shortest route from a passed Bed Down Location
'It searches from all routes in the routelist

shortLength = 100000
shortest = 0
a = 1
Do Until routelist(a, 0) = BDL + 10 Or a = Route
   a = a + 1
Loop
Do While routelist(a, 0) = BDL + 10 And a < Route + 1
      If routelist(a, 5) < shortLength Then
         shortLength = routelist(a, 5)
         shortest = a
      End If
      a = a + 1
Loop
findShortest = shortLength
End Function
Sub FigureBenefit(FOLS, TotCargo)
'This subroutine figures the benefit to each FOL for a tour
'It assigns all seven benefits (one associated with each FOL)
'to the routelist array

For ab = 1 To 7
   For fab = 1 To 7
      If routelist(Route, ab) = fab Then
         routelist(Route, fab + 5) = (TotCargo / FOLS)
      End If
   Next fab
Next ab
End Sub
Sub shift(n)
'This subroutine helps to calculate all permutations of routes

Dim x As Variant

x = GlobalVector(1)
For i = 1 To (n - 1)
   GlobalVector(i) = GlobalVector(i + 1)
Next i
```

```
   GlobalVector(n) = x
End Sub
Sub DefVehMat()

' Initialize vehicle matrix:
For cl1 = 1 To 10
   For cl2 = 1 To 10
      vehmatrix(cl1, cl2) = 0
   Next cl2
Next cl1

' Assign vehicle matrix elements from Interface sheet
For r = 1 To 6
   vehmatrix(r, 1) = Sheets("Interface").Cells(r + 3, 3)  'BDL 1
   vehmatrix(r, 2) = Sheets("Interface").Cells(r + 3, 10) 'BDL 2
   vehmatrix(r, 3) = Sheets("Interface").Cells(r + 3, 17) 'BDL 3
   vehmatrix(r, 4) = Sheets("Interface").Cells(r + 3, 24) 'BDL 4
   vehmatrix(r, 5) = Sheets("Interface").Cells(r + 3, 31) 'BDL 5
   vehmatrixa(r, 1) = Sheets("Interface").Cells(r + 3, 3)  'BDL 1
   vehmatrixa(r, 2) = Sheets("Interface").Cells(r + 3, 10) 'BDL 2
   vehmatrixa(r, 3) = Sheets("Interface").Cells(r + 3, 17) 'BDL 3
   vehmatrixa(r, 4) = Sheets("Interface").Cells(r + 3, 24) 'BDL 4
   vehmatrixa(r, 5) = Sheets("Interface").Cells(r + 3, 31) 'BDL 5

Next r

' Add in vehicle properties
For s = 2 To 6
   vehmatrix(s, 6) = Sheets("Interface").Cells(s + 3, 38) 'Veh max dist
   vehmatrix(s, 7) = Sheets("Interface").Cells(s + 3, 45) 'Veh max load
Next s

' Define Miles Left Matrix
For r = 2 To 6
   For s = 1 To 5
      milesleftmatrix(r, s) = vehmatrix(r, 6)
   Next s
Next r
End Sub
Function FigureDist(PossRoute)
Dim dist As Double

distMatrix = Sheets("DISTANCE").Range("B2:P16")

test = True
dist = 0
k = 0
   While test = True

      k = k + 1
      dist = dist + distMatrix(PossRoute(k - 1), PossRoute(k))
      If PossRoute(k + 1) = 0 Then test = False

   Wend

   ' Add the distance back to the origin.
   dist = dist + distMatrix(PossRoute(k), PossRoute(0))
   FigureDist = dist

End Function
Sub nicholas(x)

' Declare variables:
Dim Index(10) As Variant            'Specific Considered Route
Dim presentDist, TourLength As Double

' Get Distance Matrix from Distance Sheet
distMatrix = Sheets("DISTANCE").Range("B2:P16")
```

```
' Set bed down location as "zero" element of route
Index(0) = x

' Initialize vectors
For i = 1 To 9
   Index(i) = 0
   GlobalVector(i) = 0
Next i

' One FOL tours:
bob = 0.85 * maxTons
For i = 1 To 7
   Index(1) = i
   TourLength = FigureDist(Index)
   If TourLength < maxMiles Then
      Route = Route + 1
      For r = 0 To 1
         routelist(Route, r) = Index(r)
      Next r
      routelist(Route, 5) = TourLength
      FigureBenefit 1, bob
      routelist(Route, 13) = veh
      routelist(Route, 15) = Route
   End If
Next i

' Two FOL tours:
If Sheets("Interface").Cells(28, 11) > 1 Then
bob = 0.9 * maxTons
For i = 1 To 7
   Index(1) = i
   For J = (i + 1) To 7
      Index(2) = J
      TourLength = FigureDist(Index)
      If TourLength < (maxMiles - 100) Then
         Route = Route + 1
         For r = 0 To 2
            routelist(Route, r) = Index(r)
         Next r
         routelist(Route, 5) = TourLength
         FigureBenefit 2, bob
         routelist(Route, 13) = veh
         routelist(Route, 15) = Route
      End If
   Next J
Next i
End If

'Three FOL tours:
If Sheets("Interface").Cells(28, 11) > 2 Then
bob = 0.95 * maxTons
For i = 1 To 5
   For J = (i + 1) To 6
      For k = (J + 1) To 7

         'abc
         GlobalVector(0) = x
         GlobalVector(1) = i
         GlobalVector(2) = J
         GlobalVector(3) = k
         GlobalVector(4) = 0
         TourLength = FigureDist(GlobalVector)
         presentDist = TourLength
         For w = 0 To 9
            Index(w) = GlobalVector(w)
         Next w
```

```
            'Shift this tour
            For v = 1 To 2
               shift (3)
               TourLength = FigureDist(GlobalVector)
               If TourLength < presentDist Then
                  presentDist = TourLength
                  For w = 0 To 9
                     Index(w) = GlobalVector(w)
                  Next w
               End If
            Next v

            'acb
            GlobalVector(0) = x
            GlobalVector(1) = i
            GlobalVector(2) = k
            GlobalVector(3) = J
            GlobalVector(4) = 0
            TourLength = FigureDist(GlobalVector)
            If TourLength < presentDist Then
               presentDist = TourLength
               For w = 0 To 9
                  Index(w) = GlobalVector(w)
               Next w
            End If

            'Shift this tour
            For v = 1 To 2
               shift (3)
               TourLength = FigureDist(GlobalVector)
               If TourLength < presentDist Then
                  presentDist = TourLength
                  For w = 0 To 9
                     Index(w) = GlobalVector(w)
                  Next w
               End If
            Next v
            If TourLength < (maxMiles - 200) Then
               Route = Route + 1
               For r = 0 To 3
                  routelist(Route, r) = Index(r)
               Next r
               routelist(Route, 5) = TourLength
               FigureBenefit 3, bob
               routelist(Route, 13) = veh
               routelist(Route, 15) = Route
            End If
      Next k
   Next J
Next i
End If

'Four FOL tours:
If Sheets("Interface").Cells(28, 11) > 3 Then
bob = maxTons
For i = 1 To 4
   For J = (i + 1) To 5
      For k = (J + 1) To 6
         For l = (k + 1) To 7
            'abcd
            GlobalVector(0) = x
            GlobalVector(1) = i
            GlobalVector(2) = J
            GlobalVector(3) = k
            GlobalVector(4) = l
            GlobalVector(5) = 0
            TourLength = FigureDist(GlobalVector)
            presentDist = TourLength
```

```
For w = 0 To 9
   Index(w) = GlobalVector(w)
Next w

' Shift this tour
For v = 1 To 3
   shift (4)
   TourLength = FigureDist(GlobalVector)
   If TourLength < presentDist Then
      presentDist = TourLength
      For w = 0 To 9
         Index(w) = GlobalVector(w)
      Next w
   End If
Next v

'acbd
GlobalVector(0) = x
GlobalVector(1) = i
GlobalVector(2) = k
GlobalVector(3) = J
GlobalVector(4) = l
GlobalVector(5) = 0
TourLength = FigureDist(GlobalVector)
   If TourLength < presentDist Then
      presentDist = TourLength
      For w = 0 To 9
         Index(w) = GlobalVector(w)
      Next w
   End If

'Shift this tour
For v = 1 To 3
   shift (4)
   TourLength = FigureDist(GlobalVector)
   If TourLength < presentDist Then
      presentDist = TourLength
      For w = 0 To 9
         Index(w) = GlobalVector(w)
      Next w
   End If
Next v

'abdc
GlobalVector(0) = x
GlobalVector(1) = i
GlobalVector(2) = J
GlobalVector(3) = l
GlobalVector(4) = k
GlobalVector(5) = 0
TourLength = FigureDist(GlobalVector)
If TourLength < presentDist Then
   presentDist = TourLength
   For w = 0 To 9
      Index(w) = GlobalVector(w)
   Next w
End If

' Shift this tour
For v = 1 To 3
   shift (4)
   TourLength = FigureDist(GlobalVector)
   If TourLength < presentDist Then
      presentDist = TourLength
      For w = 0 To 9
         Index(w) = GlobalVector(w)
      Next w
   End If
```

```
            Next v
            If TourLength < (maxMiles - 300) Then
              Route = Route + 1
              For r = 0 To 4
                routelist(Route, r) = Index(r)
              Next r
              routelist(Route, 5) = TourLength
              FigureBenefit 4, bob
              routelist(Route, 13) = veh
              routelist(Route, 15) = Route
            End If
          Next l
        Next k
      Next J
    Next i
    End If
    End Sub
    Sub FigureAllBenefits()
    Dim eachFOL(7), ones(7) As Variant
    Dim RouteBenefit As Double
    'This subroutine applies the doproduct function to figure the actual
    'benefit for each route based on the total amount of cargo delivered
    'to each FOL so far.

    For v = 1 To 7
      ones(v) = 1
    Next v

    For i = 1 To Route
      RouteBenefit = 0
      For J = 1 To 7
        eachFOL(J) = routelist(i, J + 5)
      Next J
    If even = True Then
      RouteBenefit = doproduct(eachFOL, weight)
    Else
      RouteBenefit = doproduct(eachFOL, ones)
    End If
      cost = routelist(i, 5)
      routelist(i, 14) = RouteBenefit / cost
    Next i
    End Sub
    Sub GetRoutes()

    ' Start with weights of one
    For i = 1 To 7
      weight(i) = 1
    Next i

    ' Initialize variables
    Route = 0
    veh = 0
    maxMiles = 0
    maxTons = 0

    ' Initialize route list
    For i = 0 To 2500
      For J = 0 To 5
        routelist(i, J) = Null
      Next J
      For k = 6 To 12
        routelist(i, k) = 0
      Next k
    Next i

    ' Get vehicle matrix from Interface sheet
    DefVehMat
```

```
' Define Route List:
For BDL = 1 To 5
  If vehmatrix(1, BDL) <> "" Then
    veh = 0
    For acType = 1 To 5
    veh = acType
      If vehmatrix(acType + 1, BDL) <> 0 Then
        maxMiles = vehmatrix(acType + 1, 6)
        maxTons = vehmatrix(acType + 1, 7)
        nicholas (BDL + 10)
      End If
    Next acType
  End If
Next BDL
For i = 1 To 5
    shortVector(i) = findShortest(i)
Next i

FigureAllBenefits

End Sub
Function CheckFeas(rNum)
'This function checks the feasibility of a route.
'This feasibility is determined by the existence of vehicles of the type in
'question as well as (if there are vehicles) the amount of miles left available
'to that vehicle at the bed down location in question.

feas = True
ac = routelist(rNum, 13)
place = routelist(rNum, 0) - 10
routelength = routelist(rNum, 5)
If vehmatrix(ac + 1, place) = 0 Then feas = False Else
If routelength >= milesleftmatrix(ac + 1, place) Then feas = False
CheckFeas = feas
End Function
Sub PickBest()
'This subroutine selects the best beneift out of all of those that are feasible.

HighestBenefit = 0
pick = 0
For a = 1 To Route
  If routelist(a, 14) > HighestBenefit Then
    check = CheckFeas(a)
    If check = True Then
      pick = a
      HighestBenefit = routelist(pick, 14)
    End If
  End If
Next a
numSelected = numSelected + 1
totalmilesflown = totalmilesflown + routelist(pick, 5)
selectedRoutes(numSelected, 1) = pick
vehicletype = routelist(pick, 13)
vehicleplace = routelist(pick, 0) - 10
selectedRoutes(numSelected, 2) = (1 - vehmatrix(vehicletype + 1, vehicleplace) + vehmatrixa(vehicletype + 1,
vehicleplace))
reFigure (pick)
End Sub
Sub reFigure(chosen)
'This subroutine refigures the benefits of all routes based on the
'amount of cargo delivered so far to each FOL.

For i = 1 To 7
  delivered(i) = delivered(i) + routelist(chosen, i + 5)
Next i

vehicle = routelist(chosen, 13)
place = routelist(chosen, 0) - 10
```

```
thistourlength = routelist(chosen, 5)

' Update milesleftmatrix
milesleftmatrix(vehicle + 1, place) = milesleftmatrix(vehicle + 1, place) - thistourlength

' Check to see if this leaves too few miles for this vehicle
If milesleftmatrix(vehicle + 1, place) < shortVector(place) Then
   vehmatrix(vehicle + 1, place) = vehmatrix(vehicle + 1, place) - 1
   milesleftmatrix(vehicle + 1, place) = vehmatrix(vehicle + 1, 6)
End If

' Update weight vector
For i = 1 To 7
   weight(i) = (1 / (Exp(delivered(i) / 100000)))
Next i

' Refigure benefits
FigureAllBenefits
End Sub
Function anyvehiclesleft()
Dim totalsum As Integer

' Check to see if we should go on
' I.e., if there are any vehicles left
totalsum = 0
For ii = 1 To 5
   For jj = 1 To 5
      totalsum = totalsum + vehmatrix(ii + 1, J)
   Next jj
Next ii
If totalsum = 0 Then
   anyvehiclesleft = False
End If
End Function
Sub superSub()

ow = Time

' Initialize values and vectors
numSelected = 0
totalmilesflown = 0
For i = 1 To 7
   delivered(i) = 0
Next i
For J = 1 To 1000
   For v = 1 To 2
      selectedRoutes(J, v) = 0
   Next v
Next J

'Check if user wants to use the evening constraints
If Sheets("Interface").Cells(29, 11) = "y" Or Sheets("Interface").Cells(29, 11) = "Y" Then
even = True
Else: even = False
End If

' Figure Routes
GetRoutes

' Select Routes while there are vehicles left
Do
   PickBest
   totalsum = 0
   For ii = 1 To 5
      For jj = 1 To 5
         totalsum = totalsum + vehmatrix(ii + 1, jj)
      Next jj
   Next ii
```

```
Loop While totalsum > 0

ow = Time - ow
clearAll
displayResults

Sheets("Interface").Cells(1, 1) = ow

End Sub
Sub displayResults()

'Display amount delivered to each FOL
For d = 1 To 7
  Sheets("Interface").Cells(34, ((d - 1) * 7 + 2)) = delivered(d)
Next d

'Display total miles flown:
Sheets("Interface").Cells(18, 6) = totalmilesflown

'Display selected routes
If Sheets("Interface").Cells(26, 11) = "y" Or Sheets("Interface").Cells(26, 11) = "Y" Then
  displaySelectedRoutes
End If

'Display all routes
If Sheets("Interface").Cells(27, 11) = "y" Or Sheets("Interface").Cells(27, 11) = "Y" Then
  displayAllRoutes
End If

'Display selected routes by bed down location and order them
If Sheets("Interface").Cells(25, 11) = "y" Or Sheets("Interface").Cells(25, 11) = "Y" Then
  displayByBDL
End If


End Sub
Sub displayAllRoutes()
ClearRoutesPage
For i = 1 To Route
  For q = 0 To 19
    Sheets("Routes").Cells(i + 1, q + 1) = routelist(i, q)
  Next q
Next i
End Sub
Sub displayByBDL()

ClearBedDownLocationsPages
For i = 1 To numSelected
  fromselected = selectedRoutes(i, 1)
  vehnumber = selectedRoutes(i, 2)
  If routelist(fromselected, 0) = 11 Then
    hrow = hrow + 1
    Sheets("BDL 1, 2, & 3").Cells(hrow + 1, 1) = routelist(fromselected, 13)
    For J = 1 To 4
      Sheets("BDL 1, 2, & 3").Cells(hrow + 1, J + 2) = routelist(fromselected, J)
    Next J
    Sheets("BDL 1, 2, & 3").Cells(hrow + 1, 7) = routelist(fromselected, 5)
    Sheets("BDL 1, 2, & 3").Cells(hrow + 1, 2) = vehnumber
  End If
Next i
hrow = 0
For k = 1 To numSelected
  fromselected = selectedRoutes(k, 1)
  vehnumber = selectedRoutes(k, 2)
  If routelist(fromselected, 0) = 12 Then
    hrow = hrow + 1
    Sheets("BDL 1, 2, & 3").Cells(hrow + 1, 9) = routelist(fromselected, 13)
    For l = 1 To 4
```

```
        Sheets("BDL 1, 2, & 3").Cells(hrow + 1, l + 10) = routelist(fromselected, l)
      Next l
      Sheets("BDL 1, 2, & 3").Cells(hrow + 1, 15) = routelist(fromselected, 5)
      Sheets("BDL 1, 2, & 3").Cells(hrow + 1, 10) = vehnumber
    End If
Next k
hrow = 0
For m = 1 To numSelected
    fromselected = selectedRoutes(m, 1)
    vehnumber = selectedRoutes(m, 2)
    If routelist(fromselected, 0) = 13 Then
      hrow = hrow + 1
      Sheets("BDL 1, 2, & 3").Cells(hrow + 1, 17) = routelist(fromselected, 13)
      For n = 1 To 4
        Sheets("BDL 1, 2, & 3").Cells(hrow + 1, n + 18) = routelist(fromselected, n)
      Next n
      Sheets("BDL 1, 2, & 3").Cells(hrow + 1, 23) = routelist(fromselected, 5)
      Sheets("BDL 1, 2, & 3").Cells(hrow + 1, 18) = vehnumber
    End If
Next m
hrow = 0
For i = 1 To numSelected
    fromselected = selectedRoutes(i, 1)
    vehnumber = selectedRoutes(i, 2)
    If routelist(fromselected, 0) = 14 Then
      hrow = hrow + 1
      Sheets("BDL 4 & 5").Cells(hrow + 1, 1) = routelist(fromselected, 13)
      For J = 1 To 4
        Sheets("BDL 4 & 5").Cells(hrow + 1, J + 2) = routelist(fromselected, J)
      Next J
      Sheets("BDL 4 & 5").Cells(hrow + 1, 7) = routelist(fromselected, 5)
      Sheets("BDL 4 & 5").Cells(hrow + 1, 2) = vehnumber
    End If
Next i
hrow = 0
For k = 1 To numSelected
    fromselected = selectedRoutes(k, 1)
    vehnumber = selectedRoutes(k, 2)
    If routelist(fromselected, 0) = 15 Then
      hrow = hrow + 1
      Sheets("BDL 4 & 5").Cells(hrow + 1, 9) = routelist(fromselected, 13)
      For l = 1 To 4
        Sheets("BDL 4 & 5").Cells(hrow + 1, l + 10) = routelist(fromselected, l)
      Next l
      Sheets("BDL 4 & 5").Cells(hrow + 1, 15) = routelist(fromselected, 5)
      Sheets("BDL 4 & 5").Cells(hrow + 1, 10) = vehnumber
    End If
Next k
OrderRouteDisplay
End Sub

Sub displaySelectedRoutes()
ClearChosenRoutesPage
For howbout = 1 To numSelected
    thisroute = selectedRoutes(howbout, 1)
    For tem = 0 To 20
      Sheets("Selected Routes").Cells(howbout + 1, tem + 1) = routelist(thisroute, tem)
    Next tem
Next howbout
End Sub

Sub ClearInterfacePage()
Sheets("Interface").Cells(18, 6) = 0
For i = 1 To 7
    Sheets("Interface").Cells(34, 7 * (i - 1) + 2) = 0
Next i
End Sub
Sub ClearBedDownLocationsPages()
```

```
Sheets("BDL 1, 2, & 3").Range("A2:G1000").Clear
Sheets("BDL 1, 2, & 3").Range("I2:O1000").Clear
Sheets("BDL 1, 2, & 3").Range("Q2:W1000").Clear
Sheets("BDL 4 & 5").Range("A2:G1000").Clear
Sheets("BDL 4 & 5").Range("I2:O1000").Clear
End Sub
Sub ClearChosenRoutesPage()
Sheets("Selected Routes").Range("A2:P2500").Clear
End Sub
Sub ClearRoutesPage()
Sheets("Routes").Range("A2:P2500").Clear
End Sub
Sub OrderRouteDisplay()
'This subroutine reorders the display of the routes chosen by bed down location

  Sheets("BDL 1, 2, & 3").Select
  Columns("A:G").Select
  Selection.Sort Key1:=Range("A2"), Order1:=xlAscending, Key2:=Range("B2") _
    , Order2:=xlAscending, Key3:=Range("C2"), Order3:=xlAscending, Header:= _
    xlGuess, OrderCustom:=1, MatchCase:=False, Orientation:=xlTopToBottom
  Columns("I:O").Select
  Selection.Sort Key1:=Range("I2"), Order1:=xlAscending, Key2:=Range("J2") _
    , Order2:=xlAscending, Key3:=Range("K2"), Order3:=xlAscending, Header:= _
    xlGuess, OrderCustom:=1, MatchCase:=False, Orientation:=xlTopToBottom
  Columns("Q:W").Select
  Selection.Sort Key1:=Range("Q2"), Order1:=xlAscending, Key2:=Range("R2") _
    , Order2:=xlAscending, Key3:=Range("S2"), Order3:=xlAscending, Header:= _
    xlGuess, OrderCustom:=1, MatchCase:=False, Orientation:=xlTopToBottom
  Range("A2").Select
  Sheets("BDL 4 & 5").Select
  Columns("A:G").Select
  Selection.Sort Key1:=Range("A2"), Order1:=xlAscending, Key2:=Range("B2") _
    , Order2:=xlAscending, Key3:=Range("C2"), Order3:=xlAscending, Header:= _
    xlGuess, OrderCustom:=1, MatchCase:=False, Orientation:=xlTopToBottom
  Columns("I:O").Select
  Selection.Sort Key1:=Range("I2"), Order1:=xlAscending, Key2:=Range("J2") _
    , Order2:=xlAscending, Key3:=Range("K2"), Order3:=xlAscending, Header:= _
    xlGuess, OrderCustom:=1, MatchCase:=False, Orientation:=xlTopToBottom
  Range("A2").Select
  Sheets("Interface").Select
End Sub

Sub clearAll()
ClearRoutesPage
ClearInterfacePage
ClearChosenRoutesPage
ClearBedDownLocationsPages
End Sub
```

Bibliography

"AMC Public Web Site." http://public.scott.af.mil/hqamc. 21 February 2000.

Dantzig, George B., "Discrete-Variable Extremum Problems,"Operations Research(5): 266-277 (1957).

Gilmore, P.C. and R.E. Gomory, "The Theory and Computation of Knapsack Functions," Operations Research(14): 1045-1074 (1966).

Kan, A.H.G., L. Stougie, and C. Vercellis, "A class of generalized greedy algorithms for the multi-knapsack problem," Discrete Applied Mathematics(42): 279-290 (1993).

Martello, Silvano and Paolo Toth. Knapsack Problems Algorithms and Computer Implementations. Chinchester: John Wiley & Sons, 1990.

Pinedo, Michael. Scheduling Theory, Algorithms, and Systems. Englewood Cliffs, New Jersey: Prentice Hall, 1995.

Pirkul, Hasan, "Zero-One Knapsack Problem," Naval Research Logistics(34:2): 161-172 (1987).

Senju, Shizuo and Yoshiaki Toyoda, "An Approach to Linear Programming with 0-1 Variables," Management Science(15:4): B197-B208 (1968).

Toyoda, Yoshiaki, "A Simplified Algorithm for Obtaining Approximate Solutions to Zero-One Programming Problems," Management Science(21:12): 1417-1427 (1975).

Winston, Wayne L. Operations Research Applications and Algorithms. Belmont, California: Duxbury Press, 1994.