A MULTI-PASS CONSTRUCTION HEURISTIC FOR THE AGGREGATED AIRLIFT PROBLEM

THESIS

Stephen T. O'Leary, Captain, USAF

AFIT/GOR/ENS/06-15

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

A MULTI-PASS CONSTRUCTION HEURISTIC FOR THE AGGREGATED 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

Stephen T. O'Leary, BS

Captain, USAF

March 2006

A MULTI-PASS CONSTRUCTION HEURISTIC
FOR THE AGGREGATED AIRLIFT PROBLEM

Stephen T. O'Leary, BS
Captain, USAF

Approved:

_____	_____
Dr. Gary W. Kinney, Maj, USAF (Chairman)	date
_____	_____
Dr. James T. Moore (Member)	date

Abstract

The Air Force has over 26,000 Airmen deployed from over 80 military installations to Forward Operating Locations (FOLs) in over 21 countries providing capabilities for air mobility and refueling, precision strike, close air support, intelligence, surveillance, and reconnaissance among others. Airmen are rotated in and out of these deployed positions every 120 days as part of the AEF system. Prior to January 2005, small groups of deploying personnel made their way via ad hoc methods providing little to no visibility to planners and creating a backlog of personnel at the Points of Debarkation (PODs) where deploying airmen often waited up to 5 days before receiving tactical airlift to their FOLs. The Single Ticket process was designed to alleviate this situation by scheduling transportation for airmen from their homebase to their FOL. In-Transit Visibilty (ITV) is required to properly implement the Single Ticket process. Proper ITV is provided by aggregated airlift. With aggregated airlift personnel are aggregated into groups of 100 or more based on region of origin, POD and Required Delivery Date (RDD) so that dedicated airlift can be generated for their deployment. The dedicated airlift facilitates better ITV of personnel moving though the PODs, enables proper planning and scheduling of intratheater airlift from the POD to the FOL, and reduces the backlog of personnel and wait times at the POD. In the aggregated airlift problem we are given a Time Phased Force Deployment Document (TPFDD) containing a list of personnel scheduled for deployment, their base of origin, POD, FOL, and RDD. This research creates a multi-pass construction heuristic to solve the aggregated airlift problem. The algorithm was tested on several notional and real world TPFDDs and performed well, increasing the effectiveness over the manual process by more than 20%.

Dedication

To My Wife, who kept everything running perfectly and to my two sons, of whom I am so proud; I could not have done this without you! I love you all and appreciate the sacrifices you made throughout my course of study.

Acknowledgments

I would like to express my sincere appreciation to my faculty advisor, Maj Gary Kinney, for his guidance and support throughout the course of this thesis effort. His insight and expert knowledge of heuristics was certainly appreciated. I would, also, like to thank my sponsor, BGen Stephen L. Hoog, Commander, AEFC, for giving me the opportunity to research and help solve this problem.

I am, also, indebted to the Mr. Mitchell Berry, an analyst at AEFC, who was an excellent point of contact for all my questions on the aggregation process.

Stephen T. O'Leary

Table of Contents

Page

Abstract iv
Dedication v
Acknowledgments vi
Table of Contents vii
List of Tables viii
List of Figures ix
List of Acronyms x
Introduction: 1
1.1 Overview 1
1.2 Single Ticket Deployments 3
1.3 Problem Statement: 4
1.4 Document Outline 4
2. Background 5
2.1 The AEF Concept 5
2.2 Air Force Deployments 5
2.3 The Single Ticket Deployment Process 9
2.4 Aggregated Airlift 10
2.5 Airlift Aggregation as an Integer Program 11
2.6 The Hub Allocation Problem 14
2.7 Heuristic Search Methods 15
3. Methodology 18
3.1 Overview 18
3.2 Identifying Extraneous RDDs and Creating a Bound 18
3.3 Generating a Starting Solution 19
3.5 Combining Smaller Aggregates 22
3.6 Combining ULNs in Infeasible Aggregates 31
3.7 Output and Post Processing 31
4. Testing and Results 35
4.1 Introduction 35
4.2 Testing for performance on Notional TPFDDs 35
4.3 Testing for Performance on Real World TPFDDs 38
4.4 Analysis of Removing the Regions Constraint 40
5. Conclusion 45
5.1 Summary of Results 45
5.2 Airlift Aggregation for AEF Cycle 1/2 45
5.3 Further Research 46
Bibliography 47
Appendix A: VBA Code 49

List of Tables

Page
Table 1. Notional TPFDD 20
Table 2. Sample Output 33
Table 3. Aggregation Results for Notional TPFDDs 37
Table 4. Confidence Interval 38
Table 5. Aggregation Resuts for Real World TPFDDs 39
Table 6. Aggregation Results of Notional TPFDDs with Region Constraint Relaxed 41
Table 7. Aggregation Results of Real World TPFDDs with Region Constraint Relaxed 42
Table 8. Sample Output With and Without Region Constraint 43

List of Figures

Page

Figure 1. Sample ULN Data from JOPES 6

Figure 3. Aggregate Possibilities for a Particular RDD 19

Figure 4. Phase 1 Combination 24

Figure 5. Phase 2 Combination 25

Figure 6. Phase 3 Combination 26

Figure 7. Phase 4 Combination 27

Figure 8. Phase 5 Combination 28

Figure 9. Phase 6 Combination 29

Figure 10. Phase 7 Combination 30

Figure 11. Passenger Distributions of Real World TPFDDs 40

List of Acronyms

AEF ..... Air & Space Expeditionary Force

AEFC ..... Air & Space Expeditionary Force Center

AETF ..... Air & Space Expeditionary Task Force

AMC ..... Air Mobility Command

BWI ..... Baltimore Washington International Airport

EAD ..... Earliest Arrival Date

FOL ..... Forward Operating Location

IDO ..... Installation Deployment Officer

ITV ..... In Transit Visibility

JOPES ..... Joint Planning and Execution System

LAD ..... Latest Arrival Date

PID ..... Plan ID

POD ..... Point of Debarkation

POE ..... Point of Embarkation

PRU ..... Personnel Readiness Unit

RDD ..... Required Delivery Date

RLD ..... Ready to Load Date

TACC ..... Tanker Airlift Control Center

TMO ..... Traffic Management Office

TPFDD ..... Time Phased Force Deployment Document (Data)

TPFDDL ..... Time Phased Force Deployment Data List

UDC ..... Unit Deployment Center

ULN ..... Unit Line Number

UTC ..... Unit Type Code

A MULTI-PASS CONSTRUCTION HEURISTIC
FOR THE AGGREGATED AIRLIFT PROBLEM

Introduction:

1.1 Overview

The Air Force currently has more than 26,000 Airmen deployed in support of the Global War on Terror and Combatant Commanders. These personnel are deployed from over 80 military installations to 41 Forward Operating Locations (FOLs) in 21 different countries. They support over 300 aircraft providing capabilities for air mobility and refueling, precision strike, close air support, intelligence, surveillance, and reconnaissance among others.

To keep all of these positions filled the Air Force uses the Air & Space Expeditionary Force (AEF) concept. Under this concept the Air Force can rotate Airmen in and out of theater to support long term operations while maintaining a high level of training and expertise. Each airman is vulnerable for deployment for 4 months of a 20-month cycle which allows predictability of deployment for all Airmen and their units.

The Air & Space Expeditionary Force Center (AEFC) executes the Air Force battle rhythm and is the primary functional unit for sourcing requirements for the deployment of the United States Air Force. AEFC is responsible for filling each of the over 26,000 positions for each AEF cycle with personnel currently stationed all over the world (Herbert, 2003). AEFC works closely with Air Mobility Command (AMC) to

deploy Air Force personnel and equipment giving the Air Force the capability to rotate personnel in and out the deployed positions every 120 days.

The military deploys by Unit Type Code (UTC), a designator including everything from a large fighter squadron and all of its support personnel and equipment to a single Contracting Officer with only hand carried baggage. When deploying groups of 100 or more from the same wing, units process their personnel and cargo through the Unit Deployment Center (UDC) led by the Installation Deployment Officer (IDO). After processing, dedicated cargo and passenger planes pick them up from their Point of Embarkation (POE) and take them overseas.

Prior to January 2005, smaller deployments of less than 100 people were treated differently. Members processed through their Personnel Readiness Unit (PRU) and cargo was sent through the Traffic Management Office (TMO). Personnel flew commercial air to Baltimore Washington International Airport, boarded a "Patriot Express" contracted airlift mission to their Point of Debarkation (POD) and awaited intratheater airlift to their final destination. (AFI 10-403). Commercial airlift and "Patriot Express" missions provide planners at PODs with little to no In-Transit Visibility (ITV) or advanced notice of incoming airmen. This lack of visibility made it impossible to schedule adequate intratheater airlift to transport personnel to their FOLs. This resulted in up to a 5 day backlog at the PODs and unnecessary delays for Airmen to reach their FOLs (Hoog, 2005).

1.2 Single Ticket Deployments

Currently, AEFC is using the "Single Ticket" system on all deployments to schedule airlift for airmen from their home station all the way to their FOL (Hoog, 2005). A passenger's travel itinerary is dictated in the Time Phased Force Deployment Document (TPFDD) from their POE to their final destination or FOL (USCENTAF, 2005). The process of assigning airlift to all passengers is very time consuming and can delay publishing and updating the TPFDD.

To schedule follow-on aircraft, an accurate forecast of incoming passengers is required. Mission planning and ITV are facilitated by using dedicated airlift to move personnel. The minimum number of passengers for dedicated airlift is 100. Two groups of 40 or more (totaling more than 100) can be collected from two POEs in CONUS and two groups of 40 or more (totaling more than 100) can be delivered to two PODs in theater. This is the purpose of aggregated airlift – gathering UTCs within a region and a common Required Delivery Date (RDD) window so dedicated airlift from POE to POD can be requested. Aggregated airlift was used for AEF 7/8, AEF 9/10 and is planned for AEF 1/2. This research creates a mathematical algorithm to partially automate airlift aggregation. The solution will allow more thorough travel planning and efficient deployment of personnel. The overall goal is to shorten the backlog at PODs to less than 24 hours by giving planners at PODs an accurate forecast of passengers transiting through their ports.

1.3 Problem Statement:

The problem is given a sourced TPFDD with origins, PODs, destinations, and RDDs; place all personnel in groups of at least 100 traveling from the same region to the same POD and having RDDs within a 5 day span for Active Duty passengers and a 3 day span for Reserve/Guard personnel. A group can be composed of personnel from two regions if there are at least 40 from each region. Similarly, a group can travel to two PODs if there are at least 40 personnel traveling to each POD. If possible, passengers from the same origin with the same RDDs should be in the same group, and the span of the RDD's should be minimized. The overall objective is to minimize the number of personnel not in groups.

1.4 Document Outline

The outline of the remainder of this document is as follows. Chapter 2 provides a background on the AEF concept and Air Force deployment process, a survey of similar problems in logistics, and an introduction to heuristic problem solving. The methodology of the algorithm used to automate airlift aggregation is discussed in Chapter 3. In Chapter 4, the algorithm is tested on a set of randomly generated TPFDDs and two real world TPFDDs from AEF 7/8 and AEF 9/10. Finally in Chapter 5, summarizes the results and proposes further research opportunities pertaining to this problem.

2. Background

> "The Air & Space Expeditionary Force (AEF) is the USAF methodology for organizing, training, equipping, and sustaining rapidly responsive air and space forces to meet the defense strategy requirements. Through the AEF, the Air Force supports defense strategy requirements using a combination of both permanently assigned and rotational forces. The USAF organizes Air & Space Expeditionary Task Forces (AETF) to meet defense strategy requirements using both CONUS based and forward stationed units." – Air Force Planning Document 10-4

2.1 The AEF Concept

To adequately sustain the force and forecast deployments for Airmen, the Air Force deploys personnel on a 120-day rotation schedule as part of an Air & Space Expeditionary Task Force. AEF is the methodology or system under which the Air Force deploys. The AEF concept allows the Air Force to better schedule units for deployments and affords most Airmen predictability in their deployments. Each airman is aligned to one AEF scheduled on a 20-month rotational cycle and each AEF is eligible for deployment for 120 days during that cycle. When one AEF's deployment window ends and another's begins, large amounts of personnel are deployed and redeployed as over 25,000 positions are vacated and filled. Planning these movements and tracking personnel throughout the process is a large part of the AEF Center's responsibility. (AFPD 10-4)

2.2 Air Force Deployments

The deployment planning process involves generating a Time Phased Force Deployment Document (TPFDD). A TPFDD lists which Unit Type Codes (UTCs) are required to be deployed in support of mission objectives for a given conflict, who will

supply these capabilities, and the timing and routing details of their transport. When a TPFDD is sourced, each UTC is assigned to a unit that will be tasked to fill the requirement. The TPFDD is used as a timeline for each step of the deployment of troops, cargo, and equipment. The information in the TPFDD is from the Joint Operations Planning and Execution System (JOPES) and is "pulled" into a TPFDD List (TPFDDL) when needed. The JOPES database includes all information pertaining to each requirement. A sample of the information contained in JOPES is shown below in Figure 1.

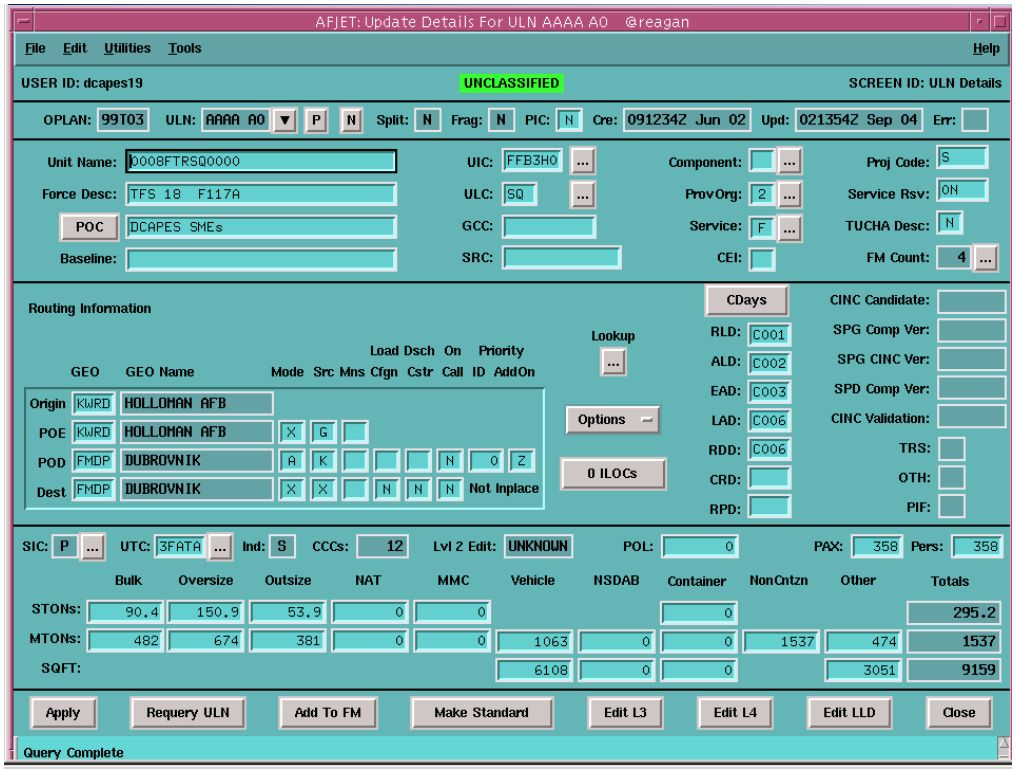

Figure 1. Sample ULN Data from JOPES

When pulling a TPFDD, desired parts of the data are chosen to form the different columns. Some sample lines from a TPFDDL are shown in Figure 2.

OPLAN	REQUID	UTC	DESCRIPTION	UIC	RLD	ALD	EAD	LAD	RDD	PAX	STONS	SERV	ORIG	POE	POD
XXXX2	F03100	3FZ99	AVIATION SENSOF	FFNL	C032	C033	C035	C037	C039	1	0	1MA	HURLBURT FL	BALTIMORE-WASH	BBBB
XXXX2	F03100	3FZ99	AVIATION SENSOF	FFFC	C032	C033	C035	C037	C039	1	0	1LA	MACDILL AFB	BALTIMORE-WASH	BBBB
XXXX2	F03200	3FZ99	AVIATION SENSOF	FFGI	C032	C033	C035	C037	C039	1	0	0RA	KADENA AB	BALTIMORE-WASH	BBBB
XXXX2	F03200	3FZ99	AVIATION SENSOF	FFLC	C032	C033	C035	C037	C039	1	0	0DA	RAMSTEIN AB	RAMSTEIN AB	BBBB
XXXX2	F0330	3FZ99	AVIATION SENSOF	FFXT	C052	C054	C055	C056	C058	1	0	1SA	MINOT AFB	BALTIMORE-WASH	BBBB
XXXX2	F03AB0	3FKD1	TFE 06 F 16 B25-3	FFL5	C006	C006	C007	C009	C009	4	0	2IG	FORT WAYNE I	FORT WAYNE INT	WWWW
XXXX2	F03AB0	3FKD1	TFE 06 F 16 B25-3	FFL5	C008	C009	C010	C012	C014	13	0	2IG	FORT WAYNE I	FORT WAYNE INT	BBBB
XXXX2	F03AB0	3FKD1	TFE 06 F 16 B25-3	FFL5	C006	C006	C007	C009	C009	0	6.9	2IG	FORT WAYNE I	FORT WAYNE INT	WWWW
XXXX2	F03AB0	3FKD1	TFE 06 F 16 B25-3	FFL5	C011	C011	C014	C016	C016	5	0	2IG	FORT WAYNE I	FORT WAYNE INT	WWWW
XXXX2	F03AB0	3FKD1	TFE 06 F 16 B25-3	FFL5	N003	N002	N001	C002	C004	1	0	2IG	FORT WAYNE I	BALTIMORE-WASH	BBBB
XXXX2	F03AB0	3FKD1	TFE 06 F 16 B25-3	FFL5	N004	N004	N001	C002	C004	1	0	2IG	FORT WAYNE I	BALTIMORE-WASH	BBBB
XXXX2	F03BB0	3FKD2	TFE 06 F 16 B25-3	FFL5	C006	C006	C007	C009	C009	1	0	2IG	FORT WAYNE I	FORT WAYNE INT	WWWW
XXXX2	F03BB0	3FKD2	TFE 06 F 16 B25-3	FFL5	C008	C009	C010	C012	C014	6	0	2IG	FORT WAYNE I	FORT WAYNE INT	BBBB
XXXX2	F03DB0	3FKD1	TFE 06 F 16 B25-3	FFL5	C006	C006	C007	C009	C009	5	0	2IG	FORT WAYNE I	FORT WAYNE INT	WWWW
XXXX2	F03DB0	3FKD1	TFE 06 F 16 B25-3	FFL5	C008	C009	C010	C012	C014	3	0	2IG	FORT WAYNE I	FORT WAYNE INT	BBBB
XXXX2	F03EX0	3FKP1	TFE 12 F 16 B40	FFD9	N004	N004	N003	C001	C001	10	0	1CA	HILL AFB	HILL AFB	WWWW
XXXX2	F03EX0	3FKP1	TFE 12 F 16 B40	FFD9	C002	C002	C003	C005	C007	19	0	1CA	HILL AFB	HILL AFB	BBBB
XXXX2	F03EX0	3FKP1	TFE 12 F 16 B40	FFD9	C003	C003	C004	C007	C007	0	14.3	1CA	HILL AFB	HILL AFB	WWWW
XXXX2	F03EX0	3FKP1	TFE 12 F 16 B40	FFD9	C004	C004	C007	C009	C009	6	0.5	1CA	HILL AFB	HILL AFB	WWWW
XXXX2	F03EX0	3FKP1	TFE 12 F 16 B40	FFJM	C001	C002	C003	C005	C007	1	0	1CA	MOUNTAIN HO	HILL AFB	BBBB
XXXX2	F03EY0	3FKP2	TFE 06 F 16 B40	FFD9	C002	C002	C003	C005	C007	9	0	1CA	HILL AFB	HILL AFB	BBBB
XXXX2	F03EY0	3FKP2	TFE 06 F 16 B40	FFD9	C003	C003	C004	C007	C007	0	0.9	1CA	HILL AFB	HILL AFB	WWWW
XXXX2	F03P10	3R4FA	SRE 04 MAE UAV	FFK3	C031	C032	C033	C035	C037	34	0	1CA	CREECH AFB	BALTIMORE-WASH	BBBB
XXXX2	F03P10	3R4FA	SRE 04 MAE UAV	FFR1	C025	C026	C027	C029	C031	4	0	1CA	CREECH AFB	BALTIMORE-WASH	BBBB
XXXX2	F03P10	3R4FA	SRE 04 MAE UAV	FFR1	C055	C056	C057	C059	C059	2	0	1CA	CREECH AFB	BALTIMORE-WASH	BBBB
XXXX2	F03SM0	3TRBL	CSR 03 HH 60G CS	FFFY	C071	C072	C073	C075	C077	20	0	0VA	NELLIS AFB	BALTIMORE-WASH	BBBB
XXXX2	F03X10	3NH4L	TAE 04 C130H	FFRT	C014	C015	C016	C018	C020	18	0	1LA	LITTLE ROCK A	LITTLE ROCK AFB	WWWW
XXXX2	F03X10	3NH4L	TAE 04 C130H	FFRT	C014	C015	C016	C018	C020	7	0	1LA	LITTLE ROCK A	LITTLE ROCK AFB	WWWW
XXXX2	F03XM	3NH22	TAE 02 C130H	FFMI	N005	N004	N003	C001	C001	13	0.7	1LG	RENO-TAHOE I	RENO-TAHOE INT	TTTT
XXXX2	F03XN	3NH22	TAE 02 C130H	FFMI	N005	N004	N003	C001	C001	1	0	1LG	RENO-TAHOE I	BALTIMORE-WASH	BBBB
XXXX2	F04000	4FPFJ	EN FIRE FIGHTER	FFD3	C047	C048	C049	C051	C059	1	0	0JA	LITTLE ROCK A	BALTIMORE-WASH	CCCC
XXXX2	F04000	4FPFJ	EN FIRE FIGHTER	FFDI	C047	C048	C049	C051	C059	1	0	1CA	WHITEMAN AF	BALTIMORE-WASH	BBBB
XXXX2	F0401	4F9XD	EN EOD SRNCO N	FFSC	C009	C010	C012	C014	C028	1	0	2IA	WASHINGTON	BALTIMORE-WASH	CCCC
XXXX2	F0402	4F9XD	EN EOD SRNCO N	FFBH	C009	C010	C012	C014	C028	1	0	0DA	SPANGDAHLEN	RAMSTEIN AB	CCCC
XXXX2	F0403	4F9XA	CES PRIME BEEF	FFD3	C009	C010	C012	C014	C028	1	0	1CA	LANGLEY AFB	BALTIMORE-WASH	CCCC
XXXX2	F04040	4F9XA	CES PRIME BEEF	FFV6	C010	C011	C012	C014	C028	1	0	1CA	MOUNTAIN HO	BALTIMORE-WASH	CCCC
XXXX2	F0405	4F9XD	EN EOD SRNCO N	FF1Z	C010	C011	C012	C014	C028	1	0	1WA	TYNDALL AFB	NORFOLK NAS	CCCC
XXXX2	F0406	4F9AF	CES PRIME BEEF	FFV6	C004	C005	C006	C008	C010	2	0.1	1CA	MOUNTAIN HO	NORFOLK NAS	BBBB
XXXX2	F04070	4F9XA	CES PRIME BEEF	FFSC	C009	C010	C012	C014	C028	1	0	2IA	WASHINGTON	BALTIMORE-WASH	CCCC
XXXX2	F0408	4F9AF	CES PRIME BEEF	FFDI	C011	C012	C013	C015	C017	2	0.1	1CA	WHITEMAN AF	NORFOLK NAS	BBBB
XXXX2	F04110	4F9DC	CES PB FL SP THF	FFS9	C015	C016	C017	C019	C024	1	0	1CA	LANGLEY AFB	NORFOLK NAS	BBBB
XXXX2	F041102	4F9DC	CES PB FL SP THF	FFSC	C017	C018	C020	C022	C024	1	0	2IA	WASHINGTON	BALTIMORE-WASH	BBBB
XXXX2	F0412	4F9AS	CES PB PAVEMEN	FFD4	C016	C017	C018	C020	C022	3	0	1CA	NELLIS AFB	NORFOLK NAS	BBBB
XXXX2	F0413	4F9AS	CES PB PAVEMEN	FFDI	C016	C017	C018	C020	C022	3	0	1CA	WHITEMAN AF	NORFOLK NAS	BBBB
XXXX2	F04300	4F9EA	CES PRIME BEEF	FFD4	C012	C013	C014	C016	C018	53	0	1CA	NELLIS AFB	NELLIS AFB	BBBB

Figure 2. Sample TPFDDL

The TPFDD above includes the OPLAN column which is the pseudo Plan ID (PID) for the operation planned by this TPFDD, the REQUID or Requirement ID is also known as a Unit Line Number (ULN) which is a unique identifier for each tasking in a TPFDD, the UTC code is listed next, and the Unit Information Code (UIC), which corresponds to the unit tasked to fill the UTC.

The travel timing columns are as follows: Ready to Load Date (RLD) when a unit is prepared to depart its origin, Available to Load Date (ALD) when a unit begins loading at the Point of Embarkation (POE), Earliest and Latest Arrival Dates (EAD and LAD) which are the earliest and latest dates a unit can arrive at its Point of Debarkation (POD)

and Required Delivery Date (RDD) which is the latest date a unit can arrive at its Forward Operating Location (FOL). The dates are in the format of the "C-Day" or "Conflict Day," and C000 is the day the conflict begins and is the standard first day for a TPFDD. Days beginning with N count backwards to N000, the day before the "C-Day." The PAX and STONS columns list the number of passengers and short tons associated with each ULN. The SERV column indicates the MAJCOM of the tasked unit. The ORIG, POE, and POD columns give the origin, POE, and the POD for the ULN. (AFI 10-400) This research does not require any further information but there is much more available in JOPES; for example, destination, point of contact, line remarks, cargo information and airlift details (channel or dedicated) are all stored and available for those requiring such information as seen in figure 1. It is generally accepted in the planning community to call a TPFDDL a TPFDD and the terms are used interchangeably for the remainder of this research.

Large deployments made up of 100 or more passengers are usually given dedicated airlift by the TPFDD. Prior to January of 2005, smaller deployments were treated differently. The passengers were sent via commercial airlift to Baltimore Washington International (BWI) airport to board a "Patriot Express" contracted channel airlift mission for transportation to their POD where they would await further airlift to their FOL (AFI 10-403). Transportation planners at the PODs did not have visibility of the passengers until they were manifested on the Patriot Express flight and were on their way (~8-10 hours notice). This did not allow for proper scheduling of airlift from POD to FOL. Members would often have to wait up to 5 days at a POD before receiving tactical airlift to their FOL. (Hoog, 2005)

2.3 The Single Ticket Deployment Process

In January 2005, the Air Force began using a Single Ticket program for deploying personnel to the U.S. Central Command area of responsibility (the majority of our current deployed positions) (Hoog, 2005). In this process, all UTCs are given an itinerary for airlift from their home station to their final destination. To make this happen, In-Transit Visibility (ITV) and airlift forecasting were needed to ensure intra-theater airlift was staged properly to accommodate the large number of personnel traveling through PODs. This additional requirement was accomplished by a joint effort between Air Mobility Command (AMC) Headquarters, Tanker Airlift Control Center (TACC) and AEFC. (AEPE, 2005)

Single Tickets are particularly useful for personnel who require tactical airlift from their POD to their FOL. The goal of the Single Ticket process is to reduce the time a member spends at a POD waiting for airlift. With additional airlift forecasting and ITV, planners at the PODs receive notice of personnel transiting through their port well in advance and are able to schedule follow-on lift for shortly after their arrival, shortening the layover time awaiting forward deployment from as much as 5 days to as little as 24 hours. Decreasing this wait time speeds up the deployment of troops and shortens the time the airman is away from home station since travel time is not included in the 120 days of a deployment. Single Ticket deployments are a must in the eyes of the Combatant Commander as well as the service member (CENTAF, 2005).

By using the single ticket program for AEF 3/4, the Air Force successfully moved 3,173 deploying personnel through the PODs within 48 hours. Of these Airmen, 86% moved within 24 hours after arriving at their POD. This success came with much effort.

Planners responsible for the single ticket program had to manually track personnel and transmit ITV consuming an enormous amount of man-hours. Because the proper information was not in the usual online database, planners had to resort to faxes, and emails of spreadsheets or phone calls to notify the planners at the POD of incoming Airmen.

For AEF 5/6, 7,215 personnel deployed with a single ticket and 4473 of them required intratheater airlift from their POD to their FOL. Even though 3266 (73%) of these passengers received forward airlift within 24 hours, 1372 of them arrived at their POD on the wrong flight or without proper ITV. Without accurate ITV, POD planners have inaccurate or no information on the schedule or number of personnel arriving at their location and are unable to schedule adequate follow-on airlift (Hoog, 2005).

2.4 Aggregated Airlift

The Single Ticket deployment system is aided by Aggregated Airlift, grouping passengers together to justify dedicated airlift from POE to POD. To aggregate airlift, the TPFDD is sorted by POD and RDD to identify groups of passengers scheduled to go to the same POD in a relatively small time window (2-5 days). When a group of 100 or more is identified, then dedicated airlift is justified. Dedicated airlift can stop at more than one POE, provided there are at least 40 passengers at that location. Similarly, a single mission can deliver to more than one POD, provided there are at least 40 passengers going to each POD. No more than two POEs and no more than two PODs can be visited in a single mission. This aggregation facilitates reduced travel times by

allowing personnel to travel on dedicated airlift and gives planners at the POD the visibility and advance notice required to schedule tactical airlift to the FOL.

Aggregated airlift planning is currently completed manually using an Excel spreadsheet containing a sourced TPFDD. The process currently takes 5 working days and planners were able to aggregate 66% of the passengers for AEF 9/10. This research involves producing a more automated and efficient way to aggregate airlift by using an algorithm to search the TPFDD for feasible aggregates. By streamlining and improving the aggregation process, more passengers will be able to use dedicated airlift for the single ticket deployment process providing more efficient use of inter-theater airlift and leading to less travel time to and from deployed locations.

2.5 Airlift Aggregation as an Integer Program

The problem of airlift aggregation can be modeled as an integer program (IP) with the following formulation:

$$Min\ Z = \sum_{i=1}^{n_1+n_2} N_i(1-\sum_{j=1}^{m} X_{ij}) \tag{2.1}$$

Subject to the following constraints:

$$X_{ij} \le Y_j \qquad j = 1\ to\ m \tag{2.2}$$

$$\sum_{j=1}^{m} X_{ij} \le 1 \qquad i = 1\ to\ n_1 + n_2 \tag{2.3}$$

$$\sum_{i=1}^{n_1+n_2} N_i X_{ij} \ge 100 Y_j \qquad j = 1\ to\ m \tag{2.4}$$

$$DA\max_j \ge D_i X_{ij} \qquad j = 1\ to\ m,\ i = 1\ to\ n_1 \tag{2.5}$$

$$DRG\max_j \geq D_i X_{ij} \qquad j = 1 \text{ to } m,\ i = n_1 + 1 \text{ to } n_1 + n_2 \tag{2.6}$$

$$D\min_j \leq D_i X_{ij} \qquad j = 1 \text{ to } m,\ i = 1 \text{ to } n_1 + n_2 \tag{2.7}$$

$$DA\max_j - D\min_j \leq 5 \qquad j = 1 \text{ to } m \tag{2.8}$$

$$DRG\max_j - D\min_j \leq 2 \qquad j = 1 \text{ to } m \tag{2.9}$$

$$B_i X_{ij} = B_i X_{kj} \qquad j = 1 \text{ to } m,\ i = 1 \text{ to } n_1 + n_2,\ k = 1 \text{ to } n_1 + n_2,\ i \neq k \tag{2.10}$$

$$P_i X_{ij} = P_i X_{kj} \qquad j = 1 \text{ to } m,\ i = 1 \text{ to } n_1 + n_2,\ k = 1 \text{ to } n_1 + n_2,\ i \neq k \tag{2.11}$$

Constants:

m = the total number of possible aggregates; $m \leq n_1 + n_2$

n_1 = the total number of Active Duty ULNs

n_2 = the total number of Reserve and Guard ULNs

i = the ULN index (The line the ULN is on in the TPFDD)

j = the aggregate index (each aggregate requires a unique index)

N_i = # of people in ULN i

D_i = RDD for ULN i

B_i = Region of origin for ULN i

P_i = POD for ULN i

Variables:

X_{ij} = 1 if ULN i is in aggregate j; 0 otherwise

Y_j = 1 if aggregate j exists; 0 otherwise

$DRG\max_j$ = the latest RDD for a Reserve or Guard ULN in aggregate j

$DA\max_j$ = the latest RDD for an Active Duty ULN in aggregate j

$D\min_j$ = the earliest RDD in aggregate j

The objective function, Equation (2.1), sums all units not in an aggregate. Equation (2.2) is the precedence constraint which forces Y_j to 1 if aggregate j is used. Equation (2.3) ensures each unit will be assigned to only one aggregate. Equation (2.4) ensures each aggregate contains at least 100 personnel. Equations (2.5), (2.6), and (2.7) set the latest and earliest RDDs for Active and Reserve/Guard units in each aggregate (each aggregate will have only one earliest RDD). Equations (2.8) and (2.9) force the RDDs for active units to be within a 5 day span and reserve units to be within a 2 day span. The last two Equations (2.10) and (2.11) force aggregates to have a common region and point of debarkation.

The problem that arises in solving this formulation is that the number of ULNs in a TPFDD is usually at minimum 2,500. This IP has $4m + m(n_1+n_2)$ variables and $4m + 2mn_1 + 2mn_2 + n_1 + n_2 + 2m(n_1 + n_2 - 1)^2$ constraints. With a 7,500 passenger 2,500 line TPFDD containing 2,125 active duty ULNs and 375 active duty ULNs and a possible 75 aggregates (i.e. planing the maximum number of 100 passenger aggregates), the model would have 187,800 variables and over 937 million constraints. A commercial solver for a problem of this size would be a very costly alternative. In addition, many real world aspects of the problem are not included in this model, such as aggregates with multiple regions or PODs. Further combinations of aggregates could be made by the planners on a

case by case basis so the optimal solution of the model is not the final solution to the real problem but only a starting point.

2.6 The Hub Allocation Problem

This problem is considered generically by viewing the ULNs as any entity needing to be grouped for shipment. The home base is the origin, the POD is a hub, and the FOL is the destination. The RDD is the required delivery date of the entity. The Reserve/Guard or Active Duty attribute represents entities with different time windows with respect to how early the entities can arrive. In this form, the problem is similar to the p-median or hub allocation problem.

The p-median problem involves locating a set of hubs relative to a set of entities so as to minimize the sum of the distances or costs between all entities and their hub. The hub allocation problem involves the location of hubs in a network to minimize the number of hubs used and maximize coverage of the network (connect all entities to a hub). Given a network of consumers with varying constraints, the solution dictates how many hubs are used and where they are placed. The passengers or "consumers" are trying to get to a hub or POE compatible with their attributes. The McTRESH model proposed by Balakrishnan and Storbeck (1991), models maximum coverage with a threshold constraint. Current and Storbeck (1994) formulated a multiobjective model which selected hub locations based on the attributes of the network area in which they were to be placed. Constraints in their formulation force all hub locations to have a minimum number of customers. They also address the issue of the different demands of customers for a particular type of hub.

These formulations do not model individual consumers but an average or projection of their demands over a particular area and time span. The aggregated airlift problem is concerned with the exact demands of the ULNs in a particular TPFDD. A large part of the decision in a hub allocation problem is deciding where to locate a hub for continued operation and there is a cost associated with opening a hub and for maintaining its operation. The additional cost for the Air Force to "create" a hub is minimal as the bases and facilities already exist. Also the Air Force does not need to maintain the hub for future use. For example, if there are enough passengers to aggregate and leave from Dyess AFB, on a particular day going to the same POD(s), Dyess AFB becomes a hub. This may be the only time Dyess AFB is used as a hub, and the next cycle may use Lackland AFB. On the other hand if UPS wants to land a plane at an airport they do not normally use there, is an extra cost. UPS must consider the long term use of a hub based on the average demand for use of the facility.

2.7 Heuristic Search Methods

The airlift aggregation problem is known as a combinatorial optimization problem and as shown in the previous section, it is related to the p-median problem. The p-median problem is proven to belong to the class of NP-hard (nondeterministic polynomial-time complete) problems by Kariv and Halkimi (1979). It is generally accepted that there is no efficient algorithm which can solve these problems to optimality.

Heuristics are a way of finding high quality solutions to NP-hard problems in a short amount of time. Heuristics use "Rules of Thumb" and the context of the problem to construct and improve upon solutions to the problem. There are no guarantees of

optimality or solution quality with most heuristics; however, in many cases heuristics have performed well in practice.

A heuristic algorithm also allows us to model real world aspects of the problem with greater fidelity. For example, if there are more than 40 personnel departing from the same origin in the same time span, the AEFC planners first objective would be to aggregate them together. This is easily accomplished by a heuristic algorithm, but it would be difficult to model in an IP formulation.

Heuristics can be viewed as belonging to one of two categories: construction or local search. Constructive methods are generally greedy and deterministic in nature and build a single feasible solution. There are two approaches to this; primal which starts with an empty solution and builds the solution without violating primal feasibility; and dual, which starts with an infeasible solution and builds or destructs the solution until it is feasible. Local search heuristics start with a feasible (or infeasible) solution and move in some improving direction. A move is defined as an operation that changes one solution into another solution. A neighborhood of a solution is comprised of all solutions connected to the current solution via one move. Local search heuristics usually stop when they reach the local optima of their neighborhood (i.e. the current solution is better than all of its neighbors).

Metaheuristics are defined by Laguna as "a master strategy that guides and modifies other heuristics to produce solutions beyond those that are normally generated in a quest for local optimality" (2000). They are usually a combination of a construction heuristics and a local search heuristics with the added ability to escape local optima, diversify their search into different neighborhoods, and intensify their search in desirable

neighborhoods. This research develops a multi-pass construction heuristic for the aggregated airlift problem.

3. Methodology

3.1 Overview

The algorithm used to construct a solution of feasible aggregates from a TPFDD is composed of three parts: generating a bound on the number of passengers able to be aggregated, creating an initial starting solution, and combining smaller aggregates to create aggregates with at least 100 passengers. The bound is created by identifying ULNs with RDDs which can never be assigned to an aggregate and subtracting them from the total amount of passengers. The initial starting solution places all ULNs in an aggregate but is infeasible because many aggregates do not contain the required 100 passengers. The starting solution is improved by combining infeasible aggregates with other aggregates, both feasible and infeasible, to create more feasible aggregates. The following sections provide model details.

3.2 Identifying Extraneous RDDs and Creating a Bound

The first step is to sort the TPFDD by RDD and remove the ULNs that will never be assigned to an aggregate. Each ULN can be a part of 5 aggregates as seen in Figure 3 below. The ULN can leave from 0 to 4 days early and therefore can be included with passengers in ULNs with RDDs up to 4 days earlier or later. RDD(r) is the day in question, the other days are those which can be in cluded in the same aggregate as RDD(r). For example, if r = 30, the ULNs in RDD(r) are required on day 30. They can be aggregated with ULNs required on day 26 through 30 (r-4 to r), 27 through 31 (r-3 to r+1), 28 through 31 (r-2 to r+2), 29 through 32 (r-1 to r+3), or 30 through 34 (r to r+4).

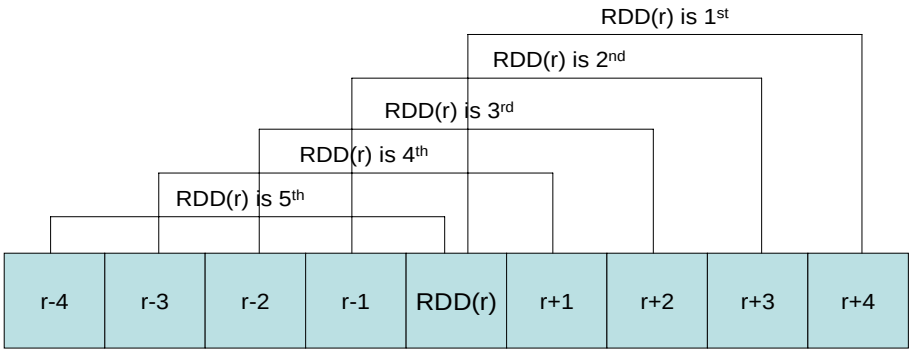

Figure 3. Aggregate Possibilities for a Particular RDD

Reserve/Guard passengers can only arrive up to 2 days early, so each aggregate can include Active Duty and Reserve/Guard passengers for the first 3 RDDs and only Active Duty for the last 2. Any ULNs that cannot be grouped with other ULNs totaling 100 passengers and leaving within 4 days before or after it (2 days for Reserve/Guard units) are removed from the list of candidates. For each RDD, the algorithm checks maximum number of passengers in all possible aggregates until it encounters an aggregate with more than 100 passengers. If a feasible aggregate can not be found for a particular RDD, all ULNs with that RDD are removed from consideration. When these ULNs are removed from consideration the model is left with an upper bound on the number of passengers it will be able to aggregate. The common region and POD constraints are not considered; therefore this bound is not expected it to be a tight bound.

3.3 Generating a Starting Solution

To generate a starting solution, each ULN is aggregated with others from the same origin going to the same POD within a feasible RDD span. Aggregates of ULNs with a common origin which are less than 75 are then disbanded so that they can be

aggregated with ULNs in the same region. A notional TPFDD, given below in Table 1, will be used to demonstrate the algorithm.

Table 1. Notional TPFDD

Index	ULN	Origin	# Pax	RDD	Regi	POD	Service
1	F9Q1X	ANDREWS	25	1	4	1	4
2	F933Q01	BEALE AF	51	1	1	2	4
3	F933R02	BEALE AF	49	1	1	2	4
4	F93BF01	DAVIS MO	30	1	1	2	4
5	FT3A1	DAVIS MO	35	1	1	3	4
6	F933H01	EDWARDS	17	1	1	3	4
7	F933L01	EDWARDS	16	1	1	3	4
8	F03EX01	HILL AFB	10	1	2	2	4
9	F0HDX01	HILL AFB	12	1	2	2	4
10	F0HEX01	HILL AFB	43	1	2	2	4
11	F0HHX01	HILL AFB	1	2	2	2	4
12	FT638	LANGLEY	1	2	4	2	4
13	F9U1A	LANGLEY	2	2	4	2	4
14	F93B5	MOODY AI	26	2	4	2	2
15	F99U3	MOODY AI	3	2	4	2	2
16	F99U4	MOODY AI	10	2	4	2	2
17	F9HB6	MOODY AI	2	3	4	2	2
18	F9PB5	MOODY AI	2	3	4	2	2
19	F0702	NELLIS AF	1	3	1	2	4
20	F0PSW0	NELLIS AF	1	3	1	2	4
21	F9X3A	ROBINS A	1	3	4	1	4
22	FNJ1A01	ROBINS A	1	3	4	1	4
23	F93AH01	SEYMOUR	4	3	4	3	4
24	F9HBJ01	SEYMOUR	4	3	4	3	4
25	F9HKJ02	SEYMOUR	74	4	4	3	4
26	F9HKK02	SEYMOUR	55	4	4	3	4
27	F9HNB	SEYMOUR	2	4	4	2	4
28	F9HRH01	SEYMOUR	4	4	4	2	4
29	F0PSV01	WASHING	1	4	4	2	4
30	FNJ1A02	WASHING	1	4	4	2	4
31	F933L02	WRIGHT-F	1	5	4	4	4
32	FNFN001	WRIGHT-F	1	5	4	4	4
33	FT63601	WRIGHT-F	1	5	4	4	4
34	FT63602	WRIGHT-F	1	5	4	4	4
35	F96MD	BARKSDA	1	5	3	2	2
36	F99UA01	BEALE AF	2	5	1	2	2
37	FNFN002	LANGLEY	1	5	4	3	2
38	F9FB5	MOODY AI	3	5	4	3	2
39	F9HB5	MOODY AI	59	5	4	2	2
40	F03XM	RENO-TAH	13	5	1	2	2
41	F03XN	RENO-TAH	1	6	1	2	2
42	F0H02	RENO-TAH	15	6	1	2	2
43	F9HBH01	SEYMOUR	10	6	4	3	2
44	F070602	TRAVIS AF	1	6	1	2	2

The algorithm starts with ULN(44) from Travis containing 1 Reserve/Guard passenger, designated as 2 in the service column, and compares it to the ULNs above it. There are no ULNs in the TPFDD with the same origin, so the algorithm stops searching when it reaches ULN(24) which is more than 2 days early and not in its feasible RDD span. Next the algorithm compares ULN(43) with those above it. The next earlier ULN with a common origin and POD is ULN(26), which is also within the feasible RDD span. ULN(43) and ULN(26) are updated with lastULN = 43 LastActiveRDD = 4, LastR/GRDD = 6 and totalpax = 65. Next ULN(25) is added and ULNs 43, 26, and, 25 are updated with totalpax = 139, and firstULN = 25. ULN(25) is updated with LastActiveRDD = 4, LastR/GRDD = 6, and lastULN = 44. The algorithm will stop when it reaches ULN(24) which is beyond the feasible RDD span for this aggregate since ULN(43) is Reserve/Guard.

After aggregating passengers from the same origin going to the same POD aggregates with less than 75 passengers are disbanded. The regional algorithm is then run to aggregate ULNs with a common POD and region. This phase would aggregate ULN(44) with ULNs 42, 41, 40, and 36 which all have a common region and POD and are within the feasible RDD span.

The regional algorithm stops adding to the aggregate when the number of passengers in the aggregate reaches 100 or more to keep the regional aggregates from leaving any earlier than necessary. Keeping large numbers of personnel from leaving too early is also the motivation for why the algorithm searches the TPFDD from the last ULN to the first ULN. For example, it would not be desirable for a 150 passenger ULN to

have to leave 3 days early to aggregate an additional 15 passengers. If possible, the additional passengers should leave with another aggregate required on an earlier day..

This portion of the algorithm completes the generation of an initial but likely infeasible solution. All ULNs are in an aggregate but many do not contain the required 100 passengers. A record is kept for those aggregates containing less than 100 passengers. The goal of the remaining passes of the heuristic is to group smaller aggregates together to create aggregates containing at least 100 passengers.

3.5 Combining Smaller Aggregates

Aggregates are combined based on the following priorities (provided by AEFC):

1. Aggregates containing passengers < 40 are combined with an aggregate in a region to the east containing passengers ≥ 100 going to the same POD
2. Aggregates containing 40 ≤ passengers < 100 are combined with other 40 ≤ passengers < 100 aggregates from a different region but going to the same POD.
3. Aggregates containing 40 ≤ passengers < 100 are combined with other aggregates containing ≥ 100 passengers from a different region but going to the same POD
4. Aggregates containing 40 ≤ passengers < 100 are combined with other aggregates containing 40 ≤ passengers < 100 from the same region but going to a different POD.

5. Aggregates containing 40 ≤ passengers < 100 are combined with aggregates containing ≥ 100 passengers from the same region but going to a different POD.
6. Aggregates containing 40 ≤ passengers < 100 are combined with other aggregates containing 40 ≤ passengers < 100 from a different region and going to a different POD.
7. Aggregates containing 40 ≤ passengers < 100 are combined with aggregates containing ≥ 100 passengers from a different region and going to a different POD.

Each phase can only combine aggregates which have not been combined in previous phases. This keeps aggregates from containing ULNs from more than 2 regions or more than 2 PODs. These phases are accomplished with the following pseudocode , broken down into phases. Each phase of the pseudocode differs only in the first two IF/THEN statements where the criteria of each phase are checked. If both aggregates meet the criteria, all ULNs belonging to the aggregates are combined into a single aggregate Examples for each phase are presented on the pages that follow.

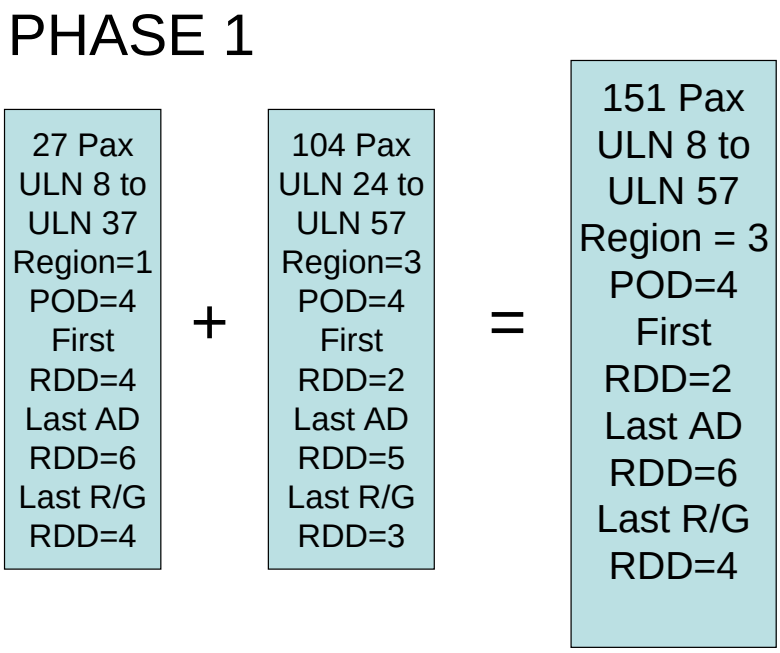

Figure 4. Phase 1 Combination

In figure 4 an aggregate with less than 40 passengers is combined with an aggregate with greater than 100 passengers to the east with the same POD. The regions are based on the time zones Pacific (1) to Eastern (4) so the greater region number is always to the east. The first aggregate contains 27 passengers from the west coast traveling to POD 4, it contains Active Duty passengers required on days 4 through 6 and Reserve/Guard passengers required on day 4. It is combined with the 104 passenger aggregate from the central region containing Active Duty passengers required on days 2 through 5 and Reserve/Guard passengers required on days 2 through 3. The combined aggregate will originate in the central region to be delivered through POD 4 by RDD 4 which will not be too early for the last Active Duty ULN required on day 6 or the last Reserve/Guard ULN required on day 4. It contains ULNs between 8 and 57.

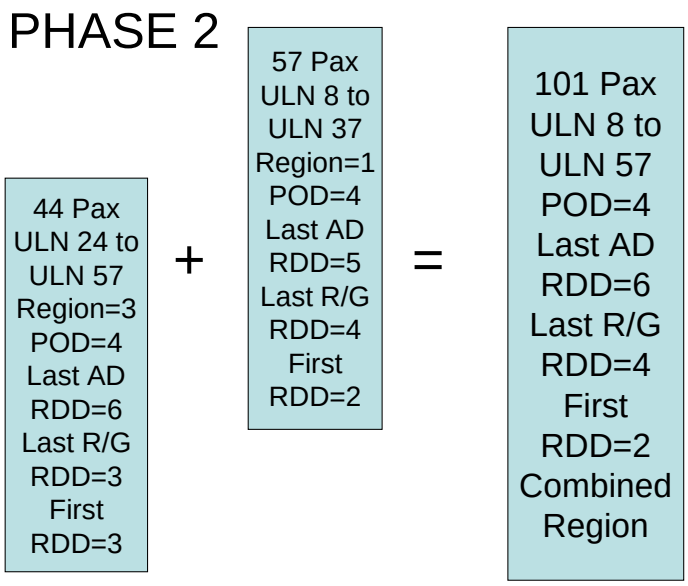

Figure 5. Phase 2 Combination

In figure 5 two aggregates from different regions, each containing 40 ≤ passengers < 100 and with a common POD are combined to form an aggregate with greater than 100 passengers. The first aggregate contains 44 passengers from the central region traveling to POD 4, it contains Active Duty passengers required on days 3 through 6 and Reserve/Guard passengers required on day 3. This aggregate is combined with the 57 passenger aggregate from the west coast region also traveling to POD 4. The 57 passenger aggregate contains Active Duty passengers required on days 2 through 5 and Reserve/Guard passengers required on days 2 through 4. The combined aggregate originates in the west coast region and stops in the central region. The combined aggregate is delivered through POD 4 by RDD 2 which is not too early for the last Active Duty ULN required on day 6 or the last Reserve/Guard ULN required on day 4. It contains ULNs between 8 and 57.

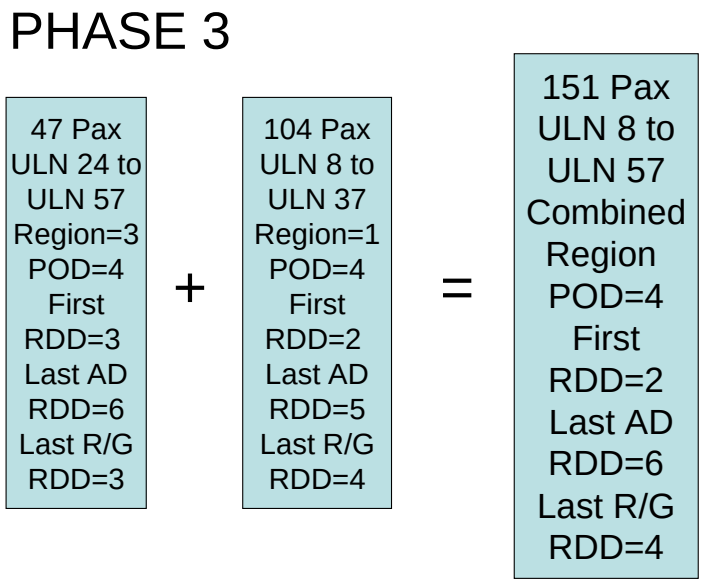

Figure 6. Phase 3 Combination

In figure 6 an aggregate containing $40 \leq$ passengers < 100 is combined with an aggregate with greater than 100 passengers going to the same POD but from a different region. The first aggregate contains 47 passengers from the central region traveling to POD 4 and includes Active Duty passengers required on days 3 through 6 and Reserve/Guard passengers required on day 3. It is combined with the 104 passenger aggregate from the west coast region containing Active Duty passengers also traveling to POD 4 who are required on days 2 through 5 and Reserve/Guard passengers required on days 2 through 4. The combined aggregate originates in the west coast region and stops in the central region. It is delivered through POD 4 by RDD 2 which is not too early for the last Active Duty ULN required on day 6 or the last Reserve/Guard ULN required on day 4. It contains ULNs between 8 and 57.

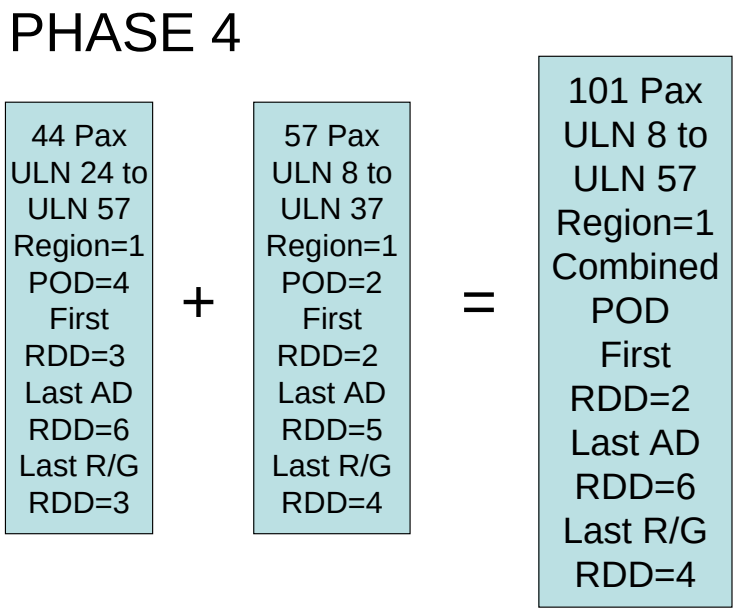

Figure 7. Phase 4 Combination

In figure 7 illustrates what happens when two aggregates containing 40 ≤ passengers < 100 from a common region but going to different PODs are combined to form a cluster with greater than 100 passengers. The first aggregate contains 44 passengers from the west coast traveling to POD 4, and includes Active Duty passengers required on days 3 through 6 and Reserve/Guard passengers required on day 3. It is combined with the 57 passenger aggregate traveling to POD 2 from the west coast region containing Active Duty passengers required on days 2 through 5 and Reserve/Guard passengers required on days 2 through 4. The combined aggregate originates in the west coast region with delivery through POD 4 and POD 2 by RDD 2 which is not too early for the last

Active Duty ULN required on day 6 or the last Reserve/Guard ULN required on day 4. It contains ULNs between 8 and 57.

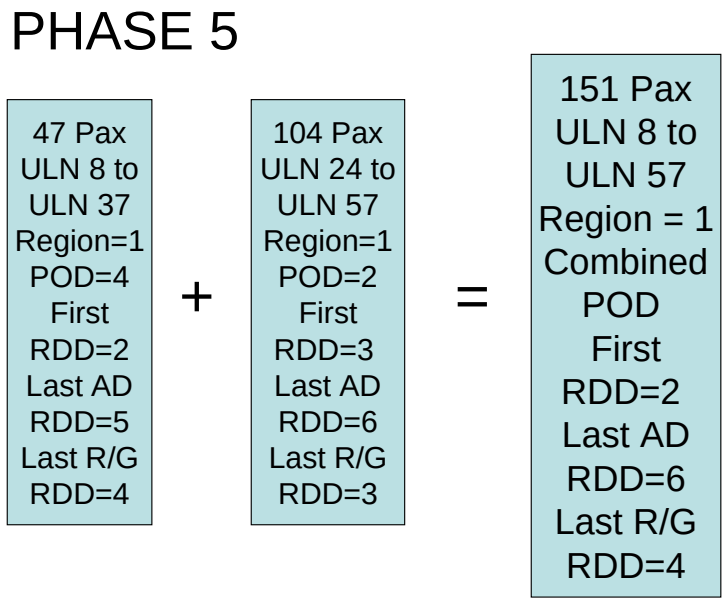

Figure 8. Phase 5 Combination

In figure 8 an aggregate with less than 40 passengers is combined with an aggregate with greater than 100 passengers from the same region but going to different PODs. The first aggregate contains 47 passengers from the west coast region traveling to POD 4, and includes Active Duty passengers required on days 2 through 5 and Reserve/Guard passengers required on day 4. It is combined with the 104 passenger aggregate also from the west coast region traveling to POD 2 containing Active Duty passengers required on days 3 through 6 and Reserve/Guard passengers required on day 3. The combined aggregate originates in the west coast region and is delivered through POD 4 and POD 2 by RDD 2

which is not too early for the last Active Duty ULN required on day 6 or the last Reserve/Guard ULN required on day 4. It contains ULNs between 8 and 57.

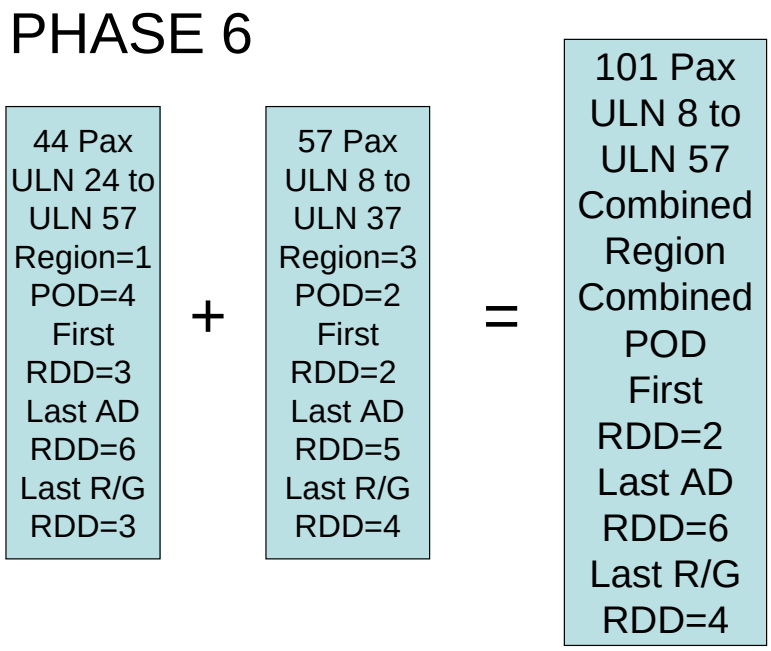

Figure 9. Phase 6 Combination

In figure 9 two aggregates containing 40 ≤ passengers < 100 from different regions and going to different PODs are combined to form an aggregate with greater than 100 passengers. The first aggregate contains 44 passengers from the west coast region traveling to POD 4, and includes Active Duty passengers required on days 3 through 6 and Reserve/Guard passengers required on day 3. It is combined with the 57 passenger aggregate from the central region and contains Active Duty passengers traveling to POD 2 who are required on days 2 through 5 and Reserve/Guard passengers who are required on days 2 through 4. The combined aggregate originates in the west coast region, stops in the central region, and is delivered through POD 4 and POD 2 by RDD 2. RDD 2 is not too

early for the last Active Duty ULN required on day 6 or the last Reserve/Guard ULN required on day 4. It contains ULNs between 8 and 57.

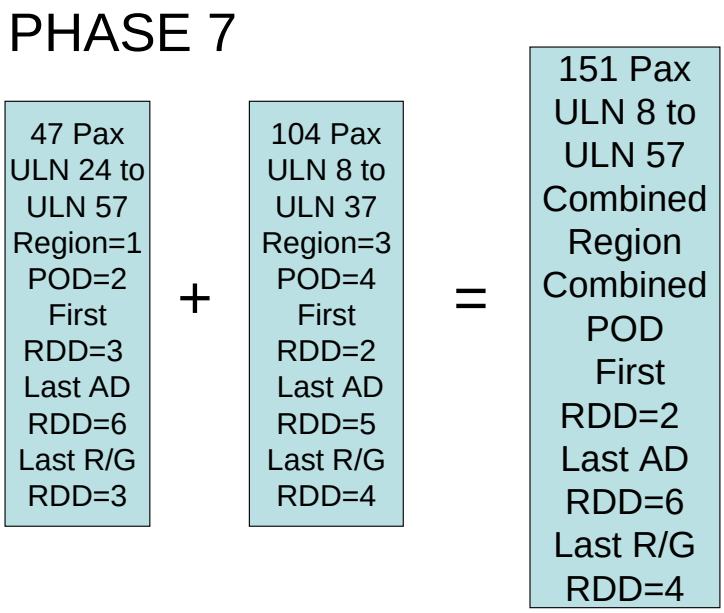

Figure 10. Phase 7 Combination

In figure 10 an aggregate containing 40 ≤ passengers < 100 is combined with an aggregate with greater than 100 passengers from a different region and going to a different POD. The first aggregate contains 47 passengers from the west coast region traveling to POD 2, and includes Active Duty passengers required on days 3 through 6 and Reserve/Guard passengers required on day 3. It is combined with the 104 passenger aggregate from the central region traveling to POD 4 that contains Active Duty passengers required on days 2 through 5 and Reserve/Guard passengers required on days 2 through 4. The combined aggregate originates in the west coast region, stops in the central region, and is delivered through POD 4 and POD 2 by RDD 2 which is not too early for the last Active Duty ULN

required on day 6 or the last Reserve/Guard ULN required on day 4. It contains ULNs between 8 and 57.

3.6 Combining ULNs in Infeasible Aggregates

At this point, any aggregate containing less than 100 passengers is disbanded and the algorithm attempts to fit the unmatched ULNs into a feasible aggregate. For each unmatched ULN, the algorithm checks all feasible aggregates going to the ULN's POD to find one with a feasible RDD span. At this point, the region of the aggregate and ULN is no longer considered since any remaining unmatched ULNs will fly out of the BWI Pax terminal in Baltimore on channel airlift.

3.7 Output and Post Processing

Post processing is accomplished by an analyst at AEFC who views the output and chooses a POE for each aggregate or a combination of aggregates. An example of the output is seen in Table 2. Each line of the output includes all of the aggregation information pertaining to a ULN. The ULN, origin, # passengers, and RDD are taken directly from the TPFDD. The region and POD are represented by a corresponding number. The service column shows whether the ULN is Active Duty (4) or Reserve/Guard (2). The ADRDD and RGRDD columns represent the latest RDD for Active Duty and Reserve/Guard ULNs, respectively, in the aggregate.

The Last and First ULN columns uniquely identify the aggregate by indicating the first and last ULN indices contained in the aggregate. The ULNs assigned to an aggregate are not necessarily listed continuously in the TPFDD; however, all ULNs with the same First and Last ULN values are assigned to the same aggregate allowing the

analyst to sort by these columns to display the aggregate. The total passenger count for the aggregate is listed next, followed by the aggregated index 0 for greater than 100 passenger aggregates and 1 for less than 100 passenger aggregates. Finally, the First RDD column indicates the earliest RDD of the aggregate. All passengers in the aggregate must move as if their RDD were the same as the First RDD of the aggregate as no ULNs can arrive late due to aggregation.

Table 2. Sample Output

Index	ULN	Origin	# Pax	RDD	Regio	POD	Service	RGRD	ADRDD	Last ULN	First UL	Pax in C	Aggre	First RD
2	F9Q1X	ANDREWS	1	1	2	1	4	0	1	2	2	1	0	1
3	F933Q01	BEALE AFI	4	1	4	2	4	2	3	94	3	133	1	1
4	F933R02	BEALE AFI	3	1	1	2	4	0	2	58	4	117	1	1
5	F93BF01	DAVIS MO	4	1	4	2	4	2	3	94	3	133	1	1
6	FT3A1	DAVIS MO	7	1	1	2	4	0	2	58	4	117	1	1
7	F933H01	EDWARDS	17	1	4	2	4	0	2	58	4	117	1	1
8	F933L01	EDWARDS	16	1	4	2	4	0	2	58	4	117	1	1
9	F03EX01	HILL AFB	10	1	4	2	4	0	2	58	4	117	1	1
10	F0HDX01	HILL AFB	12	1	4	2	4	0	2	58	4	117	1	1
11	F0HEX01	HILL AFB	43	1	1	2	4	0	2	58	4	117	1	1
12	F0HHX01	HILL AFB	1	1	1	2	4	0	2	58	4	117	1	1
13	FT638	LANGLEY	1	1	1	2	4	0	2	58	4	117	1	1
14	F9U1A	LANGLEY	2	1	4	2	4	2	3	94	3	133	1	1
15	F93B5	MOODY AF	26	1	4	2	4	2	3	94	3	133	1	1
16	F99U3	MOODY AF	3	1	4	2	4	2	3	94	3	133	1	1
17	F99U4	MOODY AF	10	1	4	2	4	2	3	94	3	133	1	1
18	F9HB6	MOODY AF	2	1	1	2	4	2	3	94	3	133	1	1
19	F9PB5	MOODY AF	2	1	3	2	4	2	3	94	3	133	1	1
20	F0702	NELLIS AF	1	1	4	2	4	2	3	94	3	133	1	1
21	F0PSW01	NELLIS AF	1	1	4	2	4	2	3	94	3	133	1	1
22	F9X3A	ROBINS AI	1	1	2	1	4	0	1	22	22	1	0	1
23	FNJ1A01	ROBINS AI	1	1	2	1	4	0	1	23	23	1	0	1
24	F93AH01	SEYMOUR	4	1	1	2	4	0	2	58	4	117	1	1
25	F9HBJ01	SEYMOUR	4	1	4	2	4	0	2	58	4	117	1	1
27	F9HKJ02	SEYMOUR	74	1	2	2	4	0	2	58	4	117	1	1
29	F9HKK02	SEYMOUR	55	1	4	2	4	0	2	58	4	117	1	1
31	F9HNB	SEYMOUR	2	1	1	2	4	0	2	58	4	117	1	1
32	F9HRH02	SEYMOUR	4	1	3	2	4	0	2	58	4	117	1	1
34	F0PSV01	WASHING	1	1	4	2	4	0	2	58	4	117	1	1
35	FNJ1A02	WASHING	1	1	3	2	4	2	3	94	3	133	1	1
36	F933L02	WRIGHT-P	1	1	4	2	4	0	2	58	4	117	1	1
37	FNFN001	WRIGHT-P	1	1	1	2	4	0	2	58	4	117	1	1
38	FT63601	WRIGHT-P	1	1	1	2	4	0	2	58	4	117	1	1
39	FT63602	WRIGHT-P	1	1	1	2	4	0	2	58	4	117	1	1
40	F96MD	BARKSDAI	1	1	4	2	2	0	2	58	4	117	1	1
41	F99UA01	BEALE AFI	2	1	4	2	2	0	2	58	4	117	1	1
43	FNFN002	LANGLEY	1	1	3	2	2	2	3	94	3	133	1	1
44	F9FB5	MOODY AF	3	1	2	2	2	2	3	94	3	133	1	1
45	F9HB5	MOODY AF	59	1	2	2	2	2	3	94	3	133	1	1
46	F03XM	RENO-TAH	13	1	4	2	2	0	2	58	4	117	1	1
47	F03XN	RENO-TAH	1	1	4	2	2	0	2	58	4	117	1	1
48	F0H02	RENO-TAH	15	1	4	2	2	0	2	58	4	117	1	1
50	F9HBH01	SEYMOUR	10	1	4	2	2	0	2	58	4	117	1	1
52	F070602	TRAVIS AF	1	1	4	2	2	0	2	58	4	117	1	1

The algorithm constructs a solution of feasible aggregates from a TPFDD by generating a bound, creating an initial starting solution, and combining smaller aggregates to create aggregates with at least 100 passengers. The initial starting solution places all ULNs in an aggregate although many aggregates do not contain the required

100 passengers. These infeasible aggregates are combined with other aggregates, both feasible and infeasible, to create more feasible aggregates. The output includes all ULNs and their aggregation information.

4. Testing and Results

4.1 Introduction

This chapter presents the testing of the algorithm on a set of notional TPFDDS and two real world TPFDDs. The effect of each phase of the algorithm on aggregation is noted. The algorithm is also tested with and without regions enforced to see the effect this added constraint has on solution quality. The number of passengers in feasible aggregates over the number of passengers given by the upper bound is used to compare solution quality for all testing. However, many other factors, such as the division of the aggregates across regions, the maximum inclusion of passengers traveling from the same origin, and the number of aggregates generated could also be used to evaluate the solution quality.

4.2 Testing for performance on Notional TPFDDs

Due to the complexity involved in declassifying a TPFDD, this research only had access to three actual TPFDDs, one of which was on a classified system and could not be fully analyzed. Therefore, a process to randomly generate notional TPFDDs was generated using the real world TPFDDs as seeds.

Since the TPFDD used is for rotation of personnel into required positions, it is assumed the actual requirements of these positions do not change significantly. It is also assumed there is some variability in the RDDs for each position from rotation to rotation and large amounts of variability in the origin as different units are tapped to fill the

requirements. The RDDs for each tasking were randomly increased or decreased by up to 3 days. The origins were selected from 80 origins with 20 in each region to evenly space taskings across the four regions. Analysis of the two fully sourced TPFDDs show 14% and 16% of taskings are filled by Reserve/Guard personnel; therefore, a ratio of 15% Reserve/Guard to 85% Active duty taskings was used to generate the notional TPFDDs. To accomplish this, 12 of the 80 origins were designated as Guard/Reserve origins. An origin was then randomly selected, based on a uniform distribution, to fill each ULN. The two real world TPFDDS were from 12 October (AEF 7/8) and 12 January (AEF 9/10). Notional TPFDDs 1-15 were generated from the 12 January TPFDD and 16-30 were generated from the 12 October TPFDD.

The first test returns the contribution of each phase of the algorithm to determine if all stages aggregate at least some ULNs. A percentage was gathered after the initial starting solution was generated (initial), after < 40 passenger aggregates were grouped with feasible aggregates (Phase 1), after > 40 passenger aggregates were grouped with each other and then feasible aggregates from different regions (Phases 2 and 3), with different PODs (Phases 4 and 5), then different regions and PODs (Phases 6 and 7), and after attempting to group all unmatched ULNs with feasible aggregates (Final Phase). Results of this testing can be seen in Table 3.

Table 3. Aggregation Results for Notional TPFDDs

	Percentage of Aggregation for Each Stage						% Bound/
TPFDD #	Initial	1	2/3	4/5	6/7	Final	Totalpax
1	23.71	30.45	70.61	72.57	74.79	86.28	97.28
2	20.36	27.14	70.06	72.51	77.52	85.69	96.71
3	19.09	23.75	68.07	72.25	72.80	86.23	97.08
4	21.69	28.36	69.02	71.77	72.93	90.23	98.22
5	21.37	24.69	69.78	72.98	73.61	88.68	98.75
6	23.88	27.21	74.00	75.53	76.79	87.11	98.33
7	23.75	28.34	66.82	69.52	71.65	84.52	97.41
8	16.10	27.19	66.76	71.86	73.13	87.41	98.51
9	24.97	29.49	69.89	73.47	73.47	89.39	96.85
10	24.51	32.47	72.74	76.84	77.93	89.84	97.10
11	20.34	28.15	70.51	72.80	75.16	89.88	97.92
12	24.55	27.93	68.83	71.57	73.33	88.27	97.09
13	17.23	26.32	71.97	74.99	76.84	92.45	96.15
14	23.30	28.89	68.40	72.21	72.68	90.09	98.28
15	29.70	31.96	66.89	75.43	77.81	91.94	97.53
16	38.65	40.03	66.95	70.78	70.78	82.03	97.83
17	30.95	36.54	67.82	68.98	68.98	77.81	96.90
18	29.77	39.51	65.42	67.28	67.28	76.31	97.44
19	33.66	36.47	66.75	68.07	68.87	78.50	96.57
20	31.47	37.13	69.06	69.06	69.66	80.46	97.52
21	35.51	38.30	64.60	66.96	66.96	79.71	97.19
22	36.31	41.90	69.27	70.69	71.26	80.43	96.92
23	27.69	36.10	64.48	67.03	67.67	82.27	96.77
24	32.79	38.34	62.63	67.31	67.31	81.50	97.72
25	34.70	41.67	65.20	68.61	68.61	77.70	97.30
26	34.20	39.73	63.62	65.15	65.78	79.44	97.95
27	33.08	41.47	66.69	68.92	68.92	81.71	96.96
28	31.46	37.04	67.61	70.09	70.82	78.68	97.17
29	29.64	39.38	65.02	68.81	68.81	78.92	97.42
30	28.43	36.76	64.32	68.27	68.27	76.00	97.61
Average	27.43	33.42	67.79	70.74	71.51	83.98	97.42
Max	38.65	41.90	74.00	76.84	77.93	92.45	98.75
Min	16.10	23.75	62.63	65.15	65.78	76.00	96.15

Each phase of aggregation did contribute to the final solution. Eliminating a phase could possibly lead to the same level of aggregation or even higher since some can

be viewed as a restriction. However, the phases are used to enforce the priorities of the decision maker and therefore provide functionality beyond the overall aggregation rate.

The algorithm performed better on the TPFDDs created using the 12 January TPFDD than those created from the 12 October TPFDD. These TPFDDs differ in the tightness of the movement window which is discussed further in Section 4.4. Overall, the algorithm performed well and significantly outperformed the manual aggregation percentage of 66%. A 95% confidence interval was calculated for the results using the interval estimator for quantiles presented by Banks (2005) using Equation (4.1) below:

$$\hat{p} \pm z_{\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{R-1}} \tag{4.1}$$

This equation takes into account that the results are a proportion. The confidence intervals for performance are presented in Table 4.

Table 4. Confidence Interval

Confidence Interval on the final Aggregation %		
Interval	Upper	Lower
95%	97.33	70.63

4.3 Testing for Performance on Real World TPFDDs

The algorithm was also tested on two real world TPFDDs. As seen below in Table 5, the algorithm was able to aggregate a greater percentage of the passengers than in the notional TPFDDs. The different performance is due to the fact that the notional TPFDDs were randomly sourced. Sourcing performed at AEFC is not done randomly. As seen in the small sample TPFDD in Chapter 2, some UTCs originate in the same location with

the same POD and RDD. This makes a logically sourced TPFDD more likely to have a higher aggregation percentage than a randomly sourced TPFDD. The 12 October TPFDD is from AEF cycle 7/8 of which 64% of the passengers were manually aggregated. The 12 January TPFDD is from AEF cycle 9/10 of which 66% of the passengers were manually aggregated.

Table 5. Aggregation Results for Real World TPFDDs

12 October TPFDD Pull:		Bound = 7232		Total Passengers = 7377		
Stage	Initial	1	2/3	4/5	6/7	Initial
Total Pax	2757	4687	6037	6079	6194	6504
% Bound	38.12	64.81	83.48	84.06	85.65	89.93
% Total	37.37	63.54	81.84	82.40	83.96	88.17
12 January TPFDD Pull:		Bound = 9087		Total Passengers = 9150		
Stage	Initial	1	2/3	4/5	6/7	Initial
Total Pax	6789	6870	8544	8608	8653	8772
% Bound	74.71	75.60	94.02	94.73	95.22	96.53
% Total	74.20	75.08	93.38	94.08	94.57	95.87

The 12 October TPFDD has 1773 less passengers than the 12 January TPFDD. Both TPFDDs have the same number of in-cycle AEF rotation personnel. The difference is in the amount of out of cycle personnel or those personnel on a 6-month rotation versus the normal 4-month AEF rotation. The 12 October TPFDD has less out of cycle personnel and has RDDs spread out farther than those in the 12 January TPFDD. This increase in the number of personnel traveling in the first 30 days of the TPFDD allowed for more aggregation in the 12 January TPFDD. Figure 11 illustrates the difference in the two TPFDDs. The 12 October TPFDD has taskings spread out over a wider range of RDDs than the 12 January TPFDD. There are no ULNs after RDD 33 that could be assigned to an aggregate in the 12 January TPFDD. However, after RDD 33, there are only 25 ULNs containing a total of 63 passengers – a small loss in the final percentage.

Conversely, there are 90 ULNs containing a total 274 passengers after RDD 33 in the 12 October TPFDD with some ULNs as late as RDD 55. TPFDDS with all passengers in a relatively short span of RDDs allows more of them to be grouped together into feasible aggregates.

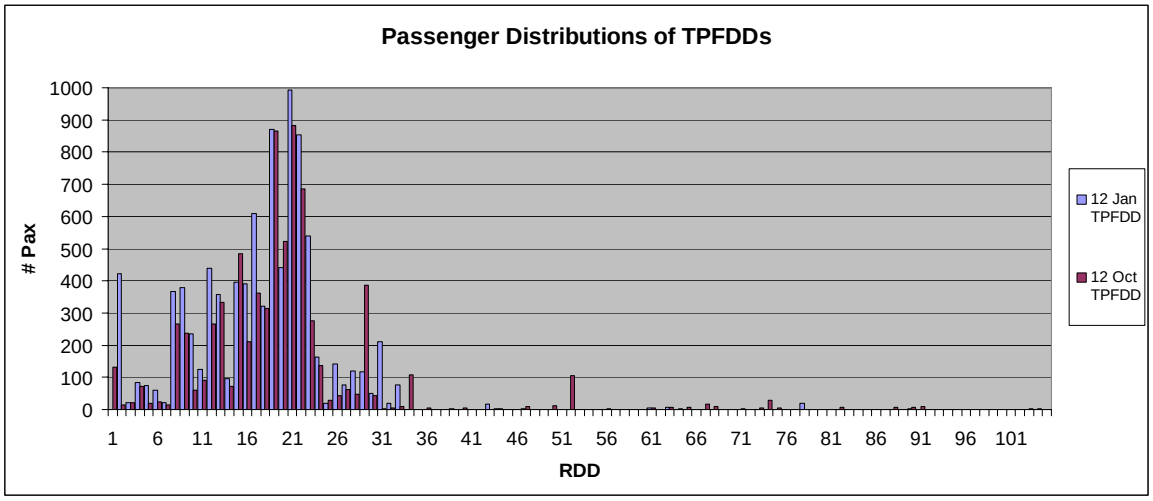

Figure 11. Passenger Distributions of Real World TPFDDs

4.4 Analysis of Removing the Regions Constraint

Finally, the effect of the region constraints on the solution is tested. To accomplish this, all ULNs were labeled as being in one region. Of course, phases 2/3 and 6/7 were had no effect as they are based on the combination of regions. The increase in percentage of aggregation is seen in Tables 6 and 7. As expected, removing the region constraint increased the aggregation percentage, reaching the optimal value of 100% of the bound on one TPFDD. When the region constraints are removed, the goal of aggregation and ITV is still achieved; however, the personnel may be required to utilize significantly more commercial airlift to get to their POE.

Table 6. Aggregation Results of Notional TPFDDs without Region Constraints

Percentage of Aggregation for Each Stage Without Region Constraint				
TPFDD	Initial	1	4/5	Final
1	46.512	70.104	89.147	99.079
2	53.927	72.008	91.796	98.576
3	54.723	69.357	94.743	100.000
4	51.641	68.332	90.531	97.307
5	52.933	71.746	93.515	97.665
6	43.992	69.557	91.619	98.333
7	54.796	72.748	92.606	98.463
8	50.566	71.644	95.329	99.079
9	54.694	72.749	91.864	99.413
10	52.977	73.236	93.438	98.762
11	45.759	68.080	91.484	97.288
12	47.479	73.368	94.169	99.381
13	52.194	68.106	94.442	98.943
14	49.416	70.544	91.771	98.021
15	55.087	71.896	92.346	98.375
16	44.520	61.147	81.779	91.811
17	50.518	63.109	79.337	94.306
18	51.906	61.644	77.476	93.934
19	42.673	63.728	83.156	92.701
20	55.796	64.137	83.611	93.022
21	51.325	63.877	82.510	91.660
22	45.776	65.357	83.972	97.455
23	48.186	64.995	83.023	94.509
24	48.218	62.089	78.166	93.439
25	43.982	67.665	81.583	94.372
26	52.491	63.562	81.843	92.306
27	54.383	69.761	84.887	93.499
28	45.801	68.122	84.208	91.057
29	53.889	66.412	85.056	93.238
30	49.924	61.033	80.017	93.418
Average	50.203	67.670	87.314	95.980
Max	55.796	73.368	95.329	100.000
Min	42.673	61.033	77.476	91.057

Table 7. Aggregation Results of Real World TPFDDs without Region Constraints

12 October TPFDD	Bound =7232		Total Passengers = 7377			Final Using
Stage	Initial	1	4/5	Final		Regions
Total Pax	4403	6310	6518	6956		6504
% Bound	60.882	87.251	90.127	96.184		89.934
% Total	59.686	85.536	88.356	94.293		88.166
12 January TPFDD	Bound = 9087			Total Passengers = 9150		
Stage	Initial	1	4/5	Final		w/Regions
Total Pax	6789	6870	8544	8946		8772
% Bound	74.711	75.602	94.024	98.448		96.534
% Total	74.197	75.082	93.377	97.77		95.869

The real world TPFDDs were also aggregated without the region constraints and the results are seen in Table 7. The aggregation percentages increased by 2-5% for the real world TPFDDs. However, without regions to guide the formation of aggregates, they were widespread across the CONUS and would be difficult to use. The difference can be seen in a sample output from the aggregation of the 12 October TPFDD with and without regions in Table 8.

Table 8. Sample Output With and Without Region Constraint

With Region Constraint				Without Region Constraint			
Origin	# Pax	RDD	POD	Origin	# Pax	RDD	POD
BEALE AFB	4	0	1	BEALE AFB	4	0	1
BEALE AFB	4	0	1	BEALE AFB	4	0	1
BEALE AFB	7	0	1	BEALE AFB	7	0	1
BEALE AFB	11	0	1	BEALE AFB	11	0	1
BEALE AFB	1	0	1	BEALE AFB	1	0	1
BEALE AFB	4	0	1	BEALE AFB	4	0	1
BEALE AFB	2	0	2	BEALE AFB	2	0	2
EDWARDS AFB	6	0	1	EDWARDS AFB	6	0	1
EDWARDS AFB	17	0	1	EDWARDS AFB	17	0	1
EDWARDS AFB	16	0	1	EDWARDS AFB	16	0	1
HILL AFB	10	0	2	HILL AFB	10	0	2
HILL AFB	5	0	2	HILL AFB	5	0	2
HILL AFB	43	0	2	HILL AFB	43	0	2
HILL AFB	1	0	2	HILL AFB	1	0	2
DAVIS MONTHAN AFB	1	1	2	WRIGHT-PATTERSON	1	0	2
NELLIS AFB	1	1	2	ANDREWS AFB	1	1	2
TRAVIS AFB	1	1	2	ANDREWS AFB	1	1	2
NELLIS AFB	2	3	2	BARKSDALE AFB	1	1	2
TRAVIS AFB	2	3	2	DAVIS MONTHAN AFB	1	1	2
BEALE AFB	1	4	2	LANGLEY AFB	1	1	2
NELLIS AFB	1	4	2	NELLIS AFB	1	1	2
NELLIS AFB	1	4	2	TRAVIS AFB	1	1	2
NELLIS AFB	1	4	2	WRIGHT-PATTERSON	1	1	1

The aggregate generated using the region constraint includes units in the Pacific and Mountain time zones. The aggregate generated without the region constraint is mostly composed of units on the west coast but also includes personnel from all other regions. The 2 passengers from Andrews AFB, for example, would have to fly to the west coast to join the aggregate. This, of course, is undesirable as AEFC would like to minimize the number of passengers having to fly east to west. The other option would be for the aggregate to have a POE on the east coast which would cause 137 passengers to fly commercial or separately contracted flights to the POE. Based on the limited gain in

aggregation in the real world TPFDDs and the preferences of the planners at AEFC, it was decided to use the region constraints.

The testing of the algorithm was performed on a set of notional TPFDDS and two real world TPFDDs. The effect of each phase of the algorithm on aggregation was noted. The effect of the region constraint on solution quality was tested as well. Conclusions drawn about these results, the impact of this research, its current application, and further research opportunities are discussed in Chapter 5.

5. Conclusion

5.1 Summary of Results

The algorithm performed well on the randomly sourced notional TPFDDs and better on the real world TPFDDs outperforming the manual process by more than 25%. It aggregates a 2500 line TPFDD in approximately 1 minute on a 1.5 GHz PC. In most instances, each phase of the algorithm is utilized to enforce the priorities of the decision maker providing a high aggregation percentage while meeting their desires in terms of the content of the aggregates. Forcing aggregates to conform to regions degrades the aggregation percentage slightly but results in aggregates that can be employed more efficiently and for less cost.

5.2 Airlift Aggregation for AEF Cycle 1/2

In January 2006, the algorithm was used to aggregate passengers for AEF cycle 1/2. Of the 5425 sourced personnel positions in the TPFDD, the algorithm was able to aggregate 4673 or 86.14%. After the remainder of the TPFDD is sourced, we believe the percentage of aggregation will increase. The AEFC planner was able to use these results to dramatically improve the airlift aggregation process for AEF Cycle 1/2 and plans to use the algorithm on future AEF Cycles as well.

A 20% increase in aggregation percentage translates to1,850 additional airmen flying dedicated airlift from their home station (or a POE in their region) to their POD each rotation or 3 times per year. The additional ITV for these airmen means they will receive intratheater airlift within 24 hours of arrival to the POD instead of the usual 72-

hour layover. The result is more airmen are provided to the Combatant Commander in a timelier manner and at less cost and with less time spent away from home.

5.3 Further Research

Further research in the area of airlift aggregation can be broken down into two categories, algorithm improvement and analysis of the constraints. The algorithm could be improved upon to select a POE for each aggregate from a list of designated military installations satisfying the necessities of a POE for aggregated airlift such as sufficient lodging, proximity to a major airport, and adequate passenger handling capability. The selection could be made based on minimizing the cost of transporting all of the aggregate's ULNs to the POE.

Parameters such as feasible RDD spans, minimum number of passengers required for dedicated airlift and minimum number of passengers required for a POE/POD were all provided by the decision maker. The values of these parameters could dramatically affect algorithm performance. Through decision and sensitivity analysis we could other alternative values and the impact on the aggregation rate.

Bibliography

AEF Center Education, Training & Partnership Branch (AEPE), "Wing Leadership Guide to the AEF." Available at: https://aefcenter.acc.af.mil/training /documents/WLG_092205_web.pdf, (2005)

Air Mobility Command. "AMC Air Channel Sequence Listing," available at: https://tacc.scott.af.mil/directorates/xog/docs/sequencelisting.pdf, (7 Jul 2005)

Baker, S., Morton, D., Rosenthal, R. & Williams, L."Optimizing military airlift", *Operations Research* 50(4), 582–602 (2002)

Balakrishnan, P. and J. Storbeck. "McTRESH: Modeling Maximum Coverage with Threshold Constraint." *Environment & Planning B* 18, 459–472. (1991)

Carter, W Brand and Joseph R. Litko. "Simulating the Air Mobility Command Channel Cargo System," *Proceedings of the 1992 Winter Simulation Conference*, 1153-1158 (Winter 1992)

CENTAF, "CENTAF AOR Personnel Routing Instructions." Available at: https://aefcenter.acc.af.mil/aefonline/deployer/CENTAF_Pax_ Routing_Instructions.pdf, (22 Nov 2005)

Current, J. and J. Storbeck. "A Multiobjective Approach to Design Franchise Outlet Networks." *Journal of Operational Research Society* 45, 71–81. (1994)

Hinton, Dave, Victor Wald, Dale Rucker, John Bugner, and James Matthews. "Efficient DoD Organic Airlift System Operations," *Lockheed Martin Integrated Systems and Solutions*, (January 2004)

Hoog, Stephen, "AEF Policy Watch, 15 May 2005." Available at: https://aefcenter.acc.af.mil/cccorner/policywatch_archive.asp, (15 May 2005)

HQ USAF/ILXX, "Air Force Instruction 10-403: Deployment Planning and Execution." Available at: http://www.e-publishing.af.mil, (5 Aug 2005)

HQ USAF/XOXW, "Air Force Instruction 10-400: Aerospace Expeditionary Force Planning." Available at: http://www.e-publishing.af.mil (16 Oct 2002)

Laguna, Manuel " Global Optimization and Meta-Heuristics," Available at: http://leeds.colorado.edu/faculty/laguna/articles/elss.pdf (2000)

Smith, Stephen F, Marcel A. Becker and Laurence A. Kramer. "Continuous Management of Airlift and Tanker Resources: A Constraint-Based Approach," *Mathematical and Computer Modeling -- Special Issue on Defense Transportation: Algorithms, Models and Applications for the 21st Century*, Vol. 39, No. 6-8, 2004, pp. 581-598

Wu, T. T., W. B. Powell and A. Whisman, "The Optimizing Simulator: An Intelligent Analysis Tool for the Military Airlift Problem," Department of Operations Research and Financial Engineering Princeton University, (July 2003)

Appendix A: VBA Code

The following is the VBA code run with Microsoft Excel:

```
Sub RDDbound()

L = 2466

N = 105


'This sequence saves the number of active duty troops leaving on RDD i in RDDset(i,0),
'the number of reserve guard troops in RDDset(i,1),
'and the total number of troops in RDDset(i,2)

For o = 0 To 14

Dim RDDset() As Integer, Totalcluster As Integer, RDDAvg() As Integer
Dim Keep() As Boolean, reserveguard() As Boolean


ReDim RDDset(N, 3), Keep(N), reserveguard(N), RDDAvg(N)

Sheets("CONUS B").Select


For i = 0 To N
   Selection.AutoFilter Field:=5 * o + 1, Criteria1:=i
     RDDset(i, 2) = Cells(L + 1, 5 * o + 2).Value
   Selection.AutoFilter Field:=5 * o + 4, Criteria1:=4
     RDDset(i, 0) = Cells(L + 1, 5 * o + 2).Value
   Selection.AutoFilter Field:=5 * o + 4
   Selection.AutoFilter Field:=5 * o + 4, Criteria1:=2
     RDDset(i, 1) = Cells(L + 1, 5 * o + 2).Value
   Selection.AutoFilter Field:=5 * o + 4
Next i
   Selection.AutoFilter Field:=5 * o + 4
   Selection.AutoFilter Field:=5 * o + 1

'Check RDD sets in the tails to see if they can be clustered

For y = 0 To 3
```

```
      If RDDset(0 + y, 0) + RDDset(1 + y, 0) + RDDset(2 + y, 0) + RDDset(3 + y, 0) +
RDDset(4 + y, 0) _
         + RDDset(0 + y, 1) + RDDset(1 + y, 1) + RDDset(2 + y, 1) > 100 Then
         Keep(0 + y) = True
         Keep(1 + y) = True
         Keep(2 + y) = True
         Keep(3 + y) = True
         Keep(4 + y) = True
      End If
Next y

For z = 0 To 3
      If RDDset(N - 4 - z, 0) + RDDset(N - 3 - z, 0) + RDDset(N - 2 - z, 0) + RDDset(N -
1 - z, 0) + _
        RDDset(N - z, 0) + RDDset(N - 4 - z, 1) + RDDset(N - 3 - z, 1) + RDDset(N - 2 -
z, 1) > 100 Then
         Keep(N - 4 - z) = True
         Keep(N - 3 - z) = True
         Keep(N - 2 - z) = True
         Keep(N - 1 - z) = True
         Keep(N - z) = True
      End If
Next z

'Check RDD sets in the middle to see if they can be clustered

   For S = 4 To N - 4
      If RDDset(S, 0) > 100 Then
         Keep(S) = True
      Else
       If RDDset(S, 2) = 0 Then
         Keep(S) = False
       Else
       For d = 0 To 4
         If RDDset((S - (4 - d)), 0) + RDDset((S - (3 - d)), 0) + RDDset((S - (2 - d)), 0) +
RDDset((S - (1 - d)), 0) + _
          RDDset((S + d), 0) + RDDset((S - (4 - d)), 1) + RDDset((S - (3 - d)), 1) +
RDDset((S - (2 - d)), 1) > 100 Then
             Keep(S) = True
          End If
       Next d
       End If
      End If
   Next S
```

```
For x = 1 To N
   RDDAvg(x) = RDDAvg(x) + RDDset(x, 2)
Next x


'Output Results of RDD sets that can possibly be clustered, this is used as a bound
Totalcluster = 0
totalnotcluster = 0
x = 0
For b = 0 To N
  If Keep(b) = True Then
    Worksheets("Data").Range("A1").Offset(x + 1, 5 * o) = b
    Worksheets("Data").Range("A1").Offset(x + 1, 5 * o + 1) = RDDset(b, 2)
    Worksheets("Data").Range("A1").Offset(x + 1, 5 * o + 2) = RDDset(b, 0)
    Worksheets("Data").Range("A1").Offset(x + 1, 5 * o + 3) = RDDset(b, 1)
    x = x + 1
    Totalcluster = Totalcluster + RDDset(b, 2)
  Else
    totalnotcluster = totalnotcluster + RDDset(b, 2)
  End If
Next b
    Worksheets("Data").Range("A1").Offset(0, 5 * o + 4) = "Total Pax Removed"
    Worksheets("Data").Range("A1").Offset(1, 5 * o + 4) = totalnotcluster
    Worksheets("Data").Range("A1").Offset(5, 5 * o + 4) = "Total Pax to Cluster"
    Worksheets("Data").Range("A1").Offset(6, 5 * o + 4) = Totalcluster

Next o

For b = 0 To N
   Worksheets("Data2").Range("A1").Offset(b, 0) = RDDAvg(N) / 15
Next b

End Sub
```

```
Sub Aggregation()
'
' Macro3 Macro
' Macro recorded 12/6/2005 by AFIT
'

N = 105        'the number of RDD's in the TPFDD
L = 2466       'the number of lines in the TPFDD


Dim ULNL() As Integer, ULNI() As Integer
Dim FirstULN, LastULN As Integer
Dim v() As Integer

o = 1 '(this can be used as a loop for multiple runs)

ReDim ULNI(L, 12), ULNL(L, 2), v(8)

' This sequence will build a ULN matrix with ULN, #Pax, RDD, Region, POD, Active or
Reserve/Guard,
' and Earliest RDD in its cluster
' ULNL includes line number and Origin
' ULNI includes #Pax, RDD, Region, Active Duty or Guard, and the Cluster that the ULN
is in

Sheets("CONUS B").Select
Range("A1").Select

For j = 2 To L - 2

If Cells(j, 5 * o + 2) > 0 Then
   region = Cells(j, 5 * o + 3).Value / 20
   ULNI(j, 0) = Cells(j, 5 * o + 2).Value
   ULNI(j, 1) = Cells(j, 5 * o + 1).Value
   ULNL(j, 1) = Cells(j, 5 * o + 3).Value
   ULNL(j, 2) = Cells(j, 5 * o + 3).Value
   ULNI(j, 2) = region
   ULNI(j, 3) = Cells(j, 5 * o + 5).Value
   ULNI(j, 4) = Cells(j, 5 * o + 4).Value


End If

Next j
```

```
'This sequence clusters ULNs with the same origin and POD leaving in a feasible RDD span
'Clusters < 40 are kept and others are broken down to ULNs

For g = 1 To 5


For k = 0 To L
If ULNI(L - k, 3) = g And ULNI(L - k, 0) <> 0 And ULNI(L - k, 9) = 0 Then
 ULNI(L - k, 7) = L - k
 ULNI(L - k, 8) = L - k
 ULNI(L - k, 9) = ULNI(L - k, 0)
 If ULNI(L - k, 4) = 2 Then
    ULNI(L - k, 5) = ULNI(L - k, 1)
 End If
 If ULNI(L - k, 4) = 4 Then
    ULNI(L - k, 6) = ULNI(L - k, 1)
 End If

 For r = k + 1 To L
 If r <> k And ULNL(L - k, 2) = ULNL(L - r, 2) And ULNI(L - r, 3) = g And ULNI(L - r, 7) = 0 Then
    If ULNI(L - k, 6) - ULNI(L - r, 1) <= 4 And ULNI(L - k, 5) - ULNI(L - r, 1) <= 2 Then

        If ULNI(L - r, 4) = 4 Then
           If ULNI(L - k, 6) = 0 Then
              ULNI(L - k, 6) = ULNI(L - r, 1)
              ULNI(L - r, 6) = ULNI(L - r, 1)
              ULNI(L - r, 5) = ULNI(L - k, 5)
           Else
              ULNI(L - r, 6) = ULNI(L - k, 6)
           End If
        End If

        If ULNI(L - r, 4) = 2 Then
           If ULNI(L - k, 5) = 0 Then
              ULNI(L - k, 5) = ULNI(L - r, 1)
              ULNI(L - r, 6) = ULNI(L - r, 1)
              ULNI(L - r, 5) = ULNI(L - k, 5)
           Else
              ULNI(L - r, 5) = ULNI(L - k, 5)
           End If
        End If
```

```
        ULNI(L - r, 7) = L - k
        ULNI(L - k, 8) = L - r
        ULNI(L - k, 9) = ULNI(L - k, 9) + ULNI(L - r, 0)
    End If
  End If
  Next r


  For u = ULNI(L - k, 8) To ULNI(L - k, 7)
    If ULNL(u, 2) = ULNL(L - k, 2) And ULNI(u, 3) = g Then
     ULNI(u, 5) = ULNI(L - k, 5)
     ULNI(u, 6) = ULNI(L - k, 6)
     ULNI(u, 8) = ULNI(L - k, 8)
     ULNI(u, 9) = ULNI(L - k, 9)
     ULNI(u, 11) = ULNI(ULNI(L - k, 8), 1)
     ULNI(u, 10) = 1
     ULNI(u, 12) = 99
    End If
  Next u
End If
Next k
Next g

'Clusters < 75 are broken into ULNs

For r = 1 To L
  If ULNI(r, 9) < 75 Then
    ULNI(r, 5) = 0
    ULNI(r, 6) = 0
    ULNI(r, 7) = 0
    ULNI(r, 8) = 0
    ULNI(r, 9) = 0
    ULNI(r, 10) = 0
    ULNI(r, 11) = 0
    ULNI(r, 12) = 0
  End If
Next r

'Output ULNs with data


For b = 0 To L
If ULNI(b, 9) >= 100 Then
    v(0) = v(0) + ULNI(b, 0)
```

```
End If
Next b

' This sequence will build clusters starting from the end of the TPFDD it names
' clusters using the variable cluster(i,j) starting on RDD cluster(i,0), ending on RDD j,
' and conntaining cluster(i,1) pax


For p = 1 To 4  '**************comment out to place all in one region***********
'p = 1
For g = 1 To 5


For k = 0 To L
If ULNI(L - k, 2) = p And ULNI(L - k, 3) = g And ULNI(L - k, 0) <> 0 And ULNI(L - k,
9) = 0 Then
 ULNI(L - k, 7) = L - k
 ULNI(L - k, 8) = L - k
 ULNI(L - k, 9) = ULNI(L - k, 0)
 If ULNI(L - k, 4) = 2 Then
   ULNI(L - k, 5) = ULNI(L - k, 1)
 End If
 If ULNI(L - k, 4) = 4 Then
   ULNI(L - k, 6) = ULNI(L - k, 1)
 End If

 For r = k + 1 To L
 If r <> k And ULNI(L - r, 2) = p And ULNI(L - r, 3) = g And ULNI(L - r, 7) = 0 Then
   If ULNI(L - k, 6) - ULNI(L - r, 1) <= 4 And ULNI(L - k, 5) - ULNI(L - r, 1) <= 2
Then
      If ULNI(L - r, 1) = ULNI(ULNI(L - k, 8), 1) And ULNL(L - r, 2) = ULNL(ULNI(L
- k, 8), 2) Then
         ULNI(L - r, 8) = 9999
      End If
   If ULNI(L - k, 9) < 100 Or ULNI(L - r, 8) = 9999 Then

      If ULNI(L - r, 4) = 4 Then
         If ULNI(L - k, 6) = 0 Then
            ULNI(L - k, 6) = ULNI(L - r, 1)
            ULNI(L - r, 6) = ULNI(L - r, 1)
            ULNI(L - r, 5) = ULNI(L - k, 5)
         Else
            ULNI(L - r, 6) = ULNI(L - k, 6)
         End If
      End If
```

```
      If ULNI(L - r, 4) = 2 Then
         If ULNI(L - k, 5) = 0 Then
            ULNI(L - k, 5) = ULNI(L - r, 1)
            ULNI(L - r, 6) = ULNI(L - r, 1)
            ULNI(L - r, 5) = ULNI(L - k, 5)
         Else
            ULNI(L - r, 5) = ULNI(L - k, 5)
         End If
      End If

      ULNI(L - r, 7) = L - k
      ULNI(L - k, 8) = L - r
      ULNI(L - k, 9) = ULNI(L - k, 9) + ULNI(L - r, 0)
   End If
 End If
End If
 Next r


 For u = ULNI(L - k, 8) To ULNI(L - k, 7)
  If ULNI(u, 2) = p And ULNI(u, 3) = g And ULNI(u, 10) = 0 Then
   ULNI(u, 5) = ULNI(L - k, 5)
   ULNI(u, 6) = ULNI(L - k, 6)
   ULNI(u, 8) = ULNI(L - k, 8)
   ULNI(u, 9) = ULNI(L - k, 9)
   ULNI(u, 11) = ULNI(ULNI(L - k, 8), 1)
   ULNI(u, 12) = 98
  End If
 Next u
End If
Next k
Next g
Next p  '************comment out to place all in one region*****************

For y = 2 To L
  If ULNI(y, 9) >= 100 Then
    ULNI(y, 10) = 1
  End If
Next y


'output ULNs with their cluster data

For b = 0 To L
```

```
If ULNI(b, 9) >= 100 Then
   v(1) = v(1) + ULNI(b, 0)
End If
Next b



'This sequence trys to combine ULNs in clusters with less than 20 pax with other clusters going to the same POD
' a group of less than 20 pax can be combined with a cluster to the East going to the same POD

For m = 2 To L
If ULNI(L - m, 0) <> 0 Then
If ULNI(L - m, 9) < 20 And ULNI(L - m, 12) > 1 Then
   For g = m To L
      If ULNI(L - g, 10) = 1 And g <> m And ULNI(L - g, 12) > 1 And _
      ULNI(L - m, 2) < ULNI(L - g, 2) And _
      ULNI(L - m, 8) <> ULNI(L - g, 8) And ULNI(L - m, 11) >= ULNI(L - g, 11) Then
       If ULNI(L - g, 3) = ULNI(L - m, 3) And _
         ULNI(L - m, 6) - ULNI(ULNI(L - g, 8), 1) <= 4 And _
         ULNI(L - m, 5) - ULNI(ULNI(L - g, 8), 1) <= 2 Then

         If ULNI(L - g, 7) > ULNI(L - m, 7) Then
            LastULN = ULNI(L - g, 7)
            Else
            LastULN = ULNI(L - m, 7)
         End If

         If ULNI(L - g, 8) < ULNI(L - m, 8) Then
            FirstULN = ULNI(L - g, 8)
            Else
            FirstULN = ULNI(L - m, 8)
         End If

         totalpax = ULNI(L - g, 9) + ULNI(L - m, 9)

         If ULNI(L - g, 5) > ULNI(L - m, 5) Then
            ULNI(L - m, 5) = ULNI(L - g, 5)
            Else
            ULNI(L - g, 5) = ULNI(L - m, 5)
         End If

         If ULNI(L - g, 6) > ULNI(L - m, 6) Then
            ULNI(L - m, 6) = ULNI(L - g, 6)
```

```
            Else
            ULNI(L - g, 6) = ULNI(L - m, 6)
          End If

          If ULNI(L - m, 8) = ULNI(L - m, 7) Then
            ULNI(L - m, 12) = m
          Else

          For z = ULNI(L - m, 8) To ULNI(L - m, 7)
            If ULNI(z, 7) = ULNI(L - m, 7) Then
              ULNI(z, 12) = m
            End If
          Next z
          End If

         For z = ULNI(L - g, 8) To ULNI(L - g, 7)
            If ULNI(z, 7) = ULNI(L - g, 7) Then
              ULNI(z, 12) = m
            End If
         Next z

          For y = FirstULN To LastULN
            If ULNI(y, 12) = m Then
              ULNI(y, 5) = ULNI(L - g, 5)
              ULNI(y, 6) = ULNI(L - g, 6)
              ULNI(y, 7) = LastULN
              ULNI(y, 8) = FirstULN
              ULNI(y, 9) = totalpax
              ULNI(y, 10) = 1
              ULNI(y, 11) = ULNI(ULNI(L - g, 8), 1)
              ULNI(y, 12) = 1
            End If
          Next y

        g = L
      End If

    End If
  Next g
 End If
End If
Next m


'output ULNs with their cluster data
```

```
For b = 0 To L
If ULNI(b, 9) >= 100 Then
   v(2) = v(2) + ULNI(b, 0)
End If
Next b



'This sequence groups clusters with > 40 pax together with clusters from other regions
'which are going to the same POD



For m = 2 To L

If ULNI(L - m, 9) < 100 And ULNI(L - m, 9) > 40 And ULNI(L - m, 12) > 2 Then
   For g = m To L
      If g <> m And ULNI(L - g, 9) + ULNI(L - m, 9) >= 100 And ULNI(L - g, 9) < 100
_
       And ULNI(L - g, 12) > 2 And ULNI(L - m, 8) <> ULNI(L - g, 8) Then
       If ULNI(L - g, 3) = ULNI(L - m, 3) And _
        ULNI(L - m, 6) - ULNI(ULNI(L - g, 8), 1) <= 4 And _
        ULNI(L - m, 5) - ULNI(ULNI(L - g, 8), 1) <= 2 Then

        If ULNI(L - g, 7) > ULNI(L - m, 7) Then
           LastULN = ULNI(L - g, 7)
           Else
           LastULN = ULNI(L - m, 7)
        End If

        If ULNI(L - g, 8) < ULNI(L - m, 8) Then
           FirstULN = ULNI(L - g, 8)
           Else
           FirstULN = ULNI(L - m, 8)
        End If

        ULNI(L - g, 9) = ULNI(L - g, 9) + ULNI(L - m, 9)

        If ULNI(L - g, 5) > ULNI(L - m, 5) Then
           ULNI(L - m, 5) = ULNI(L - g, 5)
           Else
           ULNI(L - g, 5) = ULNI(L - m, 5)
        End If
```

```
        If ULNI(L - g, 6) > ULNI(L - m, 6) Then
           ULNI(L - m, 6) = ULNI(L - g, 6)
           Else
           ULNI(L - g, 6) = ULNI(L - m, 6)
        End If


        For z = ULNI(L - m, 8) To ULNI(L - m, 7)
          If ULNI(z, 7) = ULNI(L - m, 7) Then
             ULNI(z, 12) = m
           End If
        Next z

        For z = ULNI(L - g, 8) To ULNI(L - g, 7)
           If ULNI(z, 7) = ULNI(L - g, 7) Then
              ULNI(z, 12) = m
           End If
        Next z


        For y = FirstULN To LastULN
           If ULNI(y, 12) = m Then
              ULNI(y, 5) = ULNI(L - g, 5)
              ULNI(y, 6) = ULNI(L - g, 6)
              ULNI(y, 7) = LastULN
              ULNI(y, 8) = FirstULN
              ULNI(y, 9) = ULNI(L - g, 9)
              ULNI(y, 10) = 1
              ULNI(y, 11) = ULNI(FirstULN, 1)
              ULNI(y, 12) = 2
           End If
        Next y

      g = L
     End If

   End If
  Next g
 End If
Next m

'Output Results

For b = 0 To L
If ULNI(b, 9) >= 100 Then
```

```
    v(3) = v(3) + ULNI(b, 0)
End If
Next b


' Next Clusters > 40 pax are clustered with clusters > 100 going to the same POD from
' different regions

For m = 2 To L

If ULNI(L - m, 9) < 100 And ULNI(L - m, 9) > 40 And ULNI(L - m, 12) > 3 Then
   For g = m To L
      If ULNI(L - g, 10) = 1 And ULNI(L - g, 9) + ULNI(L - m, 9) >= 100 _
        And g <> m And ULNI(L - g, 12) > 3 And ULNI(L - m, 8) <> ULNI(L - g, 8)
Then
       If ULNI(L - g, 3) = ULNI(L - m, 3) And _
         ULNI(L - m, 6) - ULNI(ULNI(L - g, 8), 1) <= 4 And _
         ULNI(L - m, 5) - ULNI(ULNI(L - g, 8), 1) <= 2 Then

         If ULNI(L - g, 7) > ULNI(L - m, 7) Then
            LastULN = ULNI(L - g, 7)
            Else
            LastULN = ULNI(L - m, 7)
         End If

         If ULNI(L - g, 8) < ULNI(L - m, 8) Then
            FirstULN = ULNI(L - g, 8)
            Else
            FirstULN = ULNI(L - m, 8)
         End If

         ULNI(L - g, 9) = ULNI(L - g, 9) + ULNI(L - m, 9)

         If ULNI(L - g, 5) > ULNI(L - m, 5) Then
            ULNI(L - m, 5) = ULNI(L - g, 5)
            Else
            ULNI(L - g, 5) = ULNI(L - m, 5)
         End If

         If ULNI(L - g, 6) > ULNI(L - m, 6) Then
            ULNI(L - m, 6) = ULNI(L - g, 6)
            Else
            ULNI(L - g, 6) = ULNI(L - m, 6)
         End If
```

```
        For z = ULNI(L - m, 8) To ULNI(L - m, 7)
           If ULNI(z, 7) = ULNI(L - m, 7) Then
              ULNI(z, 12) = m
           End If
        Next z

        For z = ULNI(L - g, 8) To ULNI(L - g, 7)
          If ULNI(z, 7) = ULNI(L - g, 7) Then
              ULNI(z, 12) = m
           End If
        Next z

         For y = FirstULN To LastULN
            If ULNI(y, 12) = m Then
               ULNI(y, 5) = ULNI(L - g, 5)
               ULNI(y, 6) = ULNI(L - g, 6)
               ULNI(y, 7) = LastULN
               ULNI(y, 8) = FirstULN
               ULNI(y, 9) = ULNI(L - g, 9)
               ULNI(y, 10) = 1
               ULNI(y, 11) = ULNI(FirstULN, 1)
               ULNI(y, 12) = 3
            End If
         Next y

       g = L
     End If

   End If
  Next g
 End If
Next m

'output ULNs with their cluster data
For b = 0 To L
If ULNI(b, 9) >= 100 Then
   v(4) = v(4) + ULNI(b, 0)
End If
Next b

'
'This sequence groups clusters with > 40 pax together with clusters from the same region
'which are going to a different POD
```

```
For m = 2 To L

If ULNI(L - m, 9) < 100 And ULNI(L - m, 9) > 40 And ULNI(L - m, 12) > 4 Then
   If ULNI(L - m, 3) = 1 Or ULNI(L - m, 3) = 2 Or ULNI(L - m, 3) = 4 Then
   For g = m To L
      If g <> m And ULNI(L - g, 9) + ULNI(L - m, 9) >= 100 And ULNI(L - g, 12) > 4 _
        And ULNI(L - g, 12) > 4 And ULNI(L - m, 8) <> ULNI(L - g, 8) _
         And ULNI(L - g, 9) < 100 Then
       If ULNI(L - m, 3) = 1 Or ULNI(L - m, 3) = 2 Or ULNI(L - m, 3) = 4 Then
       If ULNI(L - g, 2) = ULNI(L - m, 2) And _
        ULNI(L - m, 6) - ULNI(ULNI(L - g, 8), 1) <= 4 And _
        ULNI(L - m, 5) - ULNI(ULNI(L - g, 8), 1) <= 2 Then

        If ULNI(L - g, 7) > ULNI(L - m, 7) Then
           LastULN = ULNI(L - g, 7)
           Else
           LastULN = ULNI(L - m, 7)
        End If

        If ULNI(L - g, 8) < ULNI(L - m, 8) Then
           FirstULN = ULNI(L - g, 8)
           Else
           FirstULN = ULNI(L - m, 8)
        End If

        ULNI(L - g, 9) = ULNI(L - g, 9) + ULNI(L - m, 9)


        If ULNI(L - g, 5) > ULNI(L - m, 5) Then
           ULNI(L - m, 5) = ULNI(L - g, 5)
           Else
           ULNI(L - g, 5) = ULNI(L - m, 5)
        End If

        If ULNI(L - g, 6) > ULNI(L - m, 6) Then
           ULNI(L - m, 6) = ULNI(L - g, 6)
           Else
           ULNI(L - g, 6) = ULNI(L - m, 6)
        End If


        For z = ULNI(L - m, 8) To ULNI(L - m, 7)
           If ULNI(z, 7) = ULNI(L - m, 7) Then
              ULNI(z, 12) = m
           End If
```

```
            Next z

            For z = ULNI(L - g, 8) To ULNI(L - g, 7)
               If ULNI(z, 7) = ULNI(L - g, 7) Then
                  ULNI(z, 12) = m
               End If
            Next z


            For y = FirstULN To LastULN
               If ULNI(y, 12) = m Then
                  ULNI(y, 5) = ULNI(L - g, 5)
                  ULNI(y, 6) = ULNI(L - g, 6)
                  ULNI(y, 7) = LastULN
                  ULNI(y, 8) = FirstULN
                  ULNI(y, 9) = ULNI(L - g, 9)
                  ULNI(y, 10) = 1
                  ULNI(y, 11) = ULNI(FirstULN, 1)
                  ULNI(y, 12) = 4
               End If
            Next y

           g = L
         End If
        End If
       End If
     Next g
   End If
   End If
Next m

' Next Clusters > 40 pax are clustered with clusters > 100 going from the same region to
' different PODs

For m = 2 To L

If ULNI(L - m, 9) < 100 And ULNI(L - m, 9) > 40 And ULNI(L - m, 12) > 5 Then
   If ULNI(L - m, 3) = 1 Or ULNI(L - m, 3) = 2 Or ULNI(L - m, 3) = 4 Then
   For g = m To L
      If ULNI(L - g, 9) + ULNI(L - g, 9) >= 100 _
        And g <> m And ULNI(L - g, 12) > 5 And _
        ULNI(L - m, 8) <> ULNI(L - g, 8) Then
       If ULNI(L - g, 3) = 1 Or ULNI(L - g, 3) = 2 Or ULNI(L - g, 3) = 4 Then
       If ULNI(L - g, 2) = ULNI(L - m, 2) And _
        ULNI(L - m, 6) - ULNI(ULNI(L - g, 8), 1) <= 4 And _
```

```
ULNI(L - m, 5) - ULNI(ULNI(L - g, 8), 1) <= 2 Then

If ULNI(L - g, 7) > ULNI(L - m, 7) Then
   LastULN = ULNI(L - g, 7)
   Else
   LastULN = ULNI(L - m, 7)
End If

If ULNI(L - g, 8) < ULNI(L - m, 8) Then
   FirstULN = ULNI(L - g, 8)
   Else
   FirstULN = ULNI(L - m, 8)
End If

ULNI(L - g, 9) = ULNI(L - g, 9) + ULNI(L - m, 9)

If ULNI(L - g, 5) > ULNI(L - m, 5) Then
   ULNI(L - m, 5) = ULNI(L - g, 5)
   Else
   ULNI(L - g, 5) = ULNI(L - m, 5)
End If

If ULNI(L - g, 6) > ULNI(L - m, 6) Then
   ULNI(L - m, 6) = ULNI(L - g, 6)
   Else
   ULNI(L - g, 6) = ULNI(L - m, 6)
End If

For z = ULNI(L - m, 8) To ULNI(L - m, 7)
  If ULNI(z, 7) = ULNI(L - m, 7) Then
     ULNI(z, 12) = m
  End If
Next z

For z = ULNI(L - g, 8) To ULNI(L - g, 7)
  If ULNI(z, 7) = ULNI(L - g, 7) Then
     ULNI(z, 12) = m
  End If
Next z

For y = FirstULN To LastULN
   If ULNI(y, 12) = m Then
      ULNI(y, 5) = ULNI(L - g, 5)
      ULNI(y, 6) = ULNI(L - g, 6)
      ULNI(y, 7) = LastULN
```

```
                ULNI(y, 8) = FirstULN
                ULNI(y, 9) = ULNI(L - g, 9)
                ULNI(y, 10) = 1
                ULNI(y, 11) = ULNI(FirstULN, 1)
                ULNI(y, 12) = 5
              End If
            Next y

           g = L
         End If
        End If
      End If
    Next g
   End If
 End If
Next m

'output ULNs with their cluster data
For b = 0 To L
If ULNI(b, 9) >= 100 Then
   v(5) = v(5) + ULNI(b, 0)
End If
Next b


' Next Clusters > 40 pax are clustered with clusters going from different
' regions to different PODs

For m = 2 To L

If ULNI(L - m, 9) < 100 And ULNI(L - m, 9) > 40 And ULNI(L - m, 12) > 6 Then
   If ULNI(L - m, 3) = 1 Or ULNI(L - m, 3) = 2 Or ULNI(L - m, 3) = 4 Then
   For g = m To L
      If ULNI(L - g, 9) < 100 And ULNI(L - g, 9) + ULNI(L - m, 9) >= 100 And ULNI(L
- g, 12) > 6 _
        And g <> m And ULNI(L - g, 9) >= 40 And ULNI(L - m, 8) <> ULNI(L - g, 8)
Then
        If ULNI(L - g, 3) = 1 Or ULNI(L - g, 3) = 2 Or ULNI(L - g, 3) = 4 Then
        If ULNI(L - m, 6) - ULNI(ULNI(L - g, 8), 1) <= 4 And _
         ULNI(L - m, 5) - ULNI(ULNI(L - g, 8), 1) <= 2 Then

         If ULNI(L - g, 7) > ULNI(L - m, 7) Then
            LastULN = ULNI(L - g, 7)
            Else
            LastULN = ULNI(L - m, 7)
```

```
End If

If ULNI(L - g, 8) < ULNI(L - m, 8) Then
   FirstULN = ULNI(L - g, 8)
   Else
   FirstULN = ULNI(L - m, 8)
End If

ULNI(L - g, 9) = ULNI(L - g, 9) + ULNI(L - m, 9)

If ULNI(L - g, 5) > ULNI(L - m, 5) Then
   ULNI(L - m, 5) = ULNI(L - g, 5)
   Else
   ULNI(L - g, 5) = ULNI(L - m, 5)
End If

If ULNI(L - g, 6) > ULNI(L - m, 6) Then
   ULNI(L - m, 6) = ULNI(L - g, 6)
   Else
   ULNI(L - g, 6) = ULNI(L - m, 6)
End If

For z = ULNI(L - m, 8) To ULNI(L - m, 7)
  If ULNI(z, 7) = ULNI(L - m, 7) Then
     ULNI(z, 12) = m
  End If
Next z

For z = ULNI(L - g, 8) To ULNI(L - g, 7)
  If ULNI(z, 7) = ULNI(L - g, 7) Then
     ULNI(z, 12) = m
  End If
Next z

For y = FirstULN To LastULN
   If ULNI(y, 12) = m Then
      ULNI(y, 5) = ULNI(L - g, 5)
      ULNI(y, 6) = ULNI(L - g, 6)
      ULNI(y, 7) = LastULN
      ULNI(y, 8) = FirstULN
      ULNI(y, 9) = ULNI(L - g, 9)
      ULNI(y, 10) = 1
      ULNI(y, 11) = ULNI(FirstULN, 1)
      ULNI(y, 12) = 6
   End If
```

```
          Next y

         g = L
       End If
      End If
    End If
  Next g
  End If
 End If
Next m


'output ULNs with their cluster data
For b = 0 To L
If ULNI(b, 9) > 100 Then
   v(6) = v(6) + ULNI(b, 0)
End If
Next b


' Next Clusters > 40 pax are clustered with clusters >100 pax going from different
' regions to different PODs

For m = 2 To L

If ULNI(L - m, 9) < 100 And ULNI(L - m, 9) > 40 And ULNI(L - m, 12) > 7 Then
   If ULNI(L - m, 3) = 1 Or ULNI(L - m, 3) = 2 Or ULNI(L - m, 3) = 4 Then
   For g = m To L
      If ULNI(L - g, 10) = 1 And ULNI(L - g, 12) > 7 _
        And ULNI(L - m, 8) <> ULNI(L - g, 8) Then
       If ULNI(L - g, 3) = 1 Or ULNI(L - g, 3) = 2 Or ULNI(L - g, 3) = 4 Then
       If ULNI(L - m, 6) - ULNI(ULNI(L - g, 8), 1) <= 4 And _
        ULNI(L - m, 5) - ULNI(ULNI(L - g, 8), 1) <= 2 Then

        If ULNI(L - g, 7) > ULNI(L - m, 7) Then
           LastULN = ULNI(L - g, 7)
           Else
           LastULN = ULNI(L - m, 7)
        End If

        If ULNI(L - g, 8) < ULNI(L - m, 8) Then
           FirstULN = ULNI(L - g, 8)
           Else
           FirstULN = ULNI(L - m, 8)
        End If
```

```
        ULNI(L - g, 9) = ULNI(L - g, 9) + ULNI(L - m, 9)

        If ULNI(L - g, 5) > ULNI(L - m, 5) Then
           ULNI(L - m, 5) = ULNI(L - g, 5)
           Else
           ULNI(L - g, 5) = ULNI(L - m, 5)
        End If

        If ULNI(L - g, 6) > ULNI(L - m, 6) Then
           ULNI(L - m, 6) = ULNI(L - g, 6)
           Else
           ULNI(L - g, 6) = ULNI(L - m, 6)
        End If

       For z = ULNI(L - m, 8) To ULNI(L - m, 7)
          If ULNI(z, 7) = ULNI(L - m, 7) Then
             ULNI(z, 12) = m
          End If
       Next z

       For z = ULNI(L - g, 8) To ULNI(L - g, 7)
          If ULNI(z, 7) = ULNI(L - g, 7) Then
             ULNI(z, 12) = m
          End If
       Next z

        For y = FirstULN To LastULN
           If ULNI(y, 12) = m Then
              ULNI(y, 5) = ULNI(L - g, 5)
              ULNI(y, 6) = ULNI(L - g, 6)
              ULNI(y, 7) = LastULN
              ULNI(y, 8) = FirstULN
              ULNI(y, 9) = ULNI(L - g, 9)
              ULNI(y, 10) = 1
              ULNI(y, 11) = ULNI(FirstULN, 1)
              ULNI(y, 12) = 7
           End If
        Next y

      g = L
    End If
   End If
  End If
Next g
```

```
  End If
 End If
Next m


'output ULNs with their cluster data
For b = 0 To L
If ULNI(b, 9) >= 100 Then
   v(7) = v(7) + ULNI(b, 0)
End If
Next b


'This sequence will look at clusters adjacent to < 100 passenger clusters to see if they can
be combined
'The first sequence makes a forward pass through the TPFDD


'All clusters < 90 are broken into single ULN clusters

For r = 1 To L
   If ULNI(r, 9) < 90 And ULNI(r, 10) = 0 Then
      If ULNI(r, 4) = 2 Then
        ULNI(r, 5) = ULNI(r, 1)
        ULNI(r, 6) = 0
      Else
        ULNI(r, 5) = 0
        ULNI(r, 6) = ULNI(r, 1)
      End If

      ULNI(r, 7) = r
      ULNI(r, 8) = r
      ULNI(r, 9) = ULNI(r, 0)
      ULNI(r, 10) = 0
      ULNI(r, 11) = ULNI(r, 1)
    End If
Next r


'This sequence trys to combine unclustered ULNs with other clusters going to the same
POD
For m = 2 To L

If ULNI(L - m, 10) = 0 Then
```

```
For g = m + 1 To L
   If g <> m And ULNI(L - g, 9) + ULNI(L - m, 9) >= 100 Then
    If ULNI(L - g, 3) = ULNI(L - m, 3) And ULNI(L - g, 2) >= ULNI(L - m, 2) _
     And ULNI(L - m, 6) - ULNI(ULNI(L - g, 8), 1) <= 4 And _
     ULNI(L - m, 5) - ULNI(ULNI(L - g, 8), 1) <= 2 Then

     If ULNI(L - g, 7) > L - m Then
        LastULN = ULNI(L - g, 7)
        Else
        LastULN = L - m
     End If

     If ULNI(L - g, 8) < L - m Then
        FirstULN = ULNI(L - g, 8)
        Else
        FirstULN = L - m
     End If



     If ULNI(L - m, 5) = 0 And ULNI(L - m, 6) <> 0 _
        And ULNI(L - m, 6) > ULNI(L - g, 6) Then
        ULNI(L - g, 6) = ULNI(L - m, 6)
     ElseIf ULNI(L - m, 5) <> 0 And ULNI(L - m, 6) = 0 _
        And ULNI(L - m, 5) > ULNI(L - g, 5) Then
        ULNI(L - g, 5) = ULNI(L - m, 5)
     End If

     ULNI(L - m, 12) = m

     For z = ULNI(L - g, 8) To ULNI(L - g, 7)
        If ULNI(z, 7) = ULNI(L - g, 7) Then
           ULNI(z, 12) = m
        End If
     Next z

      ULNI(L - g, 9) = ULNI(L - g, 9) + ULNI(L - m, 0)


     For y = FirstULN To LastULN
        If ULNI(y, 12) = m Then
           ULNI(y, 5) = ULNI(L - g, 5)
           ULNI(y, 6) = ULNI(L - g, 6)
           ULNI(y, 7) = LastULN
           ULNI(y, 8) = FirstULN
```

```
                ULNI(y, 9) = ULNI(L - g, 9)
                ULNI(y, 10) = 1
                ULNI(y, 11) = ULNI(FirstULN, 1)
                ULNI(y, 12) = 8
              End If
            Next y

          g = L
        End If

      End If
    Next g
  End If
Next m


For m = 2 To L

If ULNI(m, 10) = 0 Then
   For g = m + 1 To L
      If g <> m And ULNI(g, 9) + ULNI(m, 9) >= 100 Then
        If ULNI(g, 3) = ULNI(m, 3) And ULNI(g, 2) >= ULNI(m, 2) _
          And ULNI(g, 6) - ULNI(m, 1) <= 4 And _
          ULNI(g, 5) - ULNI(m, 1) <= 2 Then

          If ULNI(g, 7) > m Then
             LastULN = ULNI(g, 7)
             Else
             LastULN = m
          End If

          If ULNI(g, 8) < m Then
             FirstULN = ULNI(g, 8)
             Else
             FirstULN = m
          End If

          If ULNI(L - m, 5) = 0 And ULNI(L - m, 6) <> 0 _
             And ULNI(L - m, 6) > ULNI(L - g, 6) Then
             ULNI(L - g, 6) = ULNI(L - m, 6)
          ElseIf ULNI(L - m, 5) <> 0 And ULNI(L - m, 6) = 0 _
             And ULNI(L - m, 5) > ULNI(L - g, 5) Then
             ULNI(L - g, 5) = ULNI(L - m, 5)
          End If
```

```
            ULNI(m, 12) = m

            For z = ULNI(g, 8) To ULNI(g, 7)
               If ULNI(z, 7) = ULNI(g, 7) Then
                  ULNI(z, 12) = m
               End If
            Next z

             ULNI(g, 9) = ULNI(g, 9) + ULNI(m, 0)


            For y = FirstULN To LastULN
               If ULNI(y, 12) = m Then
                  ULNI(y, 5) = ULNI(g, 5)
                  ULNI(y, 6) = ULNI(g, 6)
                  ULNI(y, 7) = LastULN
                  ULNI(y, 8) = FirstULN
                  ULNI(y, 9) = ULNI(g, 9)
                  ULNI(y, 10) = 1
                  ULNI(y, 11) = ULNI(FirstULN, 1)
                  ULNI(y, 12) = 9
               End If
            Next y

           g = L
         End If

      End If
    Next g
   End If
Next m

For b = 0 To L
If ULNI(b, 9) >= 100 Then
   v(8) = v(8) + ULNI(b, 0)
End If
Next b


'output ULNs with their cluster data
t = 1
For b = 0 To L
If ULNI(b, 0) <> 0 Then
   Worksheets("ULN").Range("A1").Offset(t, 15 * o + 0) = b
```

```
    Worksheets("ULN").Range("A1").Offset(t, 15 * o + 1) = ULNL(b, 0)
    Worksheets("ULN").Range("A1").Offset(t, 15 * o + 2) = ULNL(b, 1)
    Worksheets("ULN").Range("A1").Offset(t, 15 * o + 3) = ULNI(b, 0)
    Worksheets("ULN").Range("A1").Offset(t, 15 * o + 4) = ULNI(b, 1)
    Worksheets("ULN").Range("A1").Offset(t, 15 * o + 5) = ULNI(b, 2)
    Worksheets("ULN").Range("A1").Offset(t, 15 * o + 6) = ULNI(b, 3)
    Worksheets("ULN").Range("A1").Offset(t, 15 * o + 7) = ULNI(b, 4)
    Worksheets("ULN").Range("A1").Offset(t, 15 * o + 8) = ULNI(b, 5)
    Worksheets("ULN").Range("A1").Offset(t, 15 * o + 9) = ULNI(b, 6)
    Worksheets("ULN").Range("A1").Offset(t, 15 * o + 10) = ULNI(b, 7)
    Worksheets("ULN").Range("A1").Offset(t, 15 * o + 11) = ULNI(b, 8)
    Worksheets("ULN").Range("A1").Offset(t, 15 * o + 12) = ULNI(b, 0)
    Worksheets("ULN").Range("A1").Offset(t, 15 * o + 13) = ULNI(b, 10)
    Worksheets("ULN").Range("A1").Offset(t, 15 * o + 14) = ULNI(b, 11)
    Worksheets("ULN").Range("A1").Offset(t, 15 * o + 15) = ULNI(b, 12)
    t = t + 1
End If
Next b

For k = 0 To 8
    Worksheets("Results").Range("A1").Offset(k, o) = v(k)
Next k

'Next o

End Sub
```

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From – To)*
23-03-2006	**Master's Thesis**	Sep 2005 - Mar 2006

4. TITLE AND SUBTITLE
A MULTI-PASS CONSTRUCTION HEURISTIC FOR THE AGGREGATED AIRLIFT PROBLEM

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
O'Leary, Stephen T., Captain, USAF

5d. PROJECT NUMBER
ENS2006-15

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)
Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way, Building 642
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GOR/ENS/06-15

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AEFC/CC
Attn: BGen Stephen L. Hoog 575-2277 DSN: 785-4539
205 Thornell Ave;
Langley AFB, Va. 23665 e-mail: Stephen.hoog@langley.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)
AEFC/AOE

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

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

13. SUPPLEMENTARY NOTES
N/A

14. ABSTRACT

To support the capabilities for precision strike, close air support, intelligence, surveillance, reconnaissance, air mobility and air refueling the Air Force has over 26,000 Airmen deployed from over 80 military installations to Forward Operating Locations (FOLs) in over 21 countries. This research creates a multi-pass construction heuristic to solve the aggregated airlift problem associated with the transportation of these 26,000 deployed Airmen. We are given a Time Phased Force Deployment Document (TPFDD) containing a list of personnel scheduled for deployment, their base of origin, Point of Debarkation (POD), FOL, and Required Delivery Date (RDD). The goal is to aggregate the personnel into groups of 100 or more based on region of origin, POD and RDD to generate dedicated airlift for their deployment. The goal of aggregated airlift is to generate dedicated airlift facilitating better In Transit Visibility of personnel moving though the PODs. This, in turn, enables the proper planning and scheduling of intratheater airlift from the POD to the FOL reducing the backlog of personnel and wait times at the POD. The algorithm was tested on several notional and real world TPFDDs and performed well, increasing the effectiveness over the current manual process by more than 20%.

15. SUBJECT TERMS
Transportation, Heuristics, Scheduling

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Gary W. Kinney, USAF (ENS)
U	U	U	UU	87	19b. TELEPHONE NUMBER *(Include area code)* (937) 255-3636, ext 4601; e-mail: Gary.Kinney@afit.edu

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