2-1997

Optimizing Airborne Area Surveillance Asset Placement

Douglas E. Fuller

OPTIMIZING AIRBORNE AREA SURVEILLANCE ASSET PLACEMENT

THESIS

Douglas E. Fuller, Major

AFIT/ENS/GOA/97M - 05

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/ENS/GOA/97M

OPTIMIZING AIRBORNE AREA SURVEILLANCE ASSET PLACEMENT

THESIS

Douglas E. Fuller, Major

AFIT/ENS/GOA/97M - 05

AFIT/ENS/GOA/97M

OPTIMIZING AIRBORNE AREA SURVEILLANCE ASSET PLACEMENT

THESIS

Presented to the Faculty of the Graduate School of

Engineering

Air Education and Training Command

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Operations Analysis

Douglas E. Fuller, B.S., M.S.

Major, USAF

March 1997

Approved for public release, distribution unlimited

THESIS APPROVAL

Student: Douglas E. Fuller, Major, USAF

Class: GOA-97M

Title: Optimizing Airborne Area Surveillance Asset Placement

Defense Date: 18 February 1997

Committee:	Name/Title/Department	Signature
Advisor	James T. Moore, Lt. Col., USAF Associate Professor Department of Operational Sciences	James T. Moore
Reader	Richard F. Deckro Professor Department of Operational Sciences	
Reader	James W. Chrissis Associate Professor Department of Operational Sciences	

Acknowledgments

I am indebted to my thesis advisors, Lieutenant Colonel James T. Moore, Dr. Richard F. Deckro, and Dr. James W. Chrissis. Their insight and guidance during this research effort were invaluable. They provided guidance and motivation without limiting my research into this topic. The entire project would not have been possible without the expert staff of the AFIT library. Their tireless efforts in finding the obscure references and journals I needed was invaluable.

Finally, I would like to thank my wife, Julia, and my two children, Colin and Meredith, who provided unwavering support and patience during the thesis process. Without their encouragement this thesis would not have been accomplished.

Douglas E. Fuller

Table of Contents

page

page

List of Figures

List of Tables

AFIT/ENS/GOA/97M

Abstract

Currently there is no automated planning tool for the optimum positioning of USAF area surveillance assets for a theater-level campaign. Selection of orbit points is currently done by hand with little regard for optimum placement. This research seeks to find the optimum placement of the very limited USAF airborne surveillance assets against a theater level target set. Analysis of the number of aircraft required to cover a theater-level target set would provide commanders with information on the allocation of these critical assets.

The problem of finding the optimum points can be modeled as a classic maximal covering location problem (MCLP). Additional constraints on the placement of surveillance aircraft can be handled by preprocessing the potential orbit points to eliminate infeasible orbit points. Heavy emphasis is placed on preprocessing the data to reduce the problem size and hence solution time. The aggregation of both the potential orbit points and targets was accomplished without loss of locational information. An existing heuristic was used to find a solution in a very short time.

The heuristic finds the optimum orbit points for the available aircraft up to the point where total coverage occurs or it becomes impossible to cover any additional targets. Allocation decisions for these assets can then be accomplished.

OPTIMIZING AIRBORNE AREA SURVEILLANCE ASSET PLACEMENT

INTRODUCTION

1.1 General Background

Currently there is no automated planning tool for the optimal or near-optimal positioning of United States Air Force (USAF) area surveillance assets for a theater-level campaign. These assets include the Boeing E-3 Airborne Warning And Control System (AWACS) and Northrop E-8 Joint Surveillance and Target Attack Radar System (Joint STARS) aircraft. There is a very limited number of both of these aircraft in the USAF fleet; optimum use of these assets is critical. Selection of orbit points for these aircraft is currently done manually with little regard for ideal placement. Analysis of how many orbit points, and thus how many aircraft, required to maximally cover a theater-level target set is not currently accomplished. The focus of this research is to determine the optimal or near-optimal placement of airborne area surveillance orbit points to provide maximum target coverage with a minimal number of aircraft.

1.2 E-3 Aircraft Uses

The E-3 AWACS was developed by Boeing to fill the need for an airborne radar. Airborne radar provides the USAF greater threat warning of enemy aircraft, while affording USAF commanders enhanced battlefield awareness by increasing radar coverage over the battlefield. The E-3 is currently used by the USAF, North Atlantic Treaty Organization (NATO), and the Royal Saudi Air Force. Typically E-3 orbits are continuously occupied for months or years at a time. NATO AWACS aircraft have provided continuous coverage of the current Balkan crisis since 1992. USAF and Saudi Arabian AWACS have maintained an almost continuous orbit over Saudi Arabia since 1981. During the 1991 Gulf War there were times when eight orbits in the theater were continuously occupied by AWACS. This greatly strained the AWACS fleet, as aircraft had to be available to meet other crises that might develop (in particular a possible Korean conflict) and to meet continuous training, maintenance, and exercise support requirements. From the author's experience in NATO AWACS, it takes 3-4 aircraft to continuously occupy an orbit point.

The E-3 aircraft uses an onboard radar to scan for both airborne and maritime targets. Radar data is processed to give a real-time air picture of all friendly and enemy aircraft within the radar's range capabilities. Flying at altitudes of approximately 30,000 feet allows the AWACS radar to avoid blocking terrain and to see deep into enemy controlled territory. This knowledge of an enemy aircraft's position provides an invaluable threat warning to friendly forces, and it allows AWACS controllers to direct friendly aircraft to eliminate any airborne threat to allied aircraft. In addition, the radar picture can be sent to all friendly forces providing them with a real-time air picture without turning on their own radar and thus revealing their positions. Finally, the AWACS can easily move along with friendly ground forces as they advance into enemy territory, providing a real-time radar picture of the air threat to ground forces.

1.3 E-8 Aircraft Uses

The E-8 Joint STARS aircraft was developed by the USAF and US Army to undertake the ground surveillance, targeting, and battle management missions [20:137]. This revolutionary aircraft was first used in the Gulf War during Operation Desert Storm. In that conflict, the test-bed aircraft provided a real-time ground picture of the locations of both friendly and enemy ground forces. This ability to accurately know the locations of both friendly and enemy forces in real-time and to observe as they maneuvered was a first in military history. Friendly ground forces can be continuously updated on enemy troop movements. The possibility of an enemy surprise ground attack, at least on a large scale, has been virtually eliminated for US and allied forces covered by Joint STARS aircraft. Since the Gulf War, the first two production Joint STARS aircraft have deployed to Germany (in December 1995) to monitor NATO led peacekeeping operations in the Balkans [20:137]. Only 20 of these highly valuable (and expensive) aircraft are scheduled to be built. The effective use of this limited asset is critical.

1.4 Research Objectives

The objective of this research is to develop a model to effectively assign area surveillance assets to orbit points which maximally cover a selected target base. The overall goal of this study is to obtain solutions in under 30 minutes. Results should also include any alternative sites providing equal coverage. In a given scenario, non-optimal coverage can lead to a greater number of aircraft required to provide the same amount of target coverage as a few well-placed aircraft.

1.5 Limitations and Assumptions

The problem of finding orbit points for the area surveillance aircraft can be modeled as the classic maximal covering location problem (MCLP) first discussed by Church and ReVelle [4:101]. Solving this NP-complete problem for an optimal solution within the stated time limit of 30 minutes may prove impossible; thus a heuristic was used to find a solution (albeit with no guarantee of optimality). Prior to the execution of the solution procedure, every attempt was made to reduce the number of constraints and variables in the model, while maintaining the fidelity of the model.

Only one type of aircraft can be modeled at a time. The E-3 and E-8 are used to cover similar, yet different, target bases. In addition, the value assigned to coverage of certain targets by each aircraft type would be different. For example, an E-3 would assign greater value to covering enemy airfields rather than tanks; the opposite would, in general, hold for the E-8. Finally, only unclassified target sets and target values have been used in this study.

Several assumptions have been made to bring the scope of the problem to an executable but realistic level. These assumptions should not affect the quality of the solutions provided, but they make the problem solvable in a reasonable amount of time. These assumptions are:

1. No route planning. The surveillance aircraft can reach the assigned orbit point by some route. The exact route is left to the aircrews.

2. Orbit points are not selected if they are within the lethal range of known enemy surface-to-air missile (SAM) sites or if they are within a selected radius of enemy airfields.

This is a realistic constraint which avoids needlessly risking the destruction of these high-value aircraft.

3. Terrain effects are ignored. For the E-3 aircraft this is a realistic assumption since the E-3 is chiefly used to identify flying targets. The E-8 flying over rough country may experience degraded capabilities. In an operational setting, this may not be a realistic assumption. This is a potential area for further research.

4. Orbit points are fixed. The research looks at a 'snapshot' of the target base, preferably at the start of the campaign, prior to the destruction of many of the targets. As the target environment changes, the model may be re-run with the new target set and constraints updated as necessary.

1.6 Conclusion

Chapter 2 provides a background of previous work related to this topic. Chapter 3 covers the methodology used in solving the problem. A small example is included for illustration. Chapter 4 provides the results for two large scenarios in order to demonstrate the speed and accuracy of the model. Chapter 5 discusses possible extensions of the work and provides a brief conclusion.

RESEARCH BACKGROUND

2.1 Introduction to the Covering Problem

The covering problem entails locating a set of supply points that 'cover' a given set of demand points. A supply point covers a demand point if the demand point is within a given metric (usually distance or time) of the supply point. A single supply point can cover any number of demand points. The objective is to cover all the demand points with a minimum number of supply points. This chapter discusses the two versions of the covering problem, generation of candidate orbit points, and data aggregation.

2.2 MCLP and SCP

Two versions of covering problems are the set covering problem (SCP) and the maximal covering location problem (MCLP). The latter was introduced by Church and ReVelle in 1974 [4:101]. Extensive literature exists on both problems. A taxonomy compiled by Schilling, et al in 1993 [17:25-55] provides an excellent source of recent material on both types of covering problems. The SCP involves finding the minimum number of facility sites required to cover a given set of demand points. The covering constraints are usually based on some easily determined metric such as distance or time-of-travel. In mathematical form, the SCP is:

$$\text{Minimize } \sum_{j \in J} c_j \cdot x_j \qquad (1)$$

Subject To:

$$\sum_{j \in N_i} x_j \geq 1, \ldots \forall i \in I \qquad (2)$$

$$x_j \in \{0,1\} \quad \forall j \in J \qquad (3$$

where m = number of demand points

n = number of possible facility location sites

I = set of demand points

J = set of candidate facility location sites

S = maximum covering distance

d_{ij} = the distance (or some other metric) from each demand point i to each possible facility location point j

c_j = the cost of using site j, for $j = 1, \ldots, n$

$x_j = 1$, if facility site at location j is occupied, 0 otherwise

$N_i = \{j \mid d_{ij} \leq S\}$ for $i = 1, \ldots, m$. The set of possible facility location sites which cover demand point j

Constraint (2) forces the coverage of all the demand points without regard for the number of facilities required. The limited nature of most budgets can make covering all customers impractical. The MCLP attempts to address this problem by locating a limited number of facilities to cover the maximum number of, but not necessarily all, demand points. The MCLP is formulated mathematically as:

$$\text{Maximize} \quad z = \sum_{i \in I} a_i \cdot y_i \tag{4}$$

Subject To:

$$\sum_{j \in N_i} x_j \geq y_i, \ldots \forall i \in I \tag{5}$$

$$\sum_{j \in J} x_j \leq P \tag{6}$$

$$x_j \in \{0,1\} \quad \forall\, j \in J \tag{7}$$

$$y_i \in \{0,1\} \quad \forall\, i \in I \tag{8}$$

where $N_i = \{j \in J \mid d_{ij} \leq S\}$ for $\forall\, i \in I$

$x_j = 1$, if site at location j is occupied, 0 otherwise

$y_i = 1$, if the demand point at i is covered, 0 otherwise

a_i = the value of covering demand point i, for $i = 1,\ldots,$ m

P = the number of facility site locations that can be occupied

All other parameters defined as in the SCP above.

If the a_i are all equal to a value of 1, the problem finds the maximal cover given the number of facilities. Weighting the a_i parameter provides a solution which maximizes the value of the covered demand points. If all the demand points are covered by the given number of facilities, the problem is equivalent to the SCP.

Unfortunately, both problems are NP-hard [17:27]. Thus every effort must be made to reduce the size of the problem so as to reduce solution times. Heuristics, which quickly solve large realizations of the SCP and MCLP but provide no guarantee of optimality, can be used to provide effective solutions to these problems. There are many heuristics designed to solve the SCP [17:28]. There are fewer heuristics which solve the MCLP; however MCLP heuristics also solve the SCP. The placement of the airborne surveillance assets may be modeled as a MCLP since it is unlikely that total target coverage can be attained with the limited assets available. In addition, it is highly likely a commander may have various values for covering different targets. The selection of aircraft orbit points is formulated as a MCLP and then solved using a heuristic.

2.3 Generation of Candidate Orbit Points (COPs)

The generation of candidate orbit points (COPs) for aircraft placement has not received much attention. In most literature, candidate site locations were already taken as a given, based on owned land, zoning restrictions, or previously located sites (for example, existing warehouses or hospitals) [13; 16; 17]. While there has been some discussion concerning the optimum placement of one site on a continuous plane, given a set of demand points, but the multiple-site model has not been fully developed [3:25]. Thus, a method of developing COPs was needed. Two methods were explored. The first of these was the circle method discussed by Mehrez and Stulman in 1982 and extended by them in 1984 [15; 14]. The other method consists of laying a square grid over the area of interest and using the corners of each grid square as COPs.

2.3.1 Circle Method

Mehrez and Stulman developed a method to generate a finite candidate solution set on an infinite plane. They postulated that the optimal solution to the MCLP must exist on the set of all intersection points of circles drawn a radius R (the maximum distance a candidate site can be from a demand point and still cover it) around each the of demand points [14:20]. This approach generates a set of candidate sites to use in the solution of the MCLP. They noted that this solution set often places the candidate sites at the farthest possible locations from the demand points. From an airborne surveillance viewpoint, this is a good result since the ‘demand’ points are usually hostile. Unfortunately, the maximum number of intersection points generated by this method is $2\binom{m}{2}$ [14:22], where m is the number of demand points. For a theater-level target base of 1000 targets, this would mean nearly one million possible intersection points! This method was not used for computational reasons but might prove easy to implement on smaller problems.

2.3.2 Grid Method

The grid method consists of laying a grid over all the demand points. The size of the grid is determined by the size of the geographic region to be covered. The spacing between grid points is another factor which must be considered in developing the model. This spacing should be set to a distance less than the operational orbit radius of the surveillance aircraft to avoid missing any good sites. This method is used widely in the literature of covering problems [17]. It was also used by Ignizio in 1971 [12: 91] to solve a large problem involving the location of radar sites.

2.4 Data Aggregation

In order to solve the airborne surveillance problem in a reasonable amount of time, some aggregation of the demand points and the COPs is necessary. Errors due to aggregation are discussed at length in the literature and the reader is directed to three excellent papers on the subject. In 1978 Hillsman and Rhoda describe three sources of error resulting from demand aggregation in the *p*-median problem [11]. Current and Schilling extended this work a step further by applying it to both the SCP and MCLP problems [5; 6]. The former work develops three rules for aggregation of demand points. These rules were developed to minimize aggregation error.

Aggregation error in the MCLP has two sources: type A and type B (Current and Schilling [6:96]). The type A error has two cases. In case 1 a demand point is considered covered when in fact it is not and in case 2 a demand point is considered not covered when in fact it is covered. The first case occurs when an uncovered demand point is aggregated to a point closer to a COP and thus becomes 'covered' at the aggregated demand point. The second case occurs when a covered demand point is aggregated to a demand point farther away from a COP and thus becomes 'uncovered'. Both cases lead to errors in actual coverage for each COP and can lead to infeasible unaggregated solutions and significant optimality errors [5:121]. Type B errors occur when aggregation is accomplished at COPs by aggregating demand points that are outside the cover range of a COP to that COP location. This is similar to type A case 1 error with the same associated difficulties.

Current and Schilling propose three aggregation rules to reduce or eliminate type A and B errors. They are:

1. Only aggregate demand points at current demand point locations.
2. Do not aggregate demand point k to aggregated demand point j if the distance from k to j is greater than the covering distance.
3. Only aggregate demand at a demand point to an aggregated demand point if the set of COPs that cover both demand points are identical.

The first two rules eliminate type B error and type A case 2 errors. Unfortunately, type A case 1 errors can still occur. Rule 3 guarantees that any COP selected which covers an aggregated demand point also covers all the demand points aggregated at that point. Thus rule 3 retains all of the locational information present in the original network and is suitable for covering models with a single maximal covering distance [5:123].

The first two rules can lead to covering errors and/or optimality errors due to aggregation. These two rules can also lead to a solution that is not optimal for the unaggregated model [5:122]. The third rule produces an aggregation pattern that does not cause errors due to aggregation. Use of only rule 3 leads to less data aggregation than using other aggregation rules [5:123]. Rules 1 and 3 are implemented in this work to ensure no loss of locational information.

2.5 Algorithms and Heuristics

The primary algorithm used today to solve large mixed integer programs (MIPs) is the simplex algorithm with branch-and-bound. There are many commercially-available linear solvers. For a large zero/one program no IP solver can guarantee the solution of a large MIPs in a short amount of time. For this reason a heuristic was examined for use in this research. There are a number of heuristics investigated for use in this research to solve the covering problem [7; 12; 13; 16]. Ignizio presented a rather elegant and simple heuristic in his 1971 dissertation [12]. The heuristic was published in Francis and White in 1974 [7:447]. Ignizio's heuristic was specifically developed for the MCLP and has been used in selecting sites for ground radar [12:91]. His research on the heuristic obtained the optimum answer in 85% of all tested problems. This rate increased to 95% on MCLPs. His solution times, for small problems on 1971-era computers, were measured in seconds. The one large-scale problem solved, with 25,521 candidate locations, only took 13.53 minutes to terminate using FORTRAN IV for the Univac 1108 [12:90]. This heuristic was chosen for this research because of its ease of implementation, short solution times, and the high levels of coverage achieved. The heuristic is explained in detail in section 3.9.

METHODOLOGY

3.1 Introduction

This chapter covers the methodology used in solving the problem of placement of airborne surveillance assets to maximally cover a theater-level target base. The full mathematical model is discussed first, followed by a discussion of the parameter values and data generation techniques. Implementation of the preprocessing constraints is then reviewed. Subsequently, the method of aggregation of the COPs and targets is covered. Finally, the presentation of the heuristic is given followed by a small example.

3.2 Problem Description

The optimum placement of airborne surveillance assets can be modeled as a modified MCLP with a number of additional considerations and constraints. The airborne assets are aircraft which must be constantly moving. The normal operational orbit radius for E-3 and E-8 aircraft is 15 nautical miles (NM). Thus, the model must take into consideration each aircraft's ability to cover the targets from both sides of its orbit. The surveillance aircraft cannot be assigned orbit points within the lethal radius of enemy SAM sites. Both E-3 and E-8 aircraft have no inherent defensive capability against hostile attack and must avoid these situations. Surveillance aircraft cannot be placed too far from friendly fighter support, measured as the distance from the fighter aircraft's home field to the orbit point, or the surveillance aircraft becomes vulnerable to enemy fighter aircraft. The surveillance aircraft cannot be stationed too far from their home airfield, or transit time to the orbit point and back reduces the on-station time of the aircraft due to crew duty limitations, thus limiting the number of surveillance orbits that can be occupied.

These constraints and the form of the MCLP lead to the development of a mathematical model to describe this problem.

3.3 The Mathematical Model

The mathematical model can be described as an MCLP with additional constraints. The parameters, constants, and variable definitions are below, followed by the mathematical equations. Mathematical Formulation:

Parameters:

i = Number of the surveillance point.

j = Number of the target point

k = Number of the SAM site

m = Number of the friendly fighter base

n = Number of the home base of the surveillance aircraft

$COPT_j$ = The set of orbit points that cover target j

Constants:

Y_{ik} = Distance from surveillance point i to SAM site k

LR_k = Lethal radius of the SAM at site k

V_{im} = Distance from surveillance point i to base m

$V_i = \min V_{im} \; \forall \; i, m$

$DCAR_m$ = Range limit from friendly fighter base m

W_{in} = Distance from surveillance point i to home base n

$W_i = \min W_{in} \; \forall \; i, n$

ACR_n = Range limit from home base n

X_{ip} = Distance from surveillance point i to No-Fly Point p

NFP_p = No-Fly range of No-Fly Point p

ASP = Number of surveillance aircraft available

P_j = Target j value

Variables:

t_j = Is the target j covered by at least one occupied orbit point? 0 = No, 1 = Yes

cop_i = Is the orbit point i occupied by an aircraft? 0 = No, 1 = Yes

Mathematical equations:

$$\text{MAX} \sum_j P_j \cdot t_j \qquad \text{Objective Function} \tag{9}$$

Subject To:

$$\sum_i cop_i \leq ASP \qquad \text{Number of SPs available} \tag{10}$$

$$t_j \leq \sum_{i \in COPT_j} cop_i , \ \forall j \qquad \text{Covers the targets} \tag{11}$$

$$W_i \cdot cop_i \leq ACR_n , \ \forall i, n \qquad \text{Range limit from home base} \tag{12}$$

$$V_i \cdot cop_i \leq DCAR_m , \ \forall i, m \qquad \text{Range limit from friendly fighter base} \tag{13}$$

$$cop_i \leq \frac{Y_{ik}}{LR_k}, \ \forall i, k \qquad \text{Avoids lethal SAM range} \tag{14}$$

$$cop_i \leq \frac{X_{ip}}{NFP_p}, \ \forall i, p \qquad \text{Avoids No-Fly Zones} \tag{15}$$

where $cop_i \in \{0,1\}$ for all i, $t_j \in \{0,1\}$ for all j

The constraint set above is not the MCLP constraints. Constarints (12) - (15) model operational considerations in the employment of airborne surveillance assets. These constraints can be used to preprocess the COPs. Then constraints (12) - (15) can be removed, leaving the MCLP equations (9) - (11).

3.4 Data Generation

Due to the classified nature of the specific data requirements of the model, two notional scenarios have been developed and utilized. Data structures have, however, been developed such that actual data can be easily inserted in the model. Thus, all data used (target lists, ranges, etc.) was created by the researcher to mimic actual conditions but *are not* correct for the weapons and equipment modeled. Actual classified target data are obtainable from the respective theater air planning staffs.

3.4.1 Notional Database

The target database for an Iraq scenario was generated by using Jet Navigation Charts (JNC) of Iraq and selecting all the airfields shown. In addition, a selection of non-airfield targets was added to represent the ground forces and other targets that might be attacked by US-led forces. In all, 205 targets were selected for the Iraq scenario. The Far Eastern scenario targets were created by selecting potential targets from maps of the region from North Korea to Hong Kong. A total of 133 targets were selected. A notional value was also assigned to each target. Each target was identified by target number, target type, target value, latitude, longitude, name, and country location. Appendix C presents the complete target list for the Iraq scenario, and Appendix E presents the complete target list for the Far Eastern scenario.

Determining the value assigned to the coverage of a target is a separate area of research. It is expected that these values, which are critical to the solution of the problem, would be explicitly or implicitly determined by the theater commander. If the coverage value of all the targets are identical, then solution of the model finds the maximum *number* of targets covered since all targets have equal value. Otherwise solution of the model finds the maximum target *value* covered. Notional target values ranged from 50 to 1 (with 50 assigned to the most valuable targets and 1 assigned to the least valuable) in the scenarios. The model can handle any positive numerical value for the parameter P_j in Equation (9).

In the model, coverage values are additive. If some target class, for example all airfields, must have priority over other targets, setting the target coverage for that class of targets (or even a particular target) to a value greater than the sum of the remaining target values would cause the model to attempt to cover all the high-priority targets first. The model can be used in this way to solve pre-emptive goal-programming like MCLPs. There is no capability for the model to manage non-additive coverage values. For example, if the individual coverage values for target A and target B were 5 and 6, respectively, but the value of covering both was not 11 (=5+6), the model currently would not accurately represent this non-additive situation. This is a potential area for further research, provided the appropriate relations are designed and can be developed.

Parameters used in the model are summarized in Table 1. These parameters can be edited easily, using the FORTRAN program shown in Appendix B, to include either actual operational data or other experimental scenarios. The cover range is defined as the maximum distance that a COP can be from a target and still cover that target. This distance should be set at the surveillance asset's effective radar range minus the radius of the surveillance orbit. For example, if the effective range of the E-3 was 190 NM and the orbit radius was 15 NM, then the cover range should be set at 175 NM. This guarantees coverage at the far side of the surveillance asset's orbit.

Table 1: Parameter Values

PARAMETER	VALUE
Cover Range of Surveillance Aircraft	175 NM
Surveillance Aircraft Available	11
Lethal SAM ranges (varies by SAM type)	30 NM to 120 NM
Range limit from friendly fighter bases (varies by base)	300 NM to 350 NM
Range limit from surveillance aircraft home base (varies by base)	500 NM to 550 NM
Size of Solution Grid - Iraq Scenario	15° by 15°
Size of Solution Grid - Far Eastern Scenario	21° by 21°
Step size for generating COPs in Latitude	0.20 degrees
Step size for generating COPs in Longitude	0.20 degrees

3.4.2 Generation of 3D Solution Grid

The target list is not used directly to find distances between points. Each target's latitude and longitude is converted to an earth-centered coordinate system. The axis of this coordinate system is referenced as the **IJK** system zeroed at the earth's center as depicted in Figure 1. The vector **r** is the vector from the earth's center to the target.

The I axis runs from the earth's center to the intersection of the equator and the 0° longitude. The J axis forms a 90° angle with the I axis and is also in the plain of the equator. The K axis runs vertically through the North Pole. This forms the geocentric-equatorial coordinate system used (2:93-99). Each target's coordinates are converted to vectors in the **IJK** coordinate system through two Euler coordinate transformations. All the coordinates for SAM sites, bases, and no-fly areas are also converted to vectors in the **IJK** coordinate system. These vectors are then used to find the Euclidean distance between any two points. Distortion due to the curvature of the Earth is not a consideration because of the short length of the covering radius.

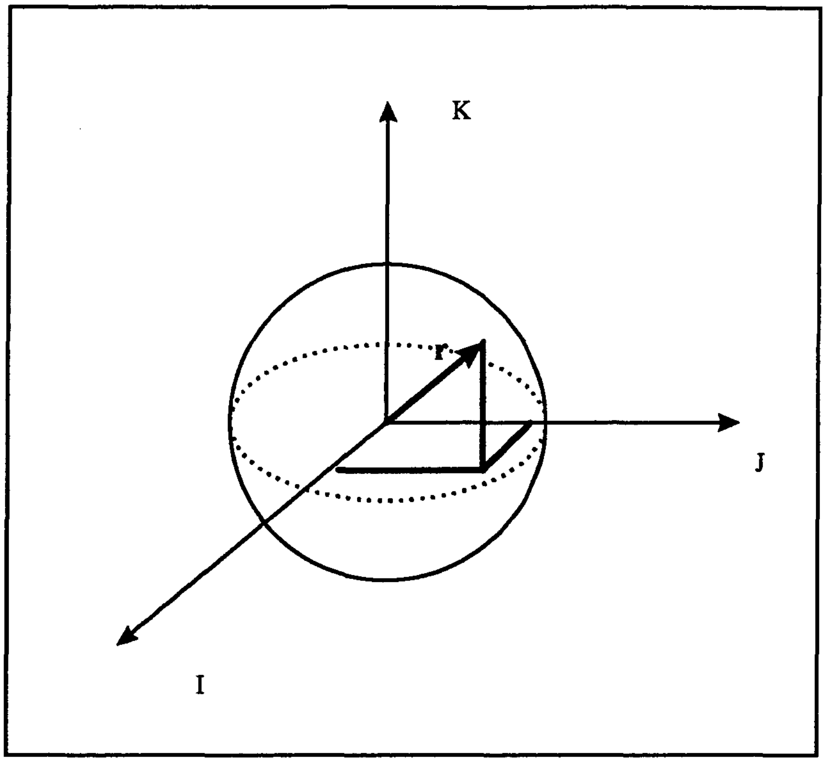

Figure 1: ***IJK*** coordinate system [2:99]

3.5 Generation of Candidate Orbit Points (COPs)

The generation of a good set of fixed COPs requires development of sufficent points to accurately portray the ability of surveillance aircraft to occupy any point in space but not so many that solution times prove unmanageable. A grid of COPs roughly 15 NM apart was chosen to reflect the surveillance aircraft orbit radius of 15 NM. The grid was centered on the unweighted center of all the targets. This choice was somewhat arbitrary but works well in practice. For the Iraq scenario, the size of the grid is 15° on each side which is roughly 900 NM per side. If a larger grid size is desired, these parameters can be easily changed by editing the parameter list using the FORTRAN program (in Appendix

B). Once all the COP latitude and longitude coordinates are generated, each point is transformed into the **IJK** coordinate system.

3.6 Elimination of COPs

Preprocessing of the COPs to eliminate non-viable points prior to solution is critical to reducing solution times. Each COP is associated with a binary decision variable in the mathematical model. The grid size chosen produces 5776 COPs (Iraq) and 11,236 COPs (Far Eastern) for each respective scenario. Finer grids (which increase the number of COPs per degree) or a larger grid size (greater actual area covered) would produce more COPs. Elimination of COPs by imposing the requirements of avoiding SAMs, being close to both the home airfield and friendly fighter airfields, and avoiding flight into no-fly areas can significantly reduce the number of COPs thus, in general, shortening solution times. COP elimination is accomplished by checking the distance from each home airfield, friendly fighter airfield, SAM site, and no-fly point (respectively), to each COP, against each of the range restrictions. If the distance does not meet the restrictions imposed by Equations (12) - (15), respectively, then the COP is eliminated.

3.6.1 Home Airfield Range Restrictions

COPs that are too far from any surveillance aircraft's home bases can be eliminated. The distance from each COP to each home base is checked. If the COP is not within the range restriction of any of the home bases, it is removed. Mathematically, this constraint is modeled by Equation (12).

3.6.2 Fighter Cover Restrictions

Friendly fighter cover is a necessity for the survival of airborne surveillance assets in modern air warfare. Fighters have very short ranges and crew fatigue for these single manned aircraft is a definite consideration. Thus, the COPs must be relatively close to friendly fighter aircraft bases. The distance from each COP to each friendly fighter base is checked. If a COP is not within the range restriction of any friendly fighter base, it is removed. This constraint is modeled by Equation (13).

3.6.3 SAM Restrictions

COPs within the lethal range of any SAM site cannot be selected as a point for placing a surveillance aircraft. Thus, these COPs can be eliminated from the MCLP prior to solution. The distance from each COP to each SAM site is checked. When modeling the lethal ranges, the orbit radius of the surveillance aircraft must be added to each lethal range to prevent the surveillance aircraft orbit from being within the lethal range. If the distance is less than the lethal SAM range for that SAM site, the COP is removed from further consideration.

Proximity to enemy fighter bases is also a consideration. Surveillance aircraft too close to enemy fighter bases risk attack before friendly aircraft can deal with the threat. Enemy fighter bases can be modeled as SAMs sites by limiting how close the COPs can be to these bases. Mathematically this constraint is modeled by Equation (14).

3.6.4 Political Boundaries and No-Fly-Zones

Political boundaries and designated no-fly-zones are addressed by the addition of further restrictions on COPs. For example, in the Iraq scenario, surveillance aircraft cannot be assigned orbits inside Iran. In addition, designated free-fire zones (areas where all aircraft are treated as hostile and engaged) must be avoided. These restrictions are modeled by generating points with ranges and excluding those COPs within the designated ranges. This is identical to how the SAM restrictions are treated. Careful modeling of borders can prevent selection of COPs inside these no-fly zones. In the Iraq scenario, the

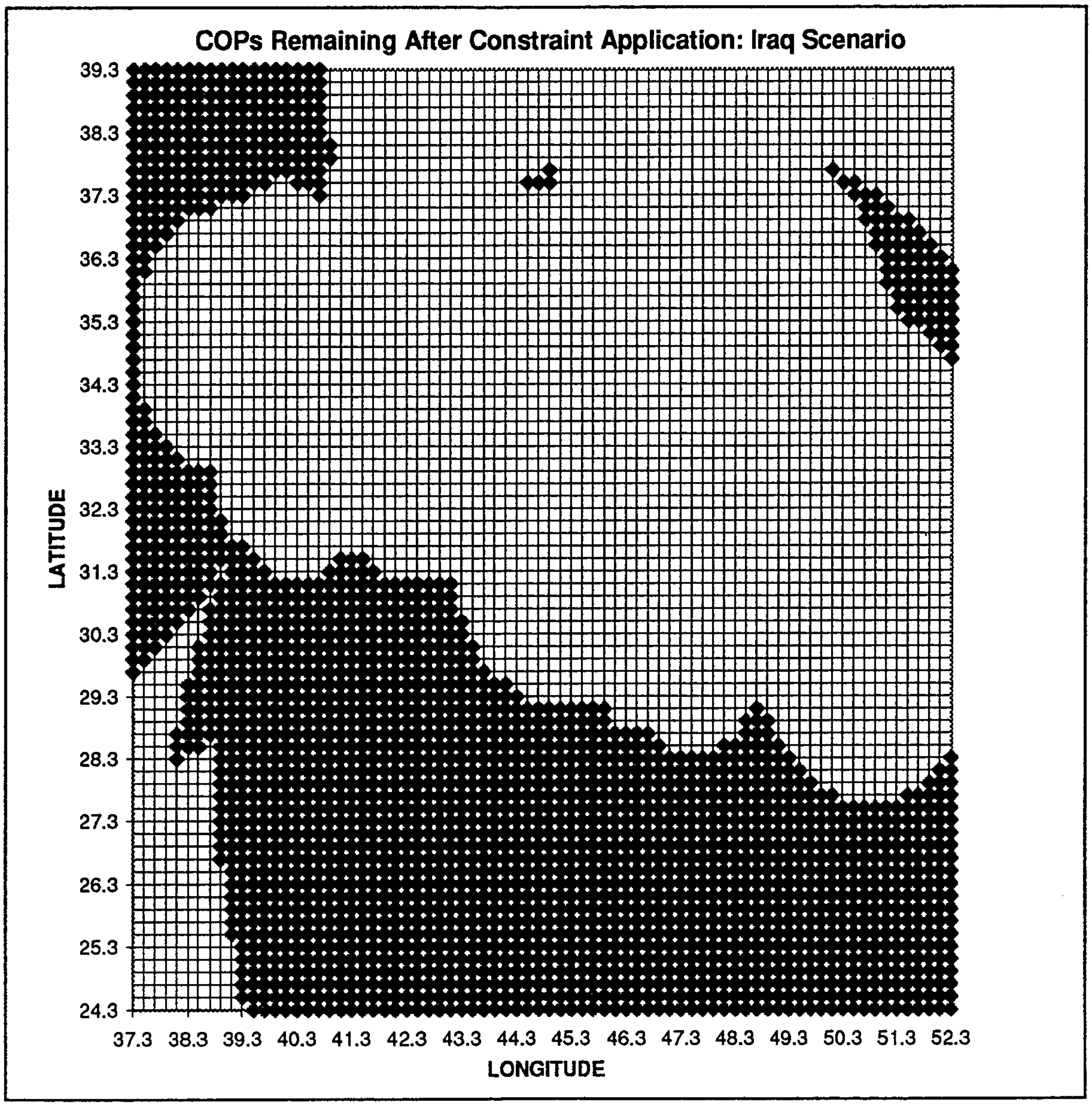

Figure 2: Iraq Scenario; COPs available prior to aggregation

Iranian, Jordanian, and Syrian airspace had to be excluded. In the Far Eastern scenario, the Chinese airspace had to be excluded. This constraint is modeled by Equation (15).

After the elimination of COPs, the number of variables is reduced substantially. For example, in the Iraq scenario the application of the operational constraints reduced the number of COPs from 5776 to 2245; a 61% reduction in the number of integer variables. Figure 2 shows graphically the reduction for this scenario. In Figure 2, every intersection represents a possible COP location at the start of the problem. The dark points are the COPs remaining after pre-processing and removing from the candidate list any points that do not satisfy constraints (12) through (15).

3.6.5 COP Elimination and Aggregation

Aggregation of COPs is undertaken to simplify the problem without loss of locational information. First, all COPs that do not cover any targets are eliminated from the model. This is equivalent to the elimination of all zero columns from the **A** matrix in the MCLP. Next, all COPs with identical coverage of targets are aggregated into one COP (the first encountered by the program). This aggregation scheme is identical to the one used for aggregating targets, and is discussed in Section 2.3 (see Current and Schilling [5]). All the COPs aggregated in this manner represent potential alternate optimal locations if the aggregated COP is selected as an orbit point. For example, if COP numbers 34, 45, 67, 68, and 127 have identical target coverage, the model keeps COP 34 and eliminates the other four COPs from the model. If COP 34 is selected as an orbit point, the other four COPs would represent alternate sites that give the same target coverage and could be selected by the decision maker without loss of overall coverage. Each COP aggregated in this manner reduces by one the number of variables in the integer program and, consequently, reduces the solution time of the problem.

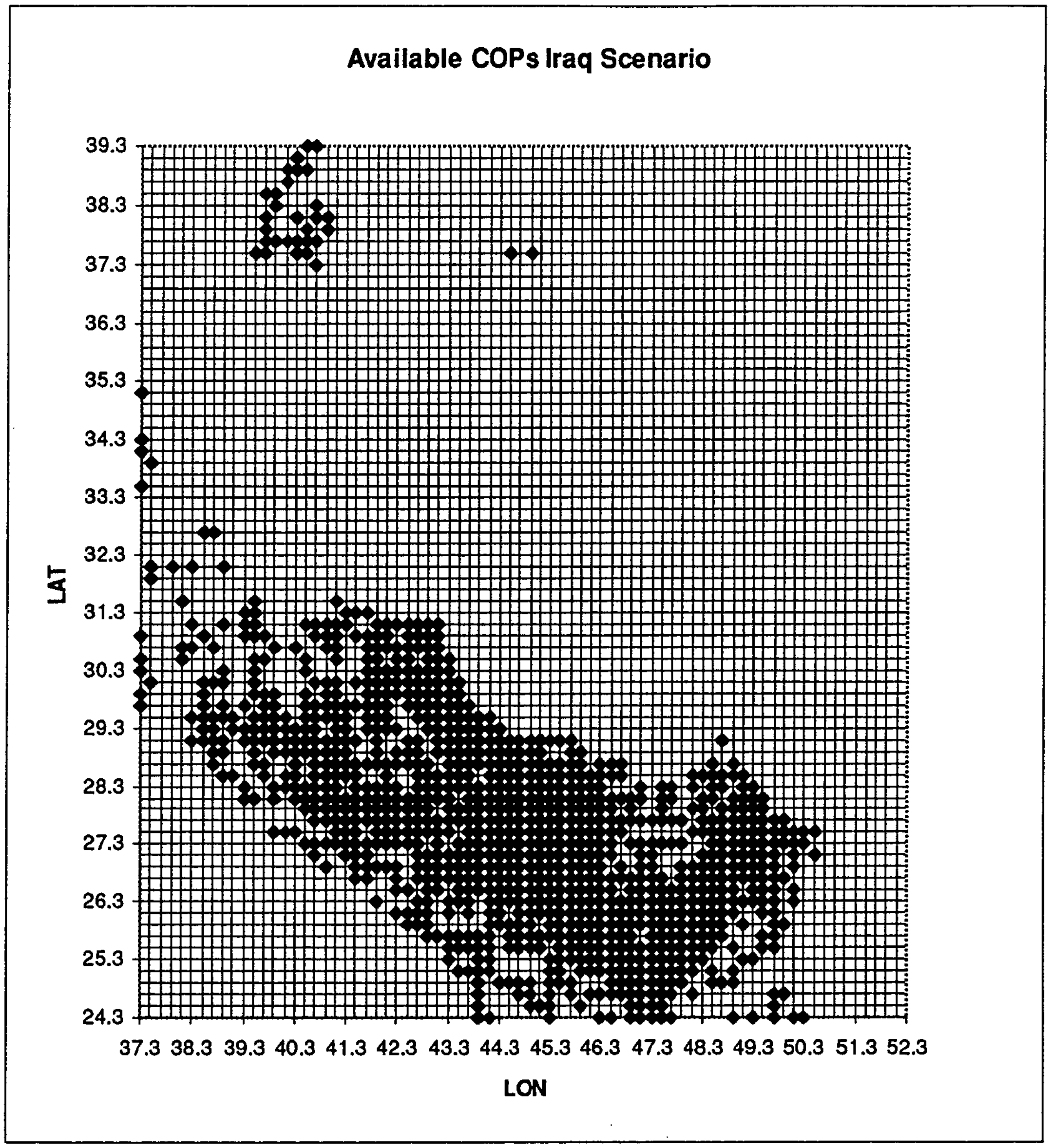

Figure 3: Iraq Scenario; COPS available after aggregation

After all elimination and aggregation was accomplished, the COPs remaining form the potential list of orbit points for placing the surveillance aircraft, as shown in Figure 3 for the Iraq scenario. Elimination and aggregation reduced the number of COPs from the remaining 2245, after the application of the constraints (14) and (15), to 1031.

3.7 Target Aggregation

Target aggregation was accomplished by applying rule 1 and rule 3 from the Current and Schilling paper to the targets available (see Section 2.3) [5]. No loss of location information occurs when these rules are applied. Target aggregation was accomplished after all COP elimination and aggregation was completed. The target value of targets selected for aggregation was summed and this value was assigned to the aggregated target. In test problems Current and Schilling obtained reductions of 36 to 92 percent in the number of targets using rules 1 and 3 [5:123]. It is expected that when using operational (and thus very large) target sets, similar results will be obtained. In actual scenarios, there could be 50 or more targets clustered around key points or population centers, which could be aggregated into a few targets producing a high degree of aggregation. Baghdad, Iraq, or P'YongYang, North Korea, provide good examples of such clustering of surveillance targets.

3.8 The Reduced Mathematical Model

After the elimination of COPs, and aggregation of COPs and targets, the resulting model can be stated as an integer programming problem in the form of an MCLP.

$$\text{Maximize} \quad z = \sum_{I} c_i \cdot y_i \tag{16}$$

$$\text{Subject To:} \quad \mathbf{A}\,\underline{x} \geq \underline{y} \tag{17}$$

$$\sum_{j} x_j \leq P \tag{18}$$

$x_j \in \{0,1\} \; \forall \, j = 1, \ldots n$

$y_i \in \{0,1\} \; \forall \, i = 1, \ldots m$

(underlined symbols represent column vectors)

where n = the number of COPs

m = the number of targets

$x_j = 1$, if an aircraft occupies COP j, 0 otherwise

$y_i = 1$, if target i is covered, 0 otherwise

A = A 0-1 matrix. $a_{ij} = 1$ if target i is within the cover range of COP j; 0 otherwise.

c_i = the value of covering target i

P = the number of aircraft available

Note that several constraints have been eliminated by preprocessing. This is now in the form of the MCLP shown in Equations (4) - (8).

3.9 Applying a Heuristic

As discussed in Chapter 2, Ignizio's heuristic was chosen to solve the MCLP developed in this research. Ignizio's heuristic utilizes a basic greedy procedure. In a greedy heuristic, the first decision variable assigned a value is the one that provides the greatest increase in the objective function value for the amount of resource used. The remaining decision variables are checked to find the decision variable, considering the targets not already covered, which offers the most improvement in the objective function. This variable is then assigned a value. This process is continued until no increase in the objective function can be obtained, or the limit on the number of decision variables assigned a value is reached. In this research model, the decision variables have values of 0 or 1. The heuristic selects three decision variables which are each assigned a value of 1. After the selection of the third decision variable, the procedure is modified by including a drop iteration after every greedy selection. Once the third variable is selected, a determination is made whether or not any one of k variables assigned a value of 1 can be dropped from the solution. A variable can be removed if the remaining k-1 variables assigned a value of 1 provide an improvement in the objective function value over the previously selected k-1 variables. For example, if after assigning three variables a value of 1, the objective function value is 58. Then, once a value is assigned to a fourth variable ($k = 4$) the objective function value increases to 65. If a group of three variables, from the four chosen, has an objective function value greater than 58, then the variable not in the group would be dropped (assigned a value of 0). A new variable is then assigned a value and added to the solution. This continues until no increase in the objective function is possible or the limit on the number of decision variables assigned value is reached. If no increase in the objective function can be obtained by assigning another variable a value, then the heuristic terminates. The detailed ten - step heuristic developed by Ignizio [12: 43-58] and modified by the author follows.

3.9.1 Step 1 - Initialization

First the problem must be formulated as a maximization problem in the form of Equations (16) - (18). Each element of the A matrix is transformed by $a_{ij} = a_{ij} \cdot c_i$ for each target i. Let θ be the ordered set of indices j of the variables or alternatives x_j set to one; initially $\theta = \{\varnothing\}$. Let k represent the number of alternatives selected thus far; initially $k =$

0. Let R be the matrix whose columns are A_{ij}, where $i = 1,...m$ and $j \in \theta$. Let A* be a column vector of the maximum R_{ij} for each $i = 1,\ldots m$, $\forall j \in \theta$.

3.9.2 Step 2 - Selection of First Alternative

For each column vector in A, calculate $T_j = \sum_{i=1}^{m} a_{ij}$. Choose the column vector A_j with the maximum T_j as the first alternative and designate it as A*. Set $x_j = 1$, place j in the ordered set θ and set $k = 1$. Remove the selected A_j from the **A** matrix. In event of a tie for maximum T_j, any one of the tied alternatives is selected. In the implementation of the heuristic the first T_j in the tie is selected. This step selects the first variable to be assigned a value.

3.9.3 Step 3 - Selection of Additional Alternatives

This step selects each additional variable to assign a value after the first. For each column A_j, where $j \notin \theta$, calculate $S_j = \sum_{i=1}^{m} \max(a_{ij} - a_i^*, 0)$. Find the maximum S_j, $S_j > 0$. For the j corresponding to max S_j, set $x_j = 1$ and place j in the $k+1$ position of θ. Set $k = k + 1$. If there is a tie for max S_j, any alternative may be selected. In the implementation of the heuristic, the first S_j in the tie is selected. If all $S_j \leq 0$, additional alternatives will not increase the objective function value; proceed to step 9.

3.9.4 Step 4 - Formation of the Best Combination

This step forms the column vector which is the current best solution. The sum of the column vector A* is equal to the objective function value at this point. Remove the column vector A_j with $j = \theta(k)$ (the j selected in step 3) from the **A** matrix. Update A*, where the *i*th element of A* is $a_i^* = \max_{j \in \theta} \{a_{ij}\}$, $i = 1, \ldots m$. If $|\theta| = 2$ repeat steps 3 and 4. Otherwise, proceed to step 5.

3.9.5 Step 5 - Combination Improvement Check

This step determines whether or not an alternative can be dropped by calculating the elimination effect. $\theta = \{\theta_1, \theta_2, \ldots \theta_k\}$, where θ_i represents the *i*th index in the ordered set θ. Let R be the matrix with columns $A(\theta_1)$, $A(\theta_2)$, ..., $A(\theta_k)$, A*. For each column of R, define E, the elimination effect, as follows:

$$E\,(A_j) = \sum_{i=1}^{m} \{\max_{s} (a_{i,s} - a_i^*)\} \text{ where } s \in \theta,\ s \neq j, \text{ for } j = \theta_1, \theta_2, \ldots \theta_k \qquad (19)$$

Define $E\,(A_{\theta_k})$ as the elimination effect of the last variable assigned a value ($j = \theta_k$). Proceed to step 6.

3.9.6 Step 6 - Elimination Check

This step determines if a variable can be removed from the solution and removes that variable if required. If the maximum $E(A_j) = E\,(A_{\theta_k})$ found in step 5, proceed to step 8. If not, remove the A_j with the maximum $E\,(A_j)$ from the matrix R (that is, remove this alternative from the solution), remove j from θ. Set $x_j = 0$. Set $k = k - 1$. Return A_j to the A matrix. Proceed to step 7.

3.9.7 Step 7 - Formulation of Improved Best Combination

Update A*, where $a_i^* = \max_{j \in \theta} \{a_{ij}\}$, $i = 1, 2, \ldots m$. Return to step 3. This step updates A* and proceeds to pick the next alternative.

3.9.8 Step 8 - Check

This check determines if the maximum number of alternatives have been selected. If $|\theta| = P$, proceed to step 9. If not, return to step 3 to select another alternative.

3.9.9 Step 9 - Assign and Terminate

For each i, set $y_i = 1$ if $a^*_i > 0$. For each i such that $y_i = 1$, assign variable y_i to alternative x_j for j corresponding to the $\max\limits_{j \in \theta} a_{ij}$. If there is a tie in the $\max\limits_{j \in \theta} a_{ij}$ then the assignment can be made to any of the tied variables. Go to step 10.

3.9.10 Step 10 - Reorder and Repeat

The A matrix is then reordered once and the heuristic is repeated. The best solution is then chosen from the two solutions generated. In the implementation of the heuristic, the A matrix is only reordered once. This reorder is accomplished by switching the last column with the first column, the second to last column with the second column, and so forth. Multiple reordering in a random fashion is an area for further research.

3.10 Example: Applying the Heuristic to MCLP

This example was developed to illustrate the workings of the heuristic using a small representative problem. The example begins with the **A** matrix already formed. Each column represents a candidate orbit point and each row represents a target to be covered. There are nine targets (y_1 to y_9) to cover from six candidate orbit points (x_1 to x_6) with three aircraft.

STEP 1: Place the problem in the proper form:

$$\text{Maximize } z = 15\, y_1 + 10\, y_2 + 10\, y_3 + 10\, y_4 + 15\, y_5 + 15\, y_6 + 3\, y_7 + 3\, y_8 + y_9 \quad (20)$$

subject to: $\sum_{j=1}^{6} x_j \leq 3$ (21)

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} \geq \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \\ y_9 \end{bmatrix} \qquad (22)$$

$x_j \in \{0,1\} \; \forall \, j = 1, \ldots 6, \qquad y_i \in \{0,1\} \; \forall \, i = 1, \ldots 9$

Transform the **A** matrix by multiplying by the c vector. The new **A** matrix is equal to $\underline{c}$**A** (original):

$$\mathbf{A} = \begin{bmatrix} 15 & 15 & 0 & 15 & 0 & 0 \\ 10 & 10 & 0 & 0 & 0 & 0 \\ 10 & 0 & 10 & 0 & 0 & 0 \\ 10 & 0 & 10 & 10 & 10 & 0 \\ 0 & 15 & 0 & 0 & 0 & 0 \\ 0 & 0 & 15 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 3 \\ 0 & 3 & 3 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

The problem is now set in tabular format for ease of illustration. Table 2 shows the example problem in tabular form

Table 2: Step 1 - A matrix for maximizing problem

	Candidate Orbit Points					
Targets	x_1	x_2	x_3	x_4	x_5	x_6
y_1	15	15	0	15	0	0
y_2	10	10	0	0	0	0
y_3	10	0	10	0	0	0
y_4	10	0	10	10	10	0
y_5	0	15	0	0	0	0
y_6	0	0	15	0	0	0
y_7	0	0	0	3	0	3
y_8	0	3	3	0	0	0
y_9	1	0	1	0	1	0

STEP 2 - Selection of first COP (Table 3)

Table 3: Step 2

	Candidate Orbit Points					
Targets	x_1	x_2	x_3	x_4	x_5	x_6
y_1	15	15	0	15	0	0
y_2	10	10	0	0	0	0
y_3	10	0	10	0	0	0
y_4	10	0	10	10	10	0
y_5	0	15	0	0	0	0
y_6	0	0	15	0	0	0
y_7	0	0	0	3	0	3
y_8	0	3	3	0	0	0
y_9	1	0	1	0	1	0
T_j	**46**	43	39	28	11	3

max $T_j = 46, x_1 = 1, \theta = \{1\}, k = 1$

Step 3 - Selection of additional COP; calculate $S_j = \sum_{i=1}^{m} \max(a_{ij} - a_i^*, 0)$ (Table 4).

Table 4: Step 3

	Candidate Orbit Points					
Targets	**A***	x_2	x_3	x_4	x_5	x_6
y_1	15	15	0	15	0	0
y_2	10	10	0	0	0	0
y_3	10	0	10	0	0	0
y_4	10	0	10	10	10	0
y_5	0	15	0	0	0	0
y_6	0	0	15	0	0	0
y_7	0	0	0	3	0	3
y_8	0	3	3	0	0	0
y_9	1	0	1	0	1	0
S_j	-	**18**	18	3	0	3

max $S_j = 18, x_2 = 1, \theta = \{1,2\}, k = 2$

Step 4 - Formation of the Best Combination (Table 5)

Table 5: Step 4

	Candidate Orbit Points				
Targets	**A***	x_3	x_4	x_5	x_6
y_1	15	0	15	0	0
y_2	10	0	0	0	0
y_3	10	10	0	0	0
y_4	10	10	10	10	0
y_5	15	0	0	0	0
y_6	0	15	0	0	0
y_7	0	0	3	0	3
y_8	3	3	0	0	0
y_9	1	1	0	1	0

Since $|\theta| = 2$, repeat steps 3 (Table 6) and 4 (Table 7).

Table 6: Step 3 - Repeat

	Candidate Orbit Points				
Targets	**A***	x_3	x_4	x_5	x_6
y_1	15	0	15	0	0
y_2	10	0	0	0	0
y_3	10	10	0	0	0
y_4	10	10	10	10	0
y_5	15	0	0	0	0
y_6	0	15	0	0	0
y_7	0	0	3	0	3
y_8	3	3	0	0	0
y_9	1	1	0	1	0
S_j	-	**15**	3	0	3

max $S_j = 15$, $x_3 = 1$, $\theta = \{1,2,3\}$, $k=3$

Table 7: Step 4 - Repeat

	Candidate Orbit Points			
Targets	**A***	x_4	x_5	x_6
y_1	15	15	0	0
y_2	10	0	0	0
y_3	10	0	0	0
y_4	10	10	10	0
y_5	15	0	0	0
y_6	15	0	0	0
y_7	0	3	0	3
y_8	3	0	0	0
y_9	1	0	1	0

Step 5 - Combination Improvement Check; calculate $E\,(A_j) = \sum_{i=1}^{m} \{ \max_{s} (a_{i,s} - a_i^*) \}$ (Table 8).

Table 8: Step 5

R matrix	**Candidate Orbit Points**			
Targets	x_1	x_2	x_3	**A***
y_1	15	15	0	15
y_2	10	10	0	10
y_3	10	0	10	10
y_4	10	0	10	10
y_5	0	15	0	15
y_6	0	0	15	15
y_7	0	0	0	0
y_8	0	3	3	3
y_9	1	0	1	1
$E(A_j)$ =	**0**	-15	-15	-

max $E(A_j) = 0$, $x_1 = 0$, $\theta = \{2,3\}$

Step 6 - Elimination Check. The maximum $E(A_j) \neq E(A_{\theta(k)})$. Thus, remove x_1 from R, remove 1 from θ, so $\theta=\{2,3\}$. Set $x_1 = 0$, $k = 2$, and proceed to step 7.

Step 7 - Formulation of Improved Best Combination (Table 9). Update A* for the new θ. Return to step 3. Tables 11 - 14 show the repeat of steps 3 - 6.

Table 9: Step 7

	Candidate Orbit Points				
Targets	**A***	x_1	x_4	x_5	x_6
y_1	15	15	15	0	0
y_2	10	10	0	0	0
y_3	10	10	0	0	0
y_4	10	10	10	10	0
y_5	15	0	0	0	0
y_6	15	0	0	0	0
y_7	0	0	3	0	3
y_8	3	0	0	0	0
y_9	1	1	0	1	0

Table 10: Step 3 - Repeat

	Candidate Orbit Points				
Targets	**A***	**x_1**	**x_4**	**x_5**	**x_6**
y_1	15	15	15	0	0
y_2	10	10	0	0	0
y_3	10	10	0	0	0
y_4	10	10	10	10	0
y_5	15	0	0	0	0
y_6	15	0	0	0	0
y_7	0	0	3	0	3
y_8	3	0	0	0	0
y_9	1	1	0	1	0
S_j	-	0	3	0	3

max $S_j = 3$, $x_4 = 1$, $\theta = \{2,3,4\}$, k=3

Table 11: Step 4 - Repeat

	Candidate Orbit Points			
Targets	**A***	**x_1**	**x_5**	**x_6**
y_1	15	15	0	0
y_2	10	10	0	0
y_3	10	10	0	0
y_4	10	10	10	0
y_5	15	0	0	0
y_6	15	0	0	0
y_7	3	0	0	3
y_8	3	0	0	0
y_9	1	1	1	0

Table 12: Step 5 and 6

R matrix	Candidate Orbit Points			
Targets	x_2	x_3	x_4	A*
y_1	15	0	15	15
y_2	10	0	0	10
y_3	0	10	0	10
y_4	0	10	10	10
y_5	15	0	0	15
y_6	0	15	0	15
y_7	0	0	3	3
y_8	3	3	0	3
y_9	0	1	0	1
$E(A_j) =$	-25	-26	-3	-

max $E(A_j) = -3$, max $E(A_j) = E(A_{\theta(k)})$ - proceed to step 8

Step 8 - Check. Since $|\theta| = 3$ proceed to step 9. The maximum number of aircraft have been selected.

Step 9 - Assign and Terminate. Since $\theta = \{2,3,4\}$, $x_2 = x_3 = x_4 = 1$, all others equal 0. Since $a^*_i > 0$ for all i, then $y_i = 1$, $i = 1,\ldots 9$, and the objective function value is 82. Since all $y_i = 1$, all targets are covered. This is also the optimal solution for the integer program given in Equations (20) - (22). Target assignment to a COP begins by assigning target y_1 to x_2 or x_4, y_2 to x_2, y_3 to x_3, etc. The heuristic terminates at this point.

Step 10 - Reorder and Repeat. The A matrix is then reordered and the problem is resolved. Again all targets are covered using three aircraft.

This example was also solved using a linear solver. Lindo® solved the problem in 22 pivots and obtained the same solution, on a Gateway® Pentium, 133Mhz computer. Chapter 4 shows the result of applying the heuristic to the two scenarios.

RESULTS

4.1 Introduction

This chapter discusses the results of applying this methodology to two large scenarios. The Iraq scenario is set up to simulate a possible coverage problem of targets in Iraq. The Far Eastern scenario simulates a covering problem posed by threats in both Korea and in and around Taiwan. Utilizing Ignizio's heuristic coded in FORTRAN 77 to solve either scenario took less than a minute including read and write times on a Sun Sparc 20. A sample of the computer output from the Far Eastern scenario is presented in Appendix G.

4.2 Iraq Scenario

The Iraq scenario had 205 targets inside a 900 by 900 NM cover grid. The cover radius used was 175 NM. Eleven aircraft were available for the scenario. Each COP was located 0.20 degrees, approximately 15 NM, apart for a total of 5776 COPs generated. The initial cover matrix was 205 rows by 5776 columns. The target list for the Iraq Scenario is in Appendix C and the restriction lists are in Appendix D. Table 13 shows the results of the preprocessing and the effects of aggregation.

Table 13: Iraq Scenario Reduction

Process	Starting #	Ending #	% Reduction
COPs Elim. by Home Base Restriction	5776	5054	12.5
COPs Elim. by Fighter Base Restriction	5054	4920	2.65
COPs Elim. by SAM Site Restriction	4920	3886	21.0
COPs Elim. by No-fly Restriction	3886	2245	42.2
COPs Elim. for No Coverage	2245	1620	27.8
COPs Aggregated	1620	1031	36.4
Total COP reduction - all sources	**5776**	**1031**	**82.2**
Targets Aggregated	205	184	10.2

The 36.4% reduction in problem size due to COP aggregation, while low, is in line with results obtained by Current and Schilling [5:123 -124]. Including target aggregation and restrictions the original **A** matrix reduced from 205 by 5776 to 184 by 1031 - an 84.0% reduction. Total coverage was obtained, given the target values used, with only four aircraft. When all targets had the same cover value of 1, the heuristic obtained total coverage with four aircraft. The solution is given in Table 14. The selection of COPs is in the order shown.

Table 14: Iraq Scenario COPs in Solution

Target Values Used			**Target Values Not-used**		
COP #	**LAT**	**LON**	**COP #**	**LAT**	**LON**
703	28.5	45.9	329	28.7	45.9
554	29.1	44.5	478	37.5	44.5
291	30.9	42.1	741	31.1	42.1
3	30.3	37.3	759	31.1	41.9

There were some alternate COP choices. For example, COP # 291 could be replaced with COP # 322 (39.1° by 40.1). This would give the planner alternate sites for placing the surveillance aircraft without any loss of target coverage. They might be useful in considering routing or in conjunction with other operational considerations. The coverage by percentage and value covered for both cases are shown in Figures 4-6. In both the target values used and not-used cases, there was no removal of previously selected COPs in favor of a better mix of COPs as the number of aircraft used increased. As shown in Figure 4, there is little difference in the percent coverage between the target values used and not-used case for this scenario. This is not surprising given the closeness of the COPs selected as shown in Table 14.

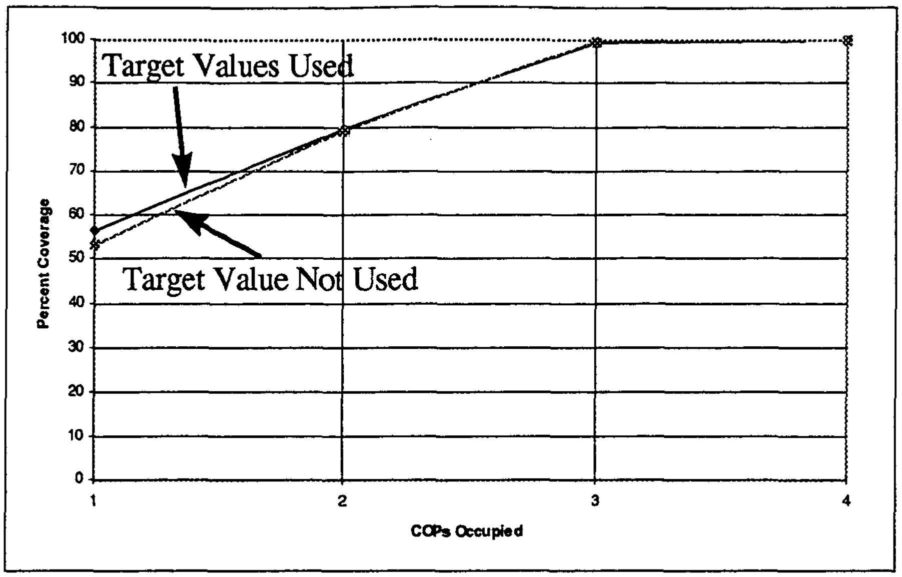

Figure 4: Iraq Scenario: Percent Target Value Covered

Figures 5 and 6 show the value of the targets covered for each COP occupied. In both cases, the last aircraft covers only one additional target. Given the limited number of assets available, this could be an important operational consideration.

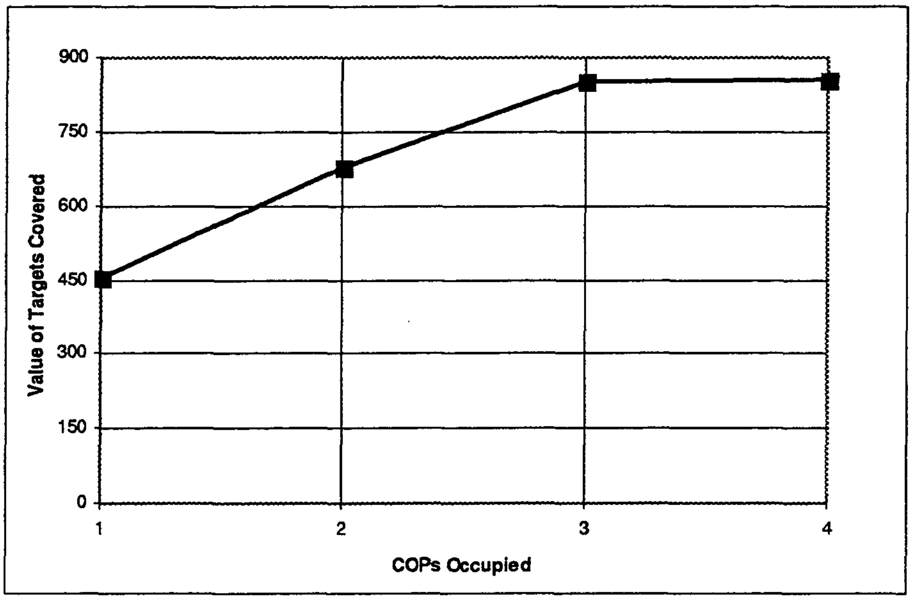

Figure 5: Target Value Used - Value of Targets Covered

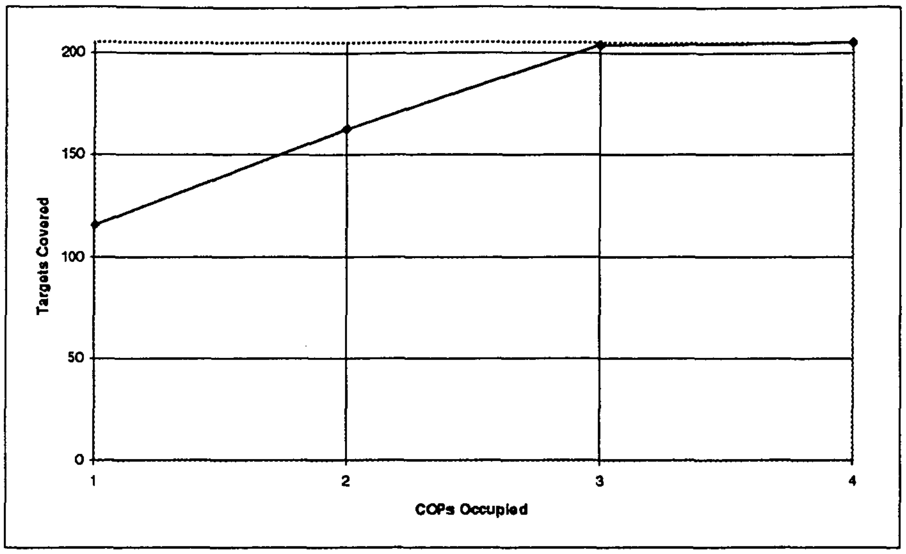

Figure 6: Target Value Not-used - Number of Targets Covered

Table 15 shows in tabular form the data in Figures 5 and 6. In both cases, over 50% of the targets are covered with just one aircraft.

Table 15: Iraq Scenario - Increases by aircraft

Target Values Used			Target Values Not-used		
COP #	%Increase	Value Increase	COP #	%Increase	Value Increase
703	53.22	455	329	56.59	116
554	26.08	223	478	22.93	47
291	20.35	174	741	20.00	41
3	0.35	3	759	0.48	1

In both cases, 99% of the required coverage was accomplished with only three aircraft. The marginal return for assigning extra surveillance aircraft to the theater is minimal. The theater commander could decide the number of aircraft used to achieve the percent of coverage. It should be again noted that, from the author's experience in NATO AWACS, it generally it takes 3 - 4 aircraft to occupy a surveillance orbit continuously. Thus, 3 - 4 orbit points represent 9 - 16 actual aircraft and associated crews and maintenance support.

Multiple coverage of some targets occurred. Most targets were only covered . once, while only a few were covered more than twice. If the workload on any aircraft is too high, then more aircraft could be assigned to reduce the load. In both cases, the last aircraft assigned greatly increased the number of targets multiply covered. The results are summarized in Table 16.

Table 16: Iraq Scenario Multiple Target Cases

	Targets Covered in Each Case	
Coverage	**TGT Values Used**	**TGT Values Not-used**
None	0	0
Single	180	147
Double	25	38
Triple	0	20

The heuristic solved the Iraq problem quickly and efficiently. The results provide the theater commander with a good approximation of the number of surveillance aircraft required for a given coverage requirement.

4.3 Far Eastern Scenario

The Far Eastern Scenario had 133 targets inside an approximately 1,260 by 1,260 NM cover grid. The cover radius used was 175 NM. Eleven aircraft were available for the scenario. Each COP was located 0.20 degrees or approximately 15 NM apart, for a total of 11,236 COPs generated. The initial cover matrix was 133 rows by 11,236 columns. The target list for the Far Eastern Scenario is in Appendix E and the restriction lists are in Appendix F. Table 17 shows the results of the preprocessing and the effects of aggregation.

Table 17: Far Eastern Scenario Reductions

Process	Starting #	Ending #	% Reduction
COPs Elim. By Home Base Restriction	11236	8191	27.1
COPs Elim. By Fighter Base Restriction	8191	5489	33.0
COPs Elim. By SAM Site Restriction	5489	5027	8.4
COPs Elim. By No-fly Restriction	5027	2963	41.1
COPs Elim. For No Coverage	2963	2733	7.8
COPs Aggregated	2733	753	72.4
Total COP reduction - all sources	**11236**	**753**	**93.3**
Targets Aggregated	133	130	2.3

Note the COP aggregation of 72.4% was on the high end of the results obtained by Current and Schilling [5:123-124]. Including target aggregation and restrictions, the original **A** matrix reduced from 133 by 11,236 to 130 by 753 - a 93.4% reduction. Total coverage was not obtained, with the target values used and not-used, with six aircraft. Three targets were not covered in both cases. The solution is given in Table 18. The selection of COPs is in the order shown. Note how the final solution set of COPs is the same, but the order of selection was different due to the effects of using the target values.

Table 18: Far Eastern Scenario COPs in Solution

Target Values Used			**Target Values Not-used**		
COP #	**LAT**	**LON**	**COP #**	**LAT**	**LON**
525	35.9	127.4	2	22.1	119.6
2	22.1	119.6	525	35.9	127.4
615	37.5	129.0	91	32.1	121.8
91	32.1	121.8	615	37.5	129.0
388	24.7	125.0	271	35.9	123.4
271	35.9	123.4	388	24.7	125.0

There were some alternate COP choices. For example, COP # 271 had six alternate sites available (COPs # 272, 332, 333, 334, 335, and 336). This would give the planner alternate sites for placing the surveillance aircraft without any loss of target coverage. The coverage by percentage and value-covered for both cases are shown in Figures 7 - 9. As shown in Figure 7, there is little difference in the percent coverage between the target values used and not-used case. This is expected from the closeness of the COPs selected as shown in Table 18.

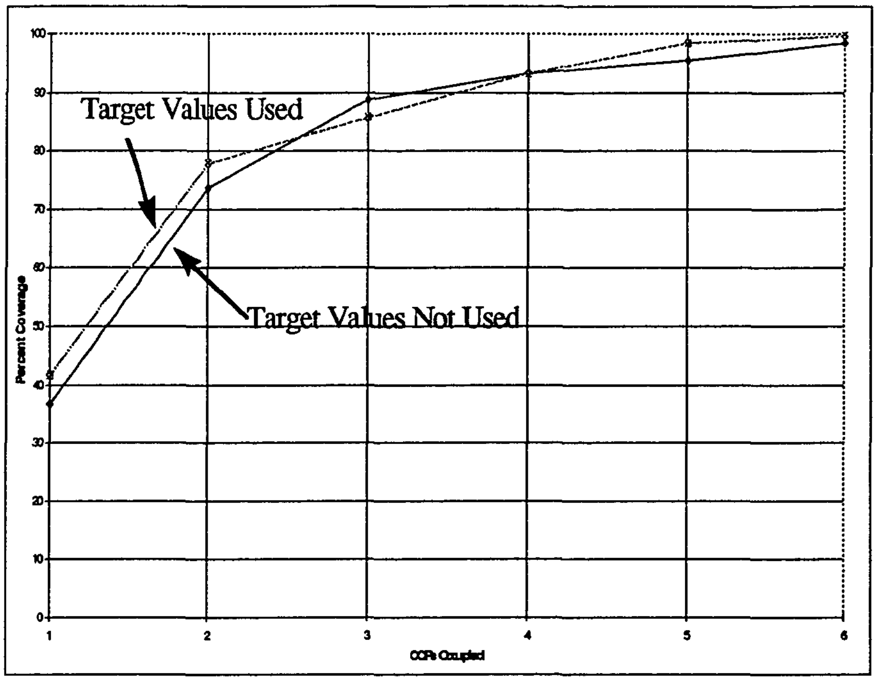

Figure 7: FE; Percent of Target Value Covered

Figures 8 and 9 show the value of the targets covered for each COP occupied. Note the diminishing returns as more aircraft are assigned.

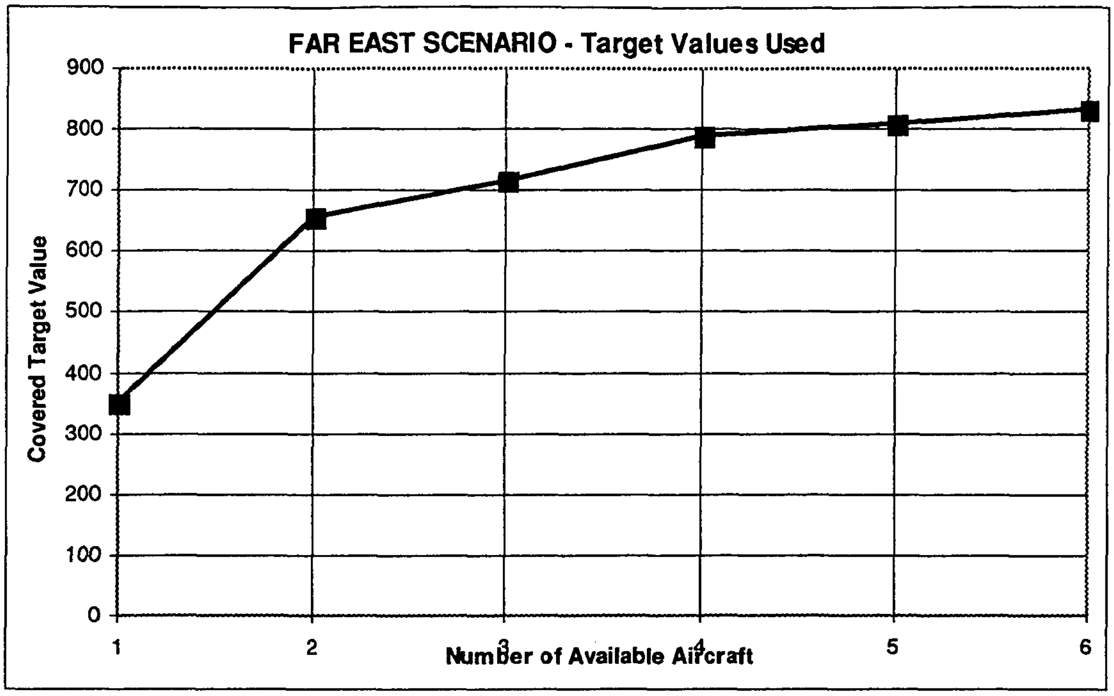

Figure 8: FE; Target Value Used - Value of Targets Covered

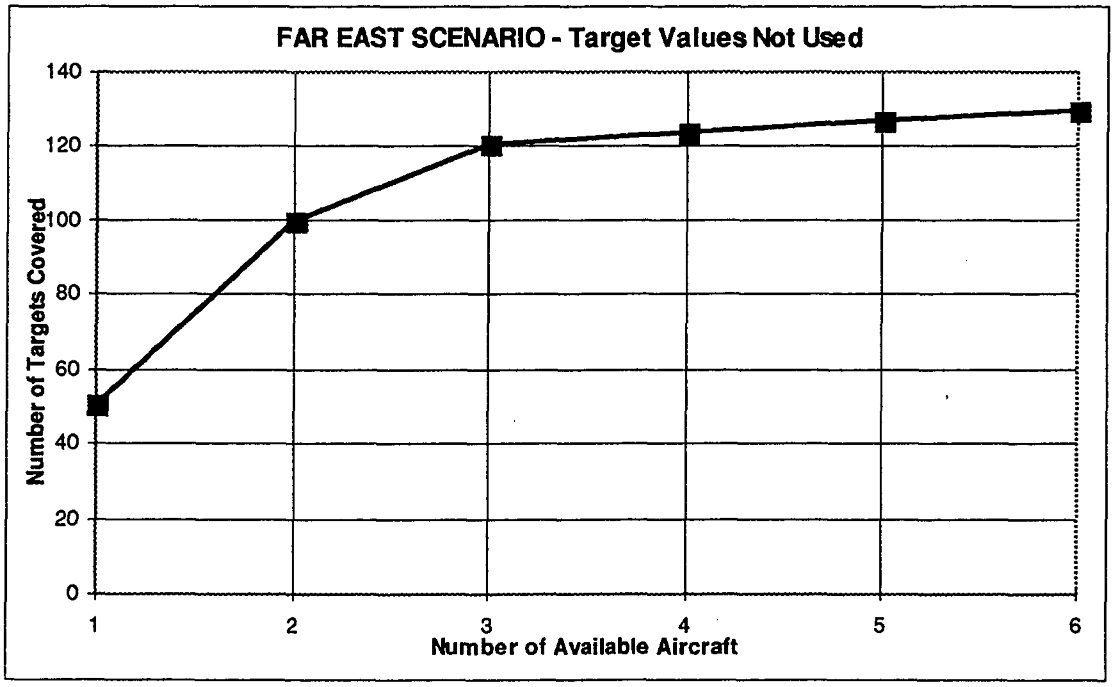

Figure 9: FE; Target Values Not-used - Number of Targets Covered

In both cases over 90% of the required coverage was accomplished with only three aircraft. The marginal return for assigning extra surveillance aircraft to the theater is minimal. The theater commander could decide the number of aircraft used to achieve the percent of coverage required. Note that 3 - 6 orbit points represent 9 - 24 actual aircraft. For a scenario using the E-8 aircraft, with only 20 projected to be bought by the USAF, a requirement of 24 aircraft would exceed the entire E-8 inventory.

Multiple coverage of some targets occurred. Most targets were only covered once and none were covered more than twice. If the workload on any aircraft is too high, then more aircraft could be assigned to reduce the load, if the aircraft are available. The results are summarized in Table 19.

Table 19: Far Eastern Scenario Multiple Target Cases

Multiple Target Coverage Target Value On or Off	
Coverage	**Number**
None	3
Single	80
Double	50
Triple	0

The value of a non-covered target y_j is c_j from Equation (9). In both cases there were three non-covered targets with a combined value of 3 and 12, respectively.

RECOMMENDATIONS

5.1 Summary

The heuristic solution to the MCLP provided quick and accurate results for two large theater-level scenarios. The heuristic can be used to find good candidate locations for placing airborne surveillance assets. The number of such assets required for a theater-level conflict can be quickly found and the solution updated as conditions change. The program can be easily modified to accommodate actual data (such as real target lists, actual coverage ranges, etc.). The research objective of solution times under 30 minutes was easily met in the test scenarios. Alternate locations, providing equal coverage, were found for the selected orbit points. The model is easy to use and could be called as a subprogram, to optimize locations as part of another program, if necessary.

The research was successful. The model found effective orbit points to place the surveillance aircraft so target coverage is maximized. Solution times were under a minute, which was well within the 30 minute limit set as the goal.

5.2 Implementation

The heuristic can be applied to any problem that can be placed in the form of a MCLP, Equations (16) - (18). Such problems can be found in the selection and placement of radar or sonar for fleet defense, placement of air or water pollution sensors, deployment of fire-fighting equipment, warehouse location problems, and, with the appropriate transformation to the dual, packing problems. The heuristic has already been applied to locating ground air defense radar in the Boston area [12: 95].

5.3 Further Research

There are two promising areas for further research. First, the heuristic can be improved to provide additional options such as double coverage of targets. Secondly, the problem setup and constraints can be refined.

The heuristic can be enhanced by adding the ability to handle requirements that some targets, or all, require multiple coverage prior to being considered as covered. This would reflect an operational requirement that some targets must be covered continuously even in the event of the loss of a surveillance aircraft. A requirement to obtain double coverage on some demand points would be the first direction of exploration. This might be accomplished by not crediting coverage of a target, requiring multiple coverage, until the required coverage is obtained.

The heuristic assigns targets to aircraft without consideration of the load on an aircraft. One area of research would be to modify the heuristic to consider the workload on any aircraft. This load-balance would act as an upper bound on the number of targets covered by a single aircraft.

The preprocessing can be enhanced by including pre-existing radar. These existing radar sites would cover targets which could be eliminated from the target list. This would enable a more comprehensive model of an entire integrated air defense system.

Modeling the borders of no-fly zones more accurately could also improve the model by eliminating more operationally infeasible COPs prior to solution. Lastly, placing realistic terrain restrictions on how far a surveillance aircraft can 'see' from a given COP in a given direction might provide a more realistic solution. This could take into account the blocking effect of tall mountain ranges. This would be particularly true if the operational theater was situated in very rugged terrain. The E-8 Joint STARS aircraft placement optimization would benefit from this more realistic treatment.

Heuristics provide quick, good solutions to a wide variety of problems. In the case of the MCLP, Ignizio's heuristic provides an excellent example. In large MCLP problems, this heuristic could be used to find a good solution very quickly and to provide a good starting basis from which to solve the problem to optimality using a commercial solver.

Appendix A

MATHEMATICAL MODEL

Parameters:

i = Number of the surveillance point.

j = Number of the target point

k = Number of the SAM site

m = Number of the friendly fight base

n = Number of the home base of the surveillance aircraft

$COPT_j$ = The set of orbit points that cover target j

Constants:

Y_{ik} = Distance from surveillance point i to SAM site k

LR_k = Lethal radius of the SAM at site k

V_{im} = Distance from surveillance point i to DCA base m

$V_i = \text{MIN } V_{im} \; \forall \; i, m$

$DCAR_m$ = Range limit from friendly DCA base m

W_{in} = Distance from surveillance point i to home base n

$W_i = \text{MIN } W_{in} \; \forall \; i, n$

ACR_n = Range limit from home base n

X_{ip} = Distance from surveillance point i to No-Fly Point p

NFP_p = No-Fly range of No-Fly Point p

ASP = Number of surveillance aircraft available

P_j = Target j value

Variables:

t_j = Is the target j covered by at least one occupied orbit point? 0 = No, 1 = Yes

cop_i = Is the orbit point i occupied by an aircraft? 0 = No, 1 = Yes

Mathematical equations:

MAX $\sum_j P_j \cdot t_j$ Objective Function

Subject To: $\sum_i cop_i \leq ASP$ Number of SPs available

$t_j \leq \sum_{i \in COPT_j} cop_i , \ \forall j$ Covers the targets

$W_i \cdot cop_i \leq ACR_n , \ \forall i, n$ Range limit from home base

$V_i \cdot cop_i \leq DCAR_m , \ \forall i, m$ Range limit from friendly fighter base

$cop_i \leq \frac{Y_{ik}}{LR_k} , \ \forall i, k$ Avoids lethal SAM range

$cop_i \leq \frac{X_{ip}}{NFRp} , \ \forall i, p$ Avoids No-Fly Zones

where $cop_i \in \{0,1\}$ for all i, $t_j \in \{0,1\}$ for all j

APPENDIX B

FORTRAN 77 Program.

```
      PROGRAM ZAWACS

C ****************************************************************
C                     VARIABLE DEFINITIONS                    *
C
C ntgt     = number of targets
C ASP      = Available Surveillance Aircraft
C cls      = # of columns in A matrix
C ncsp     = # of candidate surveillance points
C ncolA    = # of active columns in A matrix
C nrowA    = # of active rows in A matrix
C gridsize = Distance, in degrees, between csp lat. and lon.
C CTRLAT   = Center latitude of all the targets
C CTRLON   = Center longitude of all the targets
C e        = Earth's eccentricity
C Ae       = Radius of Earth at equator in NM
C Rad      = Converts degrees to radians
C esqr     = Eccentricity Squared
C CR       = Cover Radius in NM
C TOTGRDSZ = Total grid size - each side of grid box in degrees
C            Centered on the center lat. & longitude
C x, z     = Variables used to find x,y,z coordinates
C const(2) = Dummy variables
C TGTLAT   = Array of Target latitudes
C TGTLON   = Array of Target Longitudes
C TGTVAL   = Array of Target Values
C TGTx,y,z = Arrays of Target x,y,z values in NM
C CTRx,y,z = x,y,z coordinates of the Center of all the targets
C CSPTx,y,z = Arrays of the Candidate Surveillance Points (CSPT)
C            x,y,z coordinates
C CSPLAT,LON = Arrays of CSPT latitude and longitude
C COVERMAT = Cover Matrix(CM)- The A matrix, size ntgt by ncsp
C colsum   = The sum of all the entries of each column in CM
C rowsum   = The sum of all the entries of each row in CM
C colind   = Column Indicator - an array pointer to the CM columns
C rowind   = Row Indicator - an array pointer to the CM rows
C multCSP  = A ncsp by ncsp array to keep track of aggregated
C            CSPTs - These are possible alternate solution sites
C TGTNUM   = An Array of the targets numbers from 1 to ntgt
C multTGT  = A ntgt by ntgt array to keep track of aggregated
```

```
C          target sites                                   *
C *******************************************************************
C             HEURISTIC VARIABLE DEFINITIONS
C  Oset    = The Ordered Set of the columns selected in the A matrix
C  Xj      = The number of ASPs used
C  UV      = Yes/No switch for using target values 1 = Yes, 0 = No
C  Tj      = The column sum used in step 2
C  Sj      = The column sum used for selection of additional alt.
C          see the heuristic description
C  Astar   = The column vector of the last site selected
C  R       = The ordered matrix storing the column vectors of
C          the selected sites
C  EE      = The column vector used in the elimination effect
C *******************************************************************
C             PREPROCESSING VARIABLE DEFINITIONS
C  nBase   = Number of friendly home bases
C  nFtr    = Number of friendly fighter bases
C  nSAM    = Number of enemy SAM sites
C  nNoFly  = Number of No Fly points included
C  FTR     = Array of the number,x,y,z coordinates, and range of each
C          fighter base
C  SAM     = Array of the number,x,y,z coordinates, and lethal range
C          of each SAM site
C  Base    = Array of the number,x,y,z coordinates, and range
C          of each home base site
C  NoFly   = Array of the number,x,y,z coordinates, and exclusion
C          range of each NoFly point
C * Note: The four arrays hold lat, lon coordinates in the x,y positions
C  until conversion to x,y,z coordinates
C *******************************************************************
C                  PROGRAM NOTES:
C
C  The following parameters must be set prior to running the program.
C
C  VARIABLE             USE                           UNITS
C  gridsize - Defines how 'fine' the CSPT grid will be          [DEGREES]
C  TOTGRDSZ - Defines how large the grid will be        [DEGREES]
C  ntgt     - How many targets are involved             [INTEGER]
C  cls      = (TOTGRDSZ/gridsize)+1                     [INTEGER]
C  ncsp     = cls*cls                                   [INTEGER]
C  ASP      - Number of aircraft are available                  [INTEGER]
C  CR       - The Cover Radius                          [  NM  ]
C  nBase    - Number of Home Bases involved                     [INTEGER]
C  nFtr     - Number of Friendly fighter bases involved         [INTEGER]
```

```
C nSAM    - Number of SAM sites involved                  [INTEGER]
C nNoFly  - Number of No Fly points involved              [INTEGER]
C UV      - Use target Values? 1 = Yes, 0 = No       [INTEGER]
C ***                                   ***
C The following files must be created to input the correct data  ***
C UNIT   FILE NAME            USE
C 1     fetgts.txt    - Reads in the targets (#,LAT,LON,TGTVAL)
C 2     feftrbas.txt  - Reads in the friendly fighter bases in
C                 (#, LAT, LON, RANGE)
C 3     fehome.txt    - Reads in the Home base list in
C                 (#, LAT, LON, RANGE)
C 4     fenofly.txt   - Reads in the no-fly points in
C                 (#, LAT, LON, RANGE)
C 5     feSAM.txt     - Reads in the SAM sites in
C                 (#, LAT, LON, LETHAL RANGE)
C ********************************************************************
C
C                   Declarations

    INTEGER i, j, k, ntgt, ASP, cls, ncsp, ncolA, nrowA

    REAL gridsize, CTRLAT, CTRLON, e, Ae, Rad, esqr
    REAL CR, TOTGRDSZ
    REAL x, z, const, const2

    PARAMETER (gridsize = 0.2, TOTGRDSZ = 21.0)
    PARAMETER (ntgt = 133, cls = 106, ncsp = 11236)
    PARAMETER (e = 0.08182, Ae = 3443.9, Rad = 0.0174533)
    PARAMETER (ASP = 11, CR = 175.0)

    REAL TGTLAT(ntgt), TGTLON(ntgt), TGTVAL(ntgt)
    REAL TGTx(ntgt), TGTy(ntgt), TGTz(ntgt)
    REAL CTRx, CTRy, CTRz
    REAL CSPTx(ncsp), CSPTy(ncsp), CSPTz(ncsp)
    REAL CSPLAT(ncsp), CSPLON(ncsp)

    REAL COVERMAT(ntgt,ncsp), colsum(ncsp), rowsum(ntgt)
    INTEGER colind(ncsp), rowind(ntgt)

    REAL multCSP(ncsp,ncsp),TGTNUM(ntgt)
    REAL multTGT(ntgt,ntgt)

C        Heuristic Variables
```

```
      INTEGER Oset(ncsp), Xj, UV
      REAL  Tj(ncsp), Sj(ncsp), Astar(ntgt)
      REAL R(ntgt,ncsp), EE(ncsp)
      PARAMETER( UV = 0)

C        Preprocessing variables

      INTEGER nBase, nFtr, nSam, nNoFly
      PARAMETER (nBase = 4, nFtr = 16, nSam = 17, nNoFly = 6)
      REAL FTR(nFtr,5), SAM(nSam,5), Base(nBase,5), NoFly(nNoFly,5)

      CTRLAT = 0.0
      CTRLON = 0.0

C           Open Data files

      OPEN(UNIT = 1, FILE = 'fetgts.txt', STATUS = 'OLD')
      OPEN(UNIT = 2, FILE = 'feftrbas.txt', STATUS = 'OLD')
      OPEN(UNIT = 3, FILE = 'fehome.txt', STATUS = 'OLD')
      OPEN(UNIT = 4, FILE = 'fenofly.txt', STATUS = 'OLD')
      OPEN(UNIT = 5, FILE = 'fesam.txt', STATUS = 'OLD')
      OPEN(UNIT = 10, FILE = 'NVfeRES', STATUS = 'NEW')
      OPEN(UNIT = 11, FILE = 'NVfeHres', STATUS = 'NEW')
      OPEN(UNIT = 12, FILE = 'NVfeCres', STATUS = 'NEW')
      OPEN(UNIT = 13, FILE = 'NVfeZres', STATUS = 'NEW')

      WRITE(11,*)
      WRITE(11,*)'FAR EAST SCENARIO - TGT VALUES OFF'
      WRITE(11,*)
C ***
C Read in home base data and fighter base data (Blue Forces)
C and then read in SAM data and no fly area data (RED Forces)
C ***

      CALL GETBLUE(nBase,nFtr,FTR,Base)
      CALL GETRED(nSam, nNoFly,SAM,NoFly)

C ***
C Read in Target Data
C ***

       DO 10 i = 1, ntgt

          READ(1,*) TGTNUM(i),TGTLAT(i),TGTLON(i),TGTVAL(i)
```

```
         CTRLAT = CTRLAT + TGTLAT(i)
         CTRLON = CTRLON + TGTLON(i)

 10   CONTINUE

      CTRLAT = Rad * CTRLAT/FLOAT(ntgt)
      CTRLON = Rad * CTRLON/FLOAT(ntgt)

C ***
C Generate Grid to place targets on
C ***
C First find the R vector for each target
C ***
      DO 20 i = 1, ntgt

         TGTLAT(i) = TGTLAT(i) * Rad
         TGTLON(i) = TGTLON(i) * Rad

 20   CONTINUE

      esqr = e*e
      const = Ae*(1.0-(esqr))
      const2 = SQRT(1.0-(esqr*(SIN(CTRLON)*SIN(CTRLON))))

      x = (Ae/const2)*COS(CTRLON)
      z = (const/const2)*SIN(CTRLON)

      CTRx = x * COS(CTRLAT)
      CTRy = x * SIN(CTRLAT)
      CTRz = z

      DO 30 k = 1, ntgt

         const2 = SQRT(1.0-(esqr*(SIN(TGTLON(k))*SIN(TGTLON(k)))))

         x = (Ae/const2)*COS(TGTLON(k))
         z = (const/const2)*SIN(TGTLON(k))

         TGTx(k) = x * COS(TGTLAT(k))
         TGTy(k) = x * SIN(TGTLAT(k))
         TGTz(k) = z

 30   CONTINUE
```

```
C ***
C       DO 40 j = 1, ntgt
C       WRITE(13,*)TGTx(j),TGTy(j),TGTz(j), TGTNUM(j)
C 40    CONTINUE
C ***
C Finds the x, y, z coordinates of each of the constraint points
C with respect to the x,y,z coordinate system
C ***

      CALL  KILLGRID(nBase, nFTR, nSAM, nNoFly, Base, FTR, SAM, NoFly,
     +  CTRx, CTRy, CTRz)

C ****************************************************************
C ****************************************************************
C Finds the candidate surveillance points and then generates the
C 'A' matrix - the cover matrix
C *****

       CALL FINDCSP(gridsize, TOTGRDSZ, cls, ncsp, CSPTx, CSPTy,
     +  CSPTz, CTRLAT, CTRLON, CSPLAT, CSPLON)

C ***
C This subroutine processes the CSPs for each constraint
C Columns are removed from the cover matrix
C ***
       CALL ELIMCSP(CSPTx,CSPTy,CSPTz,nBase,nFTR,nSam,nNoFly,Base,FTR,
     +       SAM,NoFly,ncsp,colind, CSPLAT, CSPLON)
C ***
C This subroutine generates the 'A' matrix - COVERMAT
C ***
C
       CALL ZCOVER(CSPTx,CSPTy,CSPTz,TGTx,TGTy,TGTz,ntgt,ncsp,
     +  COVERMAT,CR)

C ***
C This subroutine aggregates the CSPs. Eliminates CSPs that do not
C cover any targets and then aggregates CSPs with identical coverage.
C CSP with identical coverage represent possible multiple solutions.
C ***

          CALL AGGCSP(COVERMAT,ntgt,colind,colsum,CSPTx,
     +  CSPTy,CSPTz,ncsp,ncolA,multCSP, CSPLAT, CSPLON)
C ***
C This subroutine aggregates the targets.  This reduces the number
```

```
C of rows in the cover matrix. Targets with the same CSP coverage
C are combined into one aggregated target. The target value is
C updated to reflect the additional coverage. No coverage errors
C are introduced.
C ***
      CALL AGGTGT(COVERMAT,ncolA,nrowA,rowind,ncsp,
   +          rowsum,multTGT,TGTNUM,ntgt,TGTVAL,UV)

C ***
C SEND THE A MATRIX TO FILE FOR GAMS USE.
C DATA IS ALL PREPROCESSED AT THIS POINT.
C ***

C ***
C Subroutine FINDSPT (find solution points) runs the heuristic
C and finds the solution.
C ***

    CALL FINDSPT(COVERMAT,ncolA,nrowA,colind,rowind,colsum,
   + rowsum,TGTVAL,ASP,Oset,Tj,Sj,Astar,R,EE,ntgt,ncsp,Xj)

C ***
C Subroutine XXX prints out the results
C ***

    CALL XXX(COVERMAT,ncolA,nrowA,TGTVAL,ASP,Oset,ntgt,
   + ncsp,Xj,CSPLAT,CSPLON,TGTLAT,TGTLON,rowind,colind)

C     CLEAN UP

      CLOSE(UNIT=1)
      CLOSE(UNIT=2)
      CLOSE(UNIT=3)
      CLOSE(UNIT=4)
      CLOSE(UNIT=5)

      END FILE(UNIT = 10)
    END FILE(UNIT = 11)
      END FILE(UNIT = 12)
      END FILE(UNIT = 13)

      CLOSE(UNIT = 10)
      CLOSE(UNIT = 11)
      CLOSE(UNIT = 12)
```

```
        CLOSE(UNIT = 13)

        STOP
        END

C ***
C ***************************************************************************
C ***************************************************************************
C ******      END MAIN PROGRAM - SUBROUTINES FOLLOW          *****
C ***************************************************************************
C ***************************************************************************
C ***
C This subroutine prints out the results
C ***
      SUBROUTINE XXX(CM,ncolA,nrowA,TV,ASP,Oset,ntgt,ncsp,Xj,
     +  csplat, csplon, tgtlat,tgtlon, rowind, colind)

      INTEGER i, j, ASP, ntgt, ncsp, ncolA, nrowA, iconst
      INTEGER Oset(ncsp), Xj, rowind(ntgt), colind(ncsp)
      REAL CM(ntgt,ncsp), TV(ntgt), csplat(ncsp),csplon(ncsp)
      REAL tgtlat(ntgt), tgtlon(ntgt), RADS, part(30), const

      REAL ttv, ttvc
      RADS = 0.0174533

      WRITE(13,*)'The final list of COPs considered'
      WRITE(13,*)'  #      LAT        LON'
      DO 49 i = 1, ncolA

      WRITE(13,*)i,'  ',csplat(i),'  ',csplon(i)

 49   CONTINUE

      WRITE(11,*)
      WRITE(11,*)' The LAT and LON of the solution are'
      WRITE(11,*)'  #      LAT        LON'
      DO 51 i = 1, Xj
      WRITE(11,*)Oset(i),'  ',(csplat(Oset(i))/RADS),' ',
     +  (csplon(Oset(i))/RADS)
 51   CONTINUE
      WRITE(11,*)
      WRITE(11,*)
      WRITE(11,*)'The following targets were not covered'
      WRITE(11,*)'LAT    LON    Value'
      DO 52 i = 1, ntgt
          rowind(i) = 0
52    CONTINUE
      DO 53 i = 1, 30
         part(i) = 0.0
53    CONTINUE

      ttv = 0.0
      ttvc = 0.0

      DO 61 i = 1, nrowA
        ttv = ttv + TV(i)
61    CONTINUE

      DO 62 i = 1, nrowA

         DO 63 j = 1, Xj

            IF (CM(i,Oset(j)).GT.0) THEN

               ttvc = ttvc + TV(i)
               GO TO 62
            ENDIF

63       CONTINUE

      WRITE(11,*)(tgtlat(i)/RADS),(tgtlon(i)/RADS),
     +          TV(i)

62    CONTINUE

C Checks to see if all targets are covered

      IF (ttvc.EQ.ttv) THEN
         WRITE(11,*)'ALL TARGETS COVERED'
      END IF

      DO 64 i = 1, Xj
        const = 0.0
        DO 65 j = 1, nrowA

          IF (rowind(j).EQ.0) THEN
             IF (CM(j,Oset(i)).GT.0.0) THEN
```

```
              const = const + TV(j)
              rowind(j) = 1

            ENDIF
          ENDIF

65        CONTINUE
         part(i) = const
64     CONTINUE

      WRITE(11,*)'Each solution gives the following % to total'
      WRITE(11,*)'Orbit    Percent    Coverage'
      WRITE(11,*)'Number   Coverage    Value'
      DO 66 i = 1, Xj
         const = 100.0*part(i)/ttv
         WRITE(11,*)Oset(i), const, part(i)
66     CONTINUE

      DO 67 j = 1, nrowA
         rowind(j) = 0
67     CONTINUE

      WRITE(11,*)'Target Coverage by each Orbit Point'
      WRITE(11,*)
      WRITE(11,*)'TARGET  TARGET  TIMES    Orbit Points'
      WRITE(11,*)'NUMBER  VALUE  COVERED   In Coverage'
      WRITE(11,*)'------------------------------'
      DO 68 i = 1, nrowA
         iconst = 0
         DO 69 j = 1, Xj
            IF (CM(i,Oset(j)).GT.0.0) THEN
               iconst = 1 + iconst
               colind(j) = Oset(j)
            ELSE
               colind(j) = 0
            ENDIF
69        CONTINUE

      WRITE(11,171)i,TV(i),iconst,(colind(k), k=1,Xj)

68     CONTINUE
171   FORMAT(I4,2X,F4.1,2X,I4,2X,20(1X,I5))
      RETURN
      END
C ***
C This subroutine reads in  the Blue data
C ***
      SUBROUTINE GETBLUE(nbase,nftr,FTR,Base)

      INTEGER i, j, k, nbase, nftr
      REAL FTR(nftr,5), Base(nbase,5)

      DO 2001 i = 1, nbase
         READ(3,*)(Base(i,j),j = 1, 4)
         Base(i,5) = Base(i,4)
 2001 CONTINUE

      DO 2002 i = 1, nftr
         READ(2,*)(FTR(i,k), k = 1, 4)
         FTR(i,5) = FTR(i,4)
 2002 CONTINUE

      RETURN
      END

C ***
C This subroutine reads in the Red data
C ***
      SUBROUTINE GETRED(nsam, nnofly, SAM, NoFly)

      INTEGER i, j, nsam, nnofly
      REAL SAM(nsam,5), NoFly(nnofly,5)

      DO 2003 i = 1, nsam
         READ(5,*)(SAM(i,j), j = 1, 4)
         SAM(i,5) = SAM(i,4)
 2003 CONTINUE

      DO 2004 i = 1, nnofly
         READ(4,*)(NoFly(i,j), j = 1, 4)
         NoFly(i,5) = NoFly(i,4)
 2004 CONTINUE

      RETURN
      END

C ***
```

```
C This subroutine finds the x, y, z coordinates of each of the constraint
C points and places them in the 2, 3, 4 spots of each matrix
C ****
      SUBROUTINE KILLGRID(nbase, nftr, nsam, nnofly, Base, FTR, SAM,
     +  NoFly, CTRx, CTRy, CTRz)

      INTEGER i, k, nbase, nftr, nsam, nnofly
      REAL Base(nbase,*), FTR(nftr,*), SAM(nsam,*), NoFly(nnofly,*)
      REAL CTRx, CTRy, CTRz, Ae, hold
      REAL RADS, esqr, e, const, const2, x, z

      Ae = 3443.9
      RADS = 0.0174533
      e = 0.08182
      esqr = e*e
      const = Ae*(1.0-esqr)

C Converts degrees to radians

      DO 2101 i = 1, nbase
         Base(i,2) = RADS * Base(i,2)
         Base(i,3) = RADS * Base(i,3)
 2101 CONTINUE

      DO 2102 i = 1, nftr
         FTR(i,2) = RADS * FTR(i,2)
         FTR(i,3) = RADS * FTR(i,3)
 2102 CONTINUE

      DO 2103 i = 1, nsam
         SAM(i,2) = RADS * SAM(i,2)
         SAM(i,3) = RADS * SAM(i,3)
 2103 CONTINUE

      DO 2104 i = 1, nnofly
         NoFly(i,2) = RADS * NoFly(i,2)
         NoFly(i,3) = RADS * NoFly(i,3)
 2104 CONTINUE

C Generates 'R' vector for all the points

      DO 2105 k = 1, nbase

        const2 = SQRT(1.0-(esqr*(SIN(Base(k,3))*SIN(Base(k,3)))))
      x = (Ae/const2)*COS(Base(k,3))
      z = (const/const2)*SIN(Base(k,3))

      hold = Base(k,2)

      Base(k,2) = x * COS(Base(k,2))
      Base(k,3) = x * SIN(hold)
      Base(k,4) = z

2105  CONTINUE

    DO 2106 k = 1, nftr

      const2 = SQRT(1.0-(esqr*(SIN(FTR(k,3))*SIN(FTR(k,3)))))
      x = (Ae/const2)*COS(FTR(k,3))
      z = (const/const2)*SIN(FTR(k,3))

      hold = FTR(k,2)

      FTR(k,2) = x * COS(FTR(k,2))
      FTR(k,3) = x * SIN(hold)
      FTR(k,4) = z

2106  CONTINUE

    DO 2107 k = 1, nsam

      const2 = SQRT(1.0-(esqr*(SIN(SAM(k,3))*SIN(SAM(k,3)))))

      x = (Ae/const2)*COS(SAM(k,3))
      z = (const/const2)*SIN(SAM(k,3))

      hold = SAM(k,2)

      SAM(k,2) = x * COS(SAM(k,2))
      SAM(k,3) = x * SIN(hold)
      SAM(k,4) = z

2107  CONTINUE

    DO 2108 k = 1, nnofly

      const2 = SQRT(1.0-(esqr*(SIN(NoFly(k,3))*SIN(NoFly(k,3)))))
```

```
        x = (Ae/const2)*COS(NoFly(k,3))
        z = (const/const2)*SIN(NoFly(k,3))

        hold = NoFly(k,2)

        NoFly(k,2) = x * COS(NoFly(k,2))
        NoFly(k,3) = x * SIN(hold)
        NoFly(k,4) = z

 2108   CONTINUE

C Finds the x, y, z coordinates of each constraint point

       WRITE(10,*)'BASES xyz'
       DO 2109 k = 1, nbase

       WRITE(10,*)Base(k,2),Base(k,3),Base(k,4)
 2109   CONTINUE
       WRITE(10,*)'FTRs xyz'
       DO 2110 k = 1, nftr

       WRITE(10,*)FTR(k,2),FTR(k,3),FTR(k,4)
 2110   CONTINUE
       WRITE(10,*)'SAMs xyz'
       DO 2111 k = 1, nsam

       WRITE(10,*)SAM(k,2),SAM(k,3),SAM(k,4)
 2111   CONTINUE
       WRITE(10,*)'NoFlys xyz'
       DO 2112 k = 1, nnofly

       WRITE(10,*)NoFly(k,2),NoFly(k,3),NoFly(k,4)
 2112   CONTINUE

       RETURN
       END

C ***
C This subroutine generates the x,y,z coordinates for
C the candidate surveillance points and places them in
C the CSPT matrix
C ***

      SUBROUTINE FINDCSP(gs, tgs, zcs, zcsqr, cspx, cspy,
  +  cspz, clat, clon, csplat, csplon)

   INTEGER i, j, k, zcs, zcsqr
   REAL gs, tgs, cspx(zcsqr),cspy(zcsqr), cspz(zcsqr)
   REAL const, const2, clat,clon, startlat, startlon
   REAL Ae, e, RADS, esqr, x, z, csplat(zcsqr)
   REAL csplon(zcsqr)

   Ae = 3443.9
   RADS = 0.0174533
   e = 0.08182
   esqr = e*e
   const = Ae*(1.0-esqr)

   clat = clat/RADS
   clon = clon/RADS

   startlat = (FLOAT(NINT(10.0*clat))/10.0)-(tgs/2.0)
   startlon = (FLOAT(NINT(10.0*clon))/10.0)-(tgs/2.0)

   i = 0

   DO 1001 j = 1, zcs

      DO 1002 k = 1, zcs

         i = i + 1
         csplat(i) = startlat + (FLOAT(k-1)*gs)
         csplon(i) = startlon + (FLOAT(j-1)*gs)
C     WRITE(12,*)i,' ',csplat(i),' ',csplon(i)
      csplat(i) = csplat(i) * RADS
      csplon(i) = csplon(i) * RADS

1002    CONTINUE

1001  CONTINUE

      DO 1004 i = 1, zcsqr

         const2 = SQRT(1.0-(esqr*(SIN(csplon(i))
   +           *SIN(csplon(i)))))
         x = (Ae/const2)*COS(csplon(i))
         z = (const/const2)*SIN(csplon(i))
```

```
             cspx(i) = x * COS(csplat(i))
             cspy(i) = x * SIN(csplat(i))
             cspz(i) = z

 1004   CONTINUE

          WRITE(11,*)'CTRLAT= ',clat,' CTRLON= ',clon

          clat = clat * RADS
          clon = clon * RADS

C       WRITE(10,*)'CSPs generated are as follows'
C       WRITE(10,*)'#    X coord        Y coord        Z coord'
C       DO 1003 i = 1, zcsqr
C            WRITE(10,5501)i,cspx(i),cspy(i), cspz(i)
C 1003   CONTINUE
C 5501   FORMAT(I4,2X,F8.2,2X,F8.2,2X,F8.2)

        RETURN
        END
C ***
C This Subroutine eliminates CSPs that do not meet the constraints
C of avoiding enemy SAM's, range from home base limits,
C friendly fighter cover limits, and No Fly restrictions
C ***
        SUBROUTINE ELIMCSP(csptx,cspty,csptz,nbase,nftr,nsam,nnofly,
   +         Base, FTR, SAM, NoFly, ncsp, colind, csplat,csplon)

        INTEGER i, k, nbase, nftr, nsam, nnofly, ncsp
        INTEGER colind(ncsp), SUM
        REAL csptx(*), cspty(*),csptz(*),Base(nbase,5), FTR(nftr,5)
        REAL SAM(nsam,5), NoFly(nnofly,5), A, B, C, test
        REAL csplat(*), csplon(*)

        INTEGER fb,fs,ff,fn, DUM

        fb = 0
        fs = 0
        ff = 0
        fn = 0

        DO 2201 k = 1, ncsp
           colind(k) = 0
 2201   CONTINUE
```

```
      DO 2202 k = 1, ncsp

        DO 2203 i = 1, nbase

          A = (csptx(k)-Base(i,2))**2.0
          B = (cspty(k)-Base(i,3))**2.0
          C = (csptz(k)-Base(i,4))**2.0
        test = SQRT(A+B+C)

          IF (test.LE.Base(i,5)) THEN
            colind(k) = 1
          ENDIF
2203      CONTINUE

          IF (colind(k).LT.1) THEN
             fb = fb + 1
             GOTO 2202
          ENDIF

          DUM = 0

        DO 2204 i = 1, nftr

          A = (csptx(k)-FTR(i,2))**2.0
          B = (cspty(k)-FTR(i,3))**2.0
          C = (csptz(k)-FTR(i,4))**2.0
        test = SQRT(A+B+C)

          IF (test.LE.FTR(i,5)) THEN
               DUM = 1
          ENDIF
2204      CONTINUE
          IF (DUM.EQ.0) THEN
             colind(k) = 0
             ff = ff + 1
             GOTO 2202
          ENDIF

        DO 2205 i = 1, nnofly

          A = (csptx(k)-NoFly(i,2))**2.0
          B = (cspty(k)-NoFly(i,3))**2.0
          C = (csptz(k)-NoFly(i,4))**2.0
```

```
           test = SQRT(A+B+C)

              IF (test.LE.NoFly(i,5)) THEN
                  colind(k) = 0
                  fn = fn + 1
                  GOTO 2202
              ENDIF
2205         CONTINUE

          DO 2206 i = 1, nsam

              A = (csptx(k)-SAM(i,2))**2.0
              B = (cspty(k)-SAM(i,3))**2.0
              C = (csptz(k)-SAM(i,4))**2.0
           test = SQRT(A+B+C)

              IF (test.LE.SAM(i,5)) THEN
                  colind(k) = 0
                  fs = fs + 1
                  GOTO 2202
              ENDIF
2206          CONTINUE
2202     CONTINUE

      SUM = 0
      DO 2207 i = 1, ncsp
         SUM = SUM + colind(i)
2207     CONTINUE

      WRITE(11,*)'# of CSP = ',ncsp
      WRITE(11,*)'The total number of CSP remaining = ',SUM
      WRITE(11,*)'ELIM by base, ftr, SAM, NoFly'
      WRITE(11,*)'BLUE',fb,ff,'  RED',fs,fn
      WRITE(11,*)

      k = 0

      DO 2208 i = 1, ncsp

          IF (colind(i).GT.0) THEN
              k = k + 1
              csptx(k) = csptx(i)
              cspty(k) = cspty(i)
              csptz(k) = csptz(i)
              csplat(k) = csplat(i)
              csplon(k) = csplon(i)
            ENDIF
 2208     CONTINUE

          DO 2209 i = (SUM + 1), ncsp
            csptx(i) = 100000.0
            cspty(i) = 100000.0
            csptz(i) = 100000.0
            csplat(i) = 0.0
            csplon(i) = 0.0
 2209     CONTINUE

          RETURN
          END

C ***
C This subroutine generates the cover matrix
C ***

         SUBROUTINE ZCOVER(csptx,cspty,csptz,tgtx,tgty,tgtz,ntgt,
     +      ncsp,CM,CD)

          INTEGER ntgt, ncsp, i, j
          REAL csptx(ncsp),cspty(ncsp), csptz(ncsp)
          REAL tgtx(ntgt),tgty(ntgt),tgtz(ntgt)
          REAL CM(ntgt,ncsp), covdist, A , B, C, CD
          covdist = CD
          DO 1101 i = 1, ntgt

            DO 1102 j = 1, ncsp

              A = (csptx(j)-tgtx(i))**2.0
              B = (cspty(j)-tgty(i))**2.0
              C = (csptz(j)-tgtz(i))**2.0

             CM(i,j) = SQRT(A+B+C)

           IF (CM(i,j).GT.covdist) THEN
             CM(i,j) = 0.0
           ELSE
               CM(i,j) = 1.0
             ENDIF
 1102        CONTINUE
 1101     CONTINUE

       RETURN
       END
C ***
C This subroutine aggregates the CSPs
C ***

      SUBROUTINE AGGCSP(CM,ntgt,colind,colsum,csptx,cspty,
     +  csptz,ncsp,ncolA,multCSP, csplat, csplon)

      INTEGER ntgt,ncsp,i,ii,j,k,kk, ncspA
      INTEGER colind(ncsp),ncolA, numagg
      REAL CM(ntgt,ncsp), colsum(ncsp), RADS
      REAL csptx(ncsp),cspty(ncsp), csptz(ncsp)
      REAL multCSP(ncsp,ncsp), csplat(*), csplon(*)

      RADS = 0.0174533

      DO 1201 i = 1, ncsp
         colind(i) = 1
 1201 CONTINUE
C ***
C This first part eliminates columns that can see no targets
C ***
      ncolA = 0
      DO 1202 j = 1, ncsp
         colsum(j) = 0.0
         DO 1203 k = 1, ntgt
            colsum(j) = CM(k,j)+colsum(j)
 1203       CONTINUE
         IF (colsum(j).LT.1.0) THEN
            colind(j) = 0
         END IF
         ncolA = colind(j) + ncolA
 1202 CONTINUE

      i = 0
      DO 1205 j = 1, ncsp
         IF (colind(j).EQ.1) THEN
            i = i + 1
            colsum(i) = 0
            DO 1206 k = 1,ntgt
```

```
          test = SQRT(A+B+C)

            IF (test.LE.NoFly(i,5)) THEN
              colind(k) = 0
              fn = fn + 1
              GOTO 2202
            ENDIF
2205        CONTINUE

          DO 2206 i = 1, nsam

            A = (csptx(k)-SAM(i,2))**2.0
            B = (cspty(k)-SAM(i,3))**2.0
            C = (csptz(k)-SAM(i,4))**2.0
          test = SQRT(A+B+C)

            IF (test.LE.SAM(i,5)) THEN
              colind(k) = 0
              fs = fs + 1
              GOTO 2202
            ENDIF
2206        CONTINUE
2202    CONTINUE

        SUM = 0
        DO 2207 i = 1, ncsp
          SUM = SUM + colind(i)
2207    CONTINUE

        WRITE(11,*)'# of CSP = ',ncsp
        WRITE(11,*)'The total number of CSP remaining = ',SUM
        WRITE(11,*)'ELIM by base, ftr, SAM, NoFly'
        WRITE(11,*)'BLUE',fb,ff,'  RED',fs,fn
        WRITE(11,*)

        k = 0

        DO 2208 i = 1, ncsp

          IF (colind(i).GT.0) THEN
            k = k + 1
            csptx(k) = csptx(i)
            cspty(k) = cspty(i)
            csptz(k) = csptz(i)
            csplat(k) = csplat(i)
            csplon(k) = csplon(i)
         ENDIF
2208   CONTINUE

       DO 2209 i = (SUM + 1), ncsp
          csptx(i) = 100000.0
          cspty(i) = 100000.0
          csptz(i) = 100000.0
          csplat(i) = 0.0
          csplon(i) = 0.0
2209   CONTINUE

       RETURN
       END

C ***
C This subroutine generates the cover matrix
C ***

      SUBROUTINE ZCOVER(csptx,cspty,csptz,tgtx,tgty,tgtz,ntgt,
    +      ncsp,CM,CD)

       INTEGER ntgt, ncsp, i, j
       REAL csptx(ncsp),cspty(ncsp), csptz(ncsp)
       REAL tgtx(ntgt),tgty(ntgt),tgtz(ntgt)
       REAL CM(ntgt,ncsp), covdist, A , B, C, CD
       covdist = CD
       DO 1101 i = 1, ntgt

          DO 1102 j = 1, ncsp

             A = (csptx(j)-tgtx(i))**2.0
             B = (cspty(j)-tgty(i))**2.0
             C = (csptz(j)-tgtz(i))**2.0

           CM(i,j) = SQRT(A+B+C)

         IF (CM(i,j).GT.covdist) THEN
            CM(i,j) = 0.0
         ELSE
               CM(i,j) = 1.0
           ENDIF
```

```
 1102         CONTINUE
 1101      CONTINUE

         RETURN
         END
C ***
C This subroutine aggregates the CSPs
C ***

          SUBROUTINE AGGCSP(CM,ntgt,colind,colsum,csptx,cspty,
        +  csptz,ncsp,ncolA,multCSP, csplat, csplon)

          INTEGER ntgt,ncsp,i,ii,j,k,kk, ncspA
          INTEGER colind(ncsp),ncolA, numagg
          REAL CM(ntgt,ncsp), colsum(ncsp), RADS
          REAL csptx(ncsp),cspty(ncsp), csptz(ncsp)
          REAL multCSP(ncsp,ncsp), csplat(*), csplon(*)

          RADS = 0.0174533

          DO 1201 i = 1, ncsp
             colind(i) = 1
  1201  CONTINUE
C ***
C This first part eliminates columns that can see no targets
C ***
          ncolA = 0
          DO 1202 j = 1, ncsp
             colsum(j) = 0.0
             DO 1203 k = 1, ntgt
                colsum(j) = CM(k,j)+colsum(j)
  1203       CONTINUE
             IF (colsum(j).LT.1.0) THEN
                colind(j) = 0
             END IF
             ncolA = colind(j) + ncolA
  1202  CONTINUE

          i = 0
          DO 1205 j = 1, ncsp
             IF (colind(j).EQ.1) THEN
                i = i + 1
                colsum(i) = 0
                DO 1206 k = 1,ntgt
```

```
              CM(k,i) = CM(k,j)
              colsum(i) = colsum(i) + CM(k,i)
1206         CONTINUE
            csptx(i)  = csptx(j)
            cspty(i)  = cspty(j)
            csptz(i)  = csptz(j)
            colind(i) = colind(j)
            csplat(i) = csplat(j)
            csplon(i) = csplon(j)
         END IF
1205  CONTINUE
      ncspA = i

      DO 1207 i = (ncolA+1),ncsp
            csptx(i) = -10000.0
            cspty(i) = -10000.0
            csptz(i) = -10000.0
            colind(i) = 0
            colsum(i) = 0.0
            csplat(i) = 0.0
            csplon(i) = 0.0
         DO 1208 j = 1, ntgt
            CM(j,i) = 0.0
1208        CONTINUE
1207  CONTINUE

C ***
C This second part finds CSPs that cover identical target sets
C and then removes all but one of the identical coverage CSPs
C ***
      WRITE(12,*)'The aggregated CSP List'
      WRITE(12,*)'#  LAT  LON  has #  LAT   LON agg. to it'
      DO 1209 i = 1, (ncolA-1)
       IF (colsum(i).GT.0.0) THEN
        DO 1210 j = (i+1), ncolA

        IF (colsum(i).EQ.colsum(j)) THEN
           DIFF1 = 0.0

           DO 1211 k = 1, ntgt
              DIFF1 = ABS(CM(k,i) - CM(k,j))+DIFF1
1211          CONTINUE

           IF (DIFF1.EQ.0.0) THEN
```

```
            colsum(j) = 0.0
            multCSP(i,j) = 1.0
            colind(j) = 0
    WRITE(12,571)i,(csplat(i)/RADS),(csplon(i)/RADS),
   + j,(csplat(j)/RADS),(csplon(j)/RADS)
         END IF
      END IF
1210     CONTINUE

     END IF
1209  CONTINUE
571  FORMAT(I5,2X,F8.2,2X,F8.2,4X,I5,2X,F8.2,2X,F8.2)
C ***
C Update all variables for eliminated columns, redundant targets
c are maintained
C ***
    i = 0
    ii = 0
    DO 1212 j = 1, ncolA
       IF (colind(j).EQ.1) THEN
         i = i + 1
         colsum(i) = colsum(j)
         colind(i) = colind(j)
         csptx(i)  = csptx(j)
         cspty(i)  = cspty(j)
         csptz(i)  = csptz(j)
         csplat(i) = csplat(j)
         csplon(i) = csplon(j)

         DO 1213 k = 1,ntgt
            CM(k,i) = CM(k,j)
1213       CONTINUE
       ELSE
         ii = ii + 1
         colsum(ncolA+ii) = colsum(j)
         colind(ncolA+ii) = 0
         csptx(ncolA+ii) = csptx(j)
         cspty(ncolA+ii) = cspty(j)
         csptz(ncolA+ii) = csptz(j)
         csplat(ncolA+ii) = csplat(j)
         csplon(ncolA+ii) = csplon(j)

         DO 1214 k = 1, ntgt
```

```
                CM(k,(ncolA+ii)) = CM(k,j)
1214          CONTINUE
          END IF
1212  CONTINUE

      kk = ncolA
      ncolA = i
      numagg = ii

      DO 1215 i = 1, ii
        j = ncolA+i
           colsum(j) = colsum(kk+i)
           colind(j) = colind(kk+i)
           csptx(j) = csptx(kk+i)
           cspty(j) = cspty(kk+i)
           csptz(j) = csptz(kk+i)
           csplat(j) = csplat(kk+i)
           csplon(j) = csplon(kk+i)

           DO 1219 k = 1,ntgt
              CM(k,j) = CM(k,kk+i)
1219          CONTINUE
1215  CONTINUE

      WRITE(11,*)'# CSP agg = ',numagg,'  ncolA = ',ncolA
C       WRITE(10,*)'CSPs Left'
C      WRITE(10,*)'X coord   Y Coord   Z Coord'
C       DO 1218 i = 1, ncolA
C       WRITE(10,*)i,csptx(i),cspty(i),csptz(i)
C 1218  CONTINUE
      WRITE(11,*)'The number of CSPs left is   ',ncolA

      RETURN
      END
C ***
C This subroutine aggregates the targets
C ***
       SUBROUTINE AGGTGT(CM,ncolA,nrowA,rowind,ncsp,rowsum,
     +                 multTGT,TGTNUM,ntgt,TGTVAL,UV)

         INTEGER i, j, k, ntgt, count, UV
         INTEGER rowind(ntgt), ncsp, ncolA, nrowA
         REAL CM(ntgt,ncsp), DIFF2, rowsum(ntgt)
         REAL multTGT(ntgt,ntgt), TGTNUM(ntgt), TGTVAL(ntgt)
        DO 1301 i = 1, ntgt
              rowind(i) = 1
1301  CONTINUE

      IF (UV.NE.1) THEN
         DO 1321 i = 1, ntgt
           TGTVAL(i) = 1.0
1321      CONTINUE
      ENDIF

      DO 1307 i = 1,ntgt
        DO 1308 j = 1, ntgt
          multTGT(i,j) = 0.0
1308     CONTINUE
1307  CONTINUE

C ***
C This part finds all targets with identical cover constraints (the
C same CSPs coverage) and removes all but one of the targets.  Targets
C eliminated this way are stored in the multTGT matrix).
C ***

      DO 1302 i = 1, ntgt
         rowsum(i) = 0.0
         DO 1303 j = 1, ncolA
         rowsum(i) = CM(i,j) + rowsum(i)
1303      CONTINUE
1302  CONTINUE

      WRITE(10,*)
      WRITE(10,*)'Aggregated Target List'
      WRITE(10,*)'Aggregated Target #',
    +  ' Redundant Target #   TGT Value'
      DO 1304 i = 1, (ntgt-1)

        count = 0

        IF (rowind(i).GT.0) THEN

          DO 1305 j = (i+1), ntgt

          IF (rowind(j).GT.0) THEN
```

```
            IF (rowsum(i).EQ.rowsum(j)) THEN

              DIFF2 = 0.0

              DO 1306 k = 1, ncolA

                DIFF2 = ABS(CM(i,k)-CM(j,k))+DIFF2

1306             CONTINUE

              IF (DIFF2.LT.1.0) THEN

                rowind(j) = 0
                TGTVAL(i) = TGTVAL(i) + TGTVAL(j)
                multTGT(i,j) = TGTNUM(j)
           WRITE(10,*)TGTNUM(i),'       ',TGTNUM(j),
   +   '  ',TGTVAL(i)
                count = count + 1

              END IF
            END IF
          END IF
1305         CONTINUE
        END IF
C         IF (rowind(i).GT.0) THEN
C         WRITE(10,*)'Target Value for kept target# ',
C    +      TGTNUM(i),' is = ',TGTVAL(i)
C         WRITE(10,*)'The number of targets agg. on target ',
C    +       TGTNUM(i),' is ',count
C         ENDIF
1304   CONTINUE

C ***
C This section eliminates the redundant targets from the A matrix, and updates
C all variables
C ***
     nrowA = 0
     i = 0

     DO 1309 j = 1, ntgt
        IF (rowind(j).EQ.1) THEN
            i = i + 1
            nrowA = i
            rowsum(i) = rowsum(j)
```

```
              rowind(i) = rowind(j)
              TGTNUM(i) = TGTNUM(j)
              TGTVAL(i) = TGTVAL(j)

              DO 1310 k = 1, ncolA
                 CM(i,k) = CM(j,k)
 1310            CONTINUE
           ENDIF
 1309  CONTINUE
C ***
C This part sets the extra rows in the covermatrix to zero
C ***
       DO 1311 i = (nrowA+1), ntgt
           rowsum(i) = 0.0
           rowind(i) = 0
           TGTVAL(i) = 0.0
           TGTNUM(i) = 0.0
           DO 1312 j = 1, ncolA
              CM(i,j) = 0.0
 1312      CONTINUE
 1311  CONTINUE

       WRITE(10,*)'Covermatrix size (nonzero) ',nrowA,' by ',ncolA
       WRITE(10,*)'Original A matrix size ',ntgt,' by ',ncsp

       RETURN
       END

C ******************************************************
C ******************************************************
C ****** THIS SUBROUTINE RUN'S IGNIZIO'S HEURISTIC *******
C ******************************************************
C ******************************************************

       SUBROUTINE FINDSPT(CM,ncolA,nrowA,colind,rowind,colsum,
      + rowsum,TGTVAL,ASP,Oset,Tj,Sj,Astar,R,EE,ntgt,ncsp,Xj)

       INTEGER i, ii, j, k, ncolA, nrowA, ASP, ntgt, ncsp
       INTEGER colind(*), rowind(*), kk
       REAL colsum(*), rowsum(*), TGTVAL(*)
       REAL CM(ntgt,ncsp), ttv

       INTEGER aspt, Xj, Oset(*), inset, Overrun
      REAL Tj(*), Sj(*), Astar(*)
      REAL R(ntgt,ncsp), EE(*)

C ***
C This sets the cover matrix with the target values, IF
C useVAL = 1, otherwise all targets have equal value of one
C This is set in the target aggregation subroutine
C ***
C *** INITIALIZE ****

      WRITE(11,*)
      WRITE(11,*)'Starting Heuristic'
      WRITE(11,*)'The number of rows left = ', nrowA
      WRITE(11,*)'The number of columns left = ', ncolA
      WRITE(11,*)'The Cover radius = 175 nm'
      WRITE(11,*)'Grid size = 21 degree box'
      WRITE(11,*)'COPs 0.20 degrees apart'
      WRITE(11,*)

      ttv = 0.0
      Xj = 1
      ii = 1
      Overrun = 0
      DO 1480 i = 1, ntgt
         rowsum(i) = 0.0
 1480 CONTINUE

C *** STEP 1 ***

         DO 1401 i = 1, nrowA
            DO 1402 j = 1, ncolA

               CM(i,j) = CM(i,j) * TGTVAL(i)

 1402          CONTINUE
        ttv = ttv + TGTVAL(i)

 1401       CONTINUE
      WRITE(11,*)
      WRITE(11,*)'Total Target Value Covered = ',ttv
      WRITE(11,*)

C *** STEP 2 ***
```

```
      DO 1403 j = 1, ncolA
        Tj(j) = 0.0
        DO 1404 i = 1, nrowA

          Tj(j) = Tj(j) + CM(i,j)

1404      CONTINUE
1403    CONTINUE

      aspt = 1

      DO 1405 j = 2, ncolA
          IF (Tj(j).GT.Tj(aspt)) THEN
              aspt = j
          END IF
1405  CONTINUE

      DO 1406 i = 1, nrowA
          Astar(i) = CM(i,aspt)
          R(i,Xj) = Astar(i)
1406  CONTINUE
      Oset(Xj) = aspt

C *** STEP 3 ****

 990  CONTINUE

      DO 1407 j = 1, ncolA
        inset = 0
        Sj(j) = 0.0
        DO 1408 k = 1, Xj
          IF (j.EQ.Oset(k)) THEN
            inset = 1
          END IF
 1408     CONTINUE
        IF (inset.EQ.0) THEN
          DO 1409 i = 1, nrowA
            Sj(j) = MAX(CM(i,j)-Astar(i),0.0)+Sj(j)
 1409       CONTINUE
        END IF
 1407   CONTINUE

C *** Check for completion ***
```

```
      CHECK = 0
      DO 1410 j = 1, ncolA
         IF (Sj(j).GT.0.0) THEN
            CHECK = 1
         END IF

 1410 CONTINUE

      IF (CHECK.LT.1) THEN

C          CALL TERMINAT()
       WRITE(11,*)'TERMINATES ON ALL Sj <= 0 - STEP 3'
         GOTO 999
      END IF
C ***
      aspt = 1
      DO 1411 j = 2, ncolA
         IF (Sj(j).GT.Sj(aspt)) THEN
            aspt = j
         END IF
 1411 CONTINUE

      Xj = Xj + 1
      Oset(Xj) = aspt

C *** STEP 4 ***

      DO 1412 i = 1, nrowA
         Astar(i) = MAX(Astar(i),CM(i,aspt))
         R(i,Xj) = CM(i,Oset(Xj))
 1412 CONTINUE

C *** Check for infinite loop
      Overrun = Overrun + 1
      IF (Overrun.GT.100) GOTO 999
C ***
      IF (Xj.LT.3) THEN
        CALL PRELIM(CM,Oset,Xj,ncolA,nrowA,ntgt,ncsp,
     +    ttv, TGTVAL, rowind)
      ENDIF

      IF (Xj.EQ.2) GOTO 990

C *** STEP 5 ***
```

```
        DO 1413 k = 1, Xj
C *** zero out the checker - rowsum ***
           DO 1423 kk = 1, nrowA
              rowsum(kk) = 0.0
 1423        CONTINUE

           DO 1414 j = 1, Xj

              IF (j.NE.k) THEN

                 DO 1415 i = 1, nrowA
                    rowsum(i) = MAX(rowsum(i),R(i,j))
 1415               CONTINUE
              END IF
 1414        CONTINUE
           EE(k) = 0.0
           DO 1416 i = 1, nrowA
              EE(k) = EE(k) + (rowsum(i)-astar(i))
 1416        CONTINUE
 1413   CONTINUE
        WRITE(11,*)' EE(k)'
        WRITE(11,*)(NINT(EE(kk)), kk=1, Xj)
        WRITE(11,*)

        aspt = Xj
        DO 1417 k = (Xj-1), 1, -1
           IF (EE(k).GT.EE(aspt)) THEN
              aspt = k
           END IF
 1417   CONTINUE

        CALL PRELIM(CM,Oset,Xj,ncolA,nrowA,ntgt,ncsp,
      + ttv,TGTVAL, rowind)

C *** STEP 6 ***

        IF (aspt.EQ.Xj) THEN
           GOTO 998
        END IF

C *** Sends to step 8 ***

        DO 1418 k = aspt, (Xj-1)
```

```
          Oset(k) = Oset(k+1)
          DO 1419 i = 1, nrowA
             R(i,k) = R(i,(k+1))
1419       CONTINUE
1418   CONTINUE

       Xj = Xj - 1

C *** STEP 7 ***

       DO 1420 i = 1, nrowA
          Astar(i) = R(i,1)
1420   CONTINUE

       DO 1421 j = 2, Xj
          DO 1422 i = 1, nrowA
             Astar(i) = MAX(Astar(i),R(i,j))
1422      CONTINUE
1421   CONTINUE

       GOTO 990

C *** STEP 8 ***

998    CONTINUE

       IF (Xj.LT.ASP) THEN
          GOTO 990
       END IF

C *** STEP 9 ***

C      CALL TERMINAT()

999    CONTINUE

       WRITE(11,*)'Solution Accomplished'
       WRITE(11,*)'Xj = ',Xj
       WRITE(11,*)'The Solution set is columns'
       WRITE(11,*)(Oset(j), j = 1,Xj)

       RETURN
       END
```

```
C ***
C This subroutine save preliminary results of the heuristic
C ***
      SUBROUTINE PRELIM(CM,Oset,Xj,ncolA,nrowA,ntgt,ncsp,
     + ttv,TV, rowind)

      INTEGER i, j, ncolA, nrowA, ntgt, ncsp
      REAL CM(ntgt,ncsp), TV(*), part(30), const
      INTEGER Xj, Oset(*), rowind(ntgt)

      WRITE(11,*)
      WRITE(11,*)'The current solution for Xj = ',Xj
      WRITE(11,*)(Oset(j), j = 1,Xj)
      WRITE(11,*)

      DO 3304 j = 1, nrowA
         rowind(j) = 0
 3304 CONTINUE

      DO 3301 i = 1, Xj
        const = 0.0
        DO 3302 j = 1, nrowA

          IF (rowind(j).EQ.0) THEN
            IF (CM(j,Oset(i)).GT.0.0) THEN

              const = const + TV(j)
              rowind(j) = 1

            ENDIF
          ENDIF

 3302     CONTINUE
        part(i) = const
 3301   CONTINUE

      WRITE(11,*)
      WRITE(11,*)'The total Coverage Value = ',ttv
      WRITE(11,*)'Each solution gives the following % to total'
      WRITE(11,*)'Orbit    Percent     Coverage'
      WRITE(11,*)'Number   Coverage     Value'
      DO 3303 i = 1, Xj
         const = 100.0*part(i)/ttv
         WRITE(11,*)Oset(i), const, part(i)
```

```
3303   CONTINUE

       RETURN
       END
```

APPENDIX C

IRAQ SCENARIO TARGET LIST

TARGET #	TYPE	VALUE	LAT	LON	NAME	COUNT RY
1	Airports	10	35.79	43.09	Q-West	Iraq
2	Airports	10	34.71	43.54	Sahra	Iraq
3	Airports	10	33.84	42.45	Al Asad	Iraq
4	Airports	10	33.34	43.6	Al Taq	Iraq
5	Airports	10	33.24	44.23	Saddam Int.	Iraq
6	Airports	10	33.32	44.38	Baghdad Int.	Iraq
7	Airports	10	30.95	46.08	Tallil	Iraq
8	Airports	10	30.6	46.62	Lajbah-SE	Iraq
9	Airports	10	30.55	47.67	Basrah Intl	Iraq
10	Airports	10	30.57	47.75	Basrah	Iraq
11	Airports	8	36.28	42.42	Afar	Iraq
12	Airports	8	36.33	43.16	Mosel	Iraq
13	Airports	8	35.87	42.13	Sinjar	Iraq
14	Airports	8	35.77	43.26	Q-South	Iraq
15	Airports	8	35.52	44.27	K1	Iraq
16	Airports	8	35.15	44.12	Ashtah	Iraq
17	Airports	8	34.93	44.4	K2	Iraq
18	Airports	8	34.65	43.78	T-East	Iraq
19	Airports	8	34.56	43.67	T-South	Iraq
20	Airports	8	34.18	44.27	Samarra	Iraq
21	Airports	8	34.18	44.75	Salum	Iraq
22	Airports	8	34.95	44.46	Tuz	Iraq
23	Airports	8	33.52	44.25	Al Taji	Iraq
24	Airports	8	33.3	44.51	Rashead	Iraq
25	Airports	8	33.95	44.4	Balad SE	Iraq
26	Airports	8	32.83	40.3	Rutbah	Iraq
27	Airports	8	32.42	41.98	Mudaysis	Iraq
28	Airports	8	32.8	44.15	Karbala	Iraq
29	Airports	8	32.92	44.6	Shayka	Iraq
30	Airports	8	32.5	45.75	UBA-Jarrah	Iraq
31	Airports	8	32.15	46.4	Kut East	Iraq
32	Airports	8	31.95	42.12	Radif	Iraq
33	Airports	8	31.44	43.15	Al Khirr	Iraq
34	Airports	8	32.35	44.88	Shaykh HS	Iraq
35	Airports	8	31.84	46.3	Qalat-W	Iraq
36	Airports	8	31.87	47.1	Amara	Iraq
37	Airports	8	31.47	47.33	Qalat-E	Iraq

38	Airports	8	30.92	43.66	Gralaysan	Iraq
39	Airports	8	30.67	44.6	Salman-N	Iraq
40	Airports	8	30.35	47.1	Rumaylah SW	Iraq
41	Airports	8	30.25	47.48	Mufrash	Iraq
42	Airports	8	30.22	47.66	Safwan	Iraq
43	Airports	8	30.45	47.48	Shaibah-W	Iraq
44	Airports	8	30.47	47.62	Shaibah	Iraq
45	Heliport	6	30.92	46.4	H-Sug-A-S	Iraq
46	Heliport	6	30.33	46.55	Al Bus-NE	Iraq
47	Heliport	6	30.4	46.8	Qalb-A-L	Iraq
48	Heliport	6	30.6	47.4	Rumaylah	Iraq
49	Heliports	6	36.66	42.6	H-Zalah	Iraq
50	Heliports	6	36.5	43.33	Hel-3494	Iraq
51	Heliports	6	36.53	44.35	Bashur	Iraq
52	Heliports	6	35.48	44.33	Kirkuk	Iraq
53	GdnTrps	5	36.33	43.3	I-Army-Hq	Iraq
54	GdnTrps	5	35.48	44.37	I-Army-Hq	Iraq
55	GndTrps	5	30.51	47.77	Army	Iraq
56	GndTrps	5	30.1	46.09	Army	Iraq
57	Bridge	3	35.08	43.57	Bridge	Iraq
58	Bridge	3	34.46	41.94	Bridge	Iraq
59	Bridge	3	34.06	42.37	Bridge	Iraq
60	Bridge	3	33.36	43.78	Bridge	Iraq
61	Bridge	3	33.3	44.35	Bridge	Iraq
62	Bridge	3	33.37	44.44	Bridge	Iraq
63	Bridge	3	32.5	45.81	Bridge	Iraq
64	Bridge	3	32.04	44.42	Bridge	Iraq
65	Bridge	3	32.5	45.8	Bridge	Iraq
66	Bridge	3	31.88	47.12	Bridge	Iraq
67	Bridge	3	30.98	47.48	Bridge	Iraq
68	Bridge	3	30.55	47.8	Bridge	Iraq
69	GdnTrps	3	36.75	42.6	I-Corp-HQ	Iraq
70	GdnTrps	3	36.8	43.4	I-Corp-HQ	Iraq
71	GdnTrps	3	36.6	44.3	I-Corp-HQ	Iraq
72	GndTrps	3	30.42	47.29	Corp	Iraq
73	GndTrps	3	29.88	46.7	Corp	Iraq
74	GndTrps	3	29.48	46.36	Corp	Iraq
75	GndTrps	3	29.89	45.33	Corp	Iraq
76	GndTrps	3	30.49	44.61	Corp	Iraq
77	GndTrps	3	30.83	43.76	Corp	Iraq
78	GndTrps	3	31.66	41.78	Corp	Iraq
79	Heliports	3	31.08	45.73	Darraji	Iraq
80	Heliports	3	34.1	44.32	K3	Iraq

81	Heliports	3	33.07	39.63	He3-NW	Iraq
82	Heliports	3	33.35	40.62	H2	Iraq
83	Heliports	3	33.8	41.45	H1-New	Iraq
84	Heliports	3	32.8	39.33	H-strip	Iraq
85	Heliports	3	32.75	39.6	H3-SE	Iraq
86	Heliports	3	32.9	39.74	H3	Iraq
87	GdnTrps	1	37.05	42.62	I-Div-Hq	Iraq
88	GdnTrps	1	36.95	43.05	I-Div-Hq	Iraq
89	GdnTrps	1	37.05	43.45	I-Div-Hq	Iraq
90	GdnTrps	1	36.9	43.85	I-Div-Hq	Iraq
91	GdnTrps	1	36.75	44	I-Div-Hq	Iraq
92	GdnTrps	1	36.87	44.25	I-Div-Hq	Iraq
93	GdnTrps	1	36.64	44.45	I-Div-Hq	Iraq
94	GdnTrps	1	36.38	42.42	I-Div-Hq	Iraq
95	GdnTrps	1	36.25	44.02	I-Div-Hq	Iraq
96	GndTrps	1	30.35	48.27	DIV	Iraq
97	GndTrps	1	30.445	47.65	DIV	Iraq
98	GndTrps	1	30.25	47.82	DIV	Iraq
99	GndTrps	1	30.11	47.9	DIV	Iraq
100	GndTrps	1	30.15	47.72	DIV	Iraq
101	GndTrps	1	30.18	47.34	DIV	Iraq
102	GndTrps	1	30.1	47.18	DIV	Iraq
103	GndTrps	1	30.24	47.26	DIV	Iraq
104	GndTrps	1	30.6	47.11	DIV	Iraq
105	GndTrps	1	30.47	46.86	DIV	Iraq
106	GndTrps	1	30.65	46.79	DIV	Iraq
107	GndTrps	1	29.92	47.04	DIV	Iraq
108	GndTrps	1	29.8	46.92	DIV	Iraq
109	GndTrps	1	29.75	46.81	DIV	Iraq
110	GndTrps	1	29.66	46.71	DIV	Iraq
111	GndTrps	1	29.5	46.62	DIV	Iraq
112	GndTrps	1	29.43	46.67	DIV	Iraq
113	GndTrps	1	29.3	46.52	DIV	Iraq
114	GndTrps	1	29.4	46	DIV	Iraq
115	GndTrps	1	29.48	45.88	DIV	Iraq
116	GndTrps	1	29.55	45.57	DIV	Iraq
117	GndTrps	1	29.61	45.31	DIV	Iraq
118	GndTrps	1	31.6	44.6	DIV	Iraq
119	GndTrps	1	31.74	44.32	DIV	Iraq
120	GndTrps	1	32.01	44.36	DIV	Iraq
121	GndTrps	1	30.24	44.54	DIV	Iraq
122	GndTrps	1	30.442	44.56	DIV	Iraq
123	GndTrps	1	30.6	44.58	DIV	Iraq

124	GndTrps	1	30.55	43.76	DIV	Iraq
125	GndTrps	1	30.8	43.92	DIV	Iraq
126	GndTrps	1	30.85	43.63	DIV	Iraq
127	GndTrps	1	30.82	43.38	DIV	Iraq
128	GndTrps	1	32.08	42.26	DIV	Iraq
129	GndTrps	1	31.8	42.1	DIV	Iraq
130	GndTrps	1	31.445	41.62	DIV	Iraq
131	GndTrps	1	31.58	41.52	DIV	Iraq
132	GndTrps	1	31.62	41.35	DIV	Iraq
133	GndTrps	1	32.26	39.8	DIV	Iraq
134	GndTrps	1	33.05	40.3	DIV	Iraq
135	Airport	7	29.4	47.55	Al Salem	Kuwait
136	Airport	7	29.25	47.98	Kuwait Intl	Kuwait
137	Airport	7	29.05	47.8	Abdaliyah	Kuwait
138	Airport	7	28.93	47.79	Jaber	Kuwait
139	GndTrps	3	29.92	47.95	BDE	Kuwait
140	GndTrps	3	29.95	47.67	BDE	Kuwait
141	GndTrps	3	29.75	47.75	BDE	Kuwait
142	GndTrps	3	29.9	47.43	BDE	Kuwait
143	GndTrps	3	29.85	47.26	BDE	Kuwait
144	GndTrps	3	29.75	47.42	BDE	Kuwait
145	GndTrps	3	29.52	47.1	BDE	Kuwait
146	GndTrps	3	29.25	46.95	BDE	Kuwait
147	GndTrps	3	29.3	47.18	BDE	Kuwait
148	GndTrps	3	29.27	47.54	BDE	Kuwait
149	GndTrps	3	29.42	47.68	BDE	Kuwait
150	GndTrps	3	29.28	47.88	BDE	Kuwait
151	Airport	12	24.72	46.7	Riyadh Mil	SA
152	Airport	12	24.88	46.8	King Khalid	SA
153	Airport	9	27.91	45.51	KKMC	SA
154	Airport	7	30.9	41.11	Badanah	SA
155	Airport	7	29.67	43.46	Rafha	SA
156	Airport	7	27.96	45.54	Al Bat	SA
157	Airport	7	28.41	46.11	Hafar	SA
158	Airport	7	28.1	48.58	Mishab	SA
159	Airport	7	27.9	48.6	Tanajib	SA
160	GndTrps	3	28.1	48.55	BDE	SA
161	GndTrps	3	28.45	48.5	BDE	SA
162	GndTrps	3	28.46	48.3	BDE	SA
163	GndTrps	3	28.91	47.33	BDE	SA
164	GndTrps	3	28.85	47.46	BDE	SA
165	GndTrps	3	28.7	47.35	BDE	SA
166	GndTrps	3	28.13	47.34	BDE	SA

167	GndTrps	3	28.9	46.25	BDE	SA
168	GndTrps	3	28.87	46.42	BDE	SA
169	GndTrps	3	28.75	46.3	BDE	SA
170	GndTrps	3	28.26	46.05	BDE	SA
171	GndTrps	3	28.45	45.95	BDE	SA
172	GndTrps	3	28.44	45.75	BDE	SA
173	GndTrps	3	28.2	45.42	BDE	SA
174	GndTrps	3	28.77	45.75	BDE	SA
175	GndTrps	3	28.72	45.45	BDE	SA
176	GndTrps	3	28.69	45.2	BDE	SA
177	GndTrps	3	28.9	44.71	BDE	SA
178	GndTrps	3	29.06	44.5	BDE	SA
179	GndTrps	3	29.1	44.7	BDE	SA
180	GndTrps	3	29.56	43.6	BDE	SA
181	GndTrps	3	29.54	43.82	BDE	SA
182	GndTrps	3	29.8	43.5	BDE	SA
183	GndTrps	3	29.75	43.7	BDE	SA
184	GndTrps	3	29.7	43.38	BDE	SA
185	GndTrps	3	30.75	42.4	BDE	SA
186	GndTrps	3	30.72	42.61	BDE	SA
187	GndTrps	3	30.65	42.75	BDE	SA
188	GndTrps	3	30.21	42.27	BDE	SA
189	GndTrps	3	31.02	41.08	BDE	SA
190	GndTrps	3	31.08	41.26	BDE	SA
191	GndTrps	3	31.11	41.12	BDE	SA
192	GndTrps	3	31.19	40.83	BDE	SA
193	GndTrps	3	31.45	39.75	BDE	SA
194	GndTrps	3	31.9	39.68	BDE	SA
195	GdnTrps	3	37.35	42.5	US Army	Turkey
196	GdnTrps	3	37.6	42.45	US Army	Turkey
197	GdnTrps	3	37.45	42.7	US Army	Turkey
198	GdnTrps	3	37.5	42.9	US Army	Turkey
199	GdnTrps	3	37.5	43.2	US Army	Turkey
200	GdnTrps	3	37.42	43.55	US Army	Turkey
201	GdnTrps	3	37.47	43.6	US Army	Turkey
202	GdnTrps	3	37.51	44	US Army	Turkey
203	GdnTrps	3	37.15	44.3	US Army	Turkey
204	GdnTrps	3	37.18	44.5	US Army	Turkey
205	GdnTrps	3	37.5	44.27	US Army	Turkey

APPENDIX D

IRAQ SCENARIO RESTRICTIONS LIST

Notes; SA - Saudi Arabia, Gdn Trps - Ground Troops, BDE - Brigade, DIV - Division, I-Corp-HQ - Iraqi Corp Headquarters, I-Army-HQ - Iraqi Army Headquarters

HOME BASES

Number	LAT	LON	Range	Name
1	37.88	32.55	500	Konya
2	24.88	46.8	550	King Khalid

SAMS

Type	Number	LAT	LON	Lethal Range
SA-10	1	35.48	44.37	85
SA-10	2	30.47	47.55	85
SA-10	3	33.06	40.28	85
SA-10	4	31.25	45.58	85
SA-10	5	33.24	44.23	85
SA-5	6	33.95	44.40	120
SA-5	7	33.32	44.38	120
SA-5	8	32.00	45.00	120
SA-5	9	33.82	42.52	120
SA-6	10	36.75	42.60	30
SA-6	11	36.80	43.40	30
SA-6	12	36.60	44.30	30
SA-6	13	36.33	43.30	30
SA-6	14	30.42	47.29	30
SA-6	15	29.88	46.70	30
SA-6	16	29.48	46.36	30
SA-6	17	29.89	45.33	30
SA-6	18	30.10	46.09	30
Red FTR	19	33.24	44.23	75
Red FTR	20	36.33	43.16	75
Red FTR	21	33.35	40.62	75
Red FTR	22	32.50	45.75	75

FRIENDLY FTR COVER

Number	LAT	LON	Range	Name
1	28.38	36.65	350	Tabuk
2	27.91	45.51	300	KKMC
3	24.88	46.80	350	King Khalid
4	28.93	47.79	300	Jaber

5	37.23	37.34	350	Incirlik

NO FLY ZONES (Political Boundaries)

Type	Number	LAT	LON	Range	Country
Border	1	32.5	39.40	35	Jordan
Border	2	35.00	39.00	100	Syria
Border	3	36.00	40.00	70	Syria
Border	4	34.42	48.75	150	Iran
Border	5	31.50	50.75	150	Iran
Border	6	32.45	48.40	50	Iran
Border	7	35.42	47.00	60	Iran
Border	8	37.25	46.60	100	Iran

APPENDIX E

FAR EAST SCENARIO TARGET LIST

TARGET #	TYPE	VALUE	LAT	LON	NAME	COUNTRY
1	Island	10	26.3	119.92	Matsu	Taiwan
2	Island	10	24.48	118.42	Quemoy	Taiwan
3	Grd Trps	3	25.17	121.72	DIV-HQ	Taiwan
4	Grd Trps	3	25.21	121.4	DIV-HQ	Taiwan
5	Grd Trps	3	24.83	120.95	DIV-HQ	Taiwan
6	Grd Trps	3	24.17	120.5	DIV-HQ	Taiwan
7	Grd Trps	3	24.22	120.31	DIV-HQ	Taiwan
8	Grd Trps	3	25.07	121.48	Army HQ	Taiwan
9	Grd Trps	3	23	120.2	DIV-HQ	Taiwan
10	Grd Trps	3	22.62	120.33	DIV-HQ	Taiwan
11	Island	6	23.51	119.6	Makung	Taiwan
12	Airports	15	25.1	121.5	Taipei	Taiwan
13	Airports	15	25.16	121.7	Chilung	Taiwan
14	Airports	15	24.2	120.6	Taichung	Taiwan
15	Airports	12	23.02	120.22	Tainan	Taiwan
16	Airports	15	22.6	120.35	Kaohsung	Taiwan
17	US CV	50	26	121	CVN BG	US Navy
18	NavBase	7	23.3	116.81	Swanton	China
19	NavBase	10	24.5	118.11	Amoy	China
20	NavBase	12	24.97	118.71	Q-C	China
21	NavBase	12	26.08	119.47	Foochow	China
22	NavBase	7	28	120.64	Wenchow	China
23	NavBase	7	30.32	120.14	Hangchow	China
24	NavBase	10	31.4	121.52	Shanghai	China
25	NavBase	8	32.1	120.9	Nuntang	China
26	Airports	17	25.45	119.03	FTR Bases	China
27	Airports	15	24.61	117.7	FTR Bases	China
28	Airports	17	25.11	118.72	FTR Bases	China
29	Airports	13	27.2	119.87	FTR Bases	China
30	Airports	13	24.12	117.47	FTR Bases	China
31	Airports	11	26.6	118.14	FTR Bases	China
32	Airports	9	23.49	116.79	FTR Bases	China
33	Airports	9	28.02	120.62	FTR Bases	China
34	Airports	9	29.87	121.5	FTR Bases	China
35	Airports	9	31.11	121.42	FTR Bases	China
36	Airports	9	32.07	120.95	FTR Bases	China
37	Airports	9	37.52	121.25	FTR Bases	China
38	Airports	9	36.97	120.82	FTR Bases	China

39	Airports	9	36.21	120.5	FTR Bases	China
40	Airports	9	41.75	123.42	FTR Bases	China
41	Bridges	2	40.1	124.51	Bridges	China
42	Bridges	2	40.44	124.79	Bridges	China
43	Bridges	2	41.17	126.12	Bridges	China
44	Bridges	2	41.37	128.14	Bridges	China
45	NavBase	5	34.87	119.28	NavBase	China
46	NavBase	5	36.11	120.22	Tsingtao	China
47	NavBase	5	38.87	121.55	Dairen	China
48	NavBase	5	38.84	121.18	Port Arthur	China
49	Straights	2	37.4	122	Straights	China
50	Straights	2	37.6	122	Straights	China
51	Straights	2	37.8	122	Straights	China
52	Straights	2	38	122	Straights	China
53	Straights	2	38.2	122	Straights	China
54	Straights	2	38.4	122	Straights	China
55	Straights	2	38.6	122	Straights	China
56	Straights	2	38.8	122	Straights	China
57	Straights	2	39	122	Straights	China
58	Grd Trps	1	26.77	120.11	DIV-HQ	China
59	Grd Trps	1	26.71	119.91	DIV-HQ	China
60	Grd Trps	1	26.32	119.9	DIV-HQ	China
61	Grd Trps	2	26.94	119.82	CORP-HQ	China
62	Grd Trps	1	26.51	119.55	DIV-HQ	China
63	Grd Trps	2	26.17	119.56	CORP-HQ	China
64	Grd Trps	3	26.03	119.47	Army HQ	China
65	Grd Trps	1	25.98	119.7	DIV-HQ	China
66	Grd Trps	1	25.5	119.81	DIV-HQ	China
67	Grd Trps	1	25.38	119.11	DIV-HQ	China
68	Grd Trps	1	25.14	118.82	DIV-HQ	China
69	Grd Trps	1	25	118.85	DIV-HQ	China
70	Grd Trps	2	24.97	118.65	CORP-HQ	China
71	Grd Trps	1	24.82	118.62	DIV-HQ	China
72	Grd Trps	1	24.6	118.59	DIV-HQ	China
73	Grd Trps	1	24.76	118.17	DIV-HQ	China
74	Grd Trps	1	24.58	118.34	DIV-HQ	China
75	Grd Trps	1	24.55	118.13	DIV-HQ	China
76	Grd Trps	1	24.27	118.14	DIV-HQ	China
77	Grd Trps	2	24.29	117.83	CORP-HQ	China
78	Grd Trps	3	24.66	117.92	Army HQ	China
79	Grd Trps	1	24.51	117.62	DIV-HQ	China
80	Grd Trps	2	38.6	128.14	CORP-HQ	North Korea
81	Grd Trps	2	38.51	128.09	CORP-HQ	North Korea
82	Grd Trps	1	38.61	128	DIV-HQ	North Korea

83	Grd Trps	2	38.58	127.89	CORP-HQ	North Korea
84	Grd Trps	3	38.62	127.87	Army HQ	North Korea
85	Grd Trps	2	39.08	127.5	CORP-HQ	North Korea
86	Grd Trps	3	38.71	127.42	CORP-HQ	North Korea
87	Grd Trps	2	38.59	127.5	CORP-HQ	North Korea
88	Grd Trps	2	38.58	127.24	CORP-HQ	North Korea
89	Grd Trps	2	38.27	126.99	CORP-HQ	North Korea
90	Grd Trps	2	38.11	126.87	CORP-HQ	North Korea
91	Grd Trps	3	38.2	126.85	Army HQ	North Korea
92	Grd Trps	2	37.95	126.5	CORP-HQ	North Korea
93	Grd Trps	3	38.09	126.47	Army HQ	North Korea
94	Grd Trps	2	38.2	126.45	CORP-HQ	North Korea
95	Grd Trps	2	37.93	126.11	CORP-HQ	North Korea
96	Grd Trps	2	38.91	125.87	CORP-HQ	North Korea
97	Airports	15	39.03	125.67	FTR Bases	North Korea
98	Airports	13	39.13	127.4	FTR Bases	North Korea
99	Airports	13	38.82	125.42	FTR Bases	North Korea
100	Airports	13	38.5	125.69	FTR Bases	North Korea
101	Airports	13	38.2	125.82	FTR Bases	North Korea
102	Airports	11	39.6	125.62	FTR Bases	North Korea
103	Airports	11	39.89	127.55	FTR Bases	North Korea
104	Airports	11	40.6	129	FTR Bases	North Korea
105	Airports	11	41.8	129.61	FTR Bases	North Korea
106	Airports	17	37.51	126.92	Seoul	South Korea
107	Airports	15	35.95	126.63	Kunsan	South Korea
108	Airports	15	35.91	128.58	Taegu	South Korea
109	Airports	15	35.11	128.97	Pusan	South Korea
110	Airports	13	36.32	127.47	Tarjon	South Korea
111	Airports	13	35.12	126.95	Kwargju	South Korea
112	Airports	10	38.5	129.4	Ulson	South Korea
113	Airports	10	34.88	126.41	Mokpo	South Korea
114	Airports	13	37.42	127.97	Wonju	South Korea
115	Airports	13	37.38	127.01	Suwon	South Korea
116	Grd Trps	7	37.71	127.06	US DIV HQ	US Army
117	Grd Trps	2	38.42	128.44	DIV-HQ	South Korea
118	Grd Trps	2	38.26	128.17	DIV-HQ	South Korea
119	Grd Trps	3	38.21	128.6	CORP-HQ	South Korea
120	Grd Trps	2	38.2	127.86	DIV-HQ	South Korea
121	Grd Trps	2	38.17	127.5	DIV-HQ	South Korea
122	Grd Trps	3	37.9	127.81	CORP-HQ	South Korea
123	Grd Trps	2	38.13	127.15	DIV-HQ	South Korea
124	Grd Trps	2	38.03	127.03	DIV-HQ	South Korea
125	Grd Trps	2	37.87	126.9	DIV-HQ	South Korea
126	Grd Trps	2	37.6	126.65	DIV-HQ	South Korea

127	Grd Trps	3	37.51	126.57	CORP-HQ	South Korea
128	Grd Trps	5	37.5	127	Army HQ	South Korea
129	Grd Trps	3	37.45	127.13	CORP-HQ	South Korea
130	Airports	20	26.24	127.75	Kadena	Japan
131	Airports	5	33.17	129.75	Sasebo	Japan
132	Airports	1	14.8	121.2	Manila	Phillipines
133	Airports	10	25.1	121.5	Taipei	Taiwan

APPENDIX F

FAR EAST SCENARIO RESTRICTIONS LIST

Home Bases

Number	LAT	LON	Range	Name
1	26.24	127.75	400	Kadena
2	33.17	129.75	400	Sasebo
3	14.8	121.2	600	Manila
4	25.1	121.5	200	Taipei

SAMS

Type	Number	LAT	LON	Lethal Range
SA-5	1	41.75	129.62	110
SA-5	2	40.4	128.83	110
SA-5	3	39.32	127.3	110
SA-5	4	39.12	125.75	110
SA-6	5	38.62	128.1	30
SA-6	6	38.57	127.4	30
SA-6	7	38.22	126.87	30
SA-6	8	37.99	126.47	30
SA-10	9	40	127.62	85
SA-10	10	39.04	127.75	85
SA-10	11	39.72	125.51	85
SA-10	12	38.74	125.3	85
SA-10	13	38.02	125.78	85

Fighter Bases

Type	Number	LAT	LON	Range	Country
Airports	1	25.1	121.5	300	Taiwan
Airports	2	25.16	121.7	300	Taiwan
Airports	3	24.2	120.6	300	Taiwan
Airports	4	23.02	120.22	300	Taiwan
Airports	5	22.6	120.35	300	Taiwan
Airports	6	37.51	126.92	200	South Korea
Airports	7	35.95	126.63	200	South Korea
Airports	8	35.91	128.58	200	South Korea
Airports	9	35.11	128.97	200	South Korea
Airports	10	36.32	127.47	200	South Korea
Airports	11	35.12	126.95	200	South Korea
Airports	12	38.5	129.4	200	South Korea

Airports	13	34.88	126.41	200	South Korea
Airports	14	37.42	127.97	200	South Korea
Airports	15	37.38	127.01	200	South Korea
CV	16	26	121	250	US Navy

NO FLY ZONES

Type	Number	LAT	LON	Range	Country
Border	1	30	112	600	China
Border	2	39.5	117	400	China
Border	3	35	114	400	China

APPENDIX G

Example FORTRAN Program Output.

The program outputs the results to four files. Each file output is designed for easy importation to MS Excell for spreadsheet analysis.

File 1

The aggregated CSP List

```
 #     LAT     LON     has #  LAT     LON    agg. to it
  3   21.90   119.80     4   22.10   119.80
  3   21.90   119.80     5   22.30   119.80
  3   21.90   119.80    10   22.10   120.00
  3   21.90   119.80    11   22.30   120.00
  6   22.50   119.80     7   22.70   119.80
  6   22.50   119.80     8   22.90   119.80
  9   21.90   120.00    17   22.10   120.20
  9   21.90   120.00    18   22.30   120.20
 25    21.90   120.40    26   22.10   120.40
 28    22.50   120.40    29   22.70   120.40
 30    22.90   120.40    31   23.10   120.40
 30    22.90   120.40    32   23.30   120.40
... etc...
```

File 2 - This file presents the results of running the heuristic

```
FAR EAST SCENARIO - TGT VALUES ON

CTRLAT=    32.3529 CTRLON=    123.075
# of CSP =  11236
The total number of CSP remaining =  2963
ELIM by base,    ftr,          SAM,  NoFly
BLUE    3045  2702   RED  462   2064
 # CSP agg =  1980
The number of CSPs left is    753

Starting Heuristic
The number of rows left =  130
The number of columns left =  753
The Cover radius = 175 nm
Grid size = 21 degree box
COPs 0.20 degrees apart
```

```
Total Target Value Covered =    846.000

The current solution for Xj =  2
 36  525

... etc...

TERMINATES ON ALL Sj <= 0 - STEP 3

Solution Accomplished
 Xj =   6

 The Solution set is columns
  525  2  615  91  388  271

  The LAT and LON of the solution are
   #        LAT         LON
  525      35.9000     127.400
  2        22.1000     119.600
  615      37.5000     129.000
  91       32.1000     121.800
  388      24.7000     125.000
  271      35.9000     123.400

 The following targets were not covered
     LAT        LON        Value
   41.7500   123.420    12.0000

Each solution gives the following % to total
 Orbit      Percent      Coverage
 Number    Coverage     Value
  525        36.0520      305.000
  2          41.1348      348.000
  615          2.83688    24.0000
  91         12.6478      107.000
  388          2.36407     20.0000
  271          3.54610     30.0000
 Target Coverage by each Orbit Point

 TARGET  TARGET  TIMES          Orbit Points
 NUMBER  VALUE  COVERED        In Coverage
 ------------------------------
  1              10.0         1           0    2    0    0    0    0
  2              10.0         1           0    2    0    0    0    0
```

```
3        3.0     1       0    2    0    0    0    0
4        3.0     1       0    2    0    0    0    0
5        3.0     1       0    2    0    0    0    0
```

... etc...

```
125      2.0     2       525   0   615   0    0    0
126      3.0     2       525   0   615   0    0    0
127      5.0     2       525   0   615   0    0    0
128      3.0     2       525   0   615   0    0    0
129     20.0     1       0     0    0    0   388   0
130      5.0     2       525   0   615   0    0    0
```

File 3

```
Aggregated Target List
Aggregated Target #        Redundant Target #   TGT Value
   12.0000                  133.000              25.0000
   40.0000                 43.0000               11.0000
   40.0000                 132.000               12.0000

Covermatrix size (nonzero)  130 by   753
Original A matrix size   133 by   11236
```

File 4

```
The final list of COPs considered
  #      LAT         LON     VALUE
  1    21.9000    119.600    339.000
  2    22.1000    119.600    348.000
  3    21.9000    119.800    332.000
  4    22.5000    119.800    348.000
  5    21.9000    120.000    321.000
```

... etc ...

```
 750    22.5000    119.800    348.000
 751    21.9000    119.800    332.000
 752    22.1000    119.600    348.000
 753    21.9000    119.600    339.000
```

Bibliography

1. Adams, W.P. and Sherali, H.D. "Linearization and an Algorithm for Zero-One Mixed-Integer Programming Problems," Operations Research. 38(2) 1990, 217 - 226.

2. Bate, R., Mueller, D., and White, J., Fundamentals of Astrodynamics. New York: Dover Publications, Inc., 1971, Chapter 2.

3. Chan, Yupo. "Real-Time Information and Transportation Decisions: An Analysis of Spatial Data," Military Operations Research. Winter 1996, 23 - 48.

4. Church, R and ReVelle, C. "The Maximal Covering Location Problem," Papers of the Regional Science Association. Vol. 32, 1974, 101-118.

5. Current, J.R. and Schilling D.A. "Analysis of errors due to demand data aggregation in the set covering and maximal covering location problems," Geographical Analysis. Vol. 22, 1990, 116-126.

6. Current, J.R. and Schilling, D.A. "Elimination of Source A and B Errors in the *p*-Median Location Problems," Geographical Analysis. Vol. 19, 1987, 95-110.

7. Francis, R.L. and White, J.A. Facility Layout and Location. New Jersey: Prentice-Hall, Inc., 1974, Chapter 10.

8. Glover, Fred and Woolsey, Eugene. "Converting the 0-1 Polynomial Programming Problem to a 0-1 Linear Program," Operations Research. 22, 1982, 180 - 182.

9. Gunawardane, Gamini. "Dynamic Versions of Set Covering Type Public Facility Location Problems," European Journal of Operations Research. Vol. 10, Number 2, Jun 1982.

10. Hall, Nicholas G. and Hochbaum, Dorit S. "The Multicovering Problem," European Journal of Operations Research. Vol. 62, Number 3, Nov 9, 1992, 323 - 339.

11. Hillsman, E. and Rhoda, R. "Errors in Measuring Distances from Populations to Service Centers," Annals of the Regional Science Association. Vol. 12, 1978, 74-88.

12 Ignizio, J.P. A Heuristic Solution to Generalized Covering Problems. PhD dissertation. Virginia Polytechnic Institute and State University, Blacksburg, Virginia, 1971.

13. Love, R., Morris, J., and Wesolowsky, G. Facilities Location Models & Methods. New York: Elsevier Science Publishing Co., Inc. 1988, Chapters 1, 4-8.

14. Mehrez, A. and Stulman, A. "An Extended Continuous Maximal Covering Location Problem with Facility Placement," Computers & Operations Research. Vol. 11, No. 1, 1984, 19 - 23.

15. Mehrez, A. and Stulman, A. "The Maximal Covering Location Problem with Facility Placement on the Entire Plane," Journal of Regional Science. Vol. 22, No. 3, 1982, 361 - 365.

16. Mirchandani, P.B. and Francis, R.L. Discrete Location Theory. New York: John Wiley and Sons, Inc., 1990, Chapters 1, 6-8.

17. Schilling, D., Jayaraman, V., and Barkhi, R. "A Review of Covering Problems in Facility Location," Location Science. Vol. 1, No. 1, 1993, 25-55.

18. Watters, L. "Reduction of Integer Polynomial Programming Problems to Zero-One Linear Programming Problems," Operations Research. 15, 1967, 1171 - 1174.

19. White, J. and Case, K. "On Covering Problems and the Central Facilities Location Problem," Geographical Analysis. Vol. 6, 1974, 281-293.

20. Young, H. and Taylor, J. "Gallery of USAF Weapons," Air Force Magazine. May 1996, 131-148.

Vita

Major Douglas E. Fuller[redacted]. He graduated from Western Reserve Academy in 1980 and entered undergraduate studies at the United States Air Force Academy in Colorado Springs, Colorado. He graduated with a Bachelor of Science degree in Astronautical Engineering in May 1984. He received his commission on 30 May 1984 and proceeded to undergraduate pilot training (UPT) at Columbus AFB, Mississippi.

UPT led to his selection to the B-52 Stratofortress. His second assignment was to Carswell AFB, Texas. There he held many jobs culminating in assignment to the Chief of Standardization/Evaluation Bomber Branch. He also earned a Master of Science in Aerospace Engineering from the University of Texas at Arlington in 1992. He then spent three years flying E-3A AWACS at Geilenkirchen AB, Germany. In August 1995, he entered the Graduate School of Engineering, Air Force Institute of Technology.

[redacted]

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

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE February 1997	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE

Optimizing Airborne Area
Surveillance Asset Placement

5. FUNDING NUMBERS

6. AUTHOR(S)

Douglas E. Fuller, Major, USAF

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

Air Force Institute of Technology/ENS
2750 P Street
Wright-Patterson AFB, Ohio 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GOA/ENS/97-10

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

HQ AF/XOOC (Checkmate)
Pentagon ADM, VA 20330

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for Public Release; Distribution is Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Currently there is no automated planning tool for the optimum positioning of USAF area surveillance assets for a theater-level campaign. This research seeks to find the optimum or near optimum placement of the limited USAF airborne surveillance assets against a theater-level target set. The problem of finding the optimum orbit points can be modeled as a classic maximal covering location problem (MCLP). Operational constraints on the placement of surveillance aircraft can be handled by preprocessing the potential orbit points to eliminate infeasible orbit points. Heavy emphasis is placed on preprocessing the data to reduce problem size and hence solution time. The aggregation of both the potential orbit points and targets was accomplished without loss of locational information. An existing heuristic was used to find a solution in a very short time. The heuristic finds the optimum orbit points for the available aircraft and any alternate solutions. Allocation decisions can then be accomplished.

14. SUBJECT TERMS

Maximal Covering Location Problem, Location Problem, Integer Programming, Aggregation, Integer Heuristics, Set Covering, Airborne Radar Positioning

15. NUMBER OF PAGES

16. PRICE CODE

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