Fall 2017

Implementation of a Multi-Criteria Decision Analysis (MCDA) Toolkit to Aid in Ranking Naval Mission Vessel Combinations With Uncertainty

Andrew R. Miller
Old Dominion University

IMPLEMENTATION OF A MULTI-CRITERIA DECISION ANALYSIS (MCDA) TOOLKIT TO AID IN RANKING NAVAL MISSION VESSEL COMBINATIONS WITH UNCERTAINTY

by

Andrew R. Miller
B.S. May 2009, Old Dominion University
M.E. May 2012, Old Dominion University

A Doctoral Project Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF ENGINEERING

SYSTEMS ENGINEERING

OLD DOMINION UNIVERSITY
December 2017

Approved by:

____
Resit Unal (Chairman)

____
Charles Keating (Member)

____
Adrian Gheorghe (Member)

____
Ariel Pinto (Member)

ABSTRACT

IMPLEMENTATION OF A MULTI-CRITERIA DECISION ANALYSIS (MCDA) TOOLKIT TO AID IN RANKING NAVAL MISSION VESSEL COMBINATIONS WITH UNCERTAINTY

Andrew R. Miller
Old Dominion University, 2017
Chairman: Dr. Resit Unal

United States (U.S.) military bases have largely been constructed outside of the contiguous United States (OCONUS) due to the need of close support logistics for conflicts and wars. In contrast, military bases within the contiguous United States (CONUS) have been constructed mostly due to economic and other related monetary factors. In addition to monetary concerns for the placement of military bases (specifically naval bases), there exists tactical, environmental, cultural, climate, logistical, and geographical issues that need to be fully considered before deciding on a naval installation location and the vessels to be stationed there. I will present a new toolkit to aid in the decision making process for placing naval vessels to maximize their strategic advantage—while reducing and managing risks—increasing the ability to protect and deter unforeseen threats and—if necessary—fight a future war while adhering to the Department of Defense’s (DoD) maritime strategy. The modification of a current integer linear program by introducing the Elimination Et Choix Traduisant la Realité (Electre) III MCDA model will be used to simulate a variety of naval fleet placement factors, weights, and decision maker (DM) preferences to aid in selection of mission scenarios.

This project is dedicated to my dad who always pushed me to be better and not give up on my dreams.

NOMENCLATURE

a_i	Alternative *i*
AOR	Area of responsibility
C(a, b)	Overall concordance index for alternatives *a* and *b*
CG	Guided Missile Cruiser
$c_j(a, b)$	Individual concordance index for alternatives *a* and *b* for criteria *j*
COA	Course of Action
CONUS	Contiguous United States
CSV	Comma-separated value (file)
CVN	Aircraft Carrier
DDG	Guided Missile Destroyer
$D_j(a, b)$	Individual discordance index for alternatives *a* and *b* for criteria *j*
DM	Decision Maker
DoD	Department of Defense
ELECTRE	Elimination Et Choix Traduisant la Realité
g_j	Criterion of alternative *j*
$g_j(a_i)$	Evaluation of alternative a_i on the criterion g_j
GUI	Graphical user interface
KML	Keyhole Markup Language
LCS	Littoral Combat Ship
MCDA	Multi-Criteria Decision Analysis
MCM	Mine Countermeasures Ship

nm	Nautical miles
NMP	Navy Mission Planner
NPS	Naval Postgraduate School
OAT	One-factor-at-a-time (sensitivity analysis)
OCONUS	Outside the Contiguous United States
p_j	Preference threshold of alternative j
q_j	Indifference threshold of alternative j
$S(a, b)$	Credibility score between alternatives a and b
$s(\lambda_0)$	Discrimination threshold at the maximum level of outranking λ_0
SSBN	Ballistic Missile Submarine
SSGN	Guided Missile Submarine
SSN	Fast Attack Submarine
v_j	Veto threshold of alternative j
WOP	Waypoints Operational Planner
λ_0	The maximum value of $S(a, b)$
λ_1	Cutoff level of the largest outranking score just less than the maximum minus a discrimination threshold

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

CHAPTER 1

INTRODUCTION

The complex global environment in which decision makers (DMs) must evaluate dynamic information to accurately plan and coordinate Naval missions requires tools that can aid in making these decisions by allowing the input of varying levels of data (from minute to imprecise or vague). The objective of a decision making tool is to be able to accept a conglomerate set of information and be able to output results to be applied to a desired situation. The importance of providing a method for quantifying results allows for decisions to be credible, persuasive, and convincing.

The rationale used by the Department of Defense (DoD)—more succinctly the U.S. Navy—to station its fleet in a limited number of locations may or may not be the best strategic or tactical decision for defending the U.S. from adversaries, scheduling and executing missions with varying requirements, or for responding to conflicts or natural disasters that occur around the world. Basing a large group of surface and subsurface vessels in a limited number of locations around the U.S. can limit the ability to effectively protect against and deter threats as well as make it difficult to exhaustively plan missions with available vessels. A suitable tool should be designed to allow for DMs to thoroughly examine all possible Naval mission combinations and allow for the discovery of any improper spread of naval vessel classes across naval bases by allowing for fuzzy and imprecise logic from the DM perspective[1].

An analysis of the current Naval vessel positions and mission decision factors will be used to tune and show how the implementation of a new tool might positively impact the DM by allowing for the evaluation of qualitative and quantitative factors in order to rank Naval mission decisions so that those decisions can be supported by data. Current Naval homeports will be ranked objectively with respect to the deterrence ability, forward defense ability, and the overall effectiveness of the proposed mission scenarios[1].

The modification of an integer linear program, developed by Robert Silva, and the partial integration of methods implemented by Benjamin Pearlswig to express mission requirements in terms of Naval vessel preclusions[2], will be combined with a MCDA method (ELECTRE III) to rank the missions scenarios by: homeport, current vessel location(s) to starting point of a mission, current vessel location(s) to the center of a mission area, and the ability for a mission to be completely fulfilled by available vessels. Robert Silva modified an integer linear program to rank surface and subsurface U.S. Navy vessel employment schedules based on area of need, capabilities of each vessel class, and resources needed to complete the mission. This decision-aid-based program is modified to focus on additional factors: the number of vessels of each class and number of classes at each homeport, relative distance to the next nearest military base, population of the area the base supports, size of the naval base, relative distance to the nearest hostile country, relative distance to the nearest ally, and the unknown chance that the naval base would actually be attacked. The planner program that Silva created in Microsoft Excel and Visual Basic was tested against multiple real-world scenarios. The outcomes of the program were shown to be as good as, if not better, than the real world scenarios[3].

---

[1] Brooks, L. (1986). *Naval power and national security: The case for the maritime strategy* [Electronic version]. International Security, 11(2), 58-88

[2] Pearlswig, Benjamin C. (2013). *Heuristic route generation for the Navy Mission Planner*. Retrieved from Naval Postgraduate School (https://calhoun.nps.edu/handle/10945/37690)

[3] Silva, Robert A. (2009). *Optimizing multi-ship, multi-mission operational planning for the joint force maritime component commander*. Retrieved from Defense Technical Information Center (http://www.dtic.mil/dtic/tr/fulltext/u2/a501491.pdf)

Though Silva's methods were focused on assisting with planning a ship's employment schedule, I believe it can be generalized to fit many other scenarios aside from military ones. The placement and employment of aircraft and airports around the world, container ship scheduling and port placements, or the placement of hospitals relative to crime rates and population densities are all other scenarios that I feel can be modeled with Silva's methodology that I will reformulate. In the case that modeling and simulating Navy surface and subsurface assets becomes overly problematic, I will have an opportunity for implementing my methodology on another industry.

1.1 Background of Navy Planning

The DoD follows structural and procedural steps to plan and execute operations across the world. The Naval Planning Guide (NWP 5-01) backs this assessment by stating that

> The characteristics of today's complex global environment have created the conditions where the U.S. Navy must be prepared for a wide range of dynamic situations ... Navy planning of today has migrated more toward mission based rather than threat-based planning. However, due to the nature of naval operations, forces at sea, unlike the other Services, require specific degrees of threat-based planning coupled with planning for specified missions. The specific degree of threat-based planning is a function of the mission, environment, and threat scenario[4].

The process of naval planning is broken into six major parts: mission analysis, course of action (COA) development, COA analysis (wargaming), COA comparison and decision, plan or

[4] Department of the Navy. (2013). *Navy Planning.* Retrieved from U.S. Naval War College (https://www.usnwc.edu/getattachment/171afbf3-a1e2-46b3-b1e9-d1fa4b0fec5a/5-01_(Dec_2013)_(NWP)-(Promulgated).aspx)

order development, and transition. This continuous cycle (seen in Figure 1-1) begins at the highest levels of the Naval command structure and ends with the person or persons completing the task(s).

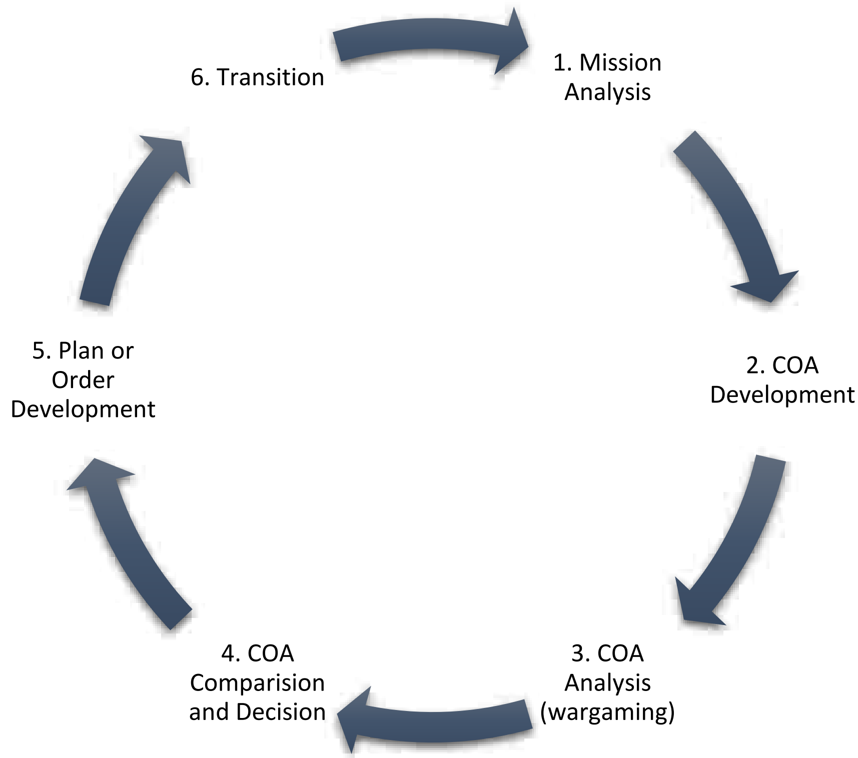

Fig. 1-1. Navy Planning Process (NPP)

The first step of the NPP (mission analysis) serves as the driving point of the rest of the process. Its purpose is to allow the DM to gain knowledge and understanding about the situation. The next step, COA development, allows mission planners to take all guidance and factors to develop multiple COAs and verify and validate the feasibility with respect to the situation at hand. COA analysis (wargaming) comes next where any weaknesses in the developed COAs are identified and planners can adjust and refine any criteria and factors to develop the best solutions. After the COAs are settled upon, they are presented to commanders for a decision in step 4. Next, Naval staff use the decision made by the commander to develop orders that can be issued and followed by the fleet. The turnover of these orders to those that will primarily execute them is performed in the transition step. The cycle then can start over with the next situation that must be resolved.

CHAPTER 2

WAYPOINTS OPERATIONAL PLANNER

The goal of this project is to expand upon the originally developed Navy Mission Planner (NMP) that was developed at the Naval Postgraduate School (NPS) by Kevin Dugan[5] in 2007, then further refined by Robert Silva in 2009, and later by Benjamin Pearlswig in 2013. The NMP was developed in Microsoft Excel with Visual Basic for enhancement of features. The goal of the NMP is to aid in the decision making process and assist the COA development and wargaming steps of the NPP. The paper-based process of assigning components to missions and then units to missions is exactly what the NMP was designed to help improve. Figures 2-1 and 2-2 are examples of the planning process, which is solely performed on paper. The two figures were extracted from NWP 5-01.

[5] Dugan, Kevin. (2007). *Navy Mission Planner*. Retrieved from Naval Postgraduate School (https://calhoun.nps.edu/handle/10945/3317)

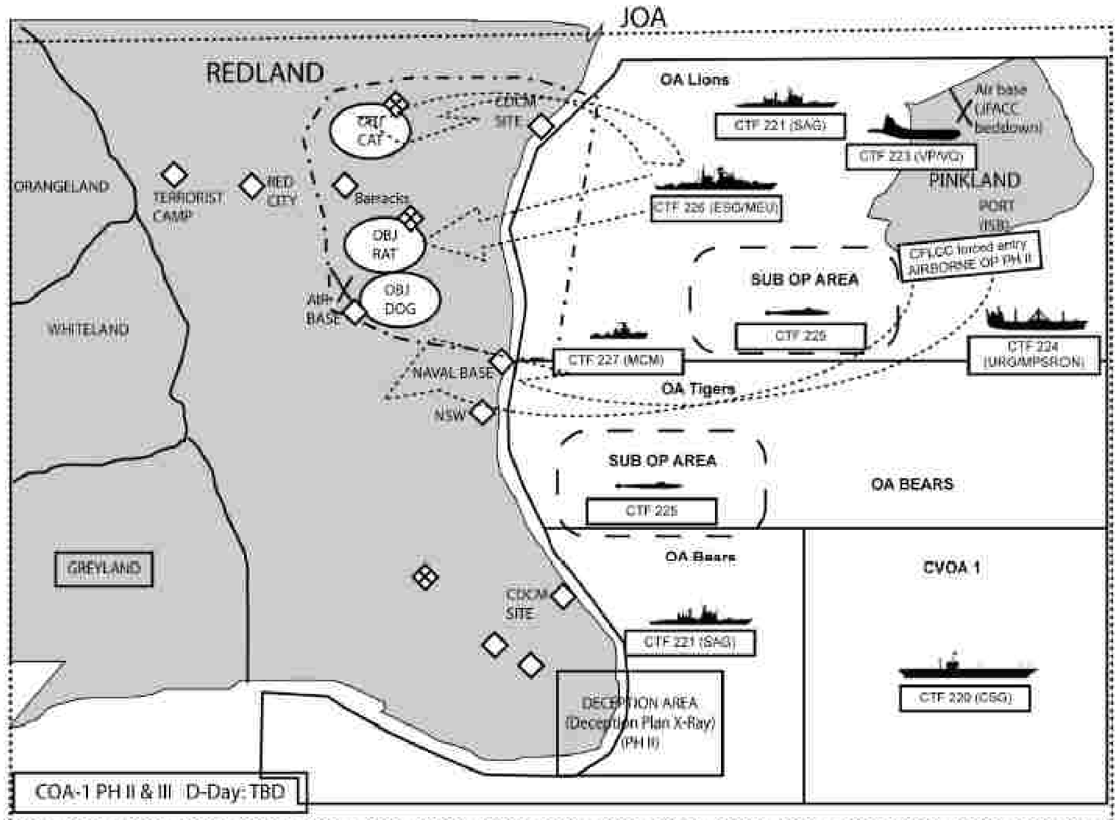

Fig. 2-1. Example of a Commander Course of Action Sketch

SUBORDINATES	CTF 226 (ESG)	T: Position in OA Lions P: Prepare for future OPS T2: Conducts rehearsals for possible amphibious operations P2: Prepare for possible amphibious operations.	Continue to conduct ISR to establish/maintain RMP. Sortie surface forces to contest threat positioning in territorial waters. Sortie subsurface fleet to contest emerging threat in TTW. Sortie air assets to contest threat positioning in TTW.	Position outside CDCM range if feasible. Defend the force.	
	CTF 222 (MIW)	T: Position IVO OAs Lions and Tigers and conduct mine hunting P: Determine the presence or absence of mines	No change; see above	Maintain positioning and current operations. Defend the force.	
	CTF 227 (MPRA)	T: Position in Pinkland and conduct patrols throughout the JOA P: Support ISR and targeting as well as providing SUW/ASW protection to the force	Increased strip alert for fighters. Sortie air assets to contest threat air posture.	Continue ISR. Coordinate with adjacent components and CTFs for I&W of Redland fighter activity.	
	CTF 224 (SUB)	T: Position in selected SUB OP AREA and provides SUW/ASW and ISR support P: Protect the force and facilitate future operations	Close selected littoral areas using defensive mining to limit threat force freedom of movement.	Maintain aggressive ASW OPS – ensure possible amphibious landing area free of Redland SS.	
OPERATIONAL FUNCTIONS	INTELLIGENCE	Provide I&W on Redland maritime strike forces and ISR in support of SUW and USW efforts.	Conduct deception to mask movement of strike force.	Activate selected named area of interest (NAI) as activity occurs.	
	FIRES	Execute planned initial strikes against selected Redland forces and capabilities to set the conditions for future actions and shape the environment.	Move aircraft to bunkers or disperse. Move ships to dispersal bases. Move CDCMs.		
	SUSTAINMENT	Flow forces and sustainment into Pinkland ISB to build combat power. Conduct UNREP as required.	Monitor logistics shipping. Weigh options to engage.	Provide additional protection for logistics and pre-positioned shipping.	
	COMMAND AND CONTROL	JTF supported component: JFMCC JFMCC Main Effort: CTF 225 JFMCC located in CVOA 1.		No change	
	PROTECTION	JFMCC accepts risk to force in selected SLOCs approaching the JOA and monitors only rather than positioning forces to cover.		Station SAG AEGIS ships IOT BPT support BMD requirement.	

Fig. 2-2. Example Wargaming Worksheet

2.1 Waypoints Operational Planner Overview

The waypoints operational planner (WOP) developed for this project has aspects from the NMP that are expanded upon (or ignored) as well as completely new tools and functions to assist in the COA development and wargaming steps of the NPP. The WOP was created in C++ (using functionality from C++11) in order to be more portable and extensible to different operating systems and to provide for a more robust libraries that could be used to quickly build a working program. Several concepts are key to the mission selection and evaluation process as implemented in the WOP: (1) waypoints, (2) country border waypoints and/or centroids, (3) mission types, (4) homeports, (5) preclusion matrices, (6) Naval units, and (7) mission type to unit mapping. All key inputs are imported by the WOP through external text files so that they can be easily maintained and loaded again should any information change.

Waypoints, as implemented in the WOP, are used for path guidance when executing a mission and determining distance to a starting or ending location. Waypoints consist of latitude and longitude decimal coordinates. The WOP provides for the auto generation of intermediary waypoints (the number of which is specified by the user) between the starting and ending waypoints which will follow the proper great circle navigation route around the Earth. Given the bearing, and starting waypoint, and a distance, the next waypoint will be calculated using the pseudocode presented in Tables 2-1 and 2-2.

```
Find bearing between, inputs of point A, point B:

    latitudeA = latitude of point A (in radians)
    longitudeA = longitude of point A (in radians)

    latitudeB = latitude of point B (in radians)
    longitudeB = longitude of point B (in radians)

    theta = longitudeB – longitudeA (in radians)
    y = sin(theta) * cos(latitudeB)
    x = cos(latitudeA) * sin(latitudeB) – sing(latitudeA) * cos(theta)

    bearing = atan2(y, x)

output bearing as divisional remainder of (bearing (in degrees) + 360)/360
```

Table 2-1. Calculation of bearing between two points

```
Calculate next waypoint, inputs of start point, distance, bearing:

    latitudeS = latitude of start point (in radians)
    longitudeS = longitude of start point (in radians)

    d = distance / 3440 (radius of Earth in nautical miles)

    b = bearing (in radians)

    s = sin(latitudeS) * cos(d) + cost(latitudeS) * sin(d) * cos(b)
```

```
    latitudeSS = asin(s)
    longitudeSS = latitudeSS + atan2(sin(b) * sin(d) * cos(latitudeS), cos(d) – sin(latitudeS) * s)

output waypoint latitude as latitudeSS (in degrees)
output waypoint longitude as divisional remainder of  ((longitudeSS (in degrees) + 540)/360)-
180
```

Table 2-2. Calculation of next waypoint given a starting point, bearing, and distance

The pseudocode presented in the Table 2-1 was based on Equation 2-1 for finding the initial bearing between two waypoints. For purposes of the WOP, the initial bearing is the same as the final bearing because the WOP is calculating the bearing between midpoints along a section of a large path and not the entire path itself. The pseudocode in Table 2-2 was based on Equation 2-2 for finding a destination point given a distance, bearing, and a starting point.

$$\theta = atan2(\sin\Delta\lambda * \cos\varphi_2, \cos\varphi_1 * \sin\varphi_2 - \sin\varphi_1 * \cos\varphi_2 * \cos\Delta\lambda)$$

(where φ1, λ1 is the start point, φ2, λ2 is the end point, and Δλ is the difference in longitude)
Eq. 2-1. Equation for the bearing between two points

$$\varphi_2 = \sin^{-1}(\sin\varphi_1 * \cos\delta + \cos\varphi_1 * \sin\delta * \cos\theta)$$
$$\lambda_2 = \lambda_1 + atan2(\sin\theta * \sin\delta * \cos\varphi_1, \cos\delta - \sin\varphi_1 * \sin\varphi_2)$$

(where φ is the latitude, λ is the longitude, θ is the bearing, δ is the angular distance (d/R) where d is the distance travelled and R is the Earth’s radius)
Eq. 2-2. Equation for destination point given distance, bearing, and starting point

Waypoints are used throughout the WOP to determine: the path for units to follow for a mission, in progress exact location of a unit during a mission, proximity to land masses (for collision avoidance), unit homeport location, and mission stoppage (or hold time). If set up, the WOP is able to run a mission and navigate all units with country avoidance, specified by a distance (in nautical miles) to avoid land. Mission stoppage time can also be set independently for each unit if there is a need for that unit to remain at a specified waypoint for any period (in hours). The WOP executes a mission on an hour-to-hour basis with the option of a user specified manual time input.

Country border points are used in determining all edge points of any country that has at least one border on an ocean. Country centroids, another method of determining unit path avoidance, are defined as the center latitude and longitude point of a country. The distance from the edge points, extending in a user-specified diameter, and the distance from the center of a country, extending around by a specified radius, aid in navigating units around countries. The country border distance can also be specified for each country independently in the case that there is a need to avoid a specific country (or countries) at a greater distance than another country. The WOP processes country border polygons using the Open Geospatial Consortium's Keyhole Markup Language (KML) standard (version 2.2.0)[6]. If no country border file is specified, the program will ignore the land collision avoidance.

Mission types and preclusion matrices go hand in hand. Missions can be created from any combination of the following mission types: Air Defense (AD), Theater Ballistic Missile Defense (TBMD), Antisubmarine Warfare (ASW), Surface Warfare (SUW), Strike (S), Naval Surface Fire Support (NSFS), Maritime Interception Operations (MIO), Mine Countermeasures (MCM), Mine Warfare (MINE), Intelligence Collection (INTEL), or Submarine Intelligence Collection (SUBINTEL). Each mission can have a value of [0 .. *n*] capabilities. Equation 2-3 can be used as the basis for specifying mission capabilities required[2].

---

[6] Open Geospatial Consortium. (2008). *OGC KML*. (07-147r2). Retrieved from http://portal.opengeospatial.org/files/?artifact_id=27810

$$0 \leq n \leq N$$

(where n is the amount of capability and N is the total number of units available)
Eq. 2-3. Mission capability range

In the above equation, the value of n can be any decimal number between 0 and N. Units also have a capability amount [0.0 – 1.0] where this represents the percent of availability of a particular capability for that unit. Unit capabilities are limited by the concept of a compatibility and a preclusion matrix.

The unit compatibility matrix is a single row vector representing the maximum capability that the unit type can achieve. Table 2-3 shows an example compatibility matrix for CVNs where columns in red are mission types not supported by the CVN unit type and have a capability amount of 0.0.

	AD	ASW	SUW	S	NSFS	MINE	MCM	MIO	INTEL	TBMD	SUBINTEL
CVN	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0

Table 2-3. CVN unit compatibility matrix

The unit preclusion matrix is an n x n matrix representing all of the capabilities that a particular unit type can perform concurrently in which the values are binary. Table 2-4 presents a sample unit preclusion matrix for CGs. The leftmost column is the primary mission where the subsequent columns are the capability of the unit to perform secondary missions.

CG	**AD**	**TBMD**	**ASW**	**SUW**	**S**	**NSFS**	**MIO**	**MCM**	**MINE**	**INTEL**	**SUBINTEL**
AD	1	0	1	1	0	0	1	0	0	1	0
TBMD	0	1	1	1	0	0	0	0	0	1	0
ASW	1	0	1	1	1	0	1	0	0	1	0
SUW	1	0	1	1	1	1	1	0	0	1	0
S	1	0	0	1	1	0	1	0	0	1	0
NSFS	1	1	0	1	0	1	0	0	0	1	0
MIO	1	1	1	1	1	0	1	0	0	1	0

MCM	0	0	0	0	0	0	0	0	0	0	0
MINE	0	0	0	0	0	0	0	0	0	0	0
INTEL	1	1	1	1	1	1	1	0	0	1	0
SUBINTEL	0	0	0	0	0	0	0	0	0	0	0

Table 2-4. CG unit preclusion matrix

A list of current homeports and approximate latitude and longitude for them is needed to determine the initial starting location (given a clean program run with no previous data) of each unit. Table 2-5 lists the homeports that were used given the most up to date information from the U.S. Navy[7]. Approximate latitude and longitude values were recorded from Google Earth by zooming into the specific location[8].

Homeport	Latitude	Longitude
Newport News (Virginia)	36.9853	76.449
Bangor (Washington)	47.772	122.749
Kings Bay (Georgia)	30.7455	81.4864
Norfolk (Virginia)	36.9619	-76.3386
Groton (Connecticut)	41.3834	72.0915
San Diego (California)	32.6896	117.2316
Yokosuka (Japan)	35.3063	139.662
Manama (Bahrain)	26.1987	50.6381
Sasebo (Japan)	35.1574	129.7132
Pearl Harbor (Hawaii)	21.3558	157.9578
Rota (Spain)	36.6182	-6.3432
Little Creek (Virginia)	36.9318	76.1792
Apra Harbor (Guam)	13.4516	144.6525
Bremerton (Washington)	47.5551	122.6324
Diego Garcia (BIOT)	7.3257	72.4102
Mayport (Florida)	30.4014	-81.4103
Everett (Washington)	47.9893	122.2499
Gaeta (Italy)	41.216	13.5759

Table 2-5. List of U.S. Navy homeport locations

[7] U.S. Navy. (2017). *Vessels*. Retrieved from https://www.navy.com/about/equipment/vessels
[8] Google Earth [Computer software]. (2017). Retrieved from https://www.google.com/earth/

2.2 Waypoints Operational Planner Advanced Features

The WOP also provides more advanced features to assist in determining and analyzing mission scenarios. The modification of unit capabilities after units have been imported allows a user to quickly modify the current capabilities of units. Users are able to modify: an individual unit, all units, all units of a specified class(es), all units of specified capabilities, or a custom assortment of modifications. Importing and exporting of all data that is generated by the program is provided by prebuilt Boost C++ serialization libraries to make the saving and loading of data quicker[9].

The addition of new homeports gives a user the ability to test out whether increasing Naval homeports and spreading out the fleet can have a positive impact on mission completion and mission solution generation. Users have the ability to freely move units between homeports and perform a basic mission comparison between two or more missions. Mission comparison computes the following options for the user: slowest and quickest mission(s) to generate solutions for (in hours, minutes, and seconds), mission(s) with the least and most overages in capabilities, mission(s) with the least and most underages in capabilities, mission(s) with the most and least number of units, mission(s) with the most and least amount of possible solutions generated, mission(s) with the greatest and least total distance to the starting point, mission(s) with the greatest and least total distance to the target point, and mission(s) with the fastest and slowest total run time.

[9] Boost [Computer software]. (2017). Retrieved from http://www.boost.org/

The concept of underages and overages is a measure of how well the units chosen for a mission fulfill all objectives. Underages and overages are directly related to unit capabilities. Unit capabilities (in the range of [0.0 – 1.0]) are indicative of how capable a particular unit is for a given capability percentage wise. If a unit has a value of 0.5 for its AD capability, this would indicate that the unit has a 50% capability of performing the air defense. This quantitative value is specified by the user and can be based on ammo/weapon stores, inoperable equipment, maintenance being performed, or other reasons. An example mission, seen in Table 2-6, shows the concept of how overages and underages work when applied to mission M1 and units U1, U2, and U3 with the total row being the difference between U1, U2, and U3 combined and M1.

	AD	ASW	SUW	S	NSFS	MINE	MCM	MIO	INTEL	TBMD	SUBINTEL
M1	2.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	3.0	0.5	0.5
U1	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0
U2	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0
U3	1.0	1.0	1.0	0.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0
Totals	-1.0	1.0	2.0	0.0	0.0	1.0	0.0	1.0	-1.5	-0.5	0.5

Table 2-6. Example mission underage and overage calculation

Depending on user preference, the total values of the outcome of a mission and its units will change. In Table 2-6, it is assumed that the user has a preference toward satisfying a mission with the least amount of overage and underages but overages do not have as much as a negative effect as an underage would. That is to say, a mission could still be successfully executed with overages but could not be run with any underages. Analyzing Table 2-6 shows that mission M1 requires a capability of AD of at least 2.0 but U3 only provides a capability of 1.0 and U1 and U2 provide no capability. This leaves M1 with a deficit of 1.0 for AD. M1 requires a partial SUBINTEL capability of 0.5 and U2 provides the capability fully while U1 and U3 do not provide this capability at all. This leaves the SUBINTEL capability for M1 with an overage of

0.5 for SUBINTEL. For the same example, Table 2-7 shows a potential perfect mission with no underages or overages.

	AD	ASW	SUW	S	NSFS	MINE	MCM	MIO	INTEL	TBMD	SUBINTEL
M1	2.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	3.0	0.5	0.5
U1	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.5	0.0
U2	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.25
U3	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.25
Totals	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Table 2-7. Perfect mission fulfillment example

The WOP also includes the ability to predict potential homeports that may improve a mission as well as perform a sensitivity analysis of a mission solution that was generated using ELECTRE III. Homeport prediction presents, to the user, a list of possible locations that might be a better homeport because of distance to the mission's target area. It is up to the user to determine whether or not the locations are actually better. Homeport prediction works by taking the mission's target location, and, for each three degrees of bearing from 0 to 360 degrees (extending outward from the target location) determines if there is a different country that might be able to harbor a vessel or be a location for a new homeport. This only takes into account if the actual location is viable by means that it has a border that is on an ocean. Figure 2-3 shows an example of how this concept works.

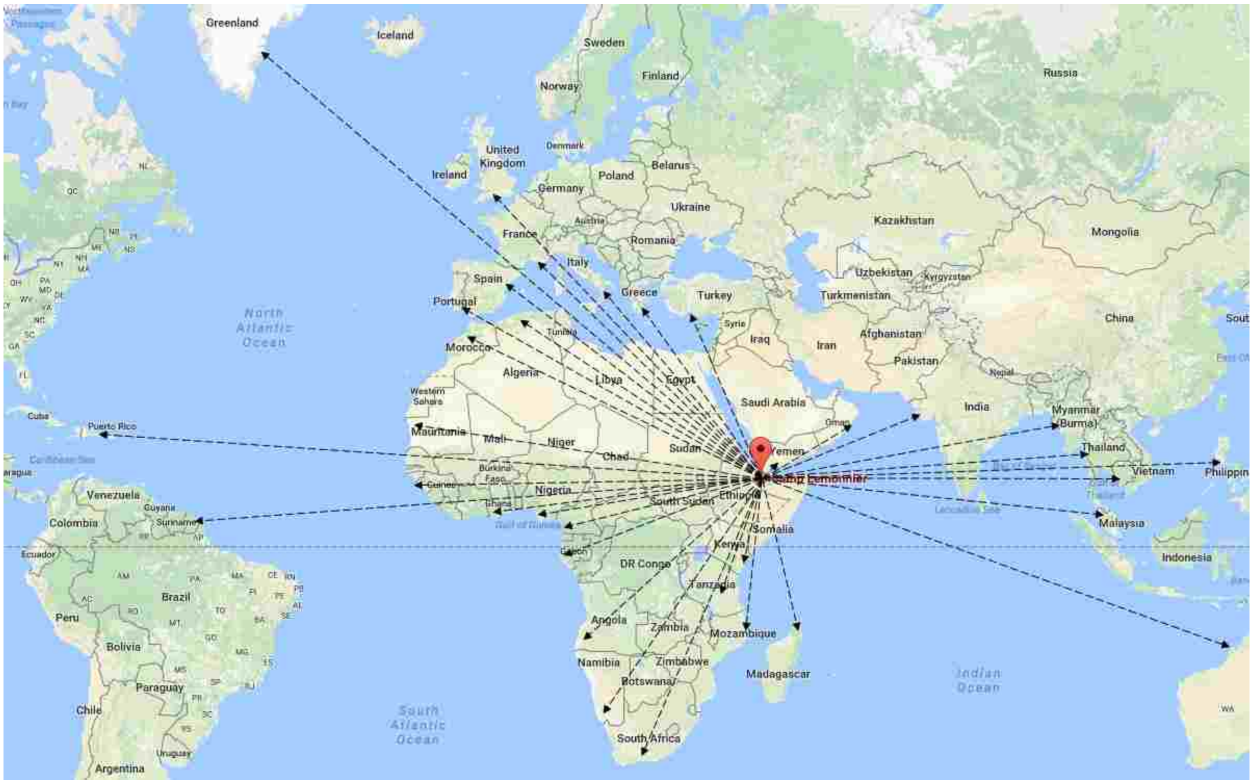

Fig. 2-3. Homeport prediction bearing example

Figure 2-3 uses a base layer from Google Maps for illustration purposes[10]. This figure is an example of how the homeport prediction logic works given the target location of Camp Lemonnier, Dijbouti. Assuming that the drawn lines lie on one of the bearings between 0 and 360, where the bearing is a multiple of three degrees (or the bearing is zero degrees), the locations where the dotted black lines point to could all be potential homeports which could provide a quicker response if units were stationed at any one of them. Based on the figure, the following countries could be suitable for a homeport: Oman, Myanmar, Thailand, Philippines, Vietnam, Malaysia, Australia, Madagascar, Mozambique, Kenya, Tanzania, South Africa,

[10] Google. (n.d.). [Google Maps base layer for countries close to Naval Station Norfolk]. Retrieved August 19, 2017, from https://goo.gl/maps/R3tq1BSELD92

Namibia, Angola, Gabon, Cameroon, Nigeria, Ghana, French Guiana, Puerto Rico, Mauritania, Morocco, Portugal, Algeria, Spain, Greenland, France, United Kingdom, Italy, Greece, or Turkey. The viability of these countries actually being a possible Naval homeport is up to the DM but the WOP presents a list of these countries sorted by closest distance to the target area.

The WOP introduces a new idea of how to solve mission scenarios using the ELECTRE III MCDA algorithm that allows for ranking and sorting possible solutions. Chapter 3 discusses this method in detail and how it is implemented to allow a DM to determine the best solution for a mission. If a mission has possible solutions generated using the ELECTRE III method, the user has the ability to perform a single variable sensitivity analysis of that mission. The WOP implements a one-factor-at-a-time (OAT) type of sensitivity analysis due to its relative low complexity to implement and that its results are easily understood. The OAT is defined by changing one factor at a time while the others remain constant[11].

The WOP program uses the original results from an ELECTRE III method of a mission as the baseline. Next, for each threshold value (preference, indifference, veto) and the criteria weights, the values are varied and the ELECTRE III procedure is re-run. If possible, for the varied threshold values, the WOP will divide the baseline threshold by ten. The baseline threshold will then be progressively added to five times by the divided threshold as well as progressively subtracted from. The ELECTRE III procedure is re-run for each addition or subtraction of each of the threshold values. The weights of the criteria are also adjusted similarly

---

[11] Saltelli, A. & Annoni, P. (2010). How to avoid a perfunctory sensitivity analysis. *Environmental Modelling & Software*, *25*(12), 1508-1517. doi: 10.1016/j.envsoft.2010.04.012

with the exception that each of the values of the weights must fall between the interval [0.0 – 1.0] and that the sum of all weights must equal to 1.0.

The results of each iteration of the ELECTRE III procedure for each threshold and weights are output to text files so that they can be analyzed in a statistical program of the user's choice. The output of the OAT sensitivity analysis includes: criteria, solution, ranked position, variable, and the variable's value. The criteria include: distance to start location, distance to target location, overages, underages, and number of units. The output solutions are a text representation of the units in the solution along with the primary preclusion for each unit. An example format for this is: CVN-68,AD,CVN-71,AD. The format shows that there are two units (CVN-68 and CVN-71) and each of them are utilizing the AD as the primary preclusion for that mission. The position is the overall rank out of all solutions where the lower the position the better the possible solution fits the mission (given the user specified thresholds). The variable and value fields output the targeted threshold or weight along with the value of that variable. Table 2-8 shows an example output for the preference threshold.

Criteria	**Solution**	**Position**	**Variable**	**Value**
Distance to Start	CVN-68,AD	25	P	900
Distance to Start	CVN-68,AD,CVN-71,AD	93	P	900
Distance to Start	CVN-68,AD,CVN-76,AD	96	P	900
Distance to Start	CVN-68,AD,CVN-76,AD,CVN-71,AD	2	P	900
Distance to Start	CVN-69,AD	139	P	900
Distance to Start	CVN-69,AD,CVN-68,AD	136	P	900
Mission Underage	CVN-68,AD	25	Q	1
Mission Underage	CVN-68,AD,CVN-71,AD	93	Q	1
Mission Underage	CVN-68,AD,CVN-76,AD	96	Q	1
Mission Underage	CVN-68,AD,CVN-76,AD,CVN-71,AD	2	Q	1
Mission Underage	CVN-69,AD	139	Q	1
Mission Underage	CVN-69,AD,CVN-68,AD	136	Q	1

Table 2-8. Example sensitivity analysis output for preference threshold

CHAPTER 3

BASIC SORTING AND OUTRANKING METHODS

When a user has specified a mission, any capabilities needed, and the WOP has generated all possible solutions, the user is presented with two options in sorting possible solutions. The first method uses the standard sort algorithm from the C++ algorithm library (with complexity of $O(N \log N)$) followed by a customized stable sort (with complexity from $O(N \log N)$ to $O(N\,(\log N) * 2)$) of all possible solutions to preserve the relative original ordering by one or more of the following values: underages, overages, total distance to starting point, and total distance to target location[12]. The WOP gives the user the choice of sorting either ascending or descending by: underages, overages, and distance to starting point, total distance to target location, and number of units. The second option for the user to sort potential mission solutions is using a specific MCDA method from the Elimination Et Choix Traduisant la Realité (ELECTRE) family called ELECTRE III.

3.1 ELECTRE III Method

The two groups of thinking for solving MCDA problems vary between a utility model (representative method) and an outranking method. The utility model is one based on weighted averages where utility functions are established

> for each single criterion, and then the utility functions will be aggregated to an overall multiutility function according to the preferential information of the decision maker. At

[12] Microsoft Visual C++ [Computer software]. (2013). Retrieved from https://msdn.microsoft.com/en-us/library/yah1y2x8(v=vs.120).aspx

last, the alternatives can be ranked from best to worst based on the overall multi-utility function value[13].

ELECTRE III is an outranking method. Outranking methods are slightly different than utility based methods in that the

> outranking relation is built through a series of pairwise comparisons of the alternatives … the concordance-discordance principle is prevalent in most Oms. It consists [of] declaring that an alternative *x* is at least as good as an alternative *y*(*xSy*) if a *majority* of the attributes supports this assertion … and if the opposition of the other attributes—the *minority*—is not "too strong."[14]

The original ELECTRE method was first presented by Bernard Roy in 1968. Since then, it has been expanded upon several times to adapt to various different MCDA problems[15]. Other variations of the original ELECTRE method includes: ELECTRE I, ELECTRE II, ELECTRE III, ELECTRE IV, ELECTRE V, and ELECTRE TRI. Of the least complex methods, ELECTRE III is the first method to introduce the ability to rank alternatives using fuzzy preference decisions from the DM. The preferential information required by ELECTRE III include criteria weights, preferences, thresholds, and vetoes (not required). From the information, alternatives can be compared with concordance and discordance functions then ranked using a credibility matrix with distillation procedures. An overview of the ELECTRE III procedure can be seen in Figure 3-1.

---

[13] Sun, Z & Han M. (2013). Multi-criteria decision making method based on Improved ELECTRE III model, presented at International Conference on Engineering, Technology, Management and Science, Nanjing, China, 2013. doi:10.2991/icetms.2013.306

[14] Bouyssou, D. (2008). Outranking Methods. *Encyclopedia of Optimization*, 2887-2893.

[15] Roy, B. (1968). Classement et choix en présence de critères multiples (la méthode ELECTRE), RIRO, 8, 57-75.

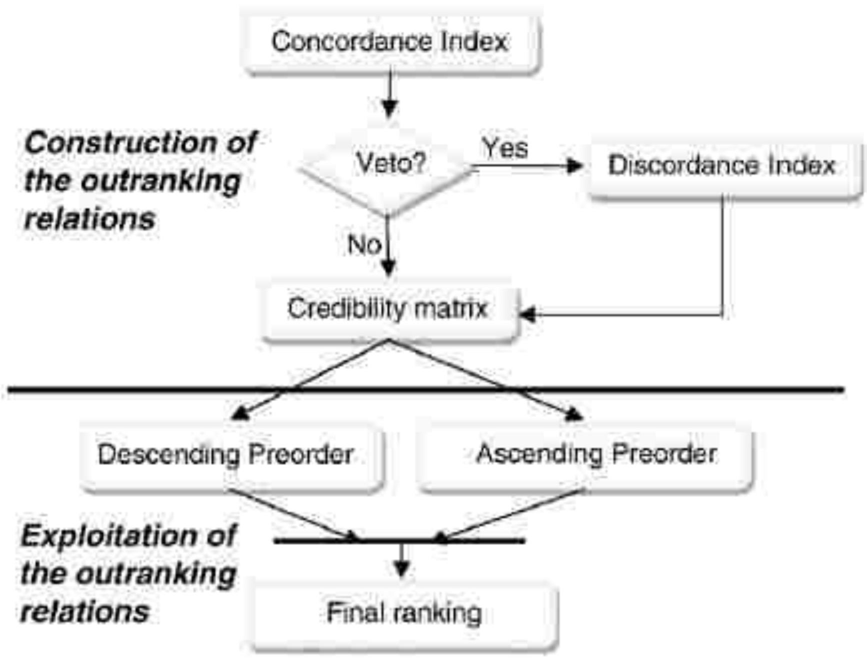

Fig. 3-1. ELECTRE III general process flow[16]

3.1.1 Concordance index

The concordance index for two alternatives (*a* and *b)*, represented by *C(a, b)*, falls within the range [0.0 – 1.0]. The lower the concordance index is, the less preferable alternative *a* is compared to alternative *b* for each criteria c_j. The overall concordance index is a weighted comparison for each of the performance values between *a* and *b*. The concordance index is computed using Equation 3-1.

[16] Giannoulis, C. & Ishizaka, A. (2010). A Web-based decision support system with ELECTRE III for a personalized ranking of British universities. *Decision Support Systems, 48*, 488-497.

$$C(a,b) = \frac{\sum_{j=1}^{n} w_j c_j(a,b)}{W}$$

Where,

$$W = \sum_{j=1}^{n} w_j$$

Eq. 3-1. Concordance index calculation[17]

Each of the indices for comparison, $c_j(a, b)$ used in the summation to attain the overall concordance index can be evaluation using one of the following equations: Equation 3-2, 3-3, or 3-4.

(1) For the case that alternative *a* is equivalent to or better than alternative *b*, minus the specified indifference threshold for criteria *j*:

For maximization of *j*	$c_j(a, b) = 1$ if $g_j(a) + q_j\left(g_j(a)\right) \geq g_j(b)$
For minimization of *j*	$c_j(a, b) = 1$ if $g_j(b) + q_j\left(g_j(a)\right) \geq g_j(a)$

Eq. 3-2. Calculation of concordance between *a* and *b*, case 1[17]

(2) For the case that alternative *a* plus the performance threshold is not as good as alternative *b* for criteria *j*:

For maximization of *j*	$c_j(a, b) = 0$ if $g_j(a) + p_j\left(g_j(a)\right) \leq g_j(b)$
For minimization of *j*	$c_j(a, b) = 0$ if $g_j(b) + p_j\left(g_j(a)\right) \leq g_j(a)$

Eq. 3-3. Calculation of concordance between *a* and *b*, case 2[17]

[17] Marzouk M. (2010). An Application of ELECTRE III to Contractor Selection, presented at Construction Research Congress, Alberta, Canada, 2010. doi:10.1061/41109(373)132

(3) If the relationship does not meet case 1 or case 2, then the relationship between alternative *a* and alternative *b* is linear for criteria *j*:

For maximization of *j*	$c_j(a,b) = \dfrac{g_j(a) - g_j(b) + p_j(g_j(a))}{p_j\left(g_j(a)\right) - q_j(g_j(a))}$
For minimization of *j*	$c_j(a,b) = \dfrac{g_j(b) - g_j(a) + p_j(g_j(a))}{p_j\left(g_j(a)\right) - q_j(g_j(a))}$

Eq. 3-4. Calculation of concordance between *a* and *b*, case 3[17]

The concordance index between alternative *a* and alternative *b* can be visualized in Figure 3-2. Zone 1 indicates the case in which alternative *a* is better than or equivalent to alternative *b* (full concordance). Zone 2 represents the case in which alternative *b* is preferred over alternative *a* by a linear amount (partial concordance). Zone 3 represents the case in which alternative *a* is not better than alternative *b* (null concordance)[16].

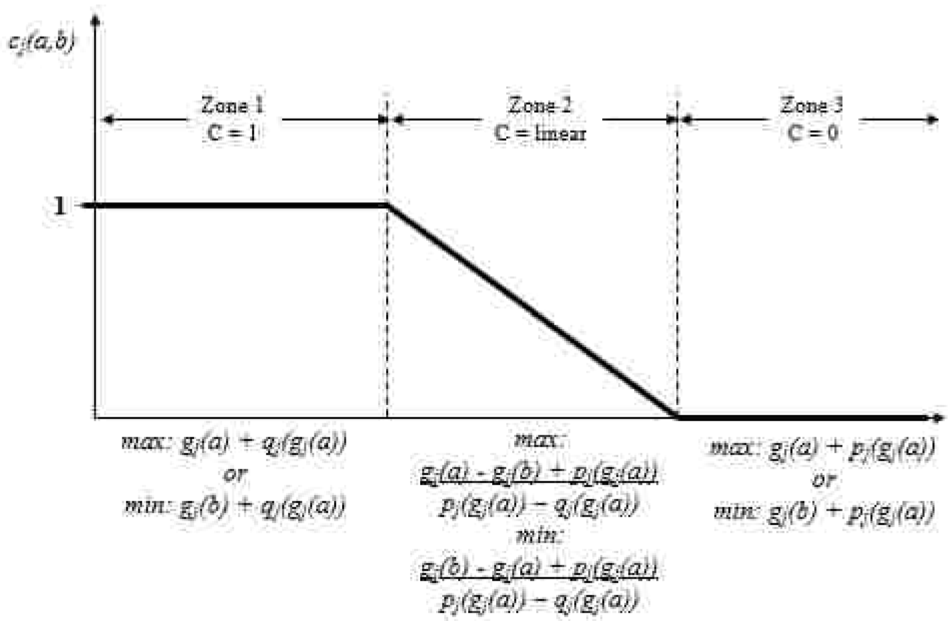

Fig. 3-2. Concordance index visualization[17]

3.1.2 Discordance index

Although alternative *a* may be better than alternative *b*, there can be one or more criteria in which alternative *a* is less preferable than alternative *b* (or alternative *b* is better than alternative *a*). The discordance index adds the ability for, even though alternative *a* may outrank alternative *b* by the value of concordance, evaluating that there may be one or more criteria *j* for which the concordance index can be overruled if the performance of alternative *b* outranks alternative *a* by at least the veto threshold[17]. Like the concordance index, the discordance index also ranges from [0.0 – 1.0]. Following the ELECTRE III process flow, if there is no veto threshold specified for criteria *j*, then the discordance index is equal to the 0.0 for all pairs of alternatives. Otherwise, the discordance index can be calculated from one of the following three equations: Equation 3-5, 3-6, or 3-7.

(1) For the case that the difference between alternative *b* and alternative *a* is not better than the preference threshold for criteria *j*:

For maximization of *j*	$D_j(a, b) = 0$ if $g_j(b) \leq g_j(a) + p_j\left(g_j(a)\right)$
For minimization of *j*	$D_j(a, b) = 0$ if $g_j(a) \leq g_j(b) + p_j\left(g_j(a)\right)$

Eq. 3-5. Calculation of discordance between *a* and *b*, case 1[17]

(2) For the case that the difference between alternative *b* and alternative *a* is greater than veto threshold for criteria *j*:

For maximization of *j*	$D_j(a, b) = 1$ if $g_j(b) \geq g_j(a) + v_j\left(g_j(a)\right)$
For minimization of *j*	$D_j(a, b) = 1$ if $g_j(a) \geq g_j(a) + v_j\left(g_j(a)\right)$

Eq. 3-6. Calculation of discordance between *a* and *b*, case 2[17]

(3) If the relationship does not meet case 1 or case 2, then the relationship between alternative *a* and alternative *b* is linear for criteria *j*:

For maximization of *j*	$D_j(a,b) = \dfrac{g_j(b) - g_j(a) - p_j(g_j(a))}{v_j\left(g_j(a)\right) - p_j(g_j(a))}$
For minimization of *j*	$D_j(a,b) = \dfrac{g_j(a) - g_j(b) - p_j(g_j(a))}{v_j\left(g_j(a)\right) - p_j(g_j(a))}$

Eq. 3-7. Calculation of discordance between *a* and *b*, case 3[17]

The discordance index between alternative *a* and alternative *b* can be visualized in Figure 3-3. Zone 1 indicates the case in which alternative *b* is not preferred to alternative *a* (no discordance). Zone 2 represents the case in which alternative *b* is preferred over alternative *a* by a linear amount (partial discordance). Zone 3 represents the case in which the difference between alternative *a* and alternative *b* exceeds the specified veto threshold amount and alternative *b* is more preferred than alternative *a* (complete discordance)[16].

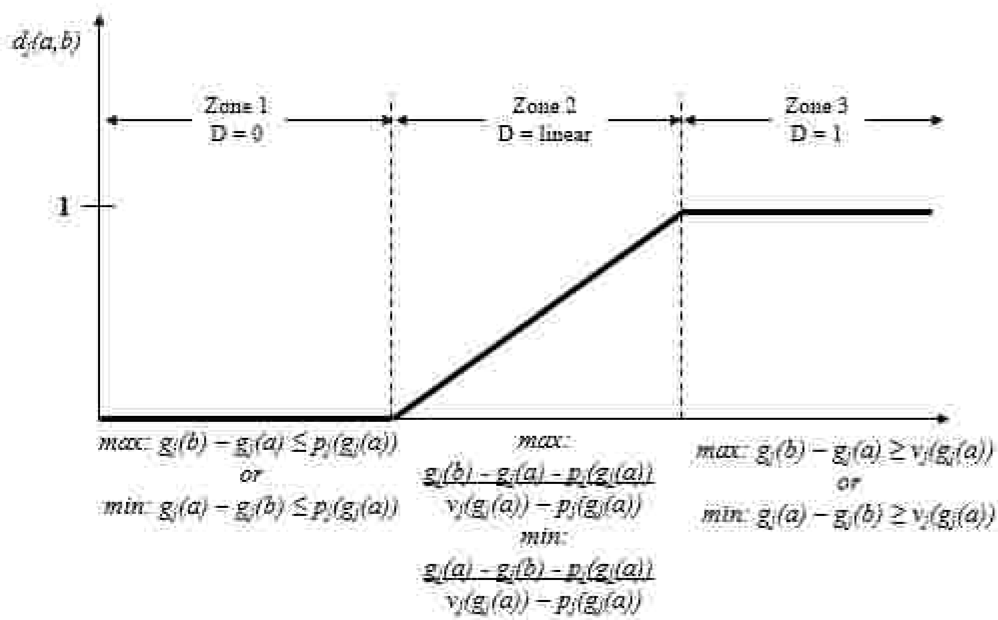

Fig. 3-3. Discordance index visualization[17]

3.1.3 Credibility

The next step in the ELECTRE III process is the computation of the credibility matrix (*S*) given the concordance and discordance indices. If there are no veto thresholds specified, then $S(a,b) = C(a,b)$, otherwise Equation 3-8 is used to determine the level of credibility in the assertion that alternative a is better than alternative b.

The degree of credibility is equal to the concordance reduced by the level of discordance for each criteria *j*

$$S(a,b) = C(a,b) \prod_{j \in \Psi(a,b)} \frac{1 - D_j(a,b)}{1 - C(a,b)}$$

Where $\Psi(a, b)$ is the set of criteria for which $D_j(a,b) > c_j(a,b)$
Eq. 3-8. Degree of credibility calculation between a and b[17]

3.1.4 Distillation

After calculation of the credibility between alternative a and alternative b, the credibility's are ranked in two separate methods (or pre-orders). The first is called descending distillation which ranks the alternatives best to worst. The second, ascending distillation, ranks the alternatives worst to best. The combined results of the two distillations gives a final ranking of alternatives. The descending and ascending distillations use the following five steps[17]:

Step 1: Set λ_0 equal to the maximum value of $S(a, b)$ in the credibility matrix *A* as in Equation 3-9.

$$\lambda_0 = \max_{a,b \in A} S(a,b)$$

Eq. 3-9. Maximum lambda value from the credibility matrix[17]

Step 2: Set λ_1 equal to the credibility score that is just less than the maximum credibility score minus a discrimination threshold as in Equation 3-10.

$$\lambda_1 = \max_{\{S(a,b)<\lambda_0 - s(\lambda_0)\}\in A} S(a,b)$$

Where $s(\lambda_0)$ is equal to the maximum level of outranking.

We can say that *a* outranks *b* (*aSb*) if *S(a, b)* is greater than the cutoff level and *S(a, b)* > *S(b, a)* by a value greater than the discrimination threshold satisfying the following:

1. $s(\lambda) = \beta - \alpha\lambda$, where α = 0.15 and β = 0.3[18]
2. *aSb* iff $S(a,b) > \lambda_1$ and $S(a,b) - S(b,a) > s(\lambda)$

Eq. 3-10. Cutoff level of outranking calculations[17]

The alpha and beta coefficients presented in Equation 3-10 are standard coefficients defined by ELECTRE III and were chosen not to be modified for the WOP. The coefficients can be tweaked in the ELECTRE III algorithm to obtain better results at the DM's preference.

Step 3: A distillation score is provided each time as a +1 every time that alternative *a* outranks alternative *b* otherwise it is given a -1. The final scores are added together to give the total qualification score.

Step 4: Descending distillation is performed, sorting the qualification scores by largest to smallest.

Step 5: Ascending distillation is performed, sorting the qualification scores by smallest to largest.

[18] Takeda, E. (2001). A method for multiple pseudo-criteria decision problems. *Computers & Operations Research 28*(14), 1427–1439.

The distillation procedures are complex but provide results for partial ranking. The set of descending distillates is formed by using progressively less restrictive rules on incrementally smaller subsets of alternatives. This allows for the assertion alternative *a* outranks alternative *b* to be truer for each progressive distillation run. After the first iteration, each subsequent iteration produces the best ranking alternative(s) from the remaining subset. The distillation procedures continue until all alternatives have either been exhausted or cannot be ranked separately from each other. The results from the two distillation procedures are combined which give a final picture ranking all alternatives[16].

The final ranking of alternatives is achieved through four possible cases[16]:

(1) Alternative *a* is higher ranked than alternative *b* in both distillations or alternative *a* is better than alternative *b* in one distillation and has the same ranking in the other distillation; *a* is better than *b*: $a\ P^{+}\ b$

(2) Alternative *a* is higher ranked than alternative *b* in one distillation but alternative *b* is higher ranked than alternative *a* in the other distillation; *a* and *b* are incomparable: *a* **R** *b*

(3) Alternative *a* has the same ranking as alternative *b* in both distillations; *a* and *b* are indifferent: *a* **I** *b*

(4) Alternative *a* is ranked lower than alternative *b* in both distillations or alternative *a* is lower than alternative *b* in one distillation and has the same rank in the other distillation; *a* is less preferable than *b*: $a\ \mathbf{P}^{-}\ b$

The total sum of $\mathbf{P}^{+}$ values for each alternative determines its rank. If alternatives are tied, they are assigned the same rank. A simple distillation example is shown in Figures 3-4 and 3-5.

Figure 3-4 illustrates a sample result of the descending and ascending distillation procedures for several alternatives. Figure 3-5 shows a final ranking that is obtained after combining the two distillation results.

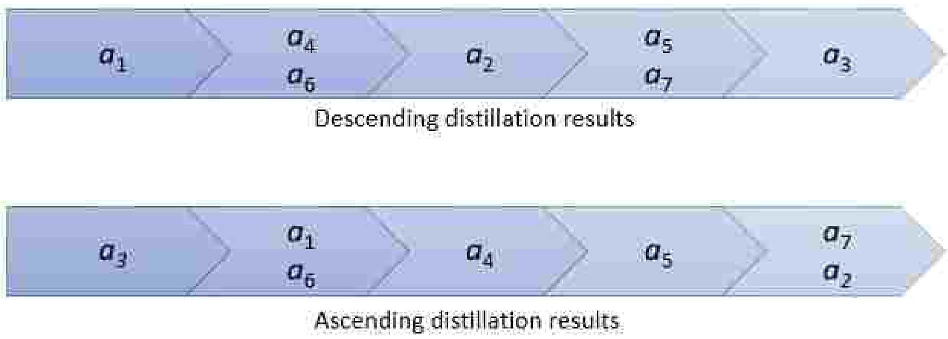

Fig. 3-4. Ascending and descending distillation example

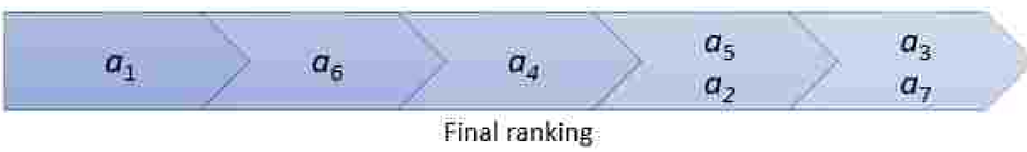

Fig. 3-5. Final ranking example

The final ranking shows that alternative a_1 is the best followed by a_6 then a_4. Alternatives a_5 and a_2 were ranked the same as well as alternatives a_3 and a_7. The final ranking results can also be seen in Table 3-1 where the total number of $\mathbf{P^+}$ values are shown in the total column.

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	**Total**
a_1		$\mathbf{P^+}$	**R**	$\mathbf{P^+}$	$\mathbf{P^+}$	$\mathbf{P^+}$	$\mathbf{P^+}$	**+5**
a_2	$\mathbf{P^-}$		**R**	$\mathbf{P^-}$	**R**	$\mathbf{P^-}$	$\mathbf{P^+}$	**+1**
a_3	**R**	**R**		**R**	**R**	**R**	**R**	**-**
a_4	$\mathbf{P^-}$	$\mathbf{P^+}$	**R**		$\mathbf{P^+}$	$\mathbf{P^-}$	$\mathbf{P^+}$	**+3**
a_5	$\mathbf{P^-}$	**R**	**R**	$\mathbf{P^-}$		$\mathbf{P^-}$	$\mathbf{P^+}$	**+1**
a_6	$\mathbf{P^-}$	$\mathbf{P^+}$	**R**	$\mathbf{P^+}$	$\mathbf{P^+}$		$\mathbf{P^+}$	**+4**
a_7	$\mathbf{P^-}$	$\mathbf{P^-}$	**R**	$\mathbf{P^-}$	$\mathbf{P^-}$	$\mathbf{P^-}$		**-**

Table 3-1. Tabular final ranking example

CHAPTER 4

WAYPOINTS OPERATIONAL PLANNER TECHNICAL OVERVIEW

The WOP was written to be modular and follows many principles of C++ classful design (with public and private member functions and variables). The program consists of 24 different source files and header files. This chapter will outline the purpose of each of the source files as well as highlight some of the major functionality that the files provide.

4.1 ELECTRE III Computation Files

The skeleton code that was built, and then highly expanded upon by me for implementing the ELECTRE III procedure, was taken from an example by Quoc Hoang from Danang Polytechnic University[19]. The original program did not take into account veto thresholds or distillation procedures and had some incorrect functions for ranking but provided a good place to start. The program also was written in an older version of C++ in which pointer management had to be carefully written into the program and it only supported the importing of ELECTRE III data via external files. Consequently, the original program was gutted and rewritten but its file and class layout was kept the same. Figure 4-1 shows the overall layout of the ELECTRE III class files where the arrows indicate dependencies between the classes.

---

[19] Hoang, Q. (2016, June 3). *Electre III method implementation in C++*. Retrieved from https://github.com/hoangddt/electreIII

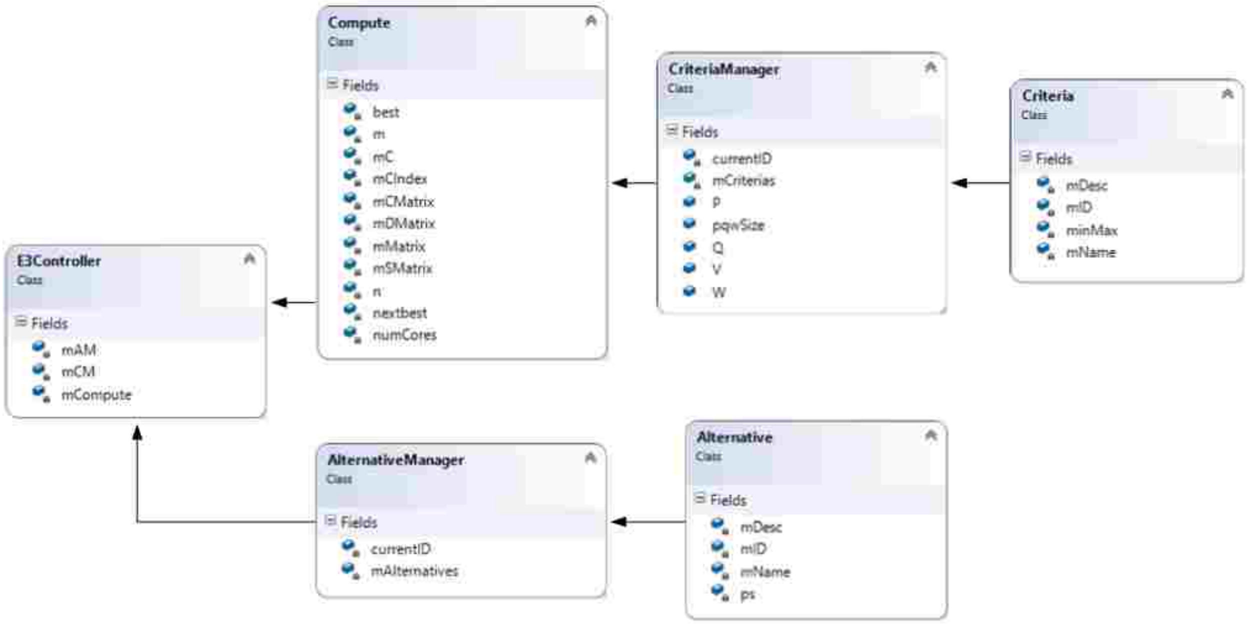

Fig. 4-1. ELECTRE III class dependencies

The ELECTRE III computational part of the WOP is made up of the source files: E3Controller, Compute, CriteriaManager, AlternativeManager, Criteria, and Alternative. The E3Controller class is the entry point into the rest of the ELECTRE III computational files. This class holds references to Compute, CriteriaManager, and AlternativeManager. It also provides the functions necessary to interact with the dependent classes and controls the start of the execution of the ELECTRE III procedure and its following steps. The CriteriaManager and Criteria classes contain all information about the criteria that will be evaluated, including: preference, indifference, and veto thresholds, criteria weights, a description, and whether or not to try to minimize or maximize the criteria.

The AlternativeManager and Alternative classes hold information about alternatives: a unique identifier, a simple name, and a pointer to the actual solution that the alternative represents. The Compute class is the workhorse of the ELECTRE III procedure. Compute holds the concordance, discordance, credibility, and alternative performance matrices, as well as the overall credibility indices. The computation procedure of ELECTRE III scales exponential with the number of alternatives and criteria that are being evaluated. The class was written to utilize up to eight threads at a time in order to reduce computation time. Table 4-1 lists the worst case complexities for the main ELECTRE III calculations performed by the Compute class.

Function	Complexity
Concordance calculation	$O(n_1) * O(n_2) * (2 * O(n_1))$
Discordance calculation	$O(n_1) * O(n_2) * O(n_1)$
Credibility calculation	$O(n_1) * O(n_1)$
Distillation calculation; runs twice for ascending then for descending	$O(n_2) * (2 * O(n_1) + O(n_1)^2 + O(n_2) + O(n_2)^2 + O(n_2))$
Ranking calculation	$2 * O(n_2) + O(n_1)^2 + [2 * O(n_2) + O(n_1)^2 + O(n_2)^2] + O(n_2)^2 + 3 * O(n_2)^2$
Where: n_1: number of criteria n_2: number of alternatives	

Table 4-1. Complexity of ELECTRE III functions in the WOP

From the complexities of the functions for the ELECTRE III procedure, it is easily seen that, for large numbers of alternatives and criteria, the concordance and discordance calculations are about equal but the distillations functions are the most time consuming to finish. The bracketed part within the ranking calculation complexity is threaded with up to eight threads so that the ranking time can be reduced. Sample total cycle calculation for various numbers of alternatives and criteria are shown in Table 4-2.

n_1	n_2	Discordance	Concordance	Distillation (x2)	Ranking	Total
4	1,000	16k	32k	~2b	~5m	~4.005b
4	10,000	160k	320k	~2t	~500m	~4.0005t
4	100,000	1.6m	3.2m	~2q	~50b	~4.0005q
8	1,000	64k	128k	~2b	~5m	~4.005b
8	100,00	640k	1.28m	~2t	~500m	~4.0005t
8	100,000	6.4m	12.8m	~2q	~50b	~4.0005q

Table 4-2. Sample complexity results with varying numbers of criteria and alternatives

It is interesting that, while the number of alternatives and criteria has a noticeable impact on the number of cycles that the discordance and concordance calculations take, the number of criteria has a negligible impact on the distillation and ranking calculations where the number of criteria far outweighs it.

4.2 Possible solution files

The possible solution (or alternative) class files represents the collection of units, capabilities used, overages, underages, and distances to the target location and to the starting location for a mission. The classes that are required for solution information are: possibleSolution, matrix, gcPoint, unit, and unit_identifier. Figure 4-2 shows the diagram of how these classes are dependent on each other.

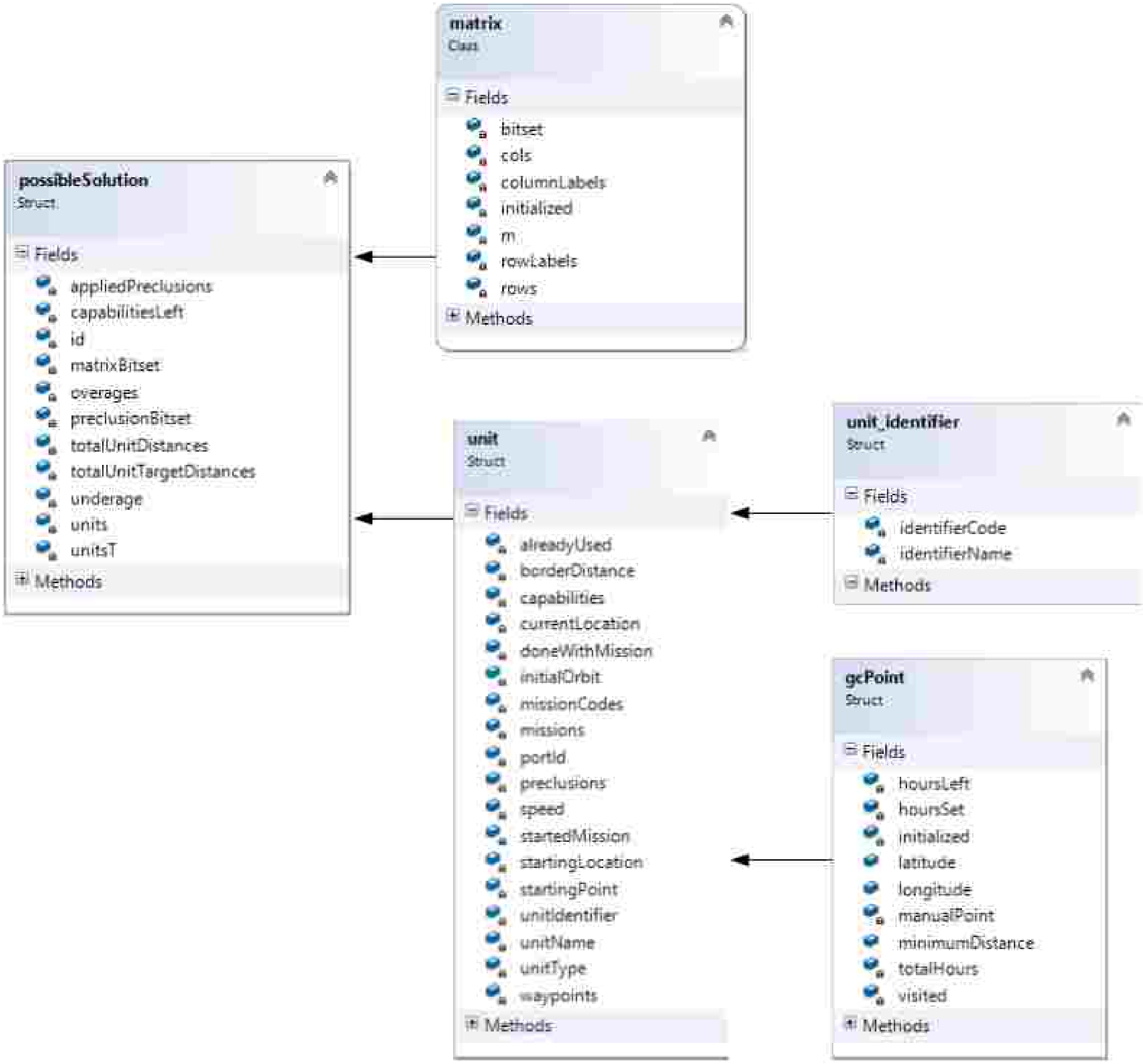

Fig. 4-2. Possible solution class dependencies

The possible solution for a given mission is a container for all units used for the solution, matrices for the capabilities not met and preclusions used for each unit, and information needed for ranking of solutions: underages, overages, total distance of all units to the starting location, total distance of all units to the target location, and number of units. The capabilities left and unit preclusion matrices are row vectors of the matrix class type. The matrix class provides basic matrix manipulation functions. The possible solution class also includes a collection of units of the unit class. The unit class contains the information about a particular unit including: distance (nm) to avoid land masses, a row vector of capabilities, the current location of the unit (gcPoint class type), a collection of one or more missions that the unit may be assigned to, the preclusion matrix for the type of unit, the maximum speed (nm) that the unit can go, and a collection of waypoints that the unit has and will visit on each mission.

4.3 Mission source files

The overall mission that is required contains one class called mission. The mission controls everything to do with unit movement and position, possible solution generation, capabilities needed versus met, all waypoints that will be followed by the units attached to the mission, and the total time elapsed when running the mission. Figure 4-3 shows the dependencies for the mission class.

Fig. 4-3. Mission class dependencies

The mission class contains the bulk of the logic in computing possible mission solutions given a variety of factors: mission prerequisites, mission starting/target areas, exclusion areas, and mission requirements. Missions can be prerequisites for other missions. If mission A is a prerequisite for mission B, then mission A must be completed before mission B can start. This allows for multiple mission scenarios to be chained together in a structured way where the missions can be run systematically, if needed, instead of having to run in parallel. When creating a new mission, the user will need to follow these steps:

1. Choose the set of imported waypoints for the mission's units to follow or create a new set.
2. Build a set of requirements for the mission. The requirements consist of a row vector of all available mission types (AD, ASW, INTEL, MCM, MINE, MIO, NSFS, S, SUBINTEL, SUW, and TBMD) in which the total of the row must be greater than zero.
3. Choose units for the mission manually or let the WOP automatically find potential solutions.
4. Choose to modify the default exclusionary start radius for units. The exclusionary start radius automatically excludes units that are 100nm (default) from the starting location to reduce the time for complete mission solution generation.
5. Choose whether or not to include any units that may have already been assigned to currently running or planned missions. Multiple missions can be created and executed simultaneously depending on the user's requirements.
6. After mission solutions have been exhaustively created, the user must determine whether to sort possible solutions manually or through the ELECTRE III method.
7. Once possible solutions are sorted, the remaining choices for the user are: whether or not to change any unit starting or ending positions, whether or not to add stoppage time for any unit(s) at any of the scheduled waypoints for the mission, and whether or not to set any path exclusions for the mission.

Based on the inputs from the user, the mission class will find all combinations of units that can satisfy the mission requirements. The pseudocode for mission solution generation can be followed in Table 4-3.

```
allUnits = allUnits minus any that are already used (depending on user preference)

allUnits = allUnits minus any that cannot at least partially reduce the requirements based on
the unit's capabilities

first loop:
loop through allUnits:
   if unit can reduce the mission requirements then
      create a possible solution with the unit

      if unit is within exclusion radius or exclusion radius is ignored then
         add possible solution to solution list
      else
         add possible solution to the extra solutions list
      end if
   end if
end loop

if there are no possible solutions and there are extra solutions then
   double the exclusionary radius
   go to first loop
end if

if there are no possible solutions and there are no extra solutions then
   end solution generation because there is no combination of units that will work
end if

start expansion routine:
   loop through all possible solutions
      loop through each possible unit for solution
         loop through each preclusion for this unit
            if preclusion is not used and preclusion can reduce mission requirements then
               create a solution with current unit and this preclusion
               add solution to the list of total solutions
            end if
         end loop
      end loop
   end loop
end routine

loop through all possible solutions
   loop through all possible units
      loop through all preclusions for this unit
         if unit with this preclusion is not already in solution
            if solution still has remaining  requirements then
```

```
          create a copy of solution
          add unit to solution copy with this preclusion
        end if
      end if
    end loop
  end loop
end loop

reduce equivalents routine:
  loop through all possible solutions[1]
    loop through all possible solutions[2]
      if possible solution[1] is not equal to possible solution[2] then
        if preclusion matrix[1] is equal to preclusion matrix[2] then
          add possible solution[2] to equivalent solutions list
          remove possible solution[2] from possible solutions
        end if
      end if
    end loop
  end loop
end routine

reduce duplicates routine:
  loop through all possible solutions[1]
    loop through all possible solutions[2]
      if possible solution[1] is not equal to possible solution[2] then
        if possible solution[1] is equal to possible solution[2] then
          remove possible solution[2] from all solutions
        end if
      end if
    end loop
  end loop
end routine

loop through all possible solutions
  if there are no solutions that full satisfy the mission requirements and there are extra units
then
    add extra units to list of all possible units
    start back at expansion routine
   else
    if there are any equivalent solutions then
      loop through all possible solutions
        loop through all equivalent solutions
          copy possible solution
          swap copy main preclusion with equivalent
          add copy to all possible solutions
        end loop
```

```
        end loop
      end if
   end if
end loop
```

Table 4-3. Mission solution generation pseudocode

After the WOP has generated all possible solutions, the user will have to execute Steps 6 and 7 to finish the creation of the mission. During Step 6 of mission creation, the user has the choice to choose either a manual method or ELECTRE III to sort missions from best to worst. If the manual method is chosen, the user will be able to sort by one or more of the following methods and whether to prefer to minimize or maximize the sort criteria: underages, overages, distance to start location, distance to target location, or number of units. The ELECTRE III method provides for a much more robust way to sort and rank possible solutions but understanding the threshold values required can be a challenge in itself.

After units are chosen for a mission, the mission class allows the user to execute a mission for a determined set of hours (or until the mission is complete). The execution of a mission is performed by navigating all units for the mission through the waypoints while navigating around land masses (or exclusion points) taking into account any stoppage points along the way. Examples of how the WOP navigates units through a set of waypoints that were generated for a mission with a distance of 100nm set to avoid land and a starting location of 32.6896, -117.232 (latitude, longitude) and a target location of -11.3212, 136.199 (latitude, longitude) can be seen in Figures 4-4, 4-5, and 4-6.

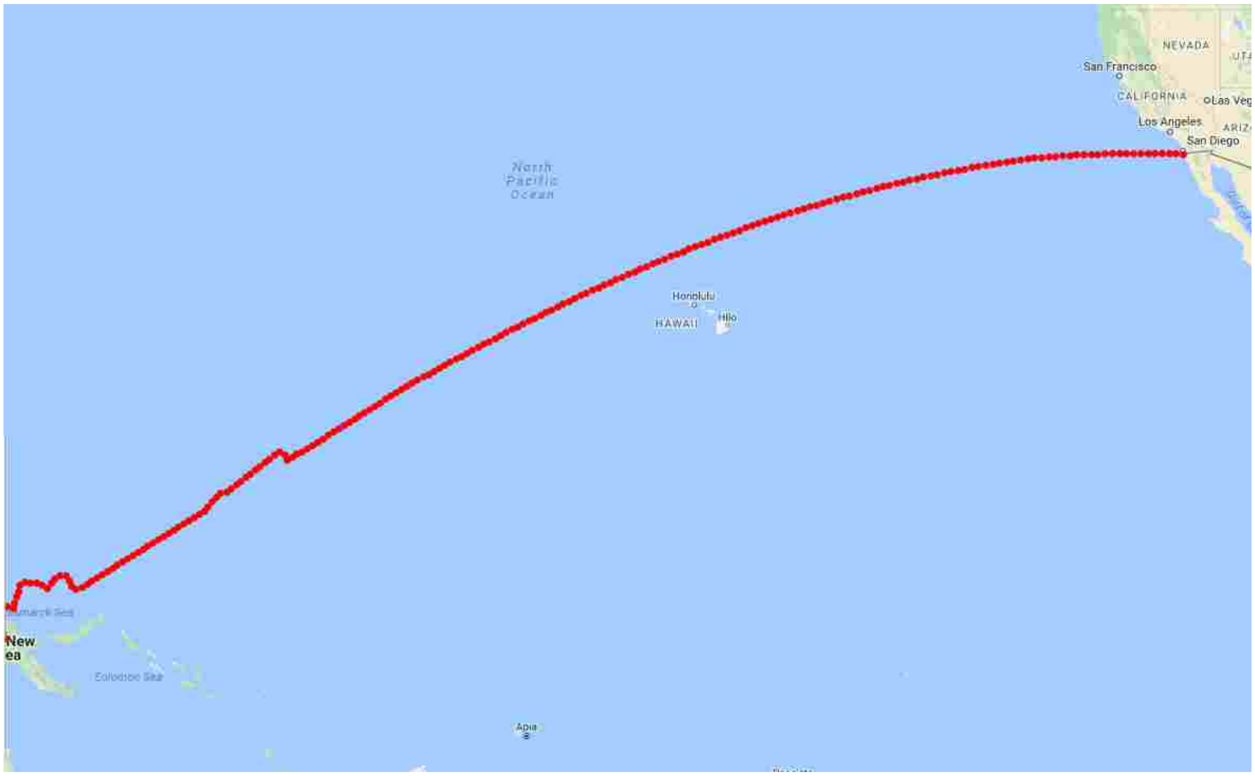

Fig. 4-4. Overall sample mission waypoint set

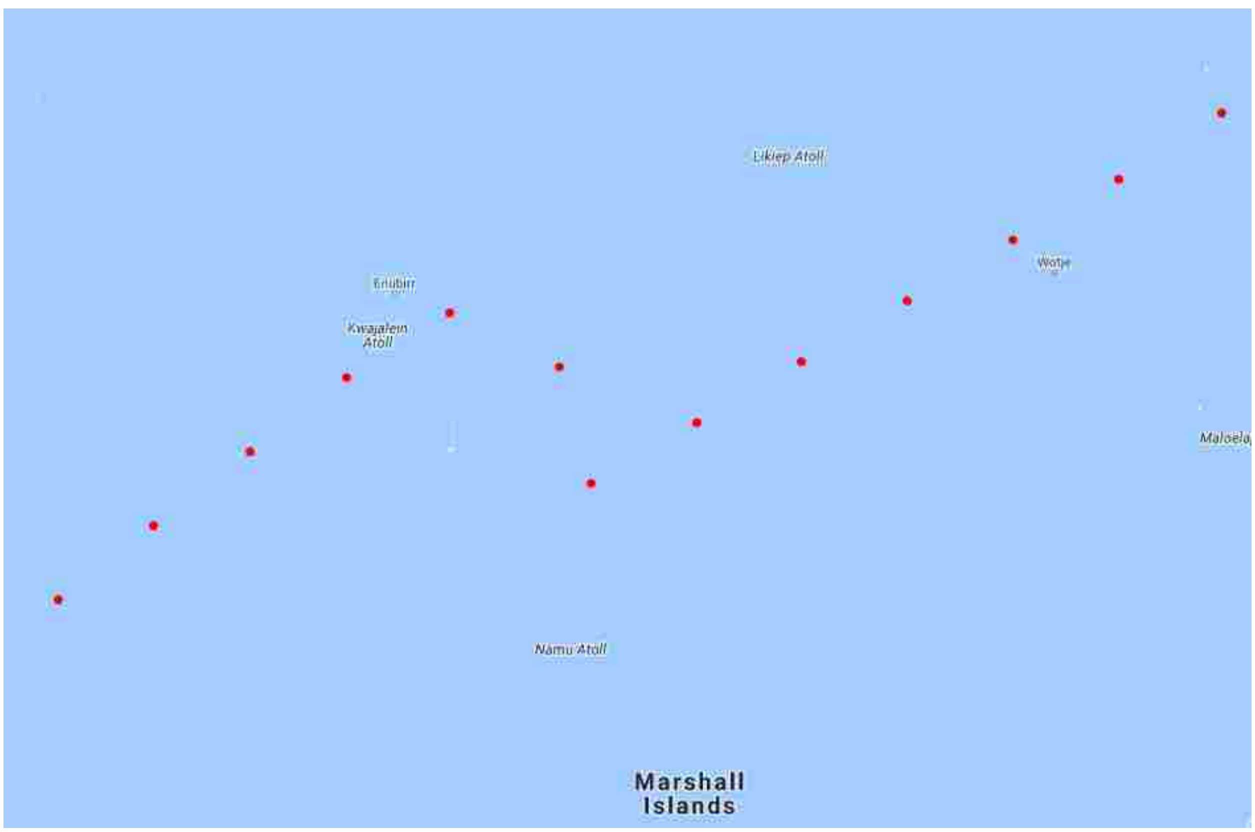

Fig. 4-5. Sample mission waypoints avoiding Marshall Islands at 100nm

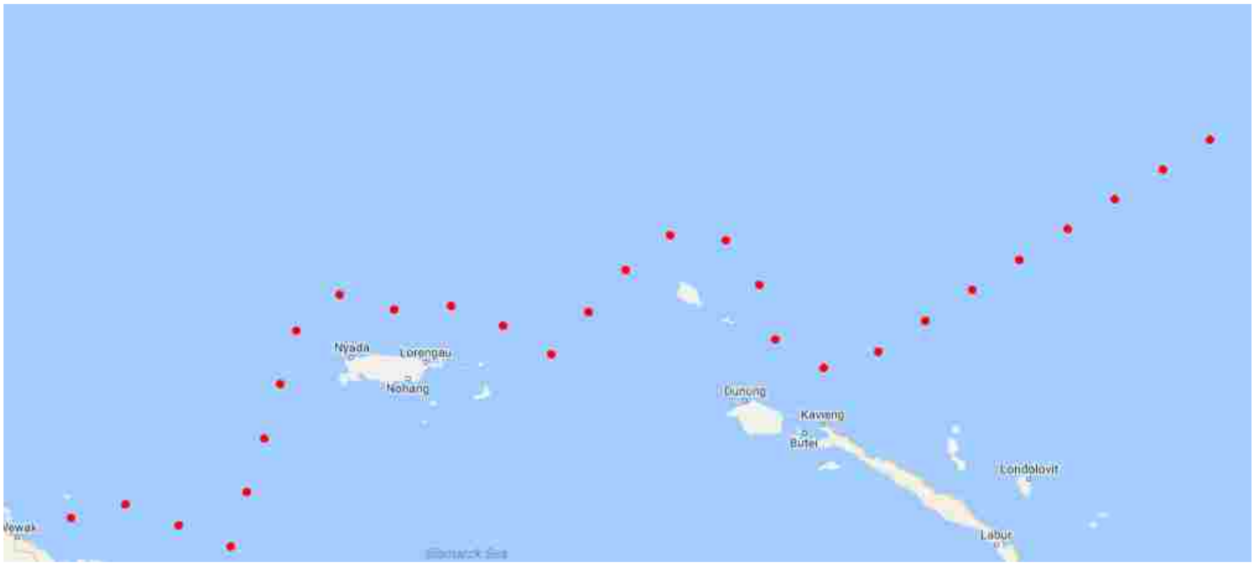

Fig. 4-6. Sample mission waypoints avoiding Bismarck Sea islands

4.4 Predicted homeport source files

The predictedHomeport class contains, per mission, each of the potential new homeports for that mission, the bearing and distance from the mission starting point to the new homeport, and the waypoint of the new homeport. While the actual homeport prediction functions are contained in the mission class, the predictedHomeport container provides a convenient way to keep track of these homeports. Figure 4-7 shows the simple layout of the predictedHomeport class.

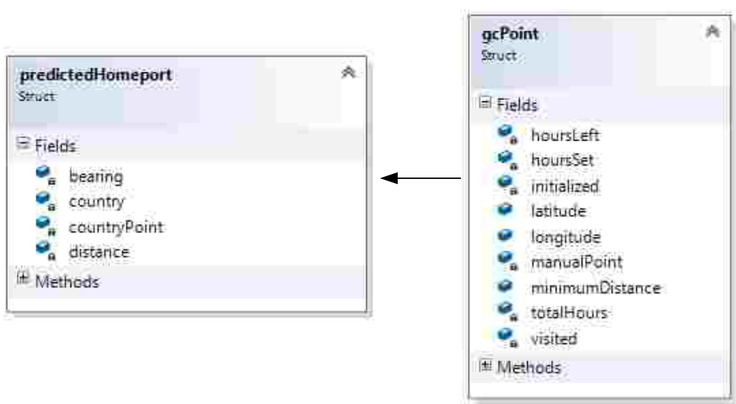

Fig. 4-7. Predicted homeport class dependencies

Even though the code for homeport prediction is within the mission class, the explanation of how the homeport function runs will be explained in this section. In Section 2.2 a general overview of how the predicted homeport algorithm works, as well as a sample graphic (Figure 2-3) of how the algorithm could generate potential homeport solutions. Table 4-4 shows the pseudocode of how homeports are predicted.

```
bearing = 0
nextPoint = mission target area
targetCountry = closest country to target area

loop while bearing is less than or equal to 360
    nextPoint = find next point from homeport with bearing, and distance of 50 (using Eq. 2-2)

    closestCountry = closest country to nextPoint
    closestCountryDistance = closest country distance to nextPoint

    if closestCountryDistance is within 25nm then
        if closestCountry is not targetCountry then
            add this country to the potential homeport list
            increase bearing by 3 degrees
            set nextPoint back to mission target area
        end if
    end if
end loop
```

Table 4-4. Potential homeport discovery pseudocode

The potential homeport code will only work if the user imports a list of countries with at least one border on an ocean in the KML v2.2 format. Homeport generation can be as accurate as the level of detail of the KML file as the homeport algorithm uses latitude and longitude border points for countries to calculate the distance between points. This algorithm assumes landlocked countries are not imported so no checking is performed when a potential country is found. Similar to Figure 2-3, Figure 4-8 shows a more accurate representation of what countries the homeport algorithm would expect when looking for potential homeports.

Fig. 4-8. Excludable countries for potential homeport generation

The shaded countries in Figure 4-8 are countries that are landlocked and should not be used for homeport generation.

4.5 Timeline source files

The timeline class controls running of all missions and holds the current step of the timeline. When the timeline is started by the user, all missions are automatically started if possible (except those with prerequisites). A timeline tick is represented as an hour of time (unless specified otherwise by the user). On each tick of the timeline, any mission(s) that might have prerequisites will be checked to see if those prerequisites are completed, and, any missions with completed prerequisites will be started. Figure 4-9 shows how the timeline class is designed with its dependencies.

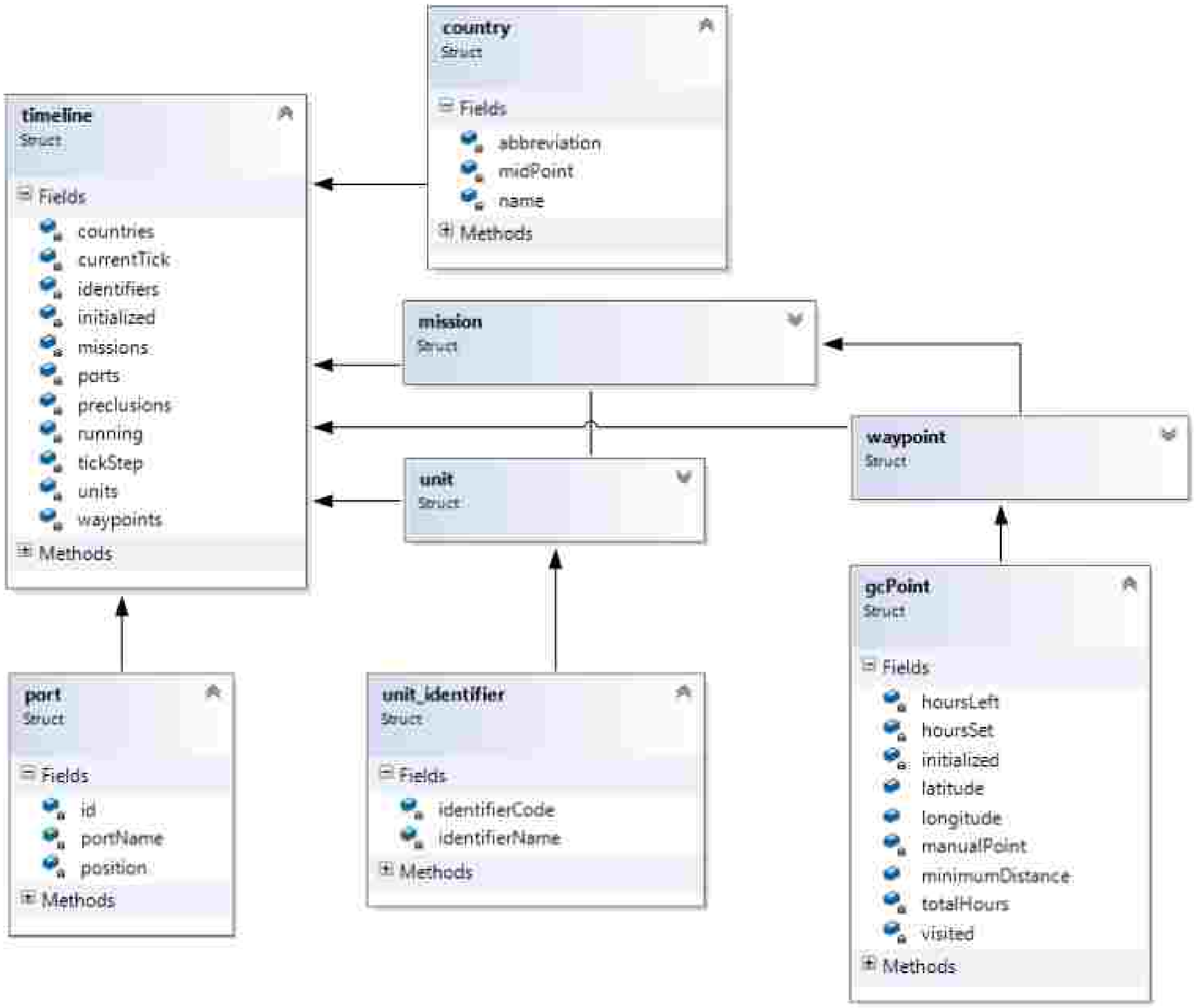

Fig. 4-9. Timeline class dependencies

It should be noted that, any time a mission is modified, the entire timeline will be reset as the timeline's missions are replaced upon any changes.

CHAPTER 5

RESULTS AND CONCLUSIONS

The WOP proves to be a useful tool in helping plan and execute Naval missions. There are many features which can be improved upon or added (see Chapter 6), but the current feature set that the WOP provides is a good baseline in acting as a toolkit for the U.S. Navy to provide a solution through steps 1-4 of the of the NPP (see Figure 1-1). This chapter provides the inputs, outputs, and an analysis of each of the major features of the WOP.

5.1 Midpoint calculation

The calculation of midpoints (latitude and longitude coordinates) between a start and an end point is crucial to the execution of a mission, the calculation of distances, for each unit, to homeports, target mission areas, and total distance travelled by each unit during a mission. The calculation of midpoints requires a start point, number of midpoints to generate, and an end point. This functionality, handled through the waypoints class, can be seen to work with the following examples. Given the starting location of Naval Station Norfolk, Virginia with the latitude of 36.9627 and longitude of -76.3307 and ending location of Cape Town, South Africa with the latitude of -33.9034 and longitude of 18.4375, Tables 5-1 and 5-2 show two examples of generated waypoints with varying numbers of midpoints.

*	36.9627	-76.3307	← Start		*	-33.9034	18.4375	← End
1	31.9447	-65.4481			6	-1.31653	-24.0752	
2	26.1004	-55.7774			7	-8.43429	-16.6378	
3	19.6754	-47.0703			8	-15.4069	-8.93254	
4	12.8643	-39.0544			9	-22.0957	-0.684357	
5	5.82324	-31.4701			10	-28.3321	9.34598	

Table 5-1. Calculated 10 midpoints between Norfolk, VA and Cape Town, South Africa

*	36.9627	-76.3307	← Start		*	-33.9034	18.4375	← End
1	24.855	-53.9676			3	-5.5975	-19.6236	
2	10.0678	-35.9828			4	-20.7882	-2.38674	

Table 5-2. Calculated 4 midpoints between Norfolk, VA and Cape Town, South Africa

The calculation of midpoints between waypoints does not take into account land masses that may be in the way of a unit if that unit were to follow the waypoints generated. The mission class which executes a mission is able to direct units around land masses. To further illustrate that the generated waypoints correctly follow the curvature of the Earth and are adequately spaced between the starting and ending points, Figures 5-1 and 5-2 visually represent Tables 5-1 and 5-2 respectively.

Fig. 5-1. First path curvature between Norfolk, VA and Cape Town, South Africa

Fig. 5-2. Second path curvature between Norfolk, VA and Cape Town, South Africa

In Figures 5-1 and 5-2, it can be seen that the distance between each point in each figure is relatively the same. For the path with ten midpoints (Figure 5-1 and Table 5-1), the distance between each point is approximately 617.06nm. For the path with four midpoints (Figure 5-2 and Table 5-2), the distance between each point is approximately 1357.53nm.

5.2 Timeline execution

The execution of the timeline is another area that is crucial to the functionality and usefulness of the WOP. The timeline requires at least one mission to be able to run. I will present two different examples of the WOP executing missions on a timeline. The first example will be a single mission with a single unit. The mission will start at Naval Station Norfolk, Virginia (36.9643, -76.3275) with a target location of Bermuda (32.37, -64.6812). The mission will be set to return to the starting location after sitting at the target location for 24 hours and there are no path exclusions set. The first mission was given a requirement of INTEL with a value of 1 and has no prerequisites. Every unit in the WOP had its capabilities set to full. Based on the closest distance to the starting location, the USS Helena (SSN-725) was chosen. This unit was set to have a maximum speed of 25 knots and it is assumed that, for ease of timeline progression, the unit will always be operating at the maximum speed. Figure 5-3 shows the entire progression of the USS Helena from the starting location, to the target location, and back to the homeport.

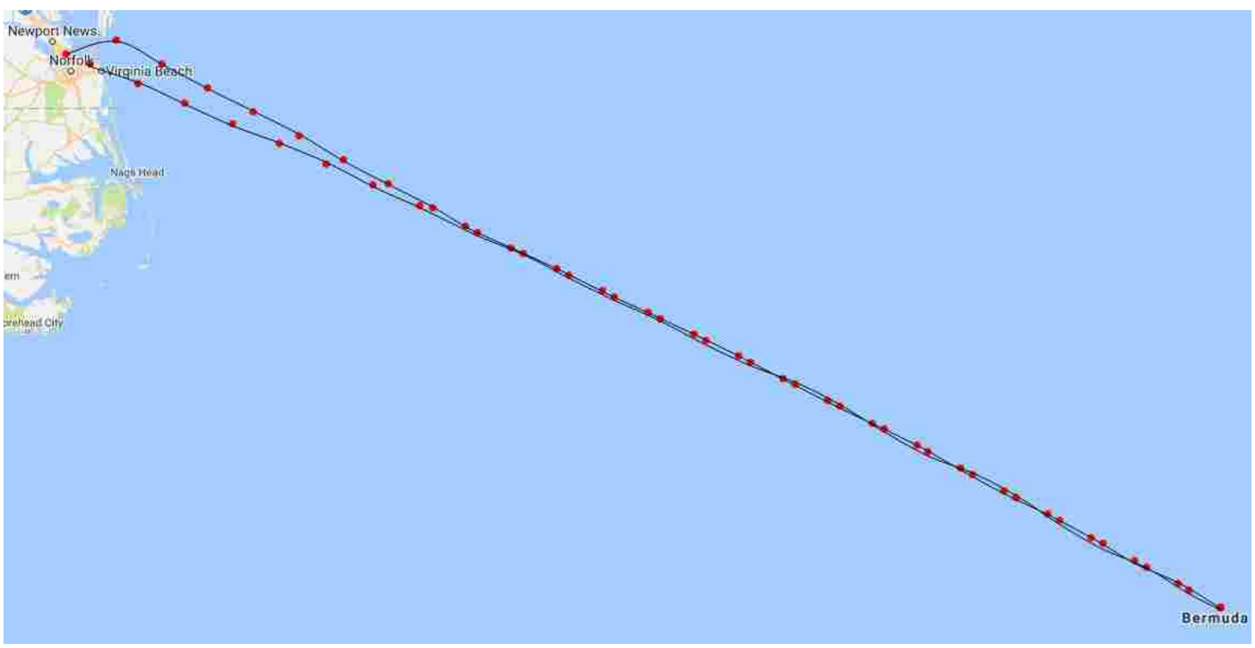

Fig. 5-3. Test mission from Naval Station Norfolk, Virginia to Bermuda and back

The sample mission seen in Figure 5-3 took 3 days, 6 hours for the USS Helena to complete (including the 24 hours of stoppage time at the target location). The timeline finished at tick 78 (where each tick is one hour). Each of the waypoints for this mission can be seen in Table 5-3.

Step	Waypoint		Step	Waypoint		Step	Waypoint
1	37.0694, -75.8348		26	32.37, -64.6812		51	32.7687, -65.5489
2	36.8801, -75.3706		27	32.37, -64.6812		52	32.9658, -65.9856
3	36.689, -74.9086		28	32.37, -64.6812		53	33.1613, -66.4242
4	36.4961, -74.449		29	32.37, -64.6812		54	33.3553, -66.8648
5	36.3015, -73.9917		30	32.37, -64.6812		55	33.5477, -67.3074
6	36.1051, -73.5367		31	32.37, -64.6812		56	33.7386, -67.7519
7	35.907, -73.0839		32	32.37, -64.6812		57	33.9279, -68.1985
8	35.7072, -72.6334		33	32.37, -64.6812		58	34.1155, -68.6469
9	35.5058, -72.1852		34	32.37, -64.6812		59	34.3015, -69.0974
10	35.332, -71.7209		35	32.37, -64.6812		60	34.4858, -69.5499
11	35.1565, -71.2585		36	32.37, -64.6812		61	34.6685, -70.0043
12	34.9792, -70.7982		37	32.37, -64.6812		62	34.8495, -70.4608
13	34.8002, -70.3399		38	32.37, -64.6812		63	35.0288, -70.9193
14	34.6195, -69.8835		39	32.37, -64.6812		64	35.2063, -71.3797
15	34.4371, -69.4292		40	32.37, -64.6812		65	35.3821, -71.8422
16	34.253, -68.9768		41	32.37, -64.6812		66	35.5561, -72.3067
17	34.0672, -68.5264		42	32.37, -64.6812		67	35.7283, -72.7732
18	33.8798, -68.078		43	32.37, -64.6812		68	35.8987, -73.2417
19	33.6899, -67.6322		44	32.37, -64.6812		69	36.0673, -73.7122
20	33.4983, -67.1884		45	32.37, -64.6812		70	36.2341, -74.1848
21	33.3052, -66.7465		46	32.37, -64.6812		71	36.3989, -74.6593
22	33.1105, -66.3066		47	32.37, -64.6812		72	36.5619, -75.1358
23	32.9142, -65.8686		48	32.37, -64.6812		73	36.723, -75.6144
24	32.7165, -65.4326		49	32.37, -64.6812		74	36.8822, -76.0949
25	32.5172, -64.9985		50	32.5701, -65.1141		75	36.9619, -76.3386

Table 5-3. Waypoints from Naval Station Norfolk, Virginia to Bermuda and back

The waypoints listed in Table 5-3 are the entirety of the waypoints for the mission. The waypoints at steps 26-49 represent the 24 hours of stoppage time at the target location (32.37, -64.6812). The second example will be two missions each with a single unit. The mission will start at San Diego, California (32.6843, -117.2302) with a target location of Australia (-11.3211, 136.1991). The mission will be set to return to the starting location with no stoppage time. The first mission was given a requirement of AD with a value of 1 and has no prerequisites. The second mission was also given the requirement of AD with a value of 1 but has the prerequisite of the first mission. Every unit in the WOP has its capabilities set to full. Based on the closest distance to the starting location, the USS Theodore Roosevelt (CVN-71) was chosen for the first mission. This unit was set to have a maximum speed of 30 knots and it is assumed that, for ease of timeline progression, the unit will always be operating at the maximum speed. The USS Spruance (DDG-111) was chosen for the second mission. The unit has a maximum speed of 30 knots as well. The second mission's prerequisite is the first mission. There is one path exclusion set for 1.2841, 167.3315 with a minimum distance of 100nm for both missions to illustrate the navigational capabilities of the WOP. Figure 5-4 shows the entire progression of the USS Theodore Roosevelt as well as the USS Spruance from the starting location, to the target location, and back to the homeport.

Fig. 5-4. Test mission from San Diego, CA to Australia and back

The first mission for the USS Theodore Roosevelt took 18 days 23 hours (455 hours) to complete from start to finish without any stops. The USS Spruance was able to start its mission at hour 456 and it also took 455 hours to complete. Both missions together took 37 days 22 hours (910 hours) to complete. It is worth noting that the path calculation (as seen in Figure 5-4) around Papua New Guinea is not perfect. This is likely due to the level of detail of the imported KML file of country borders[20].

5.3 Possible solution generation

Possible solutions are generated by fully enumerating all possible combinations of available units with all preclusions for those units that can contribute to the mission's requirements. As the number of mission requirements increase in complexity, the time to enumerate all possible solutions increases dramatically. To show this, Table 5-4 lists several different mission generation scenarios. It is assumed that all units are fully capable and have no degraded capabilities. For solutions which would require more than one unit, the exclusionary radius of determining available units was modified to WOP run time. All scenarios were run on the same machine with an Intel Core i7-4790K CPU and 16GB of ram. The WOP was configured to use up to eight threads for mission solution generation. It can be seen, in this example, a modest increase in initial solutions may not provide a substantial increase in total solutions but the generation time does not scale proportionally.

---

[20] Sivathapandia, A. (2013, May 13). *World Country Borders KML*. Retrieved from https://www.arcgis.com/home/item.html?id=7a8585998b7f470b85235dcdb560c7e2

Requirement(s)	Initial Solutions	Total Solutions	Radius (nm)	Generation Time
INTEL – 1	177	177	None	1 sec.
INTEL – 2	42	1241	2000	2 mins. 38 secs.
INTEL – 2	81	4400	3000	22 mins. 1 sec.
INTEL – 3	42	18548	2000	40 mins. 42 secs.
INTEL – 3	81	19262	3000	19 hrs. 29 mins. 39 secs.

Table 5-4. WOP mission solution generation run time examples

5.4 ELECTRE III threshold analysis

The ELECTRE III procedure is proven useful for a variety of applications including: selection of hedge fund portfolios[21], requalification of an abandoned quarry[22], or revising a gas station's inventory process[23]. The recurring theme turns out to be in the proper selection of the discrimination thresholds. Without proper guidance or extensive statistical knowledge, the selection of threshold values, by an inexperienced user, will prove ineffective in comparing alternatives. The following examples of threshold values will help to illustrate the difficulty in selecting proper threshold values for the WOP. Tables 5-5, 5-6, 5-7 and 5-8 are examples of two different alternatives based on the criteria of distance to mission target area. The tables explore the concordance and discordance values for minimization or maximization.

---

[21] Chandrasekaram, S. (2001). An Innovative Way of Selecting Portfolios: the ELECTRE Models, presented at l'Institut de Science Financière et d'Assurances pour l'obtention du diplôme d'Actuaire de l'Université de Lyon, France, 2011.

[22] Bottero, M., Ferretti, V., Figueira, J., Greco, S., & Roy, B. (2015). Dealing with a multiple criteria environmental problem with interaction effects between criteria through an extension of the Electre III method. *European Journal of Operational Research*, *245*. doi: 10.1013/j.ejro.2015.04.005

[23] Milani, A., Shanian, A., & El-Lahham, C. (2006). Using different electre methods in strategic planning of human behavioral resistance. *Journal of Applied Mathematics and Decision Sciences, 2006*, 1-19. doi: 10.115/JAMDS/2006/10936

a_i	b_i	p_j	q_j	v_j	$c_j(a, b)$ max	$c_j(a, b)$ min	$D_j(a, b)$ max	$D_j(a, b)$ min
1000	1600	1900	500	2000	0.928571	1	0	0
1000	1600	1800	500	2000	0.923077	1	0	0
1000	1600	1700	500	2000	0.916667	1	0	0
1000	1600	1600	500	2000	0.909091	1	0	0
1000	1600	1500	500	2000	0.9	1	0	0
1000	1600	1400	500	2000	0.888889	1	0	0
1000	1600	1300	500	2000	0.875	1	0	0
1000	1600	1200	500	2000	0.857143	1	0	0
1000	1600	1100	500	2000	0.833333	1	0	0
1000	1600	1000	500	2000	0.8	1	0	0
1000	1600	900	500	2000	0.75	1	0	0
1000	1600	800	500	2000	0.666667	1	0	0
1000	1600	700	500	2000	0.5	1	0	0
1000	1600	600	500	2000	0	1	0	0
1000	1600	500	500	2000	0	1	0.066667	0
1000	1600	400	500	2000	0	1	0.125	0
1000	1600	300	500	2000	0	1	0.176471	0
1000	1600	200	500	2000	0	1	0.222222	0

Table 5-5. Preference threshold variation analysis

a_i	b_i	p_j	q_j	v_j	$c_j(a, b)$ max	$c_j(a, b)$ min	$D_j(a, b)$ max	$D_j(a, b)$ min
1000	1600	1000	950	2000	1	1	0	0
1000	1600	1000	900	2000	1	1	0	0
1000	1600	1000	850	2000	1	1	0	0
1000	1600	1000	800	2000	1	1	0	0
1000	1600	1000	750	2000	1	1	0	0
1000	1600	1000	700	2000	1	1	0	0
1000	1600	1000	650	2000	1	1	0	0
1000	1600	1000	600	2000	1	1	0	0
1000	1600	1000	550	2000	0.888889	1	0	0
1000	1600	1000	500	2000	0.8	1	0	0
1000	1600	1000	450	2000	0.727273	1	0	0
1000	1600	1000	400	2000	0.666667	1	0	0
1000	1600	1000	350	2000	0.615385	1	0	0
1000	1600	1000	300	2000	0.571429	1	0	0
1000	1600	1000	250	2000	0.533333	1	0	0
1000	1600	1000	200	2000	0.5	1	0	0
1000	1600	1000	150	2000	0.470588	1	0	0
1000	1600	1000	100	2000	0.444444	1	0	0

Table 5-6. Indifference threshold variation analysis

a_i	b_i	p_j	q_j	v_j	$c_j(a, b)$ max	$c_j(a, b)$ min	$D_j(a, b)$ max	$D_j(a, b)$ min
1000	1200	100	110	0	0	1	1	0
1000	1200	100	110	100	0	1	1	0
1000	1200	100	110	200	0	1	1	0
1000	1200	100	110	300	0	1	0.5	0
1000	1200	100	110	400	0	1	0.333333	0
1000	1200	100	110	500	0	1	0.25	0
1000	1200	100	110	600	0	1	0.2	0
1000	1200	100	110	700	0	1	0.166667	0
1000	1200	100	110	800	0	1	0.142857	0
1000	1200	100	110	900	0	1	0.125	0
1000	1200	100	110	1000	0	1	0.111111	0
1000	1200	100	110	1100	0	1	0.1	0
1000	1200	100	110	1200	0	1	0.090909	0
1000	1200	100	110	1300	0	1	0.083333	0
1000	1200	100	110	1400	0	1	0.076923	0
1000	1200	100	110	1500	0	1	0.071429	0
1000	1200	100	110	1600	0	1	0.066667	0
1000	1200	100	110	1700	0	1	0.0625	0

Table 5-7. Veto threshold variation analysis

The tables show three examples of how varying the threshold values can affect the concordance and discordance indices for two alternatives. Table 5-5 shows alternative *a*, with the value of 1000nm, and alternative *b*, with a value of 1600nm, that a decreasing preference threshold makes alternative *a* less and less preferable to alternative *b* (when maximizing distance to the target) to the point where the preference threshold is 600nm at which alternative *a* not preferable to alternative *b*. As the preference threshold decreases below 600nm, alternative *a* becomes slightly more preferable to alternative *b* but not to the point where the concordance index increases above zero.

The data in Table 5-6 demonstrates how changing the indifference threshold between two alternatives affects concordance and discordance. The alternatives have the same criteria values as in Table 5-5. As the indifference threshold remains above 550nm, alternative *a* is preferable to alternative *b* (when trying to maximize the criteria). When the indifference threshold steadily decreases below 600nm, alternative *a* becomes less and less preferable to alternative *b*. When minimizing the criteria, leaving the preference threshold unchanged, alternative *a* stays preferable to alternative *b*. Table 5-7 shows how the veto threshold affects two alternatives.

Alternative *a* was given the criteria value of 1000nm while alternative *b* has a 1200nm value. With the preference and indifference thresholds of 100nm and 110nm alternative *a* is preferable to alternative *b* always when the goal is to minimize the criteria. With maximization of the criteria, alternative *a* becomes slightly more preferable to alternative *b* as the veto threshold increases. Alternative *a* never becomes preferable to alternative *b* with the values presented in the table. To show how the modification of two threshold values at the same time present a more robust picture of the how the ELECTRE III thresholds can affect the comparison of alternatives. Tables 5-8, 5-9, and 5-10 display three additional examples of changing multiple threshold values at the same time.

a_i	b_i	p_j	q_j	v_j	$c_j(a, b)$ max	$c_j(a, b)$ min	$D_j(a, b)$ max	$D_j(a, b)$ min
1600	1000	2000	2000	2000	1	1	0	0
1600	1000	1900	1900	2000	1	1	0	0
1600	1000	1800	1800	2000	1	1	0	0
1600	1000	1700	1700	2000	1	1	0	0
1600	1000	1600	1600	2000	1	1	0	0
1600	1000	1500	1500	2000	1	1	0	0
1600	1000	1400	1400	2000	1	1	0	0
1600	1000	1300	1300	2000	1	1	0	0
1600	1000	1200	1200	2000	1	1	0	0
1600	1000	1100	1100	2000	1	1	0	0
1600	1000	1000	1000	2000	1	1	0	0
1600	1000	900	900	2000	1	1	0	0
1600	1000	800	800	2000	1	1	0	0
1600	1000	700	700	2000	1	1	0	0
1600	1000	600	600	2000	1	1	0	0
1600	1000	500	500	2000	1	0	0	0.066667
1600	1000	400	400	2000	1	0	0	0.125
1600	1000	300	300	2000	1	0	0	0.176471

Table 5-8. Preference and indifference threshold variation analysis

a_i	b_i	p_j	q_j	v_j	$c_j(a, b)$ max	$c_j(a, b)$ min	$D_j(a, b)$ max	$D_j(a, b)$ min
1600	1000	2000	500	0	1	0.933333	0	0
1600	1000	1900	500	100	1	0.928571	0	0
1600	1000	1800	500	200	1	0.923077	0	0
1600	1000	1700	500	300	1	0.916667	0	0
1600	1000	1600	500	400	1	0.909091	0	0
1600	1000	1500	500	500	1	0.9	0	0
1600	1000	1400	500	600	1	0.888889	0	0
1600	1000	1300	500	700	1	0.875	0	0
1600	1000	1200	500	800	1	0.857143	0	0
1600	1000	1100	500	900	1	0.833333	0	0
1600	1000	1000	500	1000	1	0.8	0	0
1600	1000	900	500	1100	1	0.75	0	0
1600	1000	800	500	1200	1	0.666667	0	0
1600	1000	700	500	1300	1	0.5	0	0
1600	1000	600	500	1400	1	0	0	0
1600	1000	500	500	1500	1	0	0	0.1
1600	1000	400	500	1600	1	0	0	0.166667
1600	1000	300	500	1700	1	0	0	0.214286

Table 5-9. Preference and veto threshold variation analysis

a_i	b_i	p_j	q_j	v_j	$c_j(a, b)$ max	$c_j(a, b)$ min	$D_j(a, b)$ max	$D_j(a, b)$ min
1600	1000	1000	1900	0	1	1	0	0
1600	1000	1000	1800	100	1	1	0	0
1600	1000	1000	1700	200	1	1	0	0
1600	1000	1000	1600	300	1	1	0	0
1600	1000	1000	1500	400	1	1	0	0
1600	1000	1000	1400	500	1	1	0	0
1600	1000	1000	1300	600	1	1	0	0
1600	1000	1000	1200	700	1	1	0	0
1600	1000	1000	1100	800	1	1	0	0
1600	1000	1000	1000	900	1	1	0	0
1600	1000	1000	900	1000	1	1	0	0
1600	1000	1000	800	1100	1	1	0	0
1600	1000	1000	700	1200	1	1	0	0
1600	1000	1000	600	1300	1	1	0	0
1600	1000	1000	500	1400	1	0.8	0	0
1600	1000	1000	400	1500	1	0.666667	0	0
1600	1000	1000	300	1600	1	0.571429	0	0
1600	1000	1000	200	1700	1	0.5	0	0

Table 5-10. Indifference and veto threshold variation analysis

The previous three tables provide a more complete picture of how the changing of two ELECTRE III threshold variables can impact the comparison of two alternatives. Table 5-8 shows that, with the presented criteria values, alternative *a* is preferable to alternative *b* when maximizing the criteria no matter how the preference and indifference thresholds change. For the minimization of the criteria, alternative *a* is also preferable to alternative *b* only to the point where both the preference and indifference thresholds reach 500nm. As the two thresholds continue to decrease, alternative *a* becomes slightly more preferable to alternative *b* but only to the point where both the preference and indifference thresholds reach 300nm. At this point, alternative *a* starts to become progressively less preferable to alternative *b*.

When the preference and veto thresholds are modified at the same time, with the two criteria values as specified in Table 5-9, alternative *a* is always preferable to alternative *b* when the preference is maximization. For minimization, alternative *a* is preferable to a decreasing degree to the point where the preference threshold reaches 600nm and the veto threshold reaches 1400nm. As the preference continues to decrease while the veto increases, alternative *a* becomes increasingly less preferable to alternative *b*. When the indifference and veto thresholds change in the opposite directions of each other, results like Table 5-10 can occur.

The maximization preference shows that alternative *a* is always preferable to alternative *b*. When minimizing the criteria, alternative *a* is also preferable to alternative *b* but only to the point where the indifference reaches 500nm and the veto reaches 1400nm. As the indifference continues to decrease while the veto increases, alternative *a* becomes less preferable to alternative *b* but not to the point where the discordance index reaches a level above zero. The six examples of how threshold values can affect alternative comparison do not help to convince the DM of how to choose threshold values but present a picture that choosing the correct threshold values for each situation is a difficult task. The current criteria built into the WOP need to be fully understood in order to make an informed decision on what to set threshold values at.

5.5 Conventional vs ELECTRE III solution sorting

The WOP provides the user two main options in sorting possible solutions for use for a mission. The conventional method is more easily understandable (but works only with quantitative data) as it uses standard ascending and descending sorts. The WOP presents the user the ability to sort by one of more of the following quantitative options (either ascending or descending): underages, overages, distance to starting point, distance to target, and number of units. If sorting by more than one option, a custom version of a stable sort algorithm is used to preserve “the relative order of the elements with equivalent values”[24]. The stable sort implements what is more commonly known as a secondary (as well as additional levels) sort. Table 5-11 provides an example of an initial sort, secondary sort, and tertiary sort. The keys in Table 5-11 are sorted all descending in the order: Key1, Key2, and then Key3.

Original values			First Sort			Second Sort			Third Sort		
Key1	**Key2**	**Key3**	**Key1**	**Key2**	**Key3**	**Key1**	**Key2**	**Key3**	**Key1**	**Key2**	**Key3**
B	12	Z	A	33	S	A	0	S	A	0	S
A	33	S	A	0	S	A	33	S	A	33	S
C	3	A	B	12	Z	B	9	R	B	9	F
B	9	F	B	9	R	B	9	F	B	9	R
D	87	E	B	9	F	B	12	Z	B	12	Z
B	9	R	C	3	A	C	3	A	C	3	A
F	6	T	D	87	E	D	87	E	D	87	E
A	0	S	F	6	T	F	6	T	F	6	T

Table 5-11. Stable sort example

---

[24] cplusplus.com. (2017). *std::stable_sort*. Retrieved from http://www.cplusplus.com/reference/algorithm/stable_sort/

The example sort values in Table 5-11 uses arrows so that the relative positions of the rows can be more easily followed as each iteration of the sort is performed. This table is elementary but helps to understand how the WOP uses a customized stable sort to sort possible missions based on the user's preferences. In the next example with units in the WOP, the simple sorting algorithm can be very effective and may work for most scenarios. The example mission has a starting location of Naval Station Norfolk, Virginia (36.964299, -76.327480) and a target location of the Bristol Channel (51.4330593, -4.5308297) off the coast of The United Kingdom. This mission has the requirements of INTEL (2), AD (2), and MINE (1). The ideal solution would have only two total units (if the user wanted to minimize the number of units). For this mission, Table 5-12 shows the available units and their capabilities. It is worth nothing that, for ease of calculation, the WOP computes distance between points directly, ignoring actual navigational routes.

Unit	Capabilities	Distance To Start	Distance To Target
CVN-72	AD (1), INTEL(1), S (1)	5.97nm	3083.11nm
CVN-70	AD (1), INTEL (0.5), S (1)	2017.96nm	4643.92nm
CVN-77	AD (1), INTEL (0.5)	0nm	3080nm
CVN-78	AD (0.5), INTEL (1)	0nm	3080nm
MCM-3	MINE (1), MCM (1)	2017.96nm	4643.92nm
MCM-9	MINE (1), MCM (1)	6243.34nm	5080.59nm
DDG-81	AD (1), MIO (1), ASW (1)	0nm	3080nm
DDG-68	AD (1), INTEL (1)	468.86nm	3533.56nm

Table 5-12. Sample mission available units with capabilities

The units in Table 5-12 can each satisfy some part of the requirements of the sample mission. CVN-72 and CVN-70 will have overages on the S capability as it is not needed for the mission. CVN-77 and CVN-78 each have an underage of 0.5 of the INTEL and AD capabilities respectively. The MCM-3 and MCM-9 both have an overage of the MCM capability, as it is not needed. DDG-81 has overages of the MIO and ASW capabilities but DDG-68 has no overages or underages. The WOP was able to find 2653 possible mission solutions that were able to either fully or partially satisfy the requirements of this test mission. The total calculation time was 12 minutes and 21 seconds.

Using a single criteria sort method for the number of units for the generation of possible solutions, Tables 5-13 and 5-14 show truncated results for the preference of minimization and maximization of the criterion where the lower ranked the solution is, the more preferable it is.

Rank	Solution	Primary Mission	# of Units	Additional Information
1	DDG-81	AD	1	Overage: 3, Underage: 4, Total Distance: 0.55256, Total Target Distance: 3077.7
2	CVN-77	AD	1	Overage: 3, Underage: 4, Total Distance: 0.55256, Total Target Distance: 3077.7
3	CVN-70	AD	1	Overage: 3, Underage: 4, Total Distance: 2016.77, Total Target Distance: 3077.7
4	CVN-72	AD	1	Overage: 3, Underage: 4, Total Distance: 5.96357, Total Target Distance: 3077.7
5	CVN-78	AD	1	Overage: 3, Underage: 4.5, Total Distance: 0.55256, Total Target Distance: 3077.7
6	DDG-68	AD	1	Overage: 3, Underage: 4, Total Distance: 468.58, Total Target Distance: 3077.7
7	CVN-72	AD	2	Overage: 4, Underage: 3, Total Distance: 6.51613, Total Target Distance: 6155.9
	DDG-81	INTEL		
8	CVN-70	AD	2	Overage: 3, Underage: 3.5, Total Distance: 2017.33, Total Target Distance: 6155.9
	CVN-78	AD		
…	…	…	…	…
2652	CVN-72	AD	8	Overage: 6, Underage: 0, Total Distance: 10749.4, Total Target Distance: 30203.8
	CVN-70	AD		
	CVN-77	MINE		
	CVN-78	INTEL		
	DDG-68	INTEL		
	MCM-3	INTEL		
	DDG-81	INTEL		
	MCM-9	INTEL		
2653	CVN-72	MINE	8	Overage: 4, Underage: 0, Total Distance: 10749.4, Total Target Distance: 29753.6
	CVN-70	MINE		
	CVN-77	INTEL		
	CVN-78	MINE		
	DDG-68	AD		

	MCM-3	AD		
	DDG-81	INTEL		
	MCM-9	MINE		

Table 5-13. Sample mission solutions sorted by min units ascending

Rank	**Solution**	**Primary Mission**	**# of Units**	**Additional Information**
1	DDG-81	AD	1	Overage: 3, Underage: 4, Total Distance: 0.55256, Total Target Distance: 3077.7
2	CVN-77	AD	1	Overage: 3, Underage: 4, Total Distance: 0.55256, Total Target Distance: 3077.7
3	CVN-78	AD	1	Overage: 3, Underage: 4.5, Total Distance: 0.55256, Total Target Distance: 3077.7
4	CVN-72	AD	1	Overage: 3, Underage: 4, Total Distance: 5.96357, Total Target Distance: 3077.7
5	DDG-68	AD	1	Overage: 3, Underage: 4, Total Distance: 468.58, Total Target Distance: 3077.7
6	CVN-70	AD	1	Overage: 3, Underage: 4, Total Distance: 2016.77, Total Target Distance: 3077.7
7	CVN-72	AD	2	Overage: 4, Underage: 3, Total Distance: 6.51613, Total Target Distance: 6155.9
	DDG-81	INTEL		
8	CVN-70	AD	2	Overage: 3, Underage: 3.5, Total Distance: 2017.33, Total Target Distance: 6155.9
	CVN-78	AD		
…	…	…	…	…
2652	CVN-72	MINE	8	Overage: 4, Underage: 0, Total Distance: 10749.4, Total Target Distance: 29753.6
	CVN-70	MINE		
	CVN-77	INTEL		
	CVN-78	MINE		
	DDG-68	AD		
	MCM-3	AD		
	DDG-81	INTEL		
	MCM-9	MINE		
2653	CVN-72	AD	8	Overage: 6, Underage: 0, Total Distance: 10749.4, Total Target Distance: 30203.8
	CVN-70	AD		
	CVN-77	MINE		
	CVN-78	INTEL		
	DDG-68	INTEL		
	MCM-3	INTEL		
	DDG-81	INTEL		
	MCM-9	INTEL		

Table 5-14. Sample mission solutions sorted by min units and distance ascending

Though the examples in Tables 5-13 and 5-14 are not useful in terms of choosing the best possible mission solution, they show that, with minimal effort, the user can sort potential mission solutions without advanced statistical knowledge. Taking the same scenario with the requirements of INTEL (2), AD (2), and MINE (1) and the unit capabilities in Table 5-12, the next several examples will demonstrate how the possible solutions could be ranked with an ELECTRE III procedure. The thresholds and weights specified are arbitrary and were chosen in an attempt to show how varying input values for ELECTRE III can affect the ranking of possible mission solutions. For each of the distance criteria (distance to starting point and distance to target location) the WOP provides the user the ability to choose whether the ELECTRE III procedure should take into account the sum of the distance values, the single largest distance value, or the single smallest distance value. Table 5-15 shows results for one ELECTRE III run.

Criteria	Preference	*P*	*Q*	*V*	Weight
Largest distance to start	Minimize	250	750	1200	0.2
Largest distance to target	Minimize	1000	2000	3000	0.1
Overages	Minimize	0	2	4	0.05
Underages	Minimize	0	2	4	0.5
Number of units	Maximize	2	4	6	0.15

Table 5-15. Sample #1 ELECTRE III values for a mission

The results of the ELECTRE III method with the values from Table 5-15 can be difficult to interpret from a DMs perspective. This can be due to the values that were chosen for the first sample. The results of the first run show that the best possible mission solution uses six units (CVN-70, CVN-78, DDG-68, DDG-81, MCM-9, and MCM-3) with the following respective primary missions: INTEL, INTEL, MINE, AD, AD, and INTEL. The overages and underages for this mission are 3.5 and 1.5. The largest start and target distances are 6239.69nm and 5077.63nm. The worst possible mission uses six units (CVN-72, CVN-70, DDG-68, DDG-81, MCM-9, and MCM-3) with the primary mission areas of INTEL, AD, INTEL, INTEL, AD, and

AD and the overage and underage values of 4.5 and 1. The worst solution has the same distance values as the best solution. These results seem counterintuitive. The next sample run explores the impact that redistributing the criteria weights has on the ranking of potential solutions. Table 5-16 shows the values used for the ELECTRE III execution.

Criteria	**Preference**	***P***	***Q***	***V***	**Weight**
Largest distance to start	Minimize	250	750	1200	0.5
Largest distance to target	Minimize	1000	2000	3000	0.25
Overages	Minimize	0	2	4	0.05
Underages	Minimize	0	2	4	0.1
Number of units	Maximize	2	4	6	0.1

Table 5-16. Sample #2 ELECTRE III values for a mission

With the change in weights for the second sample, the best solution changed to the units CVN-72, CVN-70, DDG-68, DDG-81, MCM-9, and MCM-3 with the primary missions of INTEL, AD, AD, AD, INTEL, and INTEL. This solution still has 1 underage but has 6.5 overages. The largest distances remained unchanged. The worst solution is slightly better than the worst solution for the first sample run. The worst solution on this sample used the units CVN-70, CVN-78, DDG-68, DDG-81, MCM-9, and MCM-3 with the capabilities of INTEL, INTEL, MINE, AD, AD, and INTEL. This solution has an overage of 3.5 and an underage of 1.5. For the third sample, the weights are changed and the threshold values are also changed in an attempt to provide solutions that prefer no underages using the same criteria.

Criteria	**Preference**	***P***	***Q***	***V***	**Weight**
Largest distance to start	Minimize	250	750	1200	0.05
Largest distance to target	Minimize	1000	2000	3000	0.05
Overages	Minimize	0	1	2	0.1
Underages	Minimize	0	1	2	0.6
Number of units	Maximize	2	4	6	0.2

Table 5-17. Sample #3 ELECTRE III values for a mission

With the modified threshold values, the results from the third run were similar to that of the first two. The ELECTRE III procedure did not prefer possible solutions with no underages ahead of those solutions that could not fully meet the requirements. The next ELECTRE III sample run includes no veto threshold values. Table 5-18 uses all of the same threshold and weights as in Table 5-17 with the exception of the veto threshold.

Criteria	Preference	*P*	*Q*	*V*	Weight
Largest distance to start	Minimize	250	750	0	0.05
Largest distance to target	Minimize	1000	2000	0	0.05
Overages	Minimize	0	1	0	0.1
Underages	Minimize	0	1	0	0.6
Number of units	Maximize	2	4	0	0.2

Table 5-18. Sample #4 ELECTRE III values for a mission

Even with the removal of the veto threshold, the results of the ELECTRE III procedure did not generate results that were dissimilar to the first three iterations. In attempt to isolate a threshold variable that would have a noticeable impact the ranking of the criteria, the WOP was used to perform an OAT sensitivity analysis from the sample #4 data. Using the threshold ranges in Table 5-19, the WOP performed 300 ELECTRE III permutations per each of the four variables (P, Q, V, and weights) for each of the five criteria.

Variable	Criteria	Values
P	Largest distance to start	125, 150, 175, 200, 225, 400, 425, 450, 475, 500, 500, 600, 700, 800, 900, 1600, 1700, 1800, 1900, 2000
	Largest distance to target	125, 150, 175, 200, 225, 400, 425, 450, 475, 500, 500, 600, 700, 800, 900, 1600, 1700, 1800, 1900, 2000
	Overages	0.5, 1, 1.2, 1.4, 1.5, 1.6, 1.8, 2, 2.5, 3, 3.2, 3.4, 3.5, 3.6, 3.8, 4, 4.5, 5, 5.5, 6
	Underages	0.5, 1, 1.2, 1.4, 1.5, 1.6, 1.8, 2, 2.5, 3, 3.2, 3.4, 3.5, 3.6, 3.8, 4, 4.5, 5, 5.5, 6
	Number of units	0.5, 1, 1.2, 1.4, 1.5, 1.6, 1.8, 2, 2.5, 3, 3.2, 3.4, 3.5, 3.6, 3.8, 4, 4.5, 5, 5.5, 6
Q	Largest distance to start	675, 600, 525, 450, 375, 1200, 1275, 1350, 1425, 1500, 1800, 1600, 1400, 1200, 1000, 3200, 3400, 3600, 3800, 4000

	Largest distance to target	675, 600, 525, 450, 375, 1200, 1275, 1350, 1425, 1500, 1800, 1600, 1400, 1200, 1000, 3200, 3400, 3600, 3800, 4000
	Overages	0, 0.5, 0.6, 0.7, 0.8, 0.9, 1.6, 1.7, 1.8, 1.9, 2, 2.4, 2.8, 3.2, 3.6, 6.4, 6.8, 7.2, 7.6, 8
	Underages	0, 0.5, 0.6, 0.7, 0.8, 0.9, 1.6, 1.7, 1.8, 1.9, 2, 2.4, 2.8, 3.2, 3.6, 6.4, 6.8, 7.2, 7.6, 8
	Number of units	0, 0.5, 0.6, 0.7, 0.8, 0.9, 1.6, 1.7, 1.8, 1.9, 2, 2.4, 2.8, 3.2, 3.6, 6.4, 6.8, 7.2, 7.6, 8
V	Largest distance to start	500, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000
	Largest distance to target	500, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000
	Overages	500, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000
	Underages	0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5
	Number of units	0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5
W	Largest distance to start	0.045, 0.04, 0.035, 0.03, 0.025, 0.08, 0.085, 0.09, 0.095, 0.1
	Largest distance to target	0.045, 0.04, 0.035, 0.03, 0.025, 0.08, 0.085, 0.09, 0.095, 0.1
	Overages	0.09, 0.08, 0.07, 0.06, 0.05, 0.16, 0.17, 0.18, 0.19, 0.2
	Underages	0.54, 0.48, 0.42, 0.36, 0.3, 0.96, 1.02, 1.08, 1.14, 1.2
	Number of units	0.18, 0.16, 0.14, 0.12, 0.1, 0.32, 0.34, 0.36, 0.38, 0.4

Table 5-19. ELECTRE III sensitivity values

After each of the runs of ELECTRE III for each of the values in Table 5-19, the WOP compared the original position for each of the alternatives in the original ELECTRE III procedure and the new position given the modified threshold or weight value. With each of the values specified in Table 5-19, the positions for all of the alternatives were not affected. This could be due to the fact that: (1) ELECTRE III might be best designed for scenarios in which there exist more fuzzy or qualitative data than purely quantitative data, (2) the combination of the threshold and weights was not sufficient to cause a change in the ranking of alternatives, or (3) the number of alternatives was too narrow to fully explore the capabilities of the ELECTRE III method.

The difficultly in selecting relevant threshold values for ELECTRE III lies in the fundamental design of the algorithm. Instead of a DM choosing values that are preferred, the choosing of values in which the difference between the criteria for alternative *a* and alternative *b* is within the specified criteria. This concept of choosing the preferred and indifferent difference of two alternatives is harder to grasp than a strict preference value of one alternative over another. This change in decision making requires that a DM fully understand the criteria involved in the decision making process and how the comparison of one alternative over another is preferable (or not). Tables 5-5 through 5-10 (previously shown in this section) show how the ELECTRE III procedure can affect the concordance of one alternative over another (with regards to one criteria).

CHAPTER 6

FUTURE IMPROVEMENTS

The WOP can be beneficial if implemented correctly by a DM. There are several pieces of functionality that should be added to further improve upon its usefulness. There are also other aspects of the WOP that should be modified or improved to more quickly allow the DM to make an informed decision. The WOP, being command-line based, is more portable for different operating systems but this makes it more difficult to modify program values and execute certain functions. If implemented correctly, the conversion of the WOP into a graphical user interface (GUI) based program would improve the understandability of the WOP and allow the complicated menu structure to be redesigned in a way that the user will be able to have quicker (and more granular) access to the program's data and settings.

The implementation of a GUI would also allow for the results of the mission solution generation to be easily viewed in a structured way instead of through the current way of generated CSV files. The CSV files generated for the results of the ELECTRE III procedure become unmanageable with any real number of alternatives because the sizes of the concordance and discordance matrices are a square of the number of alternatives. The GUI would also help the user manipulate preclusions, capabilities, and ELECTRE III inputs as all of these key program settings are matrix-based. The representation and manipulation of a matrix in a command-line environment is difficult.

The WOP should also be redesigned with memory management and efficiency as the top priorities. The system's memory quickly becomes used the more alternatives there are when using ELECTRE III. The WOP implements a smart pointer system (introduced in C++11) called a shared pointer which allows the compiler to manage the destruction of pointers automatically when the pointer can no longer be reached by any running code[25]. The use of smart pointers allows the WOP to release memory as best as possible but the program suffers from the lack of ability to manage large amounts of data that needs to be loaded and used. The best example of this is in the sensitivity analysis that the WOP implements for comparing ELECTRE III results. An improvement in the looping that occurs through the key functions in the WOP (sensitivity analysis, mission solution generation, and land avoidance) is necessary to reduce the cycles and time necessary for the completion of these functions.

The WOP implements C++ threads as much as possible to reduce the total time taken to complete these functions but there are key areas that can still be threaded which will have a positive impact on the time the user spends waiting for computation to complete. The ranking of mission solutions with ELECTRE III needs to be further analyzed to better implement threads to reduce the time for completion. The determination of distillation results is not threaded and takes a considerable amount of time to complete. The distillation procedures can be threaded one of three ways: (1) creating two threads to run simultaneously (one for ascending and one for descending distillation), (2) modifying each of the distillation functions to independently use as many threads as possible for the computer the WOP is running on but have the procedures run one at a time, or (3) a combination of the first two options. The first two choices are easier to implement and the third choice is complex and might not be any better in terms of performance than the first two.

---

[25] cplusplus.com. (2017). *std::shared_ptr*. Retrieved from http://www.cplusplus.com/reference/memory/shared_ptr/

The ELECTRE III procedure is proven in studies to work well but, for the purely quantitative implementation in the current version of the WOP, it is not as quick at generating believable results as a simple sort is. The modification of the WOP to allow for dynamic ELECTRE III criteria would be beneficial as it would allow a DM to explore a range of additional qualitative and quantitative options for missions. In practical terms of defining intelligent threshold values for the quantitative criteria that is written into the WOP program, more often than not, a DM would not have enough knowledge to define these values in a way where the WOP can use ELECTRE III to be useful. As more and more scenarios are successfully planned with ELECTRE III, DMs will build up a repertoire of knowledge of how to choose threshold values and weights for varying scenarios.

The WOP could also be improved by including specific qualitative and quantitative factors more fine-grained that are of direct impact to mission requirements. The availability of food, fuel, and weapon stores should be included as additional quantitative factors into the WOP. Also, an additional qualitative input which takes into account the specific DMs preference for specific units or mission solutions depending on specific high value or sensitive targets[26]. In addition, with the qualitative factors proposed by Robert Silva, there could be additional related quantitative factors such as: distance before refueling, reduction in vessel speed while refueling, or downtime for repairs.

To validate the chosen criteria implemented in the WOP and discover criteria for a future implementation, the results and methods were discussed with a retired petty officer that was

---

[26] Silva, R. (2017, July 4). Email.

deployed on the USS George H.W. Bush (CVN-77). The petty officer identified the following criteria that would be useful and play a role in determining vessel selections for a mission: (1) minimum number of personnel that are qualified in each rate necessary for proper maintenance, operation, and protection of the vessel, (2) capabilities available and level of availability of each capability, (3) last deployment period, (4) intended schedule of overhaul or maintenance, and (5) the mission's target area[27]. Criteria #1 can be implemented either as a single factor (total percentage of personnel qualified) or individually by rate. Implementing it as a single factor would make the ELECTRE III calculations quicker and the determination of threshold values simpler.

The criteria #2 is already implemented in the WOP as the preclusion matrices. Criteria #3 can be implemented in the future as number of days since the last deployment. The determination of the appropriate amount of days between deployments can be tweaked by the DM according to Navy standards and regulations which would also prevent vessels from being selected as viable for a mission. This will allow the WOP to more quickly enumerate all available mission solutions. Criteria #4 could be difficult to implement in any capacity other than making sure that, for each vessel in each mission, the number of days before a scheduled maintenance operation minus the number of days for a mission is greater than zero. Any number less than zero for a unit would indicate that the mission would overlap the intended maintenance schedule and it could be undesirable to use that particular unit for the mission.

[27] Retired PO2. (2017, September 20). Interview.

Criteria #5 is implemented in the WOP as the distance to a mission's target area. It was suggested to have this changed to the location of the target area in either the Atlantic or Pacific fleet's area of responsibility (AOR). It was determined from my interview with the retired sailor that, for each mission, units are generally chosen based on the AOR of the mission's target area. For example, if the target area is in the Pacific fleet's AOR, units that are also based in that same AOR would be preferable to units that are in Atlantic fleet's AOR. I would suggest that the WOP be modified so that, in addition to distance to the target area, Criteria #5 also be added as an added level of customizability for the DM.

REFERENCES

Boost [Computer software]. (2017). Retrieved from http://www.boost.org/

Bottero, M., Ferretti, V., Figueira, J., Greco, S., & Roy, B. (2015). Dealing with a multiple criteria environmental problem with interaction effects between criteria through an extension of the Electre III method. *European Journal of Operational Research, 245*. doi: 10.1013/j.ejro.2015.04.005

Bouyssou, D. (2008). Outranking Methods. *Encyclopedia of Optimization*, 2887-2893.

Brooks, L. (1986). *Naval power and national security: The case for the maritime strategy* [Electronic version]. International Security, 11(2), 58-88

Chandrasekaram, S. (2001). An Innovative Way of Selecting Portfolios: the ELECTRE Models, presented at I'Institut de Science Financière et d'Assurances pour I'obtention du diplôme d'Actuaire de I'Université de Lyon, France, 2011.

cplusplus.com. (2017). *std::stable_sort.* Retrieved from http://www.cplusplus.com/reference/algorithm/stable_sort/

cplusplus.com. (2017). *std::shared_ptr.* Retrieved from http://www.cplusplus.com/reference/memory/shared_ptr/

Department of the Navy. (2013). *Navy Planning.* Retrieved from U.S. Naval War College (https://www.usnwc.edu/getattachment/171afbf3-a1e2-46b3-b1e9-d1fa4b0fec5a/5-01_(Dec_2013)_(NWP)-(Promulgated).aspx)

Dugan, Kevin. (2007). *Navy Mission Planner.* Retrieved from Naval Postgraduate School (https://calhoun.nps.edu/handle/10945/3317)

Giannoulis, C. & Ishizaka, A. (2010). A Web-based decision support system with ELECTRE III for a personalized ranking of British universities. *Decision Support Systems, 48*, 488-497.

Google. (n.d.). [Google Maps base layer for countries close to Naval Station Norfolk]. Retrieved August 19, 2017, from https://goo.gl/maps/R3tq1BSELD92

Google Earth [Computer software]. (2017). Retrieved from https://www.google.com/earth/

Hoang, Q. (2016, June 3). *Electre III method implementation in C++*. Retrieved from https://github.com/hoangddt/electreIII

Marzouk M. (2010). An Application of ELECTRE III to Contractor Selection, presented at Construction Research Congress, Alberta, Canada, 2010. doi:10.1061/41109(373)132

Milani, A., Shanian, A., & El-Lahham, C. (2006). Using different electre methods in strategic planning of human behavioral resistance. *Journal of Applied Mathematics and Decision Sciences, 2006*, 1-19. doi: 10.115/JAMDS/2006/10936

Microsoft Visual C++ [Computer software]. (2013). Retrieved from https://msdn.microsoft.com/en-us/library/yah1y2x8(v=vs.120).aspx

Open Geospatial Consortium. (2008). *OGC KML*. (07-147r2). Retrieved from http://portal.opengeospatial.org/ files/?artifact_id=27810

Pearlswig, Benjamin C. (2013). *Heuristic route generation for the Navy Mission Planner*. Retrieved from Naval Postgraduate School (https://calhoun.nps.edu/handle/10945/37690)

Retired PO2. (2017, September 20). Interview.

Roy, B. (1968). Classement et choix en présence de critères multiples (la méthode ELECTRE), RIRO, 8, 57-75.

Saltelli, A. & Annoni, P. (2010). How to avoid a perfunctory sensitivity analysis. *Environmental Modelling & Software*, *25*(12), 1508-1517. doi: 10.1016/j.envsoft.2010.04.012

Silva, Robert A. (2009). *Optimizing multi-ship, multi-mission operational planning for the joint force maritime component commander*. Retrieved from Defense Technical Information Center (http://www.dtic.mil/dtic/tr/fulltext/u2/a501491.pdf)

Silva, R. (2017, July 4). Email.

Sivathapandia, A. (2013, May 13). *World Country Borders KML*. Retrieved from https://www.arcgis.com/home/ item.html?id=7a8585998b7f470b85235dcdb560c7e2

Sun, Z & Han M. (2013). Multi-criteria decision making method based on Improved ELECTRE III model, presented at International Conference on Engineering, Technology, Management and Science, Nanjing, China, 2013. doi:10.2991/icetms.2013.30

Takeda, E. (2001). A method for multiple pseudo-criteria decision problems. *Computers & Operations Research 28*(14), 1427–1439.

U.S. Navy. (2017). *Vessels*. Retrieved from https://www.navy.com/about/equipment/vessel

APPENDICES

APPENDIX A: WOP MENU LAYOUT

Main Menu (MM)			
ID	**Choice**	**Next Menu**	**Description**
MM1	Show waypoints		Shows all waypoints in the WOP
MM2	Add manual waypoint(s)		Add new waypoints to the WOP
MM3	Change number of midpoints		Change number of midpoints for a set of waypoints
MM4	Show all countries		Shows all countries in the WOP
MM5	Show all units		Shows all units in the WOP
MM6	Show all missions		Shows all missions in the WOP
MM7	Show all homeports		Shows all homeports in the WOP
MM8	Modify unit capabilities	UCM	Modify capabilities for units
MM9	Mission menu	MSM	Modify or create a mission
MM10	Timeline menu	TLM	Run the timeline an any missions created
MM11	Advanced options	AOM	Add a homeport, compare missions, move units between homeports, generate potential new homeports, perform sensitivity analysis of a mission
MM12	Export all data		Exports all WOP data to a file
MM13	Load saved data		Imports data from an external file to replace all current WOP data
MM14	Exit		Exits WOP

Unit Capabilities Menu (UCM)			
ID	**Choice**	**Next Menu**	**Description**
UCM1	Add new homeport		Add a manual homeport
UCM2	Compare missions		Basic compare of two or more missions
UCM3	Move unit(s) between homeports		Manually move unit(s) to a different homeport
UCM4	Return units to original homeport		Return a unit to its original homeport
UCM5	Generate potential new homeports		Find a closer homeport for a mission for units to start from
UCM6	Perform sensitivity analysis		Perform OAT analysis of a mission that used ELECTRE III
UCM7	Return to main menu	MM	Go back to menu MM

Mission Menu (MSM)			
ID	**Choice**	**Next Menu**	**Description**
MSM1	Show all missions		Lists all missions in the WOP
MSM2	Create a mission	CMM	Create a new mission
MSM3	Delete a mission		Delete one or more missions
MSM4	Change mission prerequisites		Change mission prerequisites
MSM5	Alter land avoidance distance		Change distance a mission or units in a mission should avoid land
MSM6	Change units assigned to a mission	AUM	Add one or more units to a mission
MSM7		MM	Go back to menu MM

Create Mission Menu (CMM)			
ID	**Choice**	**Next Menu**	**Description**
CMM1	New mission with existing waypoints		Create a mission with waypoints already added to the WOP
CMM2	New mission with new waypoints		Create a mission with waypoints that will be specified by the user
CMM3	New mission from existing mission		Create a mission from an existing mission's requirements
CMM4	Return to mission menu	MSM	Go back to the mission menu

Mission Unit Menu (AUM)			
ID	**Choice**	**Next Menu**	**Description**
AUM1	Show available units		Show units that could help fulfill the mission
AUM2	Manually add units		Add one or more units to the mission
AUM3	Automatically add units		Set mission to find new units by lowest distance to starting location
AUM4	Show units already used		Show the units that are already used for the mission

Timeline Menu (TLM)			
ID	**Choice**	**Next Menu**	**Description**
TLM1	Show timeline status		Show a quick overview of where the timeline is in execution
TLM2	Set tick amount		Set the amount of hours the timeline runs for

TLM3	Advance timeline by tick		Execute the timeline by the hours set (default 24)
TLM4	Advance timeline by other amount		Execute the timeline by a specified amount of hours
TLM5	Finish one mission		Completely finish one mission on the timeline
TLM6	Finish all missions		Finish all missions on the timeline
TLM7	Output timeline to screen		Output the statuses and waypoints for each of the missions and units to the console
TLM8	Output timeline to a file		Output the statuses and waypoints for each of the missions and units to a file
TLM9	Return to main menu	MM	Go back to the main menu

Advanced Menu (AOM)			
ID	**Choice**	**Next Menu**	**Description**
AOM1	Add new homeport		Add a new homeport to the WOP
AOM2	Compare missions		Compare two or more missions with basic stats
AOM3	Move unit to a new homeport		Move a unit to a different homeport
AOM4	Return units to original homeports		Return one or more units to the original homeport
AOM5	Predict potential new homeport		Predict new locations for a homeport for a mission
AOM6	Perform sensitivity analysis		Perform OAT analysis of a mission that was solved using ELECTRE III
AOM7	Return to main menu	MM	Go back to the main menu

APPENDIX B: EXTERNALLY USED LIBRARIES

The WOP requires the following C++ and Boost libraries to properly compile:

algorithm
boost/archive/text_iarchive.hpp
boost/archive/text_oarchive.hpp
boost/foreach.hpp
boost/property_tree/ptree.hpp
boost/property_tree/xml_parser.hpp
boost/serialization/list.hpp
boost/serialization/map.hpp
boost/serialization/shared_ptr.hpp
boost/serialization/string.hpp
boost/serialization/vector.hpp
cmath
cstdarg
cstdio
cstdlib
cstring
ctime
fstream
future
iomanip
iostream
istream
iterator
list
map
memory
numeric
random
sstream
string
string
thread
vector

APPENDIX C: LAND-LOCKED COUNTRIES

The following countries were determined to be land-locked and were not included in the calculation for units around countries:

Botswana
Burkina Faso
Burundi
Central African Republic
Chad
Ethiopia
Lesotho
Malawi
Mali
Niger
Rwanda
Swaziland
Uganda
Zambia
Zimbabwe
Afghanistan
Azerbaijan
Bhutan
Kazakhstan
Kyrgyzstan
Laos
Mongolia
Nepal
Tajikistan
Turkmenistan
Uzbekistan
West Bank
Andorra
Armenia
Austria
Belarus
Czech Republic
Holy See (Vatican City)
Hungary
Liechtenstein
Luxembourg
Moldova
San Marino
Serbia

Slovakia
Switzerland
Macedonia
Paraguay
Bolivia

APPENDIX D: WOP CSV FILES

The following files were used for the WOP as inputs where the first row is the column labels:

Initial waypoints for testing

```
FROM_WAYPOINT_LAT,FROM_WAYPOINT_LON,TO_WAYPOINT_LAT,TO_WAYP
OINT_LON,NUM_BETWEEN,IDENTIFIER
36.964299,-76.327480,32.370036,-64.681243,2,p1
36.964299,-76.327480,51.4330593,-4.5308297,4,p2
36.962705,-76.330711,-33.903433,18.437505,10,p3
32.684311,-117.230204,-11.321189,136.199148,10,p4
```

Preclusion matrices

```
CG,AD,TBMD,ASW,SUW,S,NSFS,MIO,MCM,MINE,INTEL,SUBINTEL
AD,1,0,1,1,0,0,1,0,0,1,0
TBMD,0,1,1,1,0,0,0,0,0,1,0
ASW,1,0,1,1,1,0,1,0,0,1,0
SUW,1,0,1,1,1,1,1,0,0,1,0
S,1,0,0,1,1,0,1,0,0,1,0
NSFS,1,1,0,1,0,1,0,0,0,1,0
MIO,1,1,1,1,1,0,1,0,0,1,0
MCM,0,0,0,0,0,0,0,0,0,0,0
MINE,0,0,0,0,0,0,0,0,0,0,0
INTEL,1,1,1,1,1,1,1,0,0,1,0
SUBINTEL,0,0,0,0,0,0,0,0,0,0,0

CVN,AD,TBMD,ASW,SUW,S,NSFS,MIO,MCM,MINE,INTEL,SUBINTEL
AD,1,0,0,0,0,0,0,0,0,1,0
TBMD,0,0,0,0,0,0,0,0,0,0,0
ASW,0,0,0,0,0,0,0,0,0,0,0
SUW,0,0,0,0,0,0,0,0,0,0,0
S,0,0,0,0,1,0,0,0,0,1,0
NSFS,0,0,0,0,0,0,0,0,0,0,0
MIO,0,0,0,0,0,0,0,0,0,0,0
MCM,0,0,0,0,0,0,0,0,0,0,0
MINE,0,0,0,0,0,0,0,0,0,0,0
INTEL,1,0,0,0,1,0,0,0,0,1,0
SUBINTEL,0,0,0,0,0,0,0,0,0,0,0

DDG,AD,TBMD,ASW,SUW,S,NSFS,MIO,MCM,MINE,INTEL,SUBINTEL
AD,1,0,1,1,0,0,1,0,0,1,0
TBMD,0,1,1,1,0,0,0,0,0,1,0
ASW,1,0,1,1,1,0,1,0,0,1,0
SUW,1,0,1,1,1,1,1,0,0,1,0
S,1,0,0,1,1,0,1,0,0,1,0
```

NSFS,1,1,0,1,0,1,0,0,0,1,0
MIO,1,1,1,1,1,0,1,0,0,1,0
MCM,0,0,0,0,0,0,0,0,0,0,0
MINE,0,0,0,0,0,0,0,0,0,0,0
INTEL,1,1,1,1,1,1,1,0,0,1,0

LCS,AD,TBMD,ASW,SUW,S,NSFS,MIO,MCM,MINE,INTEL,SUBINTEL
AD,0,0,0,0,0,0,0,0,0,0,0
TBMD,0,0,0,0,0,0,0,0,0,0,0
ASW,0,0,1,0,0,0,0,0,0,1,0
SUW,0,0,0,1,0,1,1,0,0,1,0
S,0,0,0,0,0,0,0,0,0,0,0
NSFS,0,0,0,1,0,1,0,0,0,1,0
MIO,0,0,0,1,0,0,1,0,0,1,0
MCM,0,0,0,0,0,0,0,1,0,1,0
MINE,0,0,0,0,0,0,0,0,0,0,0
INTEL,0,0,1,1,0,1,1,1,0,1,0
SUBINTEL,0,0,0,0,0,0,0,0,0,0,0

MCM,AD,TBMD,ASW,SUW,S,NSFS,MIO,MCM,MINE,INTEL,SUBINTEL
AD,0,0,0,0,0,0,0,0,0,0,0
TBMD,0,0,0,0,0,0,0,0,0,0,0
ASW,0,0,0,0,0,0,0,0,0,0,0
SUW,0,0,0,0,0,0,0,0,0,0,0
S,0,0,0,0,0,0,0,0,0,0,0
NSFS,0,0,0,0,0,0,0,0,0,0,0
MIO,0,0,0,0,0,0,0,0,0,0,0
MCM,0,0,0,0,0,0,0,1,0,1,0
MINE,0,0,0,0,0,0,0,0,1,1,0
INTEL,0,0,0,0,0,0,0,1,0,1,0
SUBINTEL,0,0,0,0,0,0,0,0,0,0,0

SSGN,AD,TBMD,ASW,SUW,S,NSFS,MIO,MCM,MINE,INTEL,SUBINTEL
AD,0,0,0,0,0,0,0,0,0,0,0
TBMD,0,0,0,0,0,0,0,0,0,0,0
ASW,0,0,1,1,0,0,0,0,1,0,1
SUW,0,0,1,1,1,0,0,0,1,0,1
S,0,0,0,1,1,0,0,0,1,0,1
NSFS,0,0,0,0,0,0,0,0,0,0,0
MIO,0,0,0,0,0,0,0,0,0,0,0
MCM,0,0,0,0,0,0,0,0,0,0,0
MINE,0,0,1,1,0,0,0,0,1,0,1
INTEL,0,0,0,0,0,0,0,0,0,0,0
SUBINTEL,0,0,1,1,1,0,0,0,1,0,1

SSN,AD,TBMD,ASW,SUW,S,NSFS,MIO,MCM,MINE,INTEL,SUBINTEL

```
AD,0,0,0,0,0,0,0,0,0,0,0
TBMD,0,0,0,0,0,0,0,0,0,0,0
ASW,0,0,1,1,0,0,0,0,0,0,1
SUW,0,0,1,1,0,0,0,0,1,0,1
S,0,0,0,0,1,0,0,0,0,0,1
NSFS,0,0,0,0,0,0,0,0,0,0,0
MIO,0,0,0,0,0,0,0,0,0,0,0
MCM,0,0,0,0,0,0,0,0,0,0,0
MINE,0,0,0,1,0,0,0,0,1,0,1
INTEL,0,0,0,0,0,0,0,0,0,0,0
SUBINTEL,0,0,1,1,1,0,0,0,1,0,1
```

Units

```
UNIT_TYPE,UNIT_NAME,UNIT_IDENTIFIER,UNIT_SPEED_AVG(KNOTS),PORT_ID
CVN,USS Abraham Lincoln,CVN-72,30,p1
SSBN,USS Alabama,SSBN-731,20,p2
SSBN,USS Alaska,SSBN-732,20,p3
SSN,USS Albany,SSN-753,25,p4
SSN,USS Alexandria,SSN-757,25,p5
SSN,USS Annapolis,SSN-760,25,p5
CG,USS Antietam,CG-54,30,p7
CG,USS Anzio,CG-68,30,p4
DDG,USS Arleigh Burke,DDG-51,30,p4
SSN,USS Asheville,SSN-758,25,p6
DDG,USS Bainbridge,DDG-96,30,p4
DDG,USS Barry,DDG-52,30,p4
DDG,USS Benfold,DDG-65,30,p6
SSN,USS Boise,SSN-764,25,p4
SSN,USS Bremerton,SSN-698,25,p10
SSN,USS Buffalo,SSN-715,25,p10
DDG,USS Bulkeley,DDG-84,30,p4
CG,USS Bunker Hill,CG-52,30,p6
SSN,USS California,SSN-781,25,p5
CG,USS Cape St. George,CG-71,30,p6
CVN,USS Carl Vinson,CVN-70,30,p6
DDG,USS Carney,DDG-64,30,p11
DDG,USS Chafee,DDG-90,30,p10
CG,USS Chancellorsville,CG-62,30,p6
SSN,USS Charlotte,SSN-766,25,p10
SSN,USS Cheyenne,SSN-773,25,p10
SSN,USS Chicago,SSN-721,25,p13
CG,USS Chosin,CG-65,30,p10
DDG,USS Chung-Hoon,DDG-93,30,p10
DDG,USS Cole,DDG-67,30,p4
SSN,USS Columbia,SSN-771,25,p10
```

SSN,USS Columbus,SSN-762,25,p10
SSN,USS Connecticut,SSN-22,25,p14
LCS,USS Coronado,LCS-4,40,p6
CG,USS Cowpens,CG-63,30,p6
DDG,USS Curtis Wilbur,DDG-54,30,p7
SSN,USS Dallas,SSN-700,25,p5
DDG,USS Decatur,DDG-73,30,p6
LCS,USS Detroid,LCS-10,40,p16
DDG,USS Dewey,DDG-105,30,p6
DDG,USS Donald Cook,DDG-75,30,p11
CVN,USS Dwight D. Eisenhower,CVN-69,30,p4
DDG,USS Farragut,DDG-99,30,p16
DDG,USS Fitzgerald,DDG-62,30,p7
SSGN,USS Florida,SSGN-728,20,p3
DDG,USS Forrest Sherman,DDG-98,30,p4
LCS,USS Fort Worth,LCS-3,40,p6
LCS,USS Freedom,LCS-1,40,p6
LCS,USS Gabrielle Giffords,LCS-10,40,p6
CVN,USS George H.W. Bush,CVN-77,30,p4
CVN,USS George Washington,CVN-73,30,p4
SSGN,USS Georgia,SSGN-729,20,p3
CVN,USS Gerald R. Ford,CVN-78,30,p4
CG,USS Gettysburg,CG-64,30,p16
DDG,USS Gonzalez,DDG-66,30,p4
DDG,USS Gravely,DDG-107,30,p4
SSN,USS Greeneville,SSN-772,25,p10
DDG,USS Gridley,DDG-101,30,p6
DDG,USS Halsey,DDG-97,30,p10
SSN,USS Hampton,SSN-767,25,p6
CVN,USS Harry S. Truman,CVN-75,30,p4
SSN,USS Hartford,SSN-768,25,p5
SSN,USS Hawaii,SSN-776,25,p10
SSN,USS Helena,SSN-725,25,p4
SSBN,USS Henry M. Jackson,SSBN-730,20,p2
DDG,USS Higgins,DDG-76,30,p6
DDG,USS Hopper,DDG-70,30,p10
SSN,USS Houston,SSN-713,25,p10
DDG,USS Howard,DDG-83,30,p6
CG,USS Hue City,CG-66,30,p16
SSN,USS Illinois,SSN-786,25,p5
LCS,USS Independence,LCS-2,40,p6
LCS,USS Jackson,LCS-6,40,p6
SSN,USS Jacksonville,SSN-699,25,p10
DDG,USS James E Williams,DDG-95,30,p4
DDG,USS Jason Dunham,DDG-109,30,p4
SSN,USS Jefferson City,SSN-759,25,p6

SSN,USS Jimmy Carter,SSN-23,25,p2
CVN,USS John C. Stennis,CVN-74,30,p14
DDG,USS John Finn,DDG-113,30,p6
DDG,USS John Paul Jones,DDG-53,30,p10
DDG,USS John S McCain,DDG-56,30,p7
SSN,USS John Warner,SSN-785,25,p4
SSBN,USS Kentucky,SSBN-737,20,p2
SSN,USS Key West,SSN-722,25,p13
DDG,USS Kidd,DDG-100,30,p6
DDG,USS Laboon,DDG-58,30,p4
CG,USS Lake Champlain,CG-57,30,p6
CG,USS Lake Erie,CG-70,30,p6
DDG,USS Lassen,DDG-82,30,p7
CG,USS Leyte Gulf,CG-55,30,p4
SSBN,USS Louisiana,SSBN-743,20,p2
SSN,USS Louisville,SSN-724,25,p10
DDG,USS Mahan,DDG-72,30,p4
SSBN,USS Maine,SSBN-741,20,p2
SSBN,USS Maryland,SSBN-738,20,p3
DDG,USS Mason,DDG-87,30,p4
DDG,USS McCampbell,DDG-85,30,p7
DDG,USS McFaul,DDG-74,30,p4
DDG,USS Michael Murphy,DDG-112,30,p10
SSGN,USS Michigan,SSGN-727,20,p2
DDG,USS Milius,DDG-69,30,p6
LCS,USS Milwaukee,LCS-5,40,p16
SSN,USS Minnesota,SSN-783,25,p5
SSN,USS Mississippi,SSN-782,25,p10
SSN,USS Missouri,SSN-780,25,p5
DDG,USS Mitscher,DDG-57,30,p4
CG,USS Mobile Bay,CG-53,30,p6
DDG,USS Momsen,DDG-92,30,p17
CG,USS Monterey,CG-61,30,p4
LCS,USS Montgomery,LCS-8,40,p6
SSN,USS Montpelier,SSN-765,25,p4
DDG,USS Mustin,DDG-89,30,p7
SSBN,USS Nebraska,SSBN-739,20,p2
SSBN,USS Nevada,SSBN-733,20,p2
SSN,USS New Hampshire,SSN-778,25,p5
SSN,USS New Mexico,SSN-779,25,p5
SSN,USS Newport News,SSN-750,25,p4
CVN,USS Nimitz,CVN-68,30,p14
DDG,USS Nitze,DDG-94,30,p4
CG,USS Normandy,CG-60,30,p4
SSN,USS North Carolina,SSN-777,25,p10
SSN,USS North Dakota,SSN-784,25,p5

SSGN,USS Ohio,SSGN-726,20,p2
DDG,USS O'kane,DDG-77,30,p10
SSN,USS Oklahoma City,SSN-723,25,p13
SSN,USS Olympia,SSN-717,25,p10
DDG,USS Oscar Austin,DDG-79,30,p4
SSN,USS Pasadena,SSN-752,25,p6
DDG,USS Paul Hamilton,DDG-60,30,p10
SSBN,USS Pennsylvania,SSBN-735,20,p2
CG,USS Philippine Sea,CG-58,30,p16
DDG,USS Pinckney,DDG-91,30,p6
SSN,USS Pittsburgh,SSN-720,25,p5
CG,USS Port Royal,CG-73,30,p10
DDG,USS Porter,DDG-78,30,p11
DDG,USS Preble,DDG-88,30,p10
CG,USS Princeton,CG-59,30,p6
SSN,USS Providence,SSN-719,25,p5
DDG,USS Rafael Peralta,DDG-115,30,p6
DDG,USS Ramage,DDG-61,30,p4
SSBN,USS Rhode Island,SSBN-740,20,p3
CVN,USS Ronald Reagan,CVN-76,30,p7
DDG,USS Roosevelt,DDG-80,30,p16
DDG,USS Ross,DDG-71,30,p11
DDG,USS Russell,DDG-59,30,p6
DDG,USS Sampson,DDG-102,30,p6
SSN,USS San Francisco,SSN-711,25,p6
CG,USS San Jacinto,CG-56,30,p4
SSN,USS San Juan,SSN-751,25,p5
SSN,USS Santa Fe,SSN-763,25,p10
SSN,USS Scranton,SSN-756,25,p4
SSN,USS Seawolf,SSN-21,25,p14
CG,USS Shiloh,CG-67,30,p7
DDG,USS Shoup,DDG-86,30,p17
SSN,USS Springfield,SSN-761,25,p5
DDG,USS Spruance,DDG-111,30,p6
DDG,USS Sterett,DDG-104,30,p6
DDG,USS Stethem,DDG-63,30,p7
DDG,USS Stockdale,DDG-106,30,p6
DDG,USS Stout,DDG-55,30,p4
SSBN,USS Tennessee,SSBN-734,20,p3
SSN,USS Texas,SSN-775,25,p10
DDG,USS The Sullivans,DDG-68,30,p16
CVN,USS Theodore Roosevelt,CVN-71,30,p6
SSN,USS Toledo,SSN-769,25,p5
SSN,USS Topeka,SSN-754,25,p6
DDG,USS Truxtun,DDG-103,30,p4
SSN,USS Tucson,SSN-770,25,p10

CG,USS Vella Gulf,CG-72,30,p4
CG,USS Vicksburg,CG-69,30,p16
SSN,USS Virginia,SSN-774,25,p5
DDG,USS Wayne E. Meyer,DDG-108,30,p6
SSBN,USS West Virginia,SSBN-736,20,p3
DDG,USS William P. Lawrence,DDG-110,30,p6
DDG,USS Winston S Churchill,DDG-81,30,p4
SSBN,USS Wyoming,SSBN-742,20,p3
MCM,USS Ardent,MCM-12,14,p8
MCM,USS Warrior,MCM-10,14,p9
MCM,USS Sentry,MCM-3,14,p6
MCM,USS Scout,MCM-8,14,p8
MCM,USS Pioneer,MCM-9,14,p9
MCM,USS Patriot,MCM-7,14,p6
MCM,USS Gladiator,MCM-11,14,p8
MCM,USS Dextrous,MCM-13,14,p8
MCM,USS Devastator,MCM-6,14,p6
MCM,USS Chief,MCM-14,14,p9
MCM,USS Champion,MCM-4,14,p6

Ports

ID,PORT,LAT,LON
p1,Newport News (Virginia),36.9853,-76.449
p2,Bangor (Washington),47.772,-122.749
p3,Kings Bay (Georgia),30.7455,-81.4864
p4,Norfolk (Virginia),36.9619,-76.3386
p5,Groton (Connecticut),41.3834,-72.0915
p6,San Diego (California),32.6896,-117.2316
p7,Yokosuka (Japan),35.3063,139.662
p8,Manama (Bahrain),26.1987,50.6381
p9,Sasebo (Japan),35.1574,129.7132
p10,Pearl Harbor (Hawaii),21.3558,-157.9578
p11,Rota (Spain),36.6182,-6.3432
p12,Little Creek (Virginia),36.9318,-76.1792
p13,Apra Harbor (Guam),13.4516,144.6525
p14,Bremerton (Washington),47.5551,-122.6324
p15,Diego Garcia (BIOT),-7.3257,72.4102
p16,Mayport (Florida),30.4014,-81.4103
p17,Everett (Washington),47.9893,-122.2499
p18,Gaeta (Italy),41.216,13.5759

Country center waypoints

LAT,LONG,ABBREVIATION,SHORT_NAME
33,66,AF,Afghanistan

41,20,AL,Albania
28,3,AG,Algeria
-14.3333333,-170,AS,American Samoa
42.5,1.5,AN,Andorra
-12.5,18.5,AO,Angola
18.216667,-63.05,AV,Anguilla
17.05,-61.8,AC,Antigua and Barbuda
-34,-64,AR,Argentina
40,45,AM,Armenia
12.5,-69.966667,AA,Aruba
-15.95,-5.7,SH,Ascension
-12.416667,123.333333,AT,Ashmore and Cartier Islands
-25,135,AS,Australia
47.333333,13.333333,AU,Austria
40.5,47.5,AJ,Azerbaijan
24,-76,BF,Bahamas
26,50.5,BA,Bahrain
24,90,BG,Bangladesh
13.166667,-59.533333,BB,Barbados
-21.416667,39.7,BS,Bassas da India
53,28,BO,Belarus
50.833333,4,BE,Belgium
17.25,-88.75,BH,Belize
9.5,2.25,BN,Benin
32.333333,-64.75,BD,Bermuda
27.5,90.5,BT,Bhutan
-17,-65,BL,Bolivia
12.2,-68.25,NT,Bonaire
44.25,17.833333,BK,Bosnia and Herzegovina
-22,24,BC,Botswana
-54.433333,3.4,BV,Bouvet Island
-10,-55,BR,Brazil
-6,72,IO,British Indian Ocean Territory
18.5,-64.5,VI,British Virgin Islands
4.5,114.666667,BX,Brunei
43,25,BU,Bulgaria
13,-2,UV,Burkina Faso
22,98,BM,Burma
-3.5,30,BY,Burundi
13,105,CB,Cambodia
6,12,CM,Cameroon
60,-96,CA,Canada
16,-24,CV,Cape Verde
19.5,-80.666667,CJ,Cayman Islands
7,21,CT,Central African Republic
15,19,CD,Chad

-30,-71,CI,Chile
35,105,CH,China
-10.5,105.666667,KT,Christmas Island
10.283333,-109.216667,IP,Clipperton Island
-12,96.833333,CK,Cocos (Keeling) Islands
4,-72,CO,Colombia
-12.166667,44.25,CN,Comoros
-16.083333,-161.583333,CW,Cook Islands
-17.5,151,CR,Coral Sea Islands
10,-84,CS,Costa Rica
8,-5,IV,Cote d'Ivoire
45.166667,15.5,HR,Croatia
22,-79.5,CU,Cuba
12.166667,-69,UC,Curaçao
35,33,CY,Cyprus
49.75,15,EZ,Czech Republic
0,25,CG,Democratic Republic of the Congo
56,10,DA,Denmark
11.5,42.5,DJ,Djibouti
15.5,-61.333333,DO,Dominica
19,-70.666667,DR,Dominican Republic
-2,-77.5,EC,Ecuador
27,30,EG,Egypt
13.833333,-88.916667,ES,El Salvador
2,10,EK,Equatorial Guinea
15,39,ER,Eritrea
59,26,EN,Estonia
8,38,ET,Ethiopia
-22.333333,40.366667,EU,Europa Island
-51.75,-59.166667,FK,Falkland Islands
62,-7,FO,Faroe Islands
5,152,FM,Federated States of Micronesia
-18,178,FJ,Fiji
64,26,FI,Finland
46,2,FR,France
4,-53,FG,French Guiana
-15,-140,FP,French Polynesia
-43,67,FS,French Southern and Antarctic Lands
-1,11.75,GB,Gabon
13.5,-15.5,GA,Gambia
31.425074,34.373398,GZ,Gaza Strip
41.999981,43.499905,GG,Georgia
51.5,10.5,GM,Germany
8,-2,GH,Ghana
36.133333,-5.35,GI,Gibraltar
-11.5,47.333333,GO,Glorioso Islands

39,22,GR,Greece
72,-40,GL,Greenland
12.116667,-61.666667,GJ,Grenada
16.25,-61.583333,GP,Guadeloupe
13.4444444,144.7366667,GU,Guam
15.5,-90.25,GT,Guatemala
49.583333,-2.333333,GK,Guernsey
11,-10,GV,Guinea
12,-15,PU,Guinea-Bissau
5,-59,GY,Guyana
19,-72.416667,HA,Haiti
-53,73,HM,Heard Island and McDonald Islands
15,-86.5,HO,Honduras
22.25,114.166667,HK,Hong Kong
47,20,HU,Hungary
65,-18,IC,Iceland
20,77,IN,India
-5,120,ID,Indonesia
32,53,IR,Iran
33,44,IZ,Iraq
53,-8,EI,Ireland
54.25,-4.5,IM,Isle of Man
31.5,34.75,IS,Israel
42.833333,12.833333,IT,Italy
18.25,-77.5,JM,Jamaica
36,138,JA,Japan
49.216667,-2.116667,JE,Jersey
31,36,JO,Jordan
-17.05833,42.71667,JU,Juan de Nova Island
48,68,KZ,Kazakhstan
1,38,KE,Kenya
-5,-170,KR,Kiribati
42.583333,21,KV,Kosovo
29.5,47.75,KU,Kuwait
41,75,KG,Kyrgyzstan
18,105,LA,Laos
57,25,LG,Latvia
33.833333,35.833333,LE,Lebanon
-29.5,28.25,LT,Lesotho
6.5,-9.5,LI,Liberia
25,17,LY,Libya
47.166667,9.533333,LS,Liechtenstein
56,24,LH,Lithuania
49.75,6.166667,LU,Luxembourg
22.157778,113.559722,MC,Macau
41.833333,22,MK,Macedonia

-20,47,MA,Madagascar
-13.5,34,MI,Malawi
2.5,112.5,MY,Malaysia
3.2,73,MV,Maldives
17,-4,ML,Mali
35.916667,14.433333,MT,Malta
10,167,RM,Marshall Islands
14.666667,-61,MB,Martinique
20,-12,MR,Mauritania
-20.3,57.583333,MP,Mauritius
-12.833333,45.166667,MF,Mayotte
23,-102,MX,Mexico
47,29,MD,Moldova
43.733333,7.4,MN,Monaco
46,105,MG,Mongolia
42.5,19.3,MJ,Montenegro
16.75,-62.2,MH,Montserrat
32,-5,MO,Morocco
-18.25,35,MZ,Mozambique
-22,17,WA,Namibia
-0.533333,166.916667,NR,Nauru
28,84,NP,Nepal
52.5,5.75,NL,Netherlands
-21.5,165.5,NC,New Caledonia
-42,174,NZ,New Zealand
13,-85,NU,Nicaragua
16,8,NG,Niger
10,8,NI,Nigeria
-19.033333,-169.866667,NE,Niue
-29.033333,167.95,NF,Norfolk Island
40,127,KN,North Korea
16,146,MP,Northern Mariana Islands
62,10,NO,Norway
21,57,MU,Oman
30,70,PK,Pakistan
6,134,PS,Palau
9,-80,PM,Panama
-6,147,PP,Papua New Guinea
-22.993333,-57.996389,PA,Paraguay
-10,-76,PE,Peru
13,122,RP,Philippines
-25.066667,-130.1,PC,Pitcairn Islands
52,20,PL,Poland
39.5,-8,PO,Portugal
18.2482882,-66.4998941,PR,Puerto Rico
25.5,51.25,QA,Qatar

-1,15,CF,Republic of the Congo
-21.1,55.6,RE,Reunion
46,25,RO,Romania
60,100,RS,Russia
-2,30,RW,Rwanda
17.9,-62.833333,TB,Saint Barthelemy
17.333333,-62.75,SC,Saint Kitts and Nevis
13.883333,-60.966667,ST,Saint Lucia
18.075,-63.05833,RN,Saint Martin
46.833333,-56.333333,SB,Saint Pierre and Miquelon
13.083333,-61.2,VC,Saint Vincent and the Grenadines
-13.803096,-172.178309,WS,Samoa
43.933333,12.416667,SM,San Marino
1,7,TP,Sao Tome and Principe
25,45,SA,Saudi Arabia
14,-14,SG,Senegal
44,21,RI,Serbia
-4.583333,55.666667,SE,Seychelles
8.5,-11.5,SL,Sierra Leone
1.366667,103.8,SN,Singapore
18.04167,-63.06667,NN,Sint Maarten
48.666667,19.5,LO,Slovakia
46.25,15.166667,SI,Slovenia
-8,159,BP,Solomon Islands
6,48,SO,Somalia
-30,26,SF,South Africa
-56,-33,SX,South Georgia and South Sandwich Islands
37,127.5,KS,South Korea
8,30,OD,South Sudan
40,-4,SP,Spain
7,81,CE,Sri Lanka
16,30,SU,Sudan
4,-56,NS,Suriname
78,20,SV,Svalbard
-26.5,31.5,WZ,Swaziland
62,15,SW,Sweden
47,8,SZ,Switzerland
35,38,SY,Syria
24,121,TW,Taiwan
39,71,TI,Tajikistan
-6,35,TZ,Tanzania
15,100,TH,Thailand
-8.833333,125.75,TT,Timor-Leste
8,1.166667,TO,Togo
-9,-171.75,TL,Tokelau
-20,-175,TN,Tonga

```
11,-61,TD,Trinidad and Tobago
-15.866667,54.416667,TE,Tromelin Island
34,9,TS,Tunisia
39.059012,34.911546,TU,Turkey
40,60,TX,Turkmenistan
21.733333,-71.583333,TK,Turks and Caicos Islands
-8,178,TV,Tuvalu
2,33,UG,Uganda
49,32,UP,Ukraine
24,54,AE,United Arab Emirates
54,-4,UK,United Kingdom
39.828175,-98.5795,US,United States
-33,-56,UY,Uruguay
5.8811111,-162.0725,UM,US Minor Outlying Islands
18.3482891,-64.9834807,VI,US Virgin Islands
41.707542,63.84911,UZ,Uzbekistan
-16,167,NH,Vanuatu
41.9,12.45,VT,Vatican City
8,-66,VE,Venezuela
16.166667,107.833333,VM,Vietnam
-13.3,-176.2,WF,Wallis and Futuna
31.666667,35.25,WE,West Bank
25,-13.5,WI,Western Sahara
15.5,47.5,YM,Yemen
-15,30,ZA,Zambia
-19,29,ZI,Zimbabwe
```

Mission capabilities

```
MISSION,MISSION_TITLE
AD,Air Defense
TBMD,Theater Ballistic Missile Defense
ASW,Antisubmarine Warfare
SUW,Surface Warfare
S,Strike
NSFS,Naval Surface Fire Support
MIO,Maritime Interception Operations
MCM,Mine Countermeasures
MINE,Mine Warfare
INTEL,Intelligence Collection
SUBINTEL,Submarine Intelligence Collection
```

Unit capabilities

```
UNIT_TYPE,MISSIONS
CG,AD,ASW,SUW,S,NSFS,MIO,INTEL,TBMD
```

CVN,INTEL,AD,S
DDG,AD,ASW,SUW,S,NSFS,MIO,INTEL,TBMD
LCS,ASW,SUW,NSFS,MIO,MCM,INTEL
MCM,MCM,MINE,INTEL
SSGN,ASW,SUW,S,MINE,SUBINTEL
SSN,ASW,SUW,S,MINE,SUBINTEL
SSBN,ASW,SUW,S,MINE,SUBINTEL

VITA

Andrew Miller
Engineering Management and Systems Engineering Department

Andrew Miller was born in Columbia, South Carolina and raise throughout South Carolina, North Carolina, and Virginia much of his life. He got a B.S. in Computer Science from Old Dominion University (Norfolk, VA) in 2009 with a minor in Computer Engineering and a M.E. in Systems Engineering also from Old Dominion University in 2012. After his B.S., he began working in the federal sector performing work in the fields of database administration, enterprise architecture, systems engineering, data analytics, and information systems administration where he enjoys learning new technologies.