2022-06

A PATH ENUMERATION REFORMULATION OF THE SCHEDULE MIXED INTEGER PROGRAM SUPPORTING EXPEDITIONARY ADVANCED BASE OPERATIONS.

Mirsch, Andrew M.

Monterey, CA; Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

A PATH ENUMERATION REFORMULATION OF THE SCHEDULE MIXED INTEGER PROGRAM SUPPORTING EXPEDITIONARY ADVANCED BASE OPERATIONS

by

Andrew M. Mirsch

June 2022

Thesis Advisor:	Emily M. Craparo
Co-Advisor:	W. Matthew Carlyle
Second Reader:	Thomas W. Lucas

THIS PAGE INTENTIONALLY LEFT BLANK

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

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** June 2022	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE A PATH ENUMERATION REFORMULATION OF THE SCHEDULE MIXED INTEGER PROGRAM SUPPORTING EXPEDITIONARY ADVANCED BASE OPERATIONS	**5. FUNDING NUMBERS**
6. AUTHOR(S) Andrew M. Mirsch	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

The U.S. Marine Corps needs an accurate model for analyzing its logistical needs in support of Expeditionary Advanced Base Operations (EABO). EABO is a doctrinal method used by the U.S. Navy and Marine Corps for denying adversary forces access to the maritime global commons. Deployment and sustainment of forces engaged in EABO requires a distribution network supported by various surface and airborne connector platforms of differing capacity and speed. The Marine Corps currently has a model for analyzing its distribution networks in support of EABO, the Schedule Mixed Integer Program (S-MIP). However, the computational difficulty of S-MIP limits its usefulness in large-scale experiments. This thesis describes a path enumeration-based reformulation known as the Path Enumeration Mixed-Integer Program (PE-MIP). PE-MIP is designed to provide a less computationally difficult model than the antecedent model S-MIP. We compare the runtime of PE-MIP and the quality of its solutions with that of S-MIP model and find that PE-MIP provides faster and superior results to S-MIP. The application of PE-MIP by the research sponsor will further inform current Marine Corps and Navy operational plans, acquisition, and force structure decisions.

14. SUBJECT TERMS modeling, network flows, distribution, expeditionary advanced base operations	**15. NUMBER OF PAGES** 127
	16. PRICE CODE

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

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

A PATH ENUMERATION REFORMULATION OF THE SCHEDULE MIXED INTEGER PROGRAM SUPPORTING EXPEDITIONARY ADVANCED BASE OPERATIONS

Andrew M. Mirsch
Major, United States Marine Corps
BS, United States Naval Academy, 2010

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
June 2022

Approved by: Emily M. Craparo
Advisor

W. Matthew Carlyle
Co-Advisor

Thomas W. Lucas
Second Reader

W. Matthew Carlyle
Chair, Department of Operations Research

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The U.S. Marine Corps needs an accurate model for analyzing its logistical needs in support of Expeditionary Advanced Base Operations (EABO). EABO is a doctrinal method used by the U.S. Navy and Marine Corps for denying adversary forces access to the maritime global commons. Deployment and sustainment of forces engaged in EABO requires a distribution network supported by various surface and airborne connector platforms of differing capacity and speed. The Marine Corps currently has a model for analyzing its distribution networks in support of EABO, the Schedule–Mixed Integer Program (S-MIP). However, the computational difficulty of S-MIP limits its usefulness in large-scale experiments. This thesis describes a path enumeration-based reformulation known as the Path Enumeration Mixed-Integer Program (PE-MIP). PE-MIP is designed to provide a less computationally difficult model than the antecedent model S-MIP. We compare the runtime of PE-MIP and the quality of its solutions with that of S-MIP model and find that PE-MIP provides faster and superior results to S-MIP. The application of PE-MIP by the research sponsor will further inform current Marine Corps and Navy operational plans, acquisition, and force structure decisions.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVIATIONS

A2/AD	Anti-access/area denial
ATEM	Air Tasking and Efficiency Model
BFS	Breadth-first search
CD&I	Combat Development & Integration
CENTCOM	U.S. Central Command
DB	Deutsche Bahn
DOE	Design of experiments
DOD	Department of Defense
DON	Department of the Navy
EAB	Expeditionary advanced base
EABO	Expeditionary advanced base operations
EEZ	Exclusive economic zone
FIC	First Island Chain
HA/DR	Humanitarian aid and disaster relief
IED	Improvised explosive device
INFORMS	Institute for Operations Research and the Management Sciences
LOCE	Littoral operations in contested environments
MCAS	MEU Amphibious Connector Scheduler
MCDP	Marine Corps Doctrinal Publication
MEU	Marine Expeditionary Unit
MLR	Marine Littoral Regiment
MINLP	Mixed integer linear program
MIP	Mixed integer program
NPS	Naval Postgraduate School
OAD	Operations Analysis Division
OEF	Operation Enduring Freedom
OIF	Operation Iraqi Freedom
OPLAN	Operational Plan

OR	Operations Research
ORION	On-Road Integrated Optimization and Navigation
PE	Path enumeration
PE-MIP	Path Enumeration-Mixed Integer Program
PE-MIPv1	Path Enumeration-Mixed Integer Program Version 1
PE-MIPv2	Path Enumeration-Mixed Integer Program Version 2
PRC	People's Republic of China
RSRP	Rolling Stock Rotation Plan
SEED	Simulation Experiments and Experimental Design
S-MIP	Schedule Mixed Integer Program
SSTP	Ship-to-Shore Transportation Problem
StS	Ship-to-shore site
TACMN	Table of authorized control number
TCC	Total connector capacity
TEPF	Expeditionary fast transports
TL	Total loads
TRANSCOM	U.S. Transportation Command
TSL	Total serials legs
TSS	Total serial size
UNCLOS	United Nations Convention for the Law of the Sea
UPS	United Parcel Service
VRP	Vehicle routing problem

EXECUTIVE SUMMARY

Robust anti-access/area denial (A2/AD) capabilities developed by the People's Republic of China (PRC) and other competitors significantly challenge the United States' ability to access and exert influence over key maritime terrain worldwide. A2/AD systems are a collection of technologies designed to asymmetrically counter and negate American maritime power projection forces like carrier strike groups and amphibious task forces. Countering the asymmetric challenge of A2/AD systems is the focus of a new U.S. Navy–Marine Corps operational concept: Expeditionary Advanced Base Operations (EABO) (Department of the Navy [DON] 2021). The purpose of EABO is the interdiction and imposition of cost on any PRC forces attempting to break out of the South China Sea (SCS) into less-congested portions of the Pacific Ocean. In support of this purpose, the U.S. Marine Corps is redefining itself as the "interior force" or "stand-in force" that operates in the First Island Chain (FIC) inside the weapons engagement zone of PRC A2/AD forces (DON 2021).

The EABO concept relies on forces operating within the effective range of the enemy's A2/AD weapon systems. Dispersion of forces is crucial for successful EABO since massed forces represent a lucrative target for the enemy's A2/AD weapons. The survivability of dispersed EABO forces comes at the cost of simple and centralized distribution networks used to logistically support Marine forces conducting EABO. Therefore, the sustainment of EABO forces poses a daunting operational challenge for U.S. Navy and Marine Corps planners. The U.S. Marine Corps currently has a sophisticated optimization model for studying connector employment in support of EABO. This optimization model is named the Schedule Mixed Integer Program (S-MIP). While S-MIP produces useful results, S-MIP's current run times are undesirably slow. The computational complexity of S-MIP is such that it precludes large-scale detailed analysis of contingences in support of operational plans. Naval Postgraduate School personnel have iteratively improved S-MIP in support of a funded Naval Research Program and a previous student thesis.

This thesis studies a path enumeration-based reformulation of S-MIP to answer the following questions:

> Can a heuristic model based on path enumeration provide solutions of comparable quality to those resulting from the existing S-MIP, with lower computation times?
>
> What modification to or variations of the path enumeration heuristic model provide better solutions with less computational expense?

Research initiated with a review of the antecedent model S-MIP and its reformulation the Path Enumeration-Mixed Integer Program (PE-MIP), which was developed by Naval Postgraduate School researchers during the period of November 2021-February 2022.

Preliminary experiments focused on various means of filtering candidate (serial, path) pairs, henceforth referred to as serial paths, to expedite model runtime. Although these early efforts decreased the computational complexity of the model they did not result in the decreased model runtimes desired by the sponsor. These initial results triggered a reformulation of PE-MIP to use one of two possible termination conditions: an absolute optimality gap and a relative optimality gap. The absolute optimality gap termination condition causes PE-MIP to end upon finding a solution that delivers all input serials. The relative optimality gap termination condition allows PE-MIP to run until finding a solution within a user specified tolerance of the integer optimal solution. This reformulation resulted in two separate means of employing PE-MIP. Final experiments with PE-MIP demonstrated that use of an absolute optimality gap results in much faster model runtimes at the cost of the quality of the solution. Conversely, using a relative optimality gap as the model's termination criteria can return better results at the cost of longer computation time. The relationship between termination criteria and solution quality is illustrated in Figure 1.

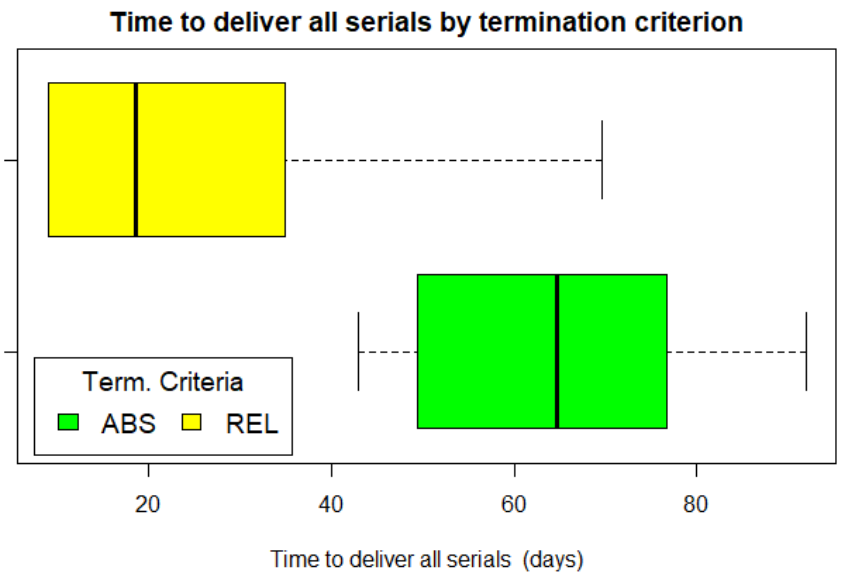

Figure 1. Boxplot depicting time to deliver all serials by termination criteria.

As shown by Figure 1, PE-MIP can return very desirable solutions regarding the time to deliver all serials. However, these superior solutions come at the cost of longer model runtimes. This trade-off illustrates two different methods of using PE-MIP for deliberate versus rapid planning applications. The computational difficulty inherent in PE-MIP prevents this model from returning solutions approaching integer optimality within tolerances in a short model runtime. Computational difficulty is best illustrated by the number of variables in each model design point. As the number of model variables increases so does the complexity of the model resulting in failures by PE-MIP to deliver all serials within a specified time limit.

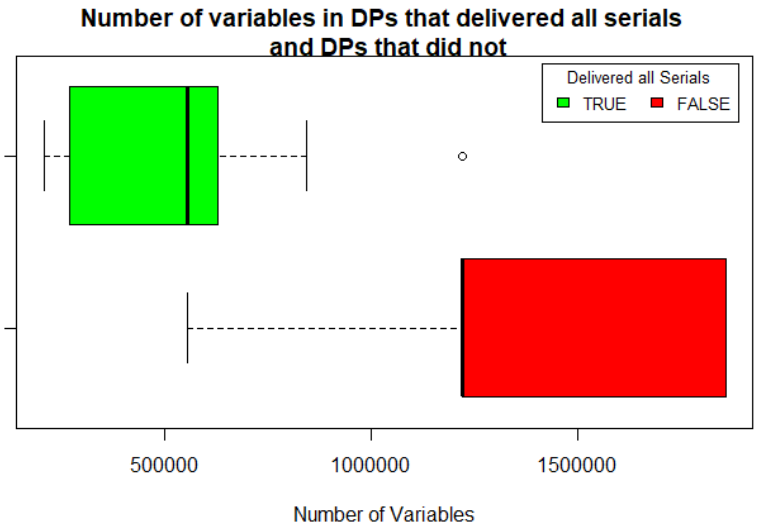

Figure 2. Boxplot depicting time to deliver all serials by termination criteria.

User inputs to PE-MIP can significantly affect the model's performance regarding the proportion of serials delivered and the time to deliver all serials. Input factors that reduce computational difficulty generally return superior results despite the flexibility given the model by inputs that result in more computational complexity.

Given the same two Marine Littoral Regiment deployment problem inputs PE-MIP returns better solutions than S-MIP in a fraction of the time required by S-MIP. Despite these improvements additional research is required to support the Marine Corps' force closure problem. Recommendations for future research include (1) developing a version of PE-MIP for supporting sustainment operations that models use of connectors on rotational routes akin to a city bus; (2) modifying PE-MIP to include adversarial interdiction considerations; and (3) improving S-MIP's heuristic serial building algorithm to create a serial construction decision support tool that features relational rules about pairing equipment types.

Reference

Department of the Navy (2021) *Tentative Manual for Expeditionary Advanced Base Operations.* Washington, DC. https://www.mcwl.marines.mil/TMEABO/

ACKNOWLEDGMENTS

Many thanks to my advisory team, Dr. Emily Caparo and Dr. Matthew Carlyle. Your support and guidance through this process was critical to the successful completion of this research. Your willingness to work with me through complex problems and share your precious time in support of this research is a testament to your professionalism and strong work ethics. I also must express an abundance of appreciation to Mr. Stephen Upton and Ms. Mary McDonald for their assistance on this project. Your support has been indispensable to this project. Finally, I must express my absolute gratitude and unending thanks to my wonderful wife and best friend, Rita. Your unending support throughout this graduate program made all my work possible. You are a fantastic wife and mother; I am truly blessed to have someone like you in my life.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

This thesis seeks to help the U.S. Navy and Marine Corps solve the problem of optimally deploying Marine forces to distributed locations in support of emerging operational concepts. In this chapter, we present the motivation for this research, its underlying questions, pertinent background, and the organization of the thesis.

A. RESEARCH PROBLEM

In the immediate aftermath of the Soviet Union's collapse, the United States enjoyed a period of unchallenged naval supremacy. Throughout this period of military ascendency, the United States projected power from the sea during numerous campaigns in the Balkans, Africa, the Middle East, and South Asia. These campaigns all featured an uncontested maritime environment in which American naval surface combatants, amphibious forces, and carrier strike groups could operate with impunity. This period of naval supremacy is over due to the reemergence of interstate military competition and the proliferation of low cost but highly effect naval weapons. Therefore, America's maritime services must now reassess their operating environment, evaluate their opponents, and develop novel solutions for new challenges.

The first chapter of Marine Corps Doctrinal Publication (MCDP) 3 Expeditionary Operations is titled "The Landscape: Chaos in the Littorals" (Department of the Navy [DON] 2018). Originally written in 1998 and unmodified for subsequent editions, this chapter describes an anarchic environment of incessant competition between many disparate factions seeking survival and supremacy in a Darwinian post-Cold War world (DON 2018). This chapter's description of the littoral environment remains relatively accurate today. However, there is something noticeably lacking for today's reader: competition between peer nation states. The People's Republic of China's (PRC) emergence as a major maritime power has added a new level of complexity to operations in the littorals. American maritime forces now face the difficult task of competing with a technologically evenly matched opponent in a complex and dynamic battlespace.

Robust anti-access/area denial (A2/AD) capabilities developed by the PRC and other competitors significantly challenge the United States' ability to access and exert influence over key maritime terrain worldwide. A2/AD systems are a collection of technologies designed to asymmetrically counter and negate American maritime power projection forces like carrier strike groups and amphibious task forces. Massive salvos of anti-ship cruise missiles, armed aerial drones, unmanned surface craft, naval mines, hypersonic munitions, and other systems are elements of the PRC's operational methods of defeating American naval involvement in a Western Pacific conflict.

The U.S. Navy and Marine Corps are reorganizing and adopting new doctrine to remain relevant and effective in the new operating environment characterizing a possible conflict with the PRC. Described as littoral operations in contested environments (LOCE), potential conflicts with the PRC in the Western Pacific are the focus of a new U.S. Navy–Marine Corps operational concept: Expeditionary Advanced Base Operations (EABO). The purpose of EABO is the interdiction and imposition of cost on any PRC forces attempting to break out of the SCS into less-congested portions of the Pacific Ocean. In support of this purpose, the U.S. Marine Corps is redefining itself as the "interior force" or "stand-in force" that operates in the First Island Chain (FIC) inside the weapons engagement zone of PRC A2/AD forces (Headquarters U.S. Marine Corps 2021).

In execution, EABO will feature small, distributed Marine detachments operating networks of sensors and anti-ship/anti-aircraft weapons fighting in support of the U.S. Navy (DON 2019). Defended by U.S. Marine infantry against PRC amphibious assault, these expeditionary advanced bases (EABs) will serve as the U.S. Navy's screening/ scouting force and comprise the "frontline" of a possible conflict with the PRC. The geography of the Western Pacific and SCS combined with the distributed nature of EABO is a significant challenge for U.S. Navy–Marine Corps logisticians. Deploying EABO forces to their operating positions and sustaining them in combat requires optimal use of limited surface and air connectors. EABs can only achieve their mission if rapidly and effectively inserted into their initial operating positions and then sustained with critical supplies.

The distributed nature of EABs implies great potential for isolation and subsequent neutralization of EABs via starvation during combat with the PRC. Such an event is not historically unprecedented in this part of the world. On several occasions during the Second World War, allied forces isolated and neutralized Japanese positions in the Pacific that were performing missions very similar to EABO. The purpose of this thesis is to provide U.S. Navy-Marine Corps planners with a useful model to optimize its use of connectors in support of EABO.

B. RESEARCH QUESTIONS

Deployment and sustainment of EAB forces is a daunting challenge for U.S. Navy-Marine Corps planners. Real-world exercises and experiments concerning sustainment of EABs are expensive and too infrequent to provide the data needed for an extensive study of optimized connector employment. Therefore, the U.S. Marine Corps and Navy need a model capable of providing meaningful insight and data on how to sustain EABO and other operational planning considerations. The U.S. Navy and Marine Corps also need data to inform decisions on what mixture of surface and air connector platforms to acquire for supporting EABO.

The U.S. Marine Corps' Operations Analysis Division (OAD), Combat Development & Integration (CD&I) currently has a sophisticated optimization model for studying connector employment in support of EABO. This optimization model, the Schedule Mixed Integer Program (S-MIP), was developed by NPS Operations Research graduate Lieutenant Colonel Nicholas Freeman in late 2019. While this model produces useful results, S-MIP's current run times are undesirably slow. The computational difficulty of S-MIP is such that it precludes large-scale detailed analysis of contingences in support of operational plans (OPLANs). Further work must be done to improve the run time of S-MIP so that it can be employed as an aid for wargaming, operational planning, and acquisitions decisions.

This thesis proposes a different model based on path enumeration to solve the connector optimization problem for which S-MIP was designed. This thesis is intended to answer the following questions:

- Can a heuristic model based on path enumeration provide solutions of comparable quality to those resulting from the existing S-MIP, with lower computation times?

- What modification to or variations of the path enumeration heuristic model provide better solutions with less computational expense?

This research employs path-based enumeration techniques combined with experimental design and data farming methods to provide insights on optimized connector employment. It also supports analysis of a sponsor-provided force closure problem. The force closure problem is the optimized deployment of a single Marine Littoral Regiment (MLR) or several MLRs from their home station to their operating positions for EABO. If successful, this research will provide the sponsor with an efficient and effective model to replace S-MIP and other optimization models or a means of improving the computational difficulty of S-MIP.

C. BACKGROUND

The FIC is the most likely location for a confrontation between the United States and the PRC. Located in the Western Pacific, the FIC is a massive geographic feature composed of several island groups and archipelagos. The FIC spans from the northern shore of Borneo and the Philippines in the south to Kyushu in the Japanese home islands in the north. The FIC is a natural barrier through which the PRC's military and commercial sea traffic must pass to reach the greater Pacific Ocean from the SCS. Several islands and archipelagos in the FIC form natural chokepoints on the sea lanes exiting and entering the SCS. These canalizing features are the key maritime terrain of a conflict in the FIC. Control or denial of use of these key waterways to adversary forces is of great importance to developing operational concepts intended for a possible conflict or confrontation in the SCS. Ongoing tensions and disputes in the SCS indicate that continued competition and possible conflict in this region are likely. There are currently several interested parties maneuvering for position and extending their claims to critical maritime terrain in the SCS.

Maritime law is complex and different portions of the ocean are often claimed by several nations. From 1973–1994, the United Nations incrementally wrote and established the United Nations Convention for the Law of the Sea (UNCLOS) to codify the jurisdiction of nations states over their adjoining oceans (Treves 2008). UNCLOS specified in 1982 that nations with shorelines have territorial waters, contiguous zones, and exclusive economic zones (EEZ) extending 12 nautical miles (nm), 24 nm, and 200 nm respectively from their shorelines (Treves 2008). While the United States is not a party to UNCLOS, 167 nations including the PRC are parties to UNCLOS (United Nations 2022). Although the PRC is a party to UNCLOS, its behavior towards its neighbors regarding maritime territorial claims is inconsistent with its treaty obligations under UNCLOS.

Tensions abound in and around the FIC due to a series of expansive maritime territorial claims made by the PRC in the SCS. The PRC's extraneous maritime claims are collectively referred to as the "nine-dashed line" policy. These claims are so named due to the dashed line of nine segments used to portray the PRC's territorial waters in PRC produced documents and media. The PRC's maritime ambitions are so central to its conception of itself and its foreign policy that the nine-dashed line even appears in the visa pages of PRC-issued passports (Beech et al. 2016). PRC maritime territorial claims infringe on the EEZs of Vietnam, the Philippines, Taiwan, Malaysia, and Indonesia as depicted in Figure 1. These competing claims have resulted in various competitive activities between the involved parties and some instances of limited violence.

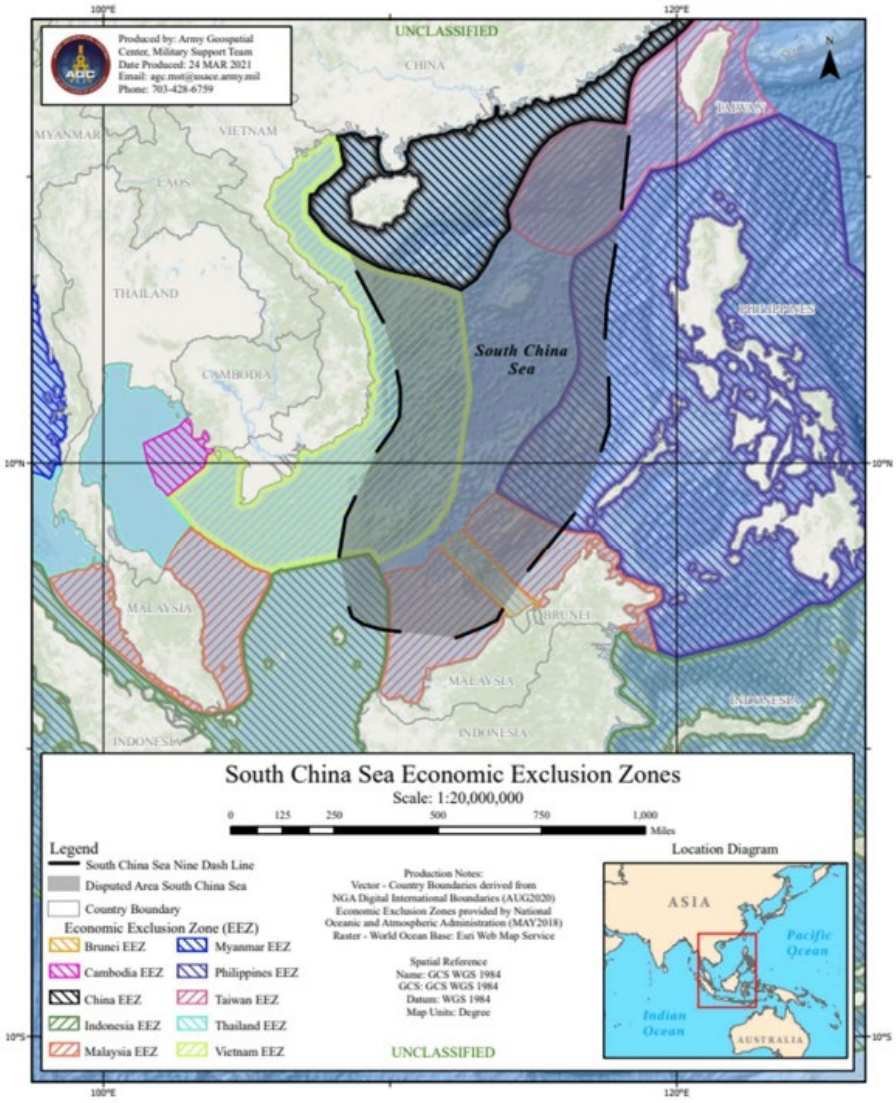

Figure 1. South China Sea Economic Exclusion Zones. Illustrated stakeholder interest within the FIC. Source: DeMarco (2021).

As tensions abound in the SCS, traditional methods of amphibious warfare are challenged and disrupted by the PRC's growing maritime and A2/AD capabilities. The last significant opposed amphibious operation in American history was Operation CHROMITE during the Korea War. Executed from 10–19 September 1950, Operation CHROMITE was a wildly successful operation that dramatically reversed the situation on the Korean peninsula. Undertaken as an operational turning movement, the successful U.S. Marine Corps and U.S. Army landings at Inchon precipitated the collapse of communist forces

besieging Pusan and initiated the United Nations' pursuit of defeated North Korean forces to the Yalu River. However, the conditions under which Operation CHROMITE was performed bear little resemblance to the current operating environment and its associated challenges.

Though limited in scale, the Falklands War of 1982 offers a better picture of what naval combat in the FIC could resemble. Fought between the United Kingdom and Argentina in a remote part of the South Atlantic, this conflict featured naval, air, and ground combat between two roughly evenly matched opponents. Critical study of the Falklands War exposes some of the limits of and challenges to traditional amphibious operations in the age of guided missile combat.

The United Kingdom lost two ships to Argentine Exocet missiles: the SS Atlantic Conveyor, a commercial cargo ship taken into naval service, and the HMS Sheffield, a Type 42 Guided Missile Destroyer (Middlebrook 2001). The HMS Glamorgan, another British destroyer, was also damaged by a shore-launched Argentine Exocet missile while providing naval surface fire support to Royal Marines fighting Argentine troops ashore. Of the British ships destroyed, the SS Atlantic Conveyor was the most important victory achieved by the Argentines during the conflict. This large ship contained critical matériel necessary for combat ashore, including ten helicopters. The loss of these critical airlift assets required British ground forces to hike 60 kilometers from their landing sites to assault the provincial capital of Port Stanley rather than seize it via air assault as originally planned (Middlebrook 2001). All these losses were inflicted on the British task force via the very limited stock of Exocet missiles (an early 1970s era munition) possessed by Argentina. The damage inflicted on the SS Atlantic Conveyor by Exocet missiles is shown in Figure 2.

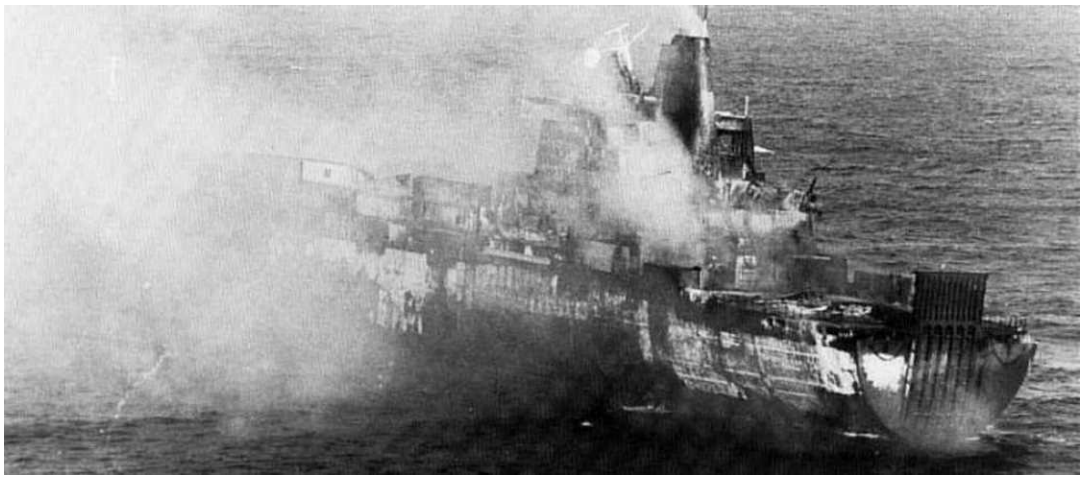

Figure 2. The SS Atlantic Conveyor after receiving two hits from Argentine Exocet missiles. Source: Think Defence (2021).

The PRC's A2/AD forces currently include massive quantities of guided missiles of much greater capability and sophistication than those fielded by the Argentine forces during the Falklands War (DON 2021). It does not require a great leap of imagination to consider the implications of such a capability concerning an American amphibious task force operating in the FIC opposed by PRC A2/AD systems. Continuing to execute amphibious operations along 20th century era modes and methods is an unacceptably hazardous proposition that risks severe attrition and mission failure for maritime forces. This is particularly true with the limited number of high-value, high-cost amphibious ships and maritime prepositioning vessels available to the U.S. Navy. The reduction of the United States' national shipbuilding industry means that any combat losses of large naval vessels cannot be quickly replaced by new ship construction or deployment from other theaters. As demonstrated by the example of the SS Atlantic Conveyor, logistical platforms are a prime target for forces seeking to disrupt amphibious operations. Therefore, careful husbanding of high-value logistical platforms and the optimized use of their associated connectors is of great concern to U.S. Navy-Marine Corps planners.

D. ORGANIZATION OF THE THESIS

The outline for the remainder of this thesis is as follows: Chapter II, the literature review, examines the applications of network theory in military and civilian logistics

optimization models. Chapter III, model formulation, discuses the formulation and implementation of the Path Enumeration-Mixed Integer Program (PE-MIP) model used in this thesis and its antecedent model, Lieutenant Colonel Freeman's S-MIP. Chapter IV, experimentation, covers the execution and results of experiments conducted during this research. Chapter V is the conclusion chapter, and it contains a summary of this research's results and recommendations for future research.

THIS PAGE INTENTIONALLY LEFT BLANK

II. LITERATURE REVIEW

This chapter contains a summary of previous relevant, selected research pertaining to logistic optimization models employing network theory for military applications.

A. MOTIVATION FOR MILITARY LOGISTICS OPTIMIZATION MODELS

"Getting there the firstest with the mostest" (sic) and "amateurs talk tactics, professionals talk logistics" are two common idioms in American military circles. These idioms reflect a broader appreciation in American military culture for the vital importance of logistics in planning and executing successful military operations. There has therefore, unsurprisingly, been a significant amount of research conducted on logistics optimization in support of military operations by America's defense research institutions.

The Naval Postgraduate School (NPS) has produced a large volume of research concerning military logistics optimization in its many forms. Military logistics is a very broad term covering many activities including maintenance, supply, transportation, and other functions. Therefore, it is necessary to limit this review to only research concerning transportation and distribution applications. Due to the inherent nature of transportation and distribution problems, these problems are most often modeled using network theory. Chapter III of this thesis, Model Formulation, contains a basic description of underlying concepts of network theory. The following sections of this chapter briefly describe some historical applications of network theory to civilian and military transportation and distribution problems.

B. NETWORK THEORY TRANSPORTATION AND DISTRIBUTION APPLICATIONS

Network theory is endemic to modern life. Transportation networks, communications networks, supply networks, social networks, and power networks affect nearly every aspect of our daily lives. Naturally, this underlying structure of modernity has elicited a significant amount of research and study in civilian and military contexts. The origins of network theory stem from the famed mathematician Leonhard Euler's 1736 paper The Seven Bridges of Konigsberg (Estrada et al. 2015). Network theory matured in

the industrial age as it was applied to various problems concerning newly built railroad networks and telegraph communications systems.

Study of transportation problems through the lens of optimizing railroad traffic is one of the oldest and most robust applications of network theory in an operations research (OR) context. In 1930, Soviet mathematician A.N. Tolstoĭ made the foundational discovery that the optimal solution for a transportation problem has no negative cost cycles in its residual graph (Schrijver 2002). Tolstoĭ's paper Methods of Finding the Minimal Total Kilometrage in Cargo-transportation Planning in Space was an early large-scale optimization problem designed to help the Soviet Union improve its industrial (and military) capacity vis-à-vis its railroad network (Schrijver 2002). Since Tolstoĭ's work, the problem size and computational resources available to researchers studying railroad transportation problems have both increased. Research on improving railroad transportation networks has continued to today with many notable projects undertaken in the last decade (Brondörfer et al. 2015).

Modern European railroad networks typically operate both passenger and freight trains dispatched from several countries across the same lines. Passenger and freight trains are scheduled and routed based on different priorities and with different requirements in terms of the times at which they operate and the frequency/regularity with which they must service different locations in Europe. In support of this problem, Germany's national railway company, Deutsche Bahn (DB), began developing in 2010 a decision support system to improve freight routing. This decision support is based on a mixed integer nonlinear program (MINLP) combined with an insertion heuristic which determines when to insert (dispatch) a freight train into Germany's railroad network (Brondörfer et al. 2015). DB's MINLP must account for high passenger train traffic during certain periods of the day (morning/evening rush hour) and then minimize the summation of a congestion cost, train running time, and length of each freight train's route. Due to its complexity, DB's MINLP provides a heuristic solution for when to dispatch freight trains. However, with a problem instance of 1,620 nodes, 5,162 arcs, and 3,350 trains, the heuristic solution was shown to be on average 2% more costly than the optimal solution (Brondörfer et al. 2015).

Additional work by DB in the last decade concerning optimization of rolling stock rotation earned DB recognition by the Institute for Operations Research and the Management Sciences (INFORMS) at the 2020 Franz Edelman competition. DB's Rolling Stock Rotation Problem (RSRP) employed the idea of a hyperarc, which accounts for arcs traversed by a train with a locomotive on both sides of the train. This modeling concept is essential for RSRP since train rolling stock optimization must account for locomotive orientation as a unique factor of railroad network efficiency because coupling and reconnecting locomotives is a time-intensive process. DB's decision support system was shown to increase productivity of its locomotives and railcars by up to 10% while simultaneously improving time savings by 80% (INFORMS, 2020).

Besides railroad optimization problems, many other distribution and transportation problems have also been successfully modeled using network theory. The vehicle routing problem (VRP) and its variations are heavily researched transportation and logistics optimization problems founded on network theory. In VRP, a distributor must service outlying customers from the distributor's depots with a finite number of vehicles. The distributor and its customers are connected in a network by arcs with some nonnegative cost imposed on each vehicle traversing an arc. In its simplest form, the mathematical formulation of a VRP is an integer program that sums the costs of dispatching a vehicle to service each costumer. This cost summation is subject the constraint of having enough vehicles to serve all customers (Laporte 1992). Study of the VRP and its derivatives has produced several models with unique features and includes both exact and heuristic algorithms (Yuen et al. 2008). The complexity and size of modern distribution problems have contributed to wide employment of heuristic algorithms in time-sensitive VRP applications like those experienced by modern delivery services.

United Parcel Service (UPS) recently received recognition for its work on the VRP by employing a heuristic algorithm to optimize its daily package delivery services. UPS is one of the largest private delivery companies worldwide, making its VRP instances very large and complex and its associated operating costs very expensive (INFORMS 2016). UPS developed the On-Road Integrated Optimization and Navigation (ORION) system to minimize its delivery costs. ORION employs clustering techniques to partition customers

into clusters of nodes easily serviced sequentially along the driver's route (INFORMS 2016). ORION also accounts for time-sensitive considerations such as business hours for commercial customers and heavy traffic times along a driver's prospective routes. These features of ORION's heuristic algorithm balance the length/cost of the route with the requirement of satisfying a customer's delivery schedule times. ORION even enables drivers to alter their rough-cut routes and schedules based on human knowledge of current route conditions not accounted for by the algorithm. In 2016, members of UPS's OR department received the Franz Edelman Award for their work on the VRP (INFORMS 2016).

C. NETWORK THEORY APPLIED TO MILITARY LOGISTICS MODELS

As demonstrated by the prevalence of network theory research in civilian applications, it is unsurprising that there are many military applications of network theory concerning logistics problems. This trend emerged as the technological advances of the industrial revolution were quickly applied to military purposes in the latter half of the nineteenth century. Growing populations, increased production of war materials, and more powerful central governments resulted in warfare transitioning from a limited activity conducted by relatively small militaries to a truly industrial-scale activity. As demonstrated by the robust, detailed mobilization and deployment plans developed by each belligerent prior to the First World War, the optimization of transportation networks soon became a matter of national survival. The shortcomings of each belligerent's August 1914 mobilization and deployment schemes further motivated the need for more quantitative and rigorous methods for evaluating transportation networks.

One of the most famous examples of military research concerning network theory was an American study of the Soviet rail network during the Cold War. In 1954, RAND scientists Ford and Fulkerson initiated research on the maximum flow, minimum capacity cut problem with their paper "Maximal Flow through a Network" (Ford and Fulkerson 1954). This work was followed in 1955 by a then classified RAND Report by T.E. Harris and F.S. Ross. This report, "Fundamentals of a Method for Evaluation Rail Net Capacities" (Harris and Ross 1955), concerned interdicting the flow of troops and material along Soviet

railways during wartime. While the focus of this research is not interdiction, the work of Ford, Fulkerson, Harris, and Ross is foundational in the field of network theory. The methods pioneered by these researchers underpin much of modern network theory and have enabled much progress in modern network theory and its applications.

NPS has produced many notable examples of logistics optimization research pertaining to transportation and distribution applications. Since NPS is first and foremost a center for naval research, there is a large volume of readily obtainable research regarding transportation and distribution networks supporting amphibious operations. Peter Ward created the Ship-to-Shore Transportation Problem (SSTP) in 2008 as a means of optimizing the employment of connectors in support of humanitarian aid and disaster relief (HA/DR) operations. Ward formulated SSTP as a mixed-integer optimization model implemented with its associated optimization tool in Xpress-MP (Ward 2008). While Ward's research is limited to a transportation network built around the T-HA class of hospital ship (USNS Comfort and Mercy), it is useful in that it contains methods for weighting different types of patients based on priority for evacuation. Ward's work also accounts for the heterogenous mixture of connectors available during amphibious operations and the varying capabilities of different types, models, and series of connectors.

In 2017, Major Robert Christafore continued research into distribution networks during amphibious operations with his study of bulk fuel distribution from ship to shore in support of Marine Expeditionary Units (MEU). Christafore developed a layered analytical planning tool, the MEU Amphibious Connector Scheduler (MCAS), for use by Navy and Marine Corps planners. MCAS is a multi-model analytical tool that employs a linear program to satisfy fuel demands of nodes ashore then inputs the results of this linear program into an assignment heuristic algorithm that assigns volumes of bulk fuel to a specific connector (Christafore 2017). MCAS then uses another linear program to schedule the arrival and departure of connectors with their embarked quantities of fuel. MCAS is noteworthy because the interaction of the assignment heuristic inputting shipments of fuel into the scheduler linear program dramatically decreases the computational difficulty of this problem. Without an assignment heuristic, this problem could have become intractable

and required a binary mixed integer assignment program with much longer runtime and associated complexity (Christafore 2017).

Major Matthew Danielson expanded the application of MCAS in 2018 to include other critical warfighting commodities, such as water and ammunition. Danielson's improvements on MCAS resulted in a temporal network flow model that optimizes round trips made by connectors from ship to shore (Danielson 2018). Another useful feature incorporated by Danielson was that his methodology generated several schedules with varying attributes allowing decision makers to select the distribution schedule most appropriate for operations ashore.

As stated in Chapter I, this thesis seeks to improve the S-MIP model developed by Lieutenant Colonel Freeman in 2019. In 2021, Captain Forest Sentinella applied design of experiment (DOE) techniques and data farming methods to analyze S-MIP for possible means of improving its solution quality and runtime (Sentinella 2021). Captain Sentinella's work largely focused on evaluating the advantages of using a single solve methodology versus a rolling horizon solve strategy during S-MIP runs. The single solve strategy has the disadvantage of a prohibitively large number of variables as the model attempts to solve all its component pieces simultaneously. Conversely, the rolling horizon strategy breaks the optimization problem into smaller component pieces and solves them sequentially, keeping previously solved for variables fixed (Sentinella 2021). Albeit with less optimal solutions, Captain Sentinella's efforts dramatically improved S-MIP's run time as demonstrated by a rolling horizon solution run time of 44.97 hours versus a single solve solution run time of 232.56 hours for the same problem inputs (Sentinella 2021).

The results of Captain Sentinella's research demonstrated that further improvements were required for S-MIP before it could be applied to large-scale force deployment problems. These results inspired the decision to experiment with the path enumeration (PE) techniques used in this thesis. PE is a means of exploring networks by explicitly defining every unique route or "path" that can be followed from source to destination within a network. PE techniques were successfully employed during Operation Iraqi Freedom and Enduring Freedom (OIF/OEF) to optimization distribution of sustainment in a contested environment.

In 2005, the rampant insurgent use of improvised explosive devices (IEDs) was inflicting an unacceptable number of casualties on logistics convoys supporting operations in Iraq and Afghanistan. The U.S. Central Command (CENTCOM) commander issued guidance that as much sustainment as possible should be airlifted to minimize the number of ground convoys exposed to IED attack. Since airlifting personnel and supplies is far more expensive than ground transportation, some type of optimization was required to make this new logistics paradigm feasible. Dr. Gerald Brown, Dr. Matthew Carlyle, and Dr. Robert Dell of the NPS OR Department, along with Mr. John Brau of U.S. Transportation Command (TRANSCOM), developed the Air Tasking and Efficiency Model (ATEM) to improve CENTCOM's air distribution network. ATEM optimized intra-theater lift in CENTCOM using an easily deployable tool written in Microsoft's Visual Basic with a Microsoft Excel user interface. ATEM modeled Iraq and Afghanistan as a network of airfields connected by the flightpaths flown by various military airlift platforms. Employing a PE heuristic, ATEM provides route guidance for aircraft schedulers to maximize the throughput of their aircraft tasking in support of theater-wide sustainment (Brown, et. al. 2011). Initially fielded in 2006 and incrementally improved upon, ATEM both improved the fiscal cost of sustaining CENTOM's air distribution network and reduced the number of ground convoys exposed to enemy attack in CENTCOM.

THIS PAGE INTENTIONALLY LEFT BLANK

III. MODEL FORMULATION AND RELATED MODELS

This chapter describes two formulations of our path-enumeration-based mixed integer program, PE-MIP. The initial formulation of PE-MIP, PE-MIPv1, is described in Section A. PE-MIPv2, a reformulated version of PE-MIPv1, is defined in Section B. The generic name PE-MIP is used when referring to attributes common to both PE-MIPv1 and PE-MIPv2. PE-MIP's required inputs and PE-MIP's relationship with the S-MIP model are also discussed in this chapter.

A. INITIAL PE-MIP MODEL FORMULATION (PE-MIPV1)

In July 2021, Dr. Emily Craparo and Dr. Matthew Carlyle of the NPS OR Department reformulated Lieutenant Colonel Freeman's S-MIP model as a PE-based approximation with the intent of improving S-MIP's solve time and the quality of the resulting solution. Like Lieutenant Colonel Freeman's original model, this reformulation was designed to optimize the delivery of equipment and personnel from source locations to initial operating positions in support of EABO. The resultant reformulation, PE-MIP, and the original S-MIP are based on network theory and the concept that transportation and distribution networks can be represented by directed networks. In a directed network, locations are known as nodes, and the connections between nodes are referred to as arcs, which are ordered pairs of direct distinct nodes (Estrada et al. 2015). Directed networks differ from other networks or graphs in that a directed network is a directed graph whose nodes and/or arcs have associated numerical values (Estrada et al. 2015). These numerical values are typically costs, capacities, and/or supplies and demands within the network. A simple directed network is shown in Figure 3.

The structure of a directed network can be easily applied to real-world distribution networks by modeling depots or other logistic sites as nodes and the routes connecting them as arcs. It is important to note that physical proximity in a network does imply connectivity in a directed network. For example, in Figure 4 nodes D and E are physically close to one another on the map, but within the network one cannot reach node D from E and vice versa directly. Instead, one must traverse several nodes and arcs to reach D from E.

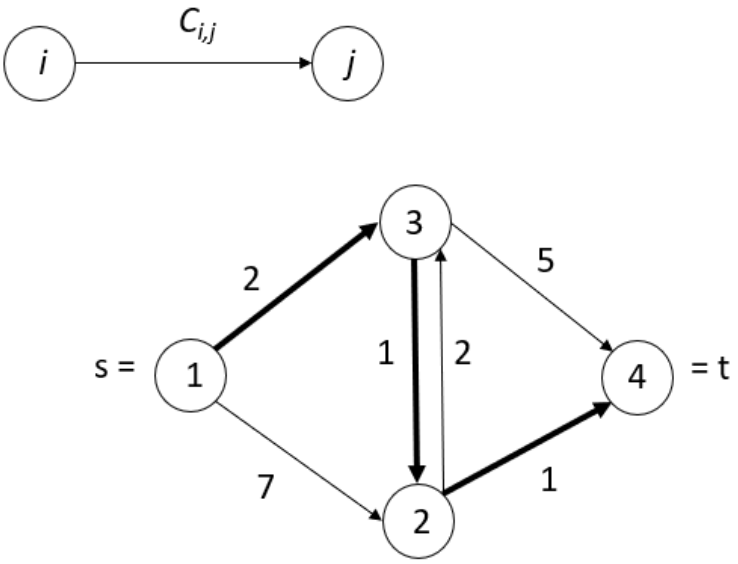

Figure 3. A simple example of a directed network. Bolded arcs depict the minimum cost path from node 1 to node 4.

Directed networks are also useful for modeling the natural effects of the world on distribution operations. For example, omitting arcs between nodes can account for physical phenomenon such as impassable reefs that bar nautical transit or other impassable terrain. This approach is on display in Figure 4 where despite the physical proximity of nodes D and E, the nodes are unconnected due to the reefs separating these nodes on their separate islands.

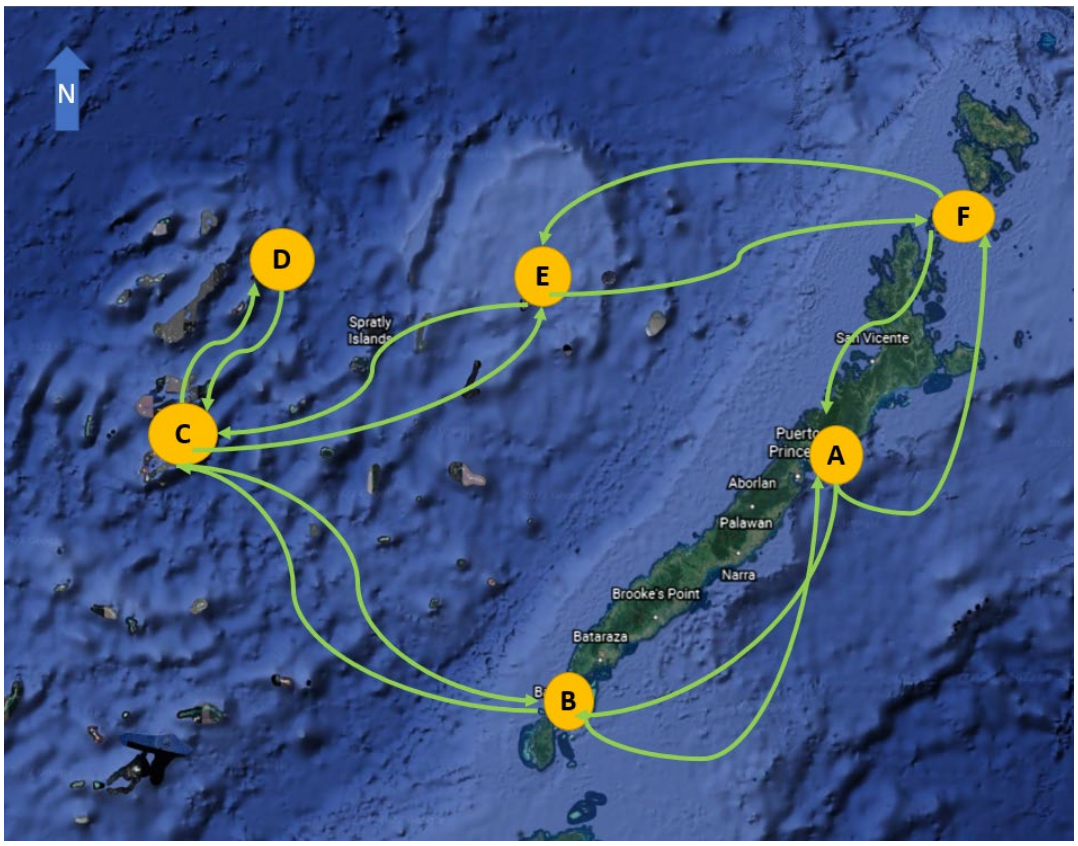

Figure 4. A simple example of a directed network overlaid on the Spratly Islands with nodes depicted in orange and arcs in green. Arc costs and capacities are omitted.

Personnel and equipment traversing the directed networks modeled by S-MIP and PE-MIP are packaged together as serials at their source node and travel together as a cohesive entity. Optimal serials are made automatically by the model using a heuristic algorithm or input by the user as will be discussed in Part B of this chapter. PE-MIP differs from S-MIP in that PE-MIP restricts the number of solutions available to the model to only those paths generated by the path enumeration code. This restriction reduces the feasible solutions available to PE-MIP, thus reducing the overall computational difficulty associated with solving the underlying optimization problem. PE-MIP's reduction in feasible solutions can result in less optimal objective function values, but with the gained benefit of much faster solution times. Therefore, given a limited model run time PE-MIP could produce a solution within specified optimality parameters whilst S-MIP continues working towards producing a better result in the given time limit. The following section

provides the complete formulation of PE-MIPv1 (which is a reformulation of Lieutenant Colonel Freeman's S-MIP) in its component sets, indices, data, decision variables, objective function, and constraints.

Key to our path-enumeration based formulation is a finite list of directed paths for serials to follow. Serials are assigned to these paths, and connectors are assigned to cover specific arcs along these paths to ensure the serials are delivered to their destinations. The arcs in each path $p \in PATHS$ are given by the tuples (i, j, p') the set $PARCS$ that have $p' = p$, and each element (s, p) in the set $SPATHS$ indicates that serial s can travel along path p. Inclusion in the set $SPATHS$ is based on whether path p starts and ends at the starting point and destination for serial s. Figure 5 depicts the full formulation of PE-MIPv1.

1. Sets and Indices

$v \in CXRS$	connectors
$s \in EQUIP$	serials
$p \in PATHS$	paths for serials
$i, j \in NODES$	nodes/locations
$(i, j) \in ARCS$	arcs
$k \in LEGS$	ordinal indices for connector legs (first leg, second leg, etc.)
$(v, k) \in CLEGS$	connector v executes at least k legs
$(v, i, j) \in TRIPS$	connector v can travel on arc (i, j)
$(v, k, i, j) \in S\_TRIPS$	connector transits
$(s, v, k, i, j) \in LOADS$	connector-serial transits
$(s, p) \in SPATHS$	serial s can travel path p
$(i, j, p) \in PARCS$	path p contains arc (i, j)

2. Data [units]

h_v	usable deck area for connector v [sq.ft]
v_s	value/priority weight for serial s
a_s	deck footprint size of serial s [sq.ft]
$t_{v,i,j}$	time required for connector v to make transit (i, j) [days]

LAYOVER	*time cost for connector to enter/leave nodes [days]*
BIG	"infinite" = |*LEGS*| * (max($t_{v,i,j}$) + *LAYOVER*) * 10
MAXLOOPS	the limit for iterative time windows allowed for the model
STEP	= 1,2,...,*MAXLOOPS*; the current successive time window
e_s	terminal/ending location (node) for serial *s*

3. Decision Variables [units]

$PATH_{s,p}$		= 1 if serial *s* is assigned to path *p*; else 0
P_s		= 1 if serial *s* not delivered to destination; else 0
$X_{v,k,i,j}$		= 1 if cxr *v* makes transit (*i*, *j*) on leg *k*; else 0
$Y_{s,v,k,i,j}$		= 1 if serial *s* loaded for transit (*v*,*k*,*i*,*j*); else 0
$W_{v,k}$	time	>= 0 at which cxr *v* completes leg *k* [days]
$Z_{s,i}$	time	>= 0 at which serial *s* arrives at node *i* [days]

4. Formulation

$$\min_{W,X,Y,Z,P,PATH} \quad z = \sum_{s \in EQUIP} v_s (Z_{s,e_s} + P_s BIG)$$

S.T.

$$\sum_{p(s,p) \in SPATHS} PATH_{s,p} + P_s = 1 \ \forall s \in EQUIP \qquad (C1)$$

$$\sum_{(v,k) \in CLEGS} Y_{s,v,k,i,j} = \sum_{p(i,j,p) \in PARCS} PATH_{s,p} \ \forall s \in EQUIP, (i,j): \exists\, p(s,p) \in SPATHS \ and \ (i,j,p) \in PARCS \qquad (C2)$$

$$\sum_{s(s,v,k,i,j) \in LOADS} a_s Y_{s,v,k,i,j} \leq h_v X_{v,k,i,j} \quad \forall (v,k,i,j) \in S\_TRIPS \qquad (C3)$$

$$\sum_{(i,j):(v,k,i,j) \in S\_TRIPS} X_{v,k,i,j} \leq 1 \quad \forall (v,k) \in CLEGS \qquad (C4)$$

$$\sum_{j':(v,j',i):(v,k,i,j) \in S\_TRIPS} X_{v,k-1,j',i} \geq X_{v,k,i,j} \quad \forall (v,k,i,j) \in S\_TRIPS: k > 1 \qquad (C5)$$

$$t_{v,i,j} X_{v,k,i,j} + W_{v,k-1} + LAYOVER \leq W_{v,k} \quad \forall (v,k,i,j) \in S\_TRIPS: k > 1 \qquad (C6)$$

$$W_{v,k} + (Y_{s,v,k,i,j} - 1) BIG + LAYOVER \leq Z_{s,i} \ \forall (s,v,k,i,j) \in LOADS \qquad (C7)$$

$$Z_{s,i} + t_{v,i,j} + (Y_{s,v,k,i,j} - 1) BIG + LAYOVER \leq W_{v,k} \ \forall (s,v,k,i,j) \in LOADS \qquad (C8)$$

$$W_{v,k}, Z_{s,i} \geq 0 \ \forall v,k,s,i \qquad (C9)$$

$$X_{v,k,i,j}, Y_{s,v,k,i,j}, P_s, PATH_{s,p} \in \{0,1\} \ \forall v,k,s,i,j,p \qquad (C10)$$

Figure 5. Initial PE-MIP formulation with objective function and constraints.

Constraint (C1) states that each serial *s* should either be assigned to one of its possible paths or marked as unassigned, and therefore undelivered.

Constraint (C2) states that for each arc (*i, j*) along the path to which serial *s* is assigned, the serial must be carried by some connector along that arc.

Constraint (C3) states that the total deck space of serials being delivered by vessel *v* on leg *k* from *i* to *j* cannot exceed h_v.

Constraint (C4) states that a connector must be assigned to at most one arc (*i,j*) per leg *k*.

Constraint (C5) states that vessel *v* can only make transit (*i,j*) on leg *k* if it also made a transit to *i* on leg *k* – 1.

Constraint (C6) states that the leg-k completion time of vessel v $W_{v,k}$ is not less than the leg-k – 1 completion time of that connector, plus the transit time of the kth leg (and associated layover).

Constraint (C7) states that the arrival time of serial s at j $Z_{s,j}$ is not less than the vessel v leg-transit (k,i,j) completion time $W_{v,k} + LAYOVER$, or zero for all trips (v,k,i,j) on which serial s does not arrive.

Constraint (C8) states that the k leg-completion time of vessel v $W_{v,k}$ is not less than the transit time $t_{v,i,j}$ plus the arrival time $Z_{s,i}$ for any cargo loaded from i to j, or zero for cargo not loaded.

Constraints (C9) and (C10) define decision variable domains.

B. PE-MIP REFORMULATION (PE-MIPV2)

During experimentation it was discovered that the initial formulation PE-MIPv1 has trouble determining feasible solutions of reasonable quality within an acceptable amount of time. The lackluster performance of PE-MIPv1 can be attributed to its constraints (C7) and (C8); these constraints cause significant difficulty for the solver as it attempts to close the optimality gap to the desired tolerance. The results returned by PE-MIPv1 caused the research team to explore a reformulation of PE-MIPv1 featuring a new objective function, new constraints, and a new method of calculating the penalty parameter *BIG*.

This reformulation of PE-MIPv1, hereafter referred to as PE-MIPv2, possesses several advantages over PE-MIPv1. The new method of calculating *BIG* improves the performance of (C7) and (C8) mitigating some of the difficulty encountered with these constraints in PE-MIPv1. The new constraints (C9) – (C11) establish valid lower bounds for the variable Z_s^{obj} which enhanced the efficiency of the first term of PE-MIPv2's objective function. Finally, in conjunction with the previously described changes, PE-MIPv2's objective function allows for the meaningful use of an absolute optimality gap termination criterion based on delivery of all serials. The results returned by PE-MIPv2 will be examined at length in Chapters IV and V. Figure 6 depicts the full formulation of PE-MIPv2.

1. Sets and Indices

The reformulation of PE-MIPv2 did not result in any changes to the sets and indices previously described in this chapter concerning PE-MIPv1.

2. Data [units]

h_v	usable deck area for cxr v [sq.ft]
v_s	value/priority weight for serial s
a_s	deck footprint size of serial s [sq.ft]
$t_{v,i,j}$	time req'd for cxr v to make transit (i,j) [days]
$LAYOVER$	time cost for cxrs to enter/leave nodes [days]
$maxhops_s$	maximum number of nodes visited by serial s on any path
$best\_possible\_time_{s,p}$	fastest transit time possible for serial s along path p, excluding layovers
$fastest\_time_s$	fastest transit time possible along any of serial s's potential paths, excluding layovers
BIG	$\sum_s 2\left(LAYOVER\left(maxhops_s - 2\right) + fastest\_time_s\right)$
e_s	terminal/ending location (node) for serial s

3. Decision Variables [units]

$PATH_{s,p}$	=1 if serial s is assigned to path p; else 0
P_s	=1 if serial s not delivered to destination; else 0
$X_{v,k,i,j}$	=1 if cxr v makes transit (i,j) on leg k; else 0
$Y_{s,v,k,i,j}$	=1 if serial s loaded for transit (v,k,i,j); else 0
$W_{v,k}$	time>=0 at which cxr v completes leg k [days]
$Z_{s,i}$	time>=0 at which serial s arrives at node i [days]
Z_s^{obj}	objective penalty for serial s [days]

4. Formulation

$$\min_{W,X,Y,Z,P,PATH} \quad z = \sum_{s \in EQUIP} v_s \left(\frac{Z_s^{obj}}{|EQUIP|} + P_s BIG\right)$$

S.T.

$$\sum_{p(s,p) \in SPATHS} PATH_{s,p} + P_s = 1 \; \forall s \in EQUIP \qquad (C1)$$

$$\sum_{(v,k) \in CLEGS} Y_{s,v,k,i,j} = \sum_{p(i,j,p) \in PARCS} PATH_{s,p} \; \forall s \in EQUIP, (i,j): \exists \, p(s,p) \in SPATHS \; and \; (i,j,p) \in PARCS \qquad (C2)$$

$$\sum_{s(s,v,k,i,j) \in LOADS} a_s Y_{s,v,k,i,j} \leq h_v X_{v,k,i,j} \quad \forall (v,k,i,j) \in S\_TRIPS \qquad (C3)$$

$$\sum_{(i,j):(v,k,i,j) \in S\_TRIPS} X_{v,k,i,j} \leq 1 \quad \forall (v,k) \in CLEGS \qquad (C4)$$

$$\sum_{j\prime:(v,j\prime,i):(v,k,i,j) \in S\_TRIPS} X_{v,k-1,j\prime,i} \geq X_{v,k,i,j} \quad \forall (v,k,i,j) \in S\_TRIPS: k > 1 \qquad (C5)$$

$$t_{v,i,j} X_{v,k,i,j} + W_{v,k-1} + LAYOVER \leq W_{v,k} \quad \forall (v,k,i,j) \in S\_TRIPS: k > 1 \qquad (C6)$$

$$W_{v,k} + (Y_{s,v,k,i,j} - 1) BIG + LAYOVER \leq Z_{s,i} \; \forall (s,v,k,i,j) \in LOADS \qquad (C7)$$

$$Z_{s,i} + t_{v,i,j} + (Y_{s,v,k,i,j} - 1) BIG + LAYOVER \leq W_{v,k} \; \forall (s,v,k,i,j) \in LOADS \qquad (C8)$$

$$Z_s^{obj} \geq Z_{s,e_s} \forall s \qquad (C9')$$

$$Z_s^{obj} \geq \sum_{p:(s,p) \in SPATHS} best\_possible\_time_{s,p} PATH_{s,p} \; \forall \, s \qquad (C10')$$

$$Z_s^{obj} \geq fastest\_time_s \; \forall s \qquad (C11)$$

$$W_{v,k}, Z_{s,i}, Z_s^{obj} \geq 0 \; \forall (v,k,s,i) \qquad (C12)$$

$$X_{v,k,i,j}, Y_{s,v,k,i,j}, P_s, PATH_{s,p} \in \{0,1\} \; \forall \, v,k,s,i,j,p \qquad (C13)$$

Figure 6. PE-MIPv2 Formulation with objective function and constraints.

Constraint (C1) states that each serial *s* should either be assigned to one of its possible paths or marked as unassigned.

Constraint (C2) states for each arc (*i, j*) along the path to which serial *s* is assigned, the serial must be carried by some connector along that arc.

Constraint (C3) states that the total deck space of serials being delivered by vessel *v* on leg *k* from *i* to *j* cannot exceed h_v.

Constraint (C4) states that a connector must be assigned to at most one arc (*i,j*) per leg *k*.

Constraint (C5) states that vessel v can only make transit (i,j) on leg k if it also made a transit to i on leg k-1.

Constraint (C6) states that the k leg-completion time of vessel v $W_{v,k}$ is not less than the k-1 leg-completion time of that connector, plus the transit time of the kth leg (and associated layover).

Constraint (C7) states that the arrival time of serial s at j $Z_{s,j}$ is not less than the vessel v leg-transit (k,i,j) completion time $W_{v,k}$ + LAYOVER, or zero for all trips (v,k,i,j) on which serial s does not arrive.

Constraint (C8) states that the k leg-completion time of vessel v $W_{v,k}$ is not less than the transit time $t_{v,i,j}$ plus the arrival time $Z_{s,i}$ for any cargo loaded from i to j, or zero for cargo not loaded.

Constraints (C9')-(C11) establish valid lower bounds on the variable used to penalize delivery time in the objective function.

Constraints (C12) and (C13) define decision variable domains.

C. PE-MIP AND S-MIP COMPARISON

PE-MIP and its antecedent model S-MIP share many similar attributes and features. The two models primarily differ in their objective functions. As shown by Figure 7, S-MIP's objective function uses the summation of three terms taking into consideration the following factors: delivery along the shortest-path distance for each serial (first term), serial delivery time by serial value/priority (second term), and a penalty for undelivered serials (third term).

$$\min_{W,X,Y,Z,P,U} \quad z = \sum_{s \in EQUIP} \left(STEP v_s U_s + v_s Z_{s,e_s} + v_s P_S BIG \right)$$

Figure 7. S-MIP objective function.

S-MIP's formulation uses its objective function and twelve associated constraints to plan transits for connectors that get serials as close to each serial's destination as possible. S-MIP's constraints are used to enforce real world phenomenon like continuity of connector/

serial movement, connector capacities, calculating time events occur, and recording serial delivers. Unlike S-MIP, PE-MIP makes no consideration in its objective function for the actual shortest-path for serial delivery. Instead, the fastest paths for serial delivery are down selected through various filters in PE-MIP's path enumeration code. These filtering methods are covered in greater detail in Chapter IV. Furthermore, S-MIP's formulation makes no consideration for path like data structures that are found in PE-MIP. Instead, S-MIP focuses on making decisions in a piecewise manner concerning each leg of a serial's journey from source to destination. The full formulation of S-MIP is found in Appendix B: S-MIP Connector Model Formulation.

D. PE-MIP INPUT DATA

Lieutenant Colonel Freeman designed the original S-MIP model to receive its input data from an Excel workbook. PE-MIP retains this feature of S-MIP and receives its inputs via an Excel workbook. Within the Excel workbook the data inputs are organized into six worksheets: WhatWhere, InputSerials, HowFar, HowFast, ConnectorData, and ExperimentDesign**.** (See Appendix A: ConnectorFarmer Manual (V0.17) (Upton 2021)). The worksheets InputSerials and HowFast are new additions to PE-MIP's inputs that were not featured in the original S-MIP inputs. As will be discussed, InputSerials and WhatWhere are two mutually exclusive worksheets that cannot be used in conjunction with one another. The following sections of this chapter discuss in detail the organization and data contained within the six input worksheets. Table 1 contains a summary of each worksheet and its associated data inputs and variables.

Table 1. Summary of input data specified via the Excel workbook if not using user specified input serials. Variables in red text indicate minimal requirements for the model to run. Adapted from Freeman (2019) and Sentinella (2021).

	WhatWhere (6)	HowFar (4)	ConnectorData (8)	ExperimentalDesign (9)	HowFast (3)
Required **Optional**	Site Type Lat Long StS Site StS Lat StS Lon IOP Source Serial Desc. TAMCN Length Width Weight Stick Value Sqft	Loc Access - Connector X Access - Connector Y Access - Connector Z	Type Square Footage Broken Stow Factor Usable Square Footage Sust. Speed - Sea State 1 Sust. Speed - Sea State 2 Sust. Speed - Sea State 3 Sust. Speed – Sea State 4	Design Point Connector X Qty Connector Y Qty Connector Z Qty SeaState Total Cap MAXK VARLIM OVERLAP mip_tolerances_mipgap timelimit SERIALSIZE FASTPATH HOPS_COEF HOPS_ADD USE_ABSMIPGAP	From To Time

1. WhatWhere

WhatWhere is the input worksheet concerning which personnel and what equipment goes from each source location to its assigned initial operating positions via a designated ship-to-shore site (StS). Each row of this worksheet represents personnel and equipment organized as modular "sticks" that can be formed by the model into serials for assignment to connectors and movement through the network. The model does not require every input column on the worksheet WhatWhere to run. Some columns are included for additional insight and situational awareness on the part of the user to understand what equipment and which personnel are assigned to each connector. The TACMN column is an example of a column included to improve the user interface but not required for the model to run. TACMN, or Table of Authorized Control Number, is a standardized administrative alphanumeric designator used to describe the many types of equipment in military service. For example, B2605 denotes a tactical water purification system, while E0947 denotes a light armored vehicle with a 25mm autocannon. Site, Lat/Long, ShiptoShore Lat/Long, Serial description, and Weight are other columns not required by the model but included in WhatWhere because they are useful for the user's situational awareness. As shown in Figure 8, PE-MIP requires the following columns from

WhatWhere: StS Site, initial operating position (IOP), Source, Length, Width, and Stick Value.

Site	Type	Lat	Lon	StS Site	StS Lat	StS Lon	IOP	Source	Serial Desc.	TAMCN	Length	Width	Weight	Stick Value	sqft
1				A			P1	D	DA-1-EL0001-MLR-3	ABCDE	710.04	12	2.19	1	59.17
1				A			P1	D	DA-1-EL0001-MLR-3	ABCDE	710.04	12	2.19	1	59.17
1				A			P1	D	DA-1-EL0001-MLR-3	ABCDE	710.04	12	2.19	1	59.17

Figure 8. A screen grab of a simple WhatWhere worksheet, best viewed in color to denote required versus not required columns. Adapted from Freeman (2019) and Sentinella (2021).

2. InputSerials

If users do not want the model to create serials via a heuristic algorithm, they have the option of inputting their own prebuilt serials via the worksheet InputSerials. Survivability of critical assets, equipment and operators pairing, and other equipment pairing considerations are all operational reasons why a user could want to build their own serials rather than allow the model to form serials for the user. For example, a user might not want every item of a specific type of critical equipment to move on the same connector in case that connector is intercepted and destroyed by the enemy or lost via an accident. Additionally, specifically trained personnel—e.g., a driver qualified to move a specific type of vehicle on/off a connector—often need to travel with their equipment to facilitate embarkation and debarkation of that equipment. InputSerials has the same columns as WhatWhere, but each row represents a serial in its entirety rather than a modular stick. Figure 9 depicts a sample of InputSerials with required variables in red.

Site	Type	Lat	Lon	StS Site	StS Lat	StS Lon	IOP	Source	Serial Desc.	Weigh	Value	Sqft
1				L			P1	D	DL-1 45 EL0001-MLR-4		1	2662.65
1				L			P1	D	DL-1 45 EL0001-MLR-4		1	2662.65
1				L			P1	D	DL-2 6 EL0002-MLR-4		1	1286.28

Figure 9. A screen grab of a simple InputSerials worksheet, best viewed in color to denote required versus not required columns. Adapted from Freeman (2019) and Sentinella (2021).

3. HowFar

The HowFar worksheet contains two separate categories of input information: connector access and a distance matrix. The connector access portion includes a row for each StS Site and a column for each connector type. The connector access row and column entry for each StS Site and connector type contains a TRUE or FALSE entry denoting whether a specific connector type can land/dock at a specific StS Site. For example, if a connector access row and column entry for a KC130 at StS Site A reads TRUE, then that means that StS Site A has a landing strip capable of accommodating a KC130 aircraft. The distance matrix portion of HowFar defines the physical distances (in nautical miles) between node pairs in the directed network. Blank entries signify that no connector can travel between the nodes specified by the row and column name.

Loc	Access - Connector X	Access - Connector Y	Access - Connector Z	A	B	D	E	G	I	J	L	M	N	O
A	TRUE	TRUE	TRUE		700	1500		430	260	650	400		840	
B	TRUE	TRUE	FALSE				1810	890	640		620		560	
D	TRUE	TRUE	TRUE				980	1470	1530	1350	1530		1370	110
E	TRUE	TRUE	FALSE							1840				940
G	TRUE	TRUE	TRUE						680	250			1210	
I	TRUE	TRUE	TRUE							880	150	1430		
J	TRUE	TRUE	TRUE								1010		1350	
L	TRUE	TRUE	TRUE									1350		
M	FALSE	FALSE	TRUE										1440	
N	TRUE	TRUE	TRUE											
O	TRUE	TRUE	FALSE											

Figure 10. A screen grab of a simple HowFar worksheet, best viewed in color. Connector access appears on the right side of this sheet and the distance matrix on the left side. Adapted from Freeman (2019) and Sentinella (2021).

4. HowFast

In the context of EABO, personnel and equipment are not usually directly delivered to the exact position where they will operate. Instead, personnel and equipment will need to move from the StS Site they debark from to a new position where they will fulfill their operational function. For example, a radar system or anti-ship missile system will normally move away from the beach, pier, or airfield where it is offloaded from a connector to a different location where it will set up and operate. The HowFast worksheet inputs this additional movement time information to capture when a serial can expect to arrive at its true destination after exiting the connector network.

5. ConnectorData

The ConnectorData worksheet encompasses data about the attributes of each connector type. The attribute data for each connector type includes the following: the square footage available on the connector to carry equipment, the broken stow factor, the usable square footage, and sustained speed under sea states 1–4. The broken stow factor is the proportion of the connector's square footage that can be used to transport serials. It follows that usable square footage (= square footage × broken stow factor) (Sentinella, 2021). Sea state is a categorical variable used to define the ocean's surface conditions due to wind, currents, and wave heights. Sea states exceeding sea state 4 preclude safe surface connector operations and are therefore excluded from consideration in this model.

Type	Square Footage	Broken Stow Factor	Usable Square Footage	Sust. Speed - Sea State 1	Sust. Speed - Sea State 2	Sust. Speed - Sea State 3	Sust. Speed - Sea State 4
Connector-X	5040	1	500	15	13	10	8
Connector-Y	328	1	328	348	348	348	348
Connector-Z	15750	1	15750	35	33	30	28

Figure 11. A screen grab of a simple ConnectorData worksheet, best viewed in color. Adapted from Freeman (2019) and Sentinella (2021).

6. ExperimentDesign

The ExperimentDesign worksheet allows users to define serval different cases with different conditions which the model can run. This feature is included so that users can compare how natural phenomena like sea state conditions and operational realities like the quantity available of different types of connectors affect serial deliveries. Each of these conditions is captured as a design point, with each row of this worksheet corresponding to a different design point. Minimum required inputs for the model are found under the columns Design Point (DP), Connector Type Quantities, SeaState, FASTPATH, HOPS_ADD, HOPS_COEF, and USE_ABSMIPGAP.

Design Point	Connector-X Qty	Connector-Y Qty	Connector-Z Qty	SeaState	timelimit	VARLIM	MAXK	OVERLAP	mip_tolerances_mipgap	SERIALSIZE	FAST_PATH	HOPS_ADD	HOPS_COEF	USE_ABSMIPGAP
1	1	3	7	4	36000	10000	13	1	0.05	62	1	0	1	0
2	2	5	3	4	36000	10000	10	1	0.05	62	1	0	1	1
3	0	9	4	4	36000	10000	5	1	0.05	62	2	1	1	0

Figure 12. A screen grab of a simple ExperimentDesign worksheet, best viewed in color. Adapted from Freeman (2019) and Sentinella (2021).

THIS PAGE INTENTIONALLY LEFT BLANK

IV. EXPERIMENTATION

This chapter covers the experiments performed on PE-MIP and S-MIP, the design of experiments, the software used to implement the experiments, methods used to capture the results of experiments, and the techniques used to conduct post-experiment analysis. This chapter also discusses the metrics of interest used to analyze solution quality, model run time, and other insights discovered by this research's experiments.

A. SOFTWARE

PE-MIP is implemented using Python's Pyomo package, an open-source optimization modeling language. All experiments used the academic version of IBM's CPLEX Optimization Studio (IBM 2006) as the model's solver. Every experiment was conducted using ConnectorFarmer (v0.17), a software package developed by Stephen Upton of the Simulation Experiments and Experimental Design (SEED) Center, More information on the SEED Center at https://nps.edu/web/seed/. ConnectorFarmer is a data farming wrapper composed of several Python scripts built to run and extract results from S-MIP. ConnectorFarmer has been incrementally improved and modified as S-MIP has changed during experimentation by OAD and NPS students. In her thesis work, Captain Forest Sentinella used ConnectorFarmer (v0.14) while studying rolling horizon and DOE effects on S-MIP (Sentinella 2021). ConnectorFarmer executes experimentation using S-MIP and PE-MIP with two primary Python scripts: ConnectorRunner and ConnectorMiner. ConnectorRunner facilitates the actual running of models on computing clusters while ConnectorMiner enables post-experiment analysis. ConnectorMiner collects and calculates several summary metrics from each experiment and then outputs these metrics to an easily digestible Excel workbook for review. ConnectorMiner can also produce several charts and graphs to help analysts visually interpret how models are performing and identify potentially problematic or nondeterministic behavior in each experiment. ConnectorFarmer is an ideal wrapper for conducting high-level, computationally expensive experiments because it can run several instances of a model on different cores of the same computer concurrently. The research team used this feature of ConnectorFarmer while

executing experiments on NPS's high-performance computing clusters Hamming and Reaper. Additional information about ConnectorFarmer is available in Appendix C, the Connector Farmer User Manual (v0.17).

B. DESIGN OF EXPERIMENTS

The research team designed initial experiments to answer specific question about the model and aid in debugging nondeterministic model behavior. Later experiments employed full factorial designs of experiment (DOE) to explore the effect of varying different model parameters.

C. INITIAL EXPERIMENTS AND PE-MIP DEVELOPMENT

This section provides a cursory summary of PE-MIP's development. Appendix B: PE-MIP Development provides a full narrative and detailed description of PE-MIP's formulation and incremental evolution from December 2021 through early March 2022. The research team began preliminary experiments with S-MIP and PE-MIPv1 in early December 2021. These early experiments focused on debugging the code associated with PE-MIPv1 and comparing PE-MIPv1 and S-MIP's performance with small scale problem instances. PE-MIP's solution quality and runtime were roughly equivalent to those returned by S-MIP during these opening experiments. Success with small scale problem instances motivated the progression to experimentation with large problem instances of more utility to the research sponsor.

1. Description of the Two MLR Deployment Problem

For large-scale experiments, the research team used inputs from OAD representing the deployment of two MLRs (4th MLR and 12th MLR) from their home stations to initial operating positions. The Marine Corps will ultimately have three MLRs permanently stationed in the Western Pacific. However, the scenario used by OAD to create inputs for this research featured 3d MLR already deployed to its initial operating positions due to regular bilateral military exercises. Therefore, the scenario used for these experiments only required the deployment of two MLRs at the beginning of a conflict.

The research team used the same input data provided by OAD for all large-scale experiments and these inputs will for the remainder of the thesis be referred to as the two MLR deployment problem. The input serials, types, and quantities of connectors available for the two MLR deployment were informed by an OAD working group that had previously evaluated and quantified what the deployment of two MLRs would entail. The two MLR deployment problem input includes 472 serials comprising 173,443.3 square feet of equipment requiring deployment. Connector quantities and types for the two MLR deployment problem were based around a U.S. Navy force structure assumption that nine Light Amphibious Warships (LAW) will be procured to habitually support each MLR. Therefore, 27 total LAWs would be available to support sealift operations in the Western Pacific not accounting for ship maintenance or other unavailability. Two high speed surface ships were also assumed as available due to current big deck fast transit ship inventories. Aircraft connector availability was driven by the assumption of four squadrons of KC-130Js, two squadrons of MV-22Bs, a limited number of serviceable CH-53Ks, and no joint heavy airlift assets, i.e., U.S. Air Force C-17s or C-5s.

All network data in the input files used by the research team is notional and nondescriptive of any real-world operational plan. In other words, the nodes within the input file's network do not represent any planned EAB sites nor should these nodes be thought of as exactly corresponding to any real-world locations in the Western Pacific or elsewhere. Instead, the 17 nodes in the network and their associated arcs lengths and connector access values are informed by real world data obfuscated to provide a useful approximation of the operating environment for unclassified experimentation.

2. Computational Difficulty and Large-Scale Experiments with PE-MIPv1 and S-MIP

Introductory experiments with larger scale problem instances indicated that PE-MIP suffered from the same computational difficulty problems and associated long model runtimes that had inhibited the practical application of S-MIP in support of the research sponsor's objectives. During these initial experiments, the research team identified that the path enumeration techniques used by PE-MIPv1 worked very quickly and efficiently. However, the sheer number of paths enumerated by PE-MIPv1 created a substantial

amount of complexity within the model which degraded its performance. The two MLR deployment problem's network includes 10,303 total possible unique paths from the source nodes to the destination nodes. These 10,303 possible unique paths result in 907,689 total possible unique serial paths that PE-MIPv1 could consider for its solution. Since many of these serial paths are nonsensical, or highly inefficient (e.g., visiting the majority of the nodes in the network when a far shorter path is available), filtering out undesirable and impracticable serial paths can help scope the possible solutions to only reasonable candidate serial paths. Therefore, the research team decided to implement a series of filters on candidate serial paths for the purpose of decreasing PE-MIPv1's computational by reducing the number of candidate serial paths considered by the model.

3. Serial Path Filters for Reducing Computational Difficulty

From its inception and earliest formulation PE-MIPv1 has included a serial path filter that removes candidate serial paths without sufficient capacity for the associated serial to traverse the path. Initial additional filtering efforts focused on sorting out candidate serial paths that visited extraneous nodes during the serial's transit from source to destination. This path length filter used a modified version of the breadth-first search (BFS) algorithm to return the length of (in nodes) of the shortest possible path from source to destination. The length of this array is stored within the code as a local variable "MINHOPS" which is used to calculate the longest allowable path length "MAXHOPS" using the following formula: MAXHOPS = HOPS_COEF * MINHOPS + HOPS_ADD. "HOPS_COEF" and "HOPS_ADD" are user-specified parameters input through the DOE input to provide flexibility in controlling the maximum allowable path length for different DPs.

The path length filter applied through MAXHOPS did improve model runtime. However, as is discussed in Appendix B, these improvements but did not reduce PE-MIPv1's overall computational difficulty enough to allow expansive experiments on large problem instances. The research team decided that an additional filter on candidate serial paths considering serial transit times could further reduce model runtime. The transit time filter calculates the total transit time of a serial from source to destination via the fastest

available connector capable of traversing a path. The transit time filter is implemented via the user input parameter "FASTPATH" which allows the user to specify how many possible serial paths PE-MIPv1 should consider from the remaining serial paths after filtering for path length. This consideration is implemented in accordance with each serial's sorted serial transit times along a serial path. In other words, if the user inputs FASTPATH = 1, then only the serial path with the fastest transit time for that serial (from among those paths remaining following the MAXHOPS filter) is considered for a solution, whereas when FASTPATH = 2, PE-MIPv1 will consider both the fastest and second fastest serial path. Like the path length filter before it, the transit time filter improved PE-MIPv1's runtime, but not to the degree required for applying PE-MIPv1 in support of OAD's requirements. The only modest improvements achieved by the path length and transit time filters resulted in exploring a reformulation of PE-MIPv1; this reformulation ultimately produced PE-MIPv2. This decision is described in detail in Chapter III, Section B. PE-MIPv2 inherited from PE-MIPv1 all the serial path filtering methods described in the preceding chapters.

The reformulation of PE-MIPv1 into PE-MIPv2 immediately returned promising results. Given simple inputs for MAXK, FASTPATH, and HOPS_ADD, PE-MIPv2 returned results in a small fraction of the time required with its previous formulation by PE-MIPv1. This dramatic improvement in model runtime motivated the research team to initiate the final set of comprehensive experiments described in the following sections of this chapter.

D. FINAL EXPERIMENTS

1. Design of Final Experiment

The final experiment revisited the same two MLR deployment problem previously explored during initial experimentation. For the final experiment the research team decided upon a full factorial DOE varying the input variables MAXK, HOPS_ADD, FASTPATH, and USE_ABSMIPGAP. Table 2 depicts the values of the different levels used for each input variable for this DOE.

Table 2. Input variable levels used for final full factorial DOE

MAXK	HOPS_ADD	FASTPATH	USE_ABSMIPGAP	Time Limit (hours)
5, 10, 15	0, 1	1, 2	0, 1 (binary)	24, 48, 72

This full factorial DOE yielded 72 total possible DPs. This final experiment was executed in batches by time limit value starting with DPs using a 24-hour time limit. This method provided considerable efficiency since many of the 72 DPs are duplicates with the only difference being the time limit used. This allowed the researchers to avoid redundant or wasted effort repeating the same deterministic experiment with a larger time limit value for a DP that would produce a known output. Minor variations for identical DPs can occur in model outputs due to the parallel execution of DPs within the high-performance computing clusters used during this research. However, the research team determined that this small variance was acceptable and did not justify the time cost of repeating otherwise identical DPs. Input variables for the number/types of connectors used, sea state conditions, VARLIM, etc., remained fixed for all DPs.

All DPs using a relative optimality gap used a MIP GAP tolerance of 20% while those using an absolute optimality gap termination criterion used the specified value of *BIG* / 2. Per Chapter III, Section D, *BIG* is each DP's surrogate value for infinity and represents the latest possible time a serial could be delivered. *BIG* is used to penalize undelivered serials in the objective function using the binary variable P_s which denotes delivery of serials when $P_s = 0$. Therefore, if all serials are delivered, PE-MIPv2's objective function quickly approaches a value less than *BIG* / 2, causing the DP to terminate. The final round of experiments began on 19 March 22 with the first batch of 24-hour DPs.

2. Final Experiment Results

On 14 April 2022 the research team received the results from the full factorial experiment that began on 19 March 2022. These results included output metrics for 53 of the 72 DPs in the full factorial DOE. Due to the batch process used to prevent duplication of DPs that had previously returned integer optimal solutions within tolerance a total of 18

DPs were not executed. For each of these 18 DPs, another DPs with an identical combination of factors, but a lower time limit, had already returned an integer optimal solution within tolerance. Table 3 summarizes the 18 DPs that were not run due to an acceptable solution having already been achieved with a lower time limit.

Table 3. Summary of DPs that were not executed or did not record results during the final experiment

DPs	MAXK	HOPS_ADD	FASTPATH	USE_ABSMIPGAP	Time Limit (hours)
13, 25	5	0	1	1	48, 72
14, 26	10	0	1	1	48, 72
15, 27	15	0	1	1	48, 72
16, 28	5	0	2	1	48, 72
17, 29	10	0	2	1	48, 72
18, 30	15	0	2	1	48, 72
19, 31	5	1	1	1	48, 72
22, 34	5	1	2	1	48, 72
32	10	1	1	1	72
33	15	1	1	1	72

Of the 24 unique DPs with 24-hour time limits, 16 DPs delivered all serials. Of the 16 DPs that delivered all serials, eight returned integer optimal solutions within tolerance. Every DP that returned an integer optimal solution within tolerance inside of 24 hours used the absolute optimality gap as its termination criteria. Table 4 summarizes the pertinent DP factors and output metrics for those DPs that returned integer optimal solutions within in 24 hours or less. It must be stated that the time the last serial is delivered is itself not an optimized model output. The time the last serial is delivered is instead a byproduct of each individual serial's optimized delivery time. The net result is that the minimized serial delivery times of each individual serial results in a lower time to deliver all serials which is recorded and returned as a model output.

Table 4. Summary of DPs that returned integer optimal solutions within tolerance in 24 hours or less.

DP	MAXK	FAST-PATHS	HOPS-ADD	USE ABS MIP GAP	Prop. Serials Del	Time last serial delivered (days)	Obj. function value	Sol. time (hours)	MIP GAP (%)
1	**5**	**1**	**0**	**1**	**1**	**43**	**11.32**	**3.34**	**93.26**
2	**10**	**1**	**0**	**1**	**1**	**64.7**	**14.12**	**3.04**	**94.6**
3	**15**	**1**	**0**	**1**	**1**	**91.9**	**25.1**	**4.84**	**96.96**
4	**5**	**2**	**0**	**1**	**1**	**43**	**11.32**	**1.81**	**93.26**
5	**10**	**2**	**0**	**1**	**1**	**64.7**	**14.12**	**2.83**	**94.6**
6	**15**	**2**	**0**	**1**	**1**	**87.7**	**26.11**	**4.69**	**97.08**
7	**5**	**1**	**1**	**1**	**1**	**46.4**	**15.17**	**1.22**	**94.97**
10	**5**	**2**	**1**	**1**	**1**	**52.35**	**17.04**	**15.03**	**95.52**

The solution times for the DPs in Table 4 represent dramatic improvement compared to the solution times returned by S-MIP for the same two MLR deployment problem. It is also noteworthy that the outputs of these DPs indicate that PE-MIPv2 yields better results in terms of time to deliver serials with smaller MAXK values. This trend is demonstrated by DPs 1, 4, and 7 which returned the fastest serial delivery times. Conversely, DPs 3 and 6 returned the slowest times to deliver all serials with MAXK = 15. The relationship between MAXK and the quality of PE-MIPv2's solution is more thoroughly discussed in Chapter V, Section B.

The DPs that delivered all their serials but did not return an integer optimal solution within the 24-hour time limit also revealed interesting insights. Table 5 summarizes the DPs that delivered all serials but failed to find an integer optimal solution within tolerance in the 24-hour time limit.

Table 5. Summary of DPs that delivered all serials but did not return integer optimal solutions within tolerance in 24 hours or less.

DP	MAXK	FAST-PATHS	HOPS-ADD	USE ABS MIP GAP	Prop. Serials Del	Time last serial delivered (days)	Obj. function value	Sol. time (hours)	MIP GAP (%)
8	**10**	**1**	**1**	**1**	**1**	**65.7**	**16.87**	**24***	**95.48**
37	**5**	**1**	**0**	**0**	**1**	**24.4**	**2.96**	**24***	**74.21**
38	**10**	**1**	**0**	**0**	**1**	**28.4**	**3.85**	**24***	**80.16**
39	**15**	**1**	**0**	**0**	**1**	**52.5**	**6.6**	**24***	**88.44**
40	**5**	**2**	**0**	**0**	**1**	**9.8**	**2.8**	**24***	**72.73**
41	**10**	**2**	**0**	**0**	**1**	**21.2**	**3.7**	**24***	**79.46**
42	**15**	**2**	**0**	**0**	**1**	**41.4**	**4.7**	**24***	**83.79**
43	**5**	**1**	**1**	**0**	**1**	**35.65**	**4.5**	**24***	**78.43**
*Denotes a DP that terminated due to exceeding its time limit									

The serial delivery time yielded by DP 40 immediately draws attention since this is the smallest time to deliver all serials for a two MLR problem set returned yet by either S-MIP, PE-MIPv1, or PE-MIPv2. With factors MAXK=5, ADD_HOPS=0, and FASTPATH = 2, DP 40 suggests that PE-MIPv2 can return very good solutions when using a limited number of connector legs and multiple short but fast (in terms of serial transit time) paths to choose from. DP 40 returned a relative optimality gap of 72.73% and the convergence rate of a mixed integer program is not guaranteed. Therefore, there is no clear way of determining whether DP 40 would have reached the specified 20% relative optimality gap with an additional 24 or 48 hours (or more) of model runtime.

15 total DPs ran with 48-hour time limits. Of these DPs, 13 delivered all serials and two DPs returned integer optimal solutions within tolerance. Both DPs that returned integer optimal solutions within tolerance in this batch used an absolute optimality gap for their termination criteria. Table 6 summarizes the DPs from the 48-hour batch that returned integer optimal solutions (within tolerance).

Table 6. Summary of DPs from the 48-hour time limit batch that returned integer optimal solutions within tolerance.

DP	MAXK	FAST-PATHS	HOPS-ADD	USE ABS MIP GAP	Prop. Serials Del.	Time last serial delivered (days)	Obj. function value	Sol. time (hours)	MIP GAP (%)
20	**10**	**1**	**1**	**1**	**1**	**64.60**	**14.82**	**12.49**	**94.85**
21	**15**	**1**	**1**	**1**	**1**	**117.75**	**25.62**	**48.1**	**97.02**

DP 20 raises some questions since it returned a solution in ~12.5 hours but its associated 24-hour time limit DP, DP 8, terminated after exceeding its 24-hour time limit. DP 8 did not return a memory error. This abnormality could be attributed to the fact that DP 8 was executed on the Reaper cluster while DP 40 was executed on Hamming. Hamming and Reaper use different versions of CPLEX, meaning that these machines could have used dramatically divergent solution paths resulting in very different solution times. The other 11 DPs that delivered all serials within the 48-hour time limit but did not return integer optimal solutions all used a relative optimality gap termination criterion. Table 7 summarizes these DPs.

Table 7. Summary of DPs from the 48-hour time limit batch that delivered all serials but did not return integer optimal solutions within tolerance.

DP	MAXK	FAST-PATHS	HOPS-ADD	USE ABS MIP GAP	Prop. Serials Del.	Time last serial delivered (days)	Obj. function value	Sol. Time (hours)	MIP GAP (%)
49	**5**	**1**	**0**	**0**	**1**	**9**	**2.28**	**48***	**60.55**
50	**10**	**1**	**0**	**0**	**1**	**9**	**2.48**	**48***	**69.26**
51	**15**	**1**	**0**	**0**	**1**	**10.5**	**2.93**	**48***	**73.99**
52	**5**	**2**	**0**	**0**	**1**	**9**	**2.35**	**48***	**61.73**
53	**10**	**2**	**0**	**0**	**1**	**9**	**2.59**	**48***	**70.59**
54	**15**	**2**	**0**	**0**	**1**	**19**	**2.83**	**48***	**73.01**
55	**5**	**1**	**1**	**0**	**1**	**9.3**	**2.93**	**48***	**65.33**
56	**10**	**1**	**1**	**0**	**1**	**34.2**	**4.62**	**48***	**83.48**
57	**15**	**1**	**1**	**0**	**1**	**69.65**	**7.5**	**48***	**89.82**
58	**5**	**2**	**1**	**0**	**1**	**28.4**	**4.76**	**48***	**83.19**
59	**10**	**2**	**1**	**0**	**1**	**47.8**	**8.46**	**48***	**90.98**
*Denotes a DP that terminated due to exceeding its time limit									

The results returned by the DPs in Table 7 also demonstrate the phenomenon identified in DP 40 that very small times to deliver all serials are possible given the correct inputs and enough processing time. It is also noteworthy that in DP 58, despite the input MAXK = 5, the factors FASTPATHS = 2 and HOPS_ADD = 1 result in triple the amount of time to deliver all serials as the other DPs with MAXK = 5 (DPs 49, 52, 55). The relationship between the input factor FASTPATH and the time of last serial delivered is explored in detail in Chapter V, Section B.

13 total DPs were executed during the 72-hour time limit batch of experiments. None of these 13 DPs returned integer optimal solutions and four DPs did not deliver all their serials before exceeding the 72-hour time limit. Table 8 summarizes the DPs in the 72-hour batch that delivered all their serials.

Table 8. Summary of DPs from the 72-hour time limit batch that delivered all serials but did not integer optimal solutions within tolerance.

DP	MAXK	FAST-PATHS	HOPS-ADD	USE ABS MIPGAP	Prop. Serials Del.	Time last serial delivered (days)	Obj. function value	Sol. time (hours)	MIP GAP (%)
61	**5**	**1**	**0**	**0**	**1**	**9**	**2.39**	**72***	**62.35**
62	**10**	**2**	**0**	**0**	**1**	**9**	**2.41**	**72***	**68.40**
63	**15**	**2**	**0**	**0**	**1**	**9**	**2.36**	**72***	**67.65**
64	**5**	**2**	**0**	**0**	**1**	**9**	**2.34**	**72***	**61.59**
65	**10**	**1**	**0**	**0**	**1**	**9**	**2.4**	**72***	**68.22**
66	**15**	**1**	**0**	**0**	**1**	**18.1**	**3.13**	**72***	**75.66**
67	**5**	**1**	**1**	**0**	**1**	**11.4**	**2.95**	**72***	**65.46**
68	**10**	**1**	**1**	**0**	**1**	**18**	**3.8**	**72***	**79.91**
69	**15**	**1**	**1**	**0**	**1**	**66.45**	**7.73**	**72***	**90.12**
70	**5**	**2**	**1**	**0**	**1**	**19.5**	**4.58**	**72***	**83.35**
*Denotes a DP that terminated due to exceeding its time limit									

Across all batches 14 total DPs did not deliver all their serials before exceeding their time limits. Table 9 summarizes DPs that did not deliver their serials within their specified time limits.

Table 9. Summary of DPs that did not deliver all serials before exceeding their time limit.

DP	MAXK	FAST-PATHS	HOPS-ADD	USEABS MIPGAP	Time limit (hours)	Prop. serials delivered	Obj. Func Value	MIP GAP (%)
9	**15**	**1**	**1**	**1**	**24***	**0.69**	**199822.28**	**100**
11	**10**	**2**	**1**	**1**	**24***	**0.00**	**650416.46**	**100**
12	**15**	**2**	**1**	**1**	**24***	**0.32**	**443719.41**	**100**
23	**10**	**2**	**1**	**1**	**48***	**0.83**	**107504.14**	**100**
24	**15**	**2**	**1**	**1**	**48***	**0.43**	**372067.68**	**100**
35	**10**	**2**	**1**	**1**	**72***	**0.61**	**254948.71**	**100**
36	**15**	**2**	**1**	**1**	**72***	**0.32**	**443719.41**	**100**
44	**10**	**1**	**1**	**0**	**24***	**0.94**	**41355.91**	**100**
45	**15**	**1**	**1**	**0**	**24***	**0.97**	**17938.77**	**100**
47	**10**	**2**	**1**	**0**	**24***	**0.00**	**650,416.46**	**100**
48	**15**	**2**	**1**	**0**	**24***	**0.32**	**443719.41**	**100**
60	**15**	**2**	**1**	**0**	**48***	**0.43**	**372067.68**	**100**
71	**10**	**2**	**1**	**0**	**72***	**0.84**	**101995.02**	**100**
72	**15**	**2**	**1**	**0**	**72***	**0.43**	**372067.68**	**100**
*Denotes a DP that terminated due to exceeding its time limit								

Table 9 yields some interesting insights about the effect the factors MAXK, FASTPATHS, and HOPS_ADD have on PE-MIP's performance. Not a single DP that failed to deliver all its serials used the input factor MAXK = 5. Furthermore, every DP that did not deliver all serials in its assigned time limit featured the input factor HOPS_ADD = 1. DPs using MAXK = 5, FASTPATH = 1, and HOPS_ADD = 0 represent the most constrained inputs in the full factorial DOE. Therefore, every DP not using these values for these factors are relaxations of the DPs using MAXK = 5, FASTPATH = 1, and HOPS_ADD = 0. Since many of the DPs relaxed in terms of MAXK, FASTPATH, and

HOPS_ADD did not return better results within their assigned time limits, this indicates that relaxation of these factors comes at the expense of additional computational complexity. This trade-off of flexibility and complexity within the space of PE-MIPv2's input factors is explored in detail in Chapter V.

THIS PAGE INTENTIONALLY LEFT BLANK

V. CONCLUSIONS AND FUTURE WORK

This chapter holistically compares the results of experiments conducted during this research project to determine what insights have been found. We draw conclusions from the post-experiment analysis of our work and make determinations about the usefulness and viability of PE-MIPv2 for solving force closure problems. This chapter ends with ideas and suggestions for possible additional research related to this thesis and its motivating problem.

A. TERMINATION CRITERIA

PE-MIPv2 can use either an absolute optimality gap termination criterion or a relative gap termination criterion. Termination criterion selection has profound implications for both the quality of PE-MIPv2's solutions and the computation time required to achieve an integer optimal solution within tolerance. As demonstrated in Figure 13, the absolute optimality gap termination criterion returns integer optimal solutions within tolerance drastically faster than the relative optimality gap termination criterion.

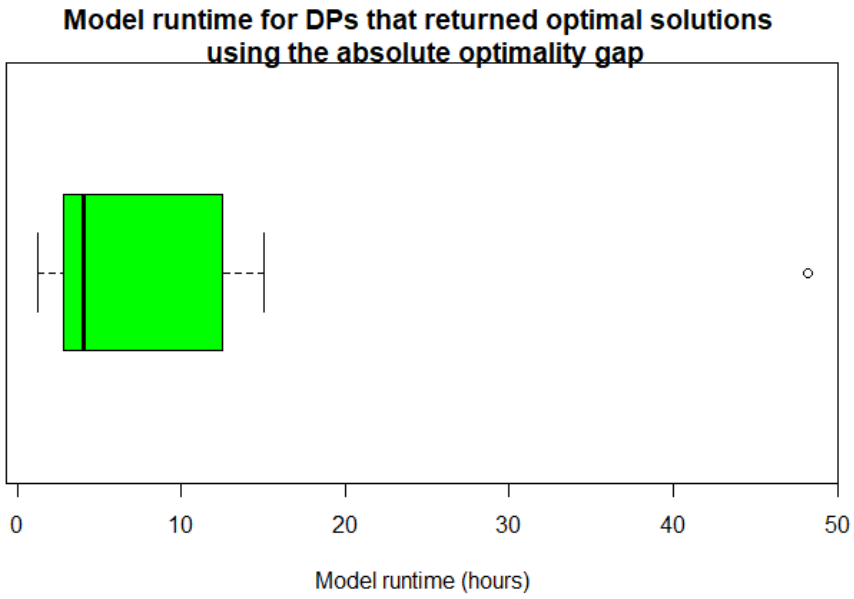

Figure 13. Boxplot depicting model runtimes for integer optimal solutions within tolerance achieved using an absolute optimality gap termination criterion.

Even with a 72-hour time limit, no DP using the relative optimality gap termination criterion returned an integer optimal solution within tolerance. However, at the cost of more computation time, DPs using the relative optimality gap termination criterion returned solutions that were much better regarding the time to deliver all serials than those using the absolute optimality gap. The relationship between solution quality and termination criteria is illustrated in Figure 14.

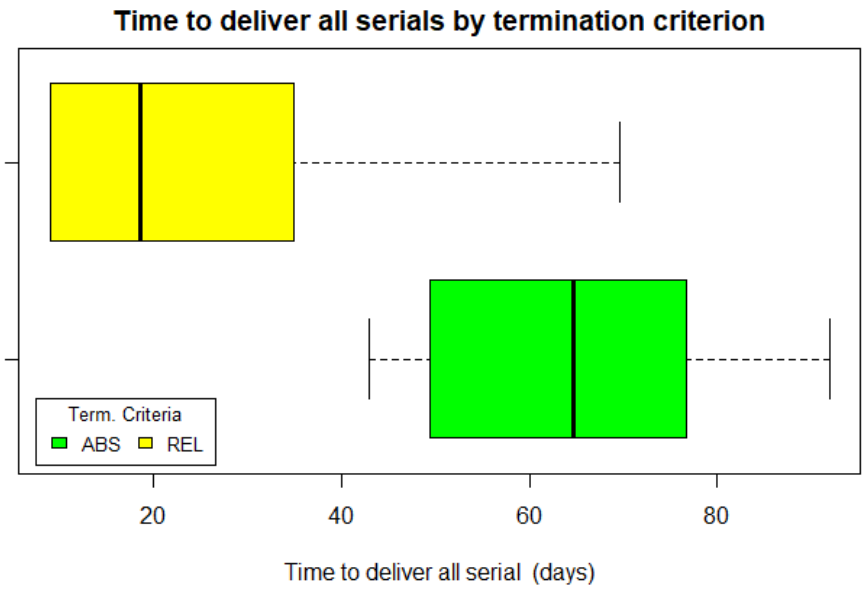

Figure 14. Boxplot depicting time to deliver all serials by termination criteria.

These results suggest different applications for PE-MIPv2 using these two different termination criteria. PE-MIPv2 can inform debate on issues requiring rapid input for time constrained decisions via expedited model runs using the absolute optimality gap termination criterion. Such an application of PE-MIPv2 produces information for a decision maker at the speed of relevance. This information comes at the cost of the quality of the solution relative to a solution obtained using the relative optimality gap termination criterion and additional runtime. On the other hand, if PE-MIPv2 is used to inform deliberate planning in a non-time constrained environment, analysts can afford allowing PE-MIPv2 to run for longer times in pursuit of better solutions.

Another possible use of PE-MIPv2 is for large-scale data farming experiments in support of acquisitions decisions concerning connectors. In this context, analysts would execute many DPs to understand the interactions between the quantities and capabilities required for prospective connectors under consideration for acquisition. The requirement of running a large number of DPs lends itself towards using the absolute optimality gap termination criterion to ensure sufficient DPs are run to fully explore the design space. This application warrants further research both in terms efficient experiment designs as well as

PE-MIPv2's natural computational complexity. As will be discussed in the next section of this chapter, computational difficulty is an aspect of PE-MIPv2 that must be reckoned with when designing future experiments.

B. INPUT FACTORS AND COMPUTATIONAL DIFFICULTY

As indicated in Chapter IV, Section D, the values used for the PE-MIPv2 input factors MAXK, FASTPATHS, and HOPS_ADD can significantly affect PE-MIPv2's outputs. These inputs primarily affect PE-MIPv2's outputs via increases in computational difficulty that inhibit PE-MIPv2 from quickly finding integer optimal solutions. Increases in computational difficulty are most easily quantified by the number of variables which PE-MIPv2 attempts to solve for a given DP. Consequently, a DP with more variables to solve for is less likely to deliver all its serials or return an integer optimal solution within tolerance. This dynamic is well demonstrated by Figure 15 and Table 10.

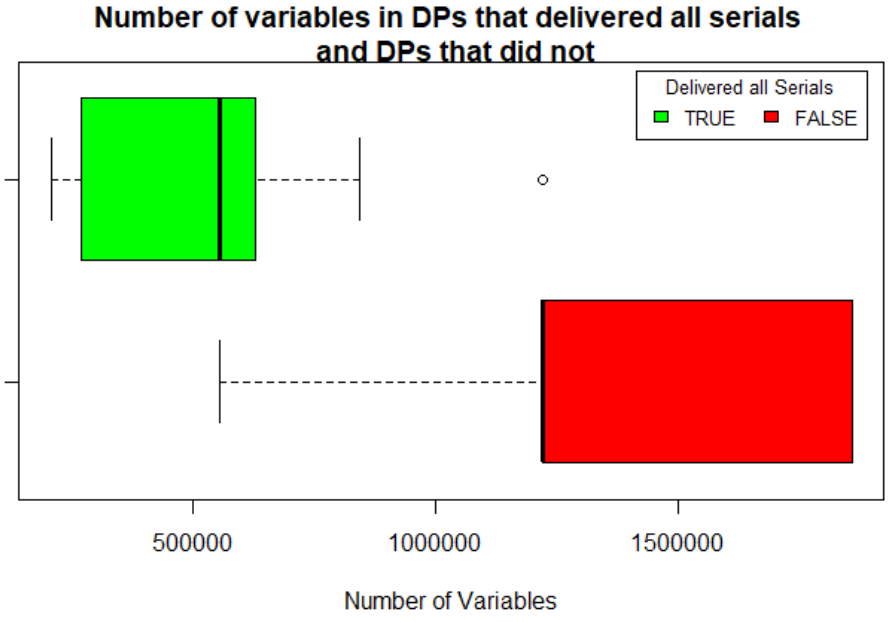

Figure 15. Boxplot depicting the number of variables for DPs that delivered all serials versus DPs that did not.

Table 10. Summary statistics for number of variables in DPs that delivered all serials versus DPs that did not.

All Delivered	Min.	1st. Quart.	Median	Mean	3rd Quart.	Max.
TRUE	208,376	270,299	556,607	492,728	627,306	1,220,271
FALSE	556,607	1,220,271	1,220,271	1,391,550	1,856,083	1,856,083

These summary statistics and visualizations of PE-MIPv2's variable counts clearly demonstrate that DPs that failed to deliver all serials generally have more variables requiring a solution by PE-MIPv2 than those DPs that deliver all serials. The number of constraints in the model also affects PE-MIPv2's computational complexity. The 13 constraints of PE-MIPv2's formulation create sets of rules that the model's decision variables must adhere to remain within the DP's feasible region. Like the number of variables in the DP, larger numbers of constraints in a DP trend towards failure to deliver all serials before exceeding the DP's time limit as shown in Figure 16 and Table 11.

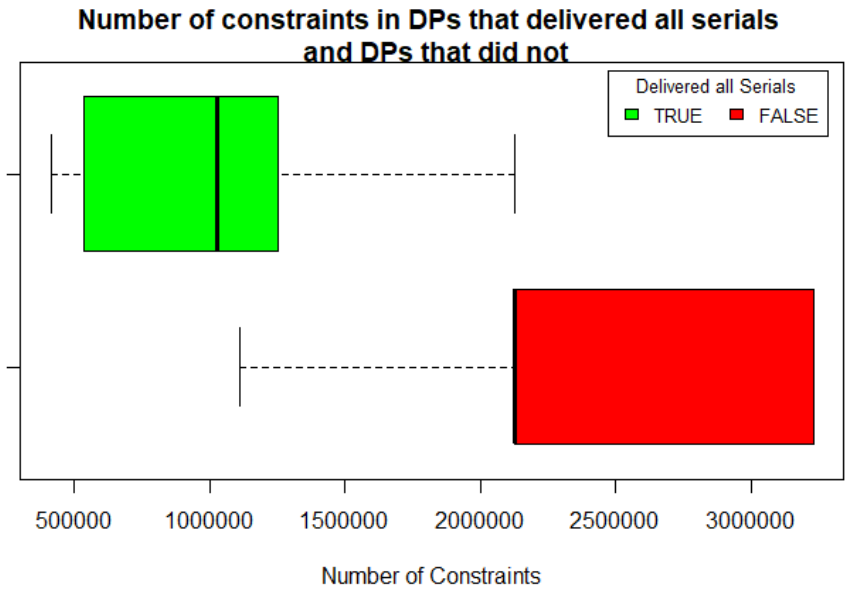

Figure 16. Boxplot depicting the number of variables for DPs that delivered all serials versus DPs that did not.

Table 11. Summary statistics for number of constraints in DPs that delivered all serials versus DPs that did not.

All Delivered	Min.	1st. Quart.	Median	Mean	3rd Quart.	Max.
TRUE	413,500	537,776	1,029,074	960,917	1,252,470	2,128,413
FALSE	1,111,655	2,128,413	2,128,413	2,464,471	3,229,164	3,229,164

Figure 16 and Table 11 further illustrate that less complex DPs suggest a higher likelihood of successful serial delivery within a specified time limit. This trend is also well illustrated by Figure 17, which uses a partition tree to depict the effect of input factors on successful serial delivery.

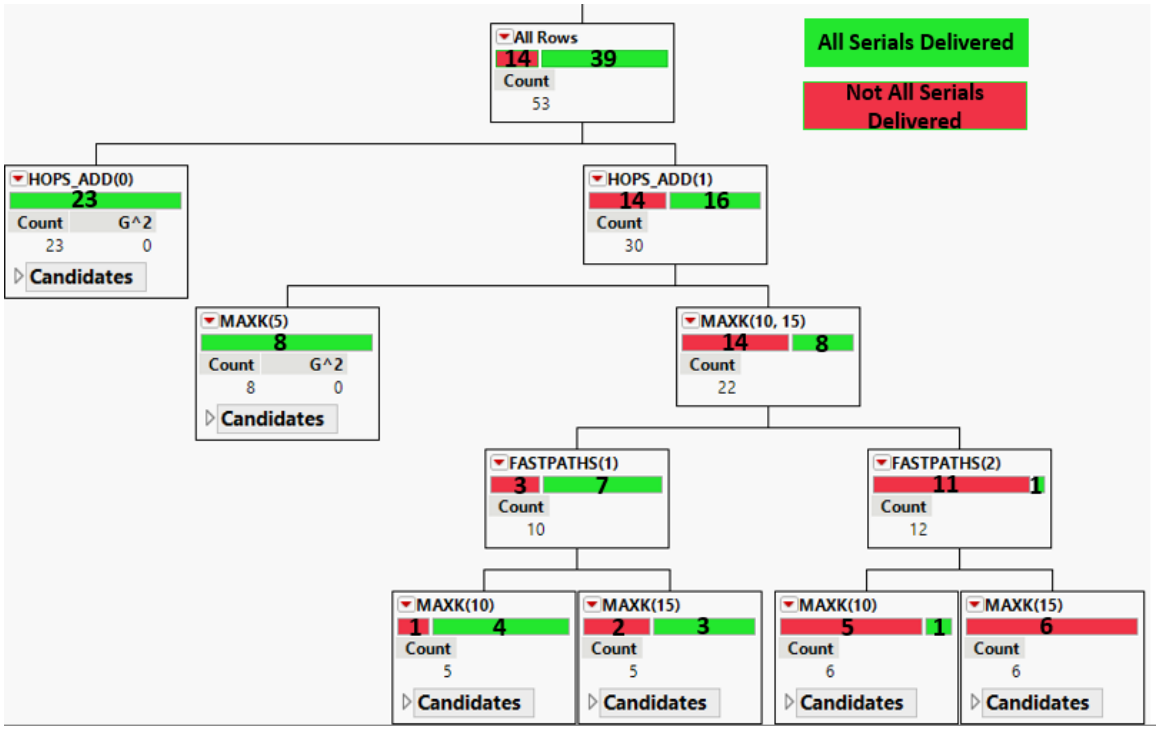

Figure 17. Partition tree depicting the effect of input factor on the successful delivery of all serials for each DP (row).

As shown in Figure 17, simpler inputs for MAXK, FASTPATHS, and HOPS_ADD result in more DPs that successfully deliver all serials. The following subsections describe

how the input factors MAXK, FASTPATHS, and HOPS_ADD influence PE-MIPv2's solutions and computation time.

1. MAXK

MAXK, the number of connector legs allowed per DP has a significant effect on the computational difficulty and performance of PE-MIPv2. The research team began to suspect that higher MAXK values detrimentally affect PE-MIP's solutions and model runtimes during experiments with PE-MIPv1 described in Appendix B: PE-MIP Development. Preliminary experiments with PE-MIPv1 returned poorer solutions with DPs using MAXK = 10 versus those DPs with MAXK = 5. These results motivated some exploratory calculations by the research team to determine how many connector legs are necessary to support the two MLR deployment problem with the given input serials and connectors. Examining the lift capacity of the connectors at the square footage of serials requiring transportation yielded several insights about relationship between MAXK and the two MLR deployment problem. These "back of the envelope calculations" are summarized below.

Total number of serials in the two MLR deployment problem: 472

Total serial size (TSS) in square feet: 173,443.3 square feet

Total connector capacity (TCC): 195,468 square feet

Approximate total loads (TL) required for two MLR deployment (TSS/ TCC): (173,443.3 square feet / 195,468 square feet) ≈ 0.887323 loads

Total serial legs required as a sum of MINHOPS: 1416

Total serial legs (TSL) accounting for IOP node and reducing to arcs: 1416-(2×472) = 472

Approximate total connector legs required as TL*TSL: 0.887323 × 472 ≈ 419

Average number of connector legs required per serial delivery: 419/472 ≈ 0.8877

The calculated approximate number of connector legs required per serial (~0.8877) suggests that using all available connectors both MLRs could deploy all their equipment in a single movement. Such a deployment would by necessity assume optimal connector packing and subdivision of serials (and equipment) to account for the absence of these considerations in the "back of the envelope" calculations. At first glance this result might seem nonsensical, but additional scrutiny of the two MLR deployment problem's inputs reveals why a small number of connector legs is feasible. Both MLRs' source nodes (D and L) are directly connected to their respective StS nodes (J and G respectively). Regarding connectors, the only connector type that cannot access each source/StS node is the CH-53K, which cannot access nodes D or L. The smallest serial is 6.72 square feet, and the 48 smallest serials can reach their destinations via a single connector leg using a MV-22B, which is the least capable connector in terms of useable square footage. Furthermore, 334 total serials (70% of all serials) can reach their destination StS node on a KC-130J using a single connector leg. Therefore, due to the quantity/types of connectors available, the composition of the serials, and the structure of the network the complete deployment of two MLRs is possible using a small value of MAXK.

Higher values of MAXK mean that a connector can traverse more arcs during its employment. This has the direct effect of providing PE-MIPv2 with much more flexibility in the solutions it generates to deliver all serials. The second order effect of this flexibility is that serials could spend much more time embarked on a connector awaiting delivery to its ultimate destination. In other words, a serial embarked on a connector executing a DP with a high MAXK value could visit many more nodes than necessary to reach its destination StS node. This visitation of extraneous nodes by embarked serials occurs because PE-MIPv2 is not constrained by a low MAXK value that forces efficient serial delivery via the limited number of connector legs available. The relationship between MAXK and time of last serial delivery is clearly visualized in Figure 18.

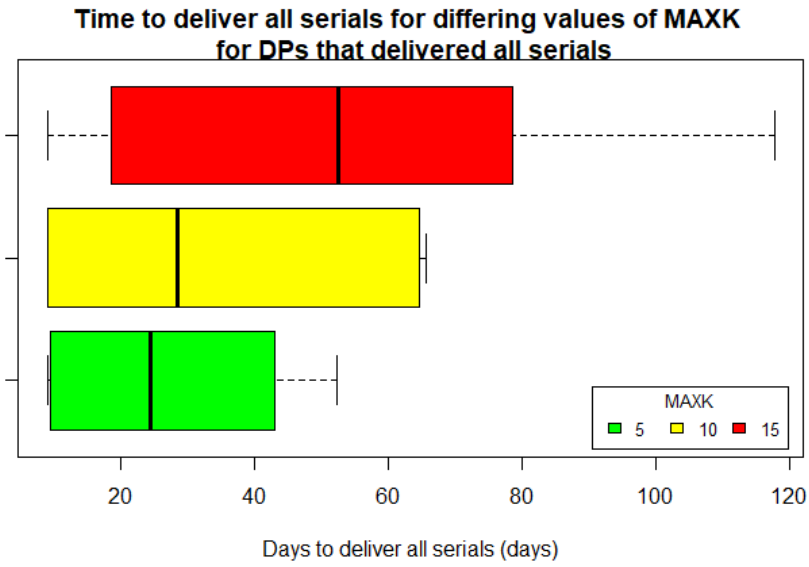

Figure 18. Boxplot depicting time to deliver all serials by MAXK value.

Table 12. Summary statistics for time to deliver all serials by MAXK value.

MAXK	Min.	1st. Quart.	Median	Mean	3rd Quart.	Max.
5	9.00	9.55	24.40	115.18	43.00	1377.45*
10	9.00	9.00	28.40	34.25	64.60	65.70
15	9.00	18.55	52.50	53.09	78.67	117.75
*DP 46 returned an outlier of 1377.45 days						

If given enough time PE-MIPv2, can find solutions approaching integer optimality within tolerance even for DPs with higher MAXK values. However, finding better solutions for higher MAXK values comes at the cost of computational difficulty and model runtime. Simply put, larger MAXK values increase the number of variables that PE-MIPv2 must solve; this phenomenon is illustrated in Figure 19. Increases in variables results in increasing the amount of time required for finding an integer optimal solution within tolerance.

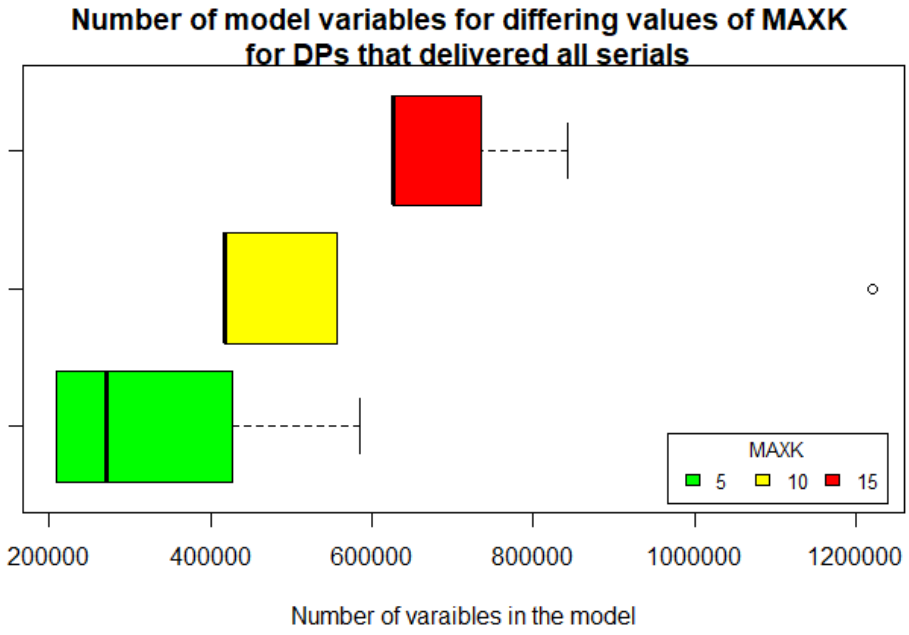

Figure 19. Boxplot depicting number of model variables by MAXK value for DPs that delivered all serials.

Table 13. Summary statistics for quantity of model variables by MAXK value for DPs that delivered all serials.

MAXK	Min.	1st. Quart.	Median	Mean	3rd Quart.	Max.
5	208,376	208,376	270,299	325,365	427,731	585,163
10	417,489	417,489	417,489	522,047	556,607	1,220,271
15	627,306	627,306	627,306	686,301	735,463	843,619

The second order effect of the increased computational difficulty inherent in large values of MAXK is that PE-MIPv2 will terminate its computation before returning a fast and efficient solution for delivering all serials. Termination before returning a fast and efficient solution while using the relative optimality gap criterion occurs because it has exceeded its time limit. For the absolute optimality gap termination criterion, PE-MIPv2 terminates because it has reached a less desirable albeit satisfactory solution for this termination criterion. Furthermore, the increased computational difficulty associated with larger MAXK value results in PE-MIPv2's failure to deliver all serials before exceeding its time limit. As shown in Table 9, located in Chapter IV, Section D, every DP that failed

to deliver all serials before exceeding its time limit had values of MAXK = 10 or MAXK = 15.

2. HOPS_ADD

HOPS_ADD is the input factor used to help define the maximum serial path length allowed for consideration by the model. HOPS_ADD is used to calculate a maximum path length in terms of nodes visited via the formula MAXHOPS = HOPS_COEF * MINHOPS + HOPS_ADD. A full description on this formula is found in Chapter IV, Section C. For the final experiment the research team used values HOPS_COEF = 1 and HOPS_ADD = 0 or 1 for the full factorial DOE. HOPS_ADD = 0 can be interpreted as meaning that for DPs with HOPS_ADD = 0 only serial paths with the shortest possible length in terms of nodes visited are considered by PE-MIPv2. For DPs with HOPS_ADD = 1, serial paths can contain one node extra in addition to the minimum number required to travel from the source node to the destination node.

DPs using HOPS_ADD = 1 exhibited greater computational difficulty and correspondingly less optimal solutions than DPs using HOPS_ADD = 0. As with MAXK, the increased computational difficulty innate in those DPs with HOPS_ADD = 1 versus HOPS_ADD = 0 is easily quantified by comparing the number of variables associated with these HOPS_ADD values as illustrated in Figure 20 and Table 14.

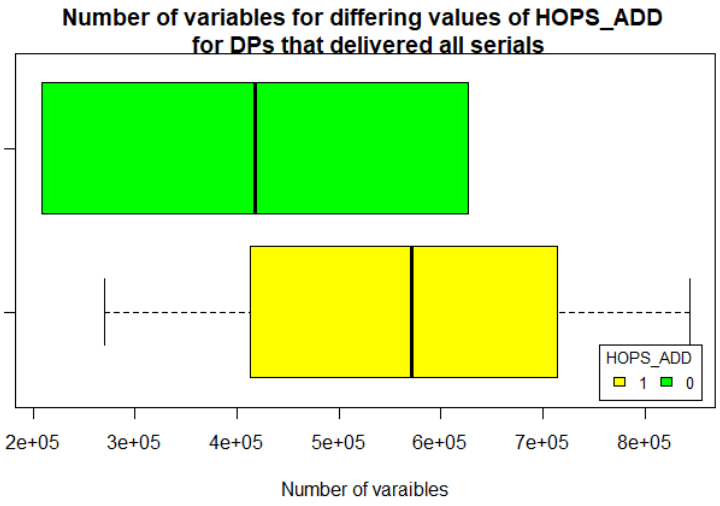

Figure 20. Boxplot depicting number of variables for DPs that delivered all serials for HOPS_ADD = 0 and HOPS_ADD = 1.

Table 14. Summary statistics for number of model variables by HOPS_ADD value.

HOPS_ADD	Min.	1st. Quart.	Median	Mean	3rd Quart.	Max.
0	208,376	208,376	417,489	426,826	627,306	627,306
1	270,299	485,030	570,885	587,463	649,777	1,220,271

The visualization and summary statistics in Figure 20 and Table 14 clearly capture how a PE-MIPv2 model instance becomes more complex using HOPS_ADD = 1. As shown in the Figure 21 and Table 15, this added complexity returns degraded solutions regarding the key model output metric time to deliver all serials.

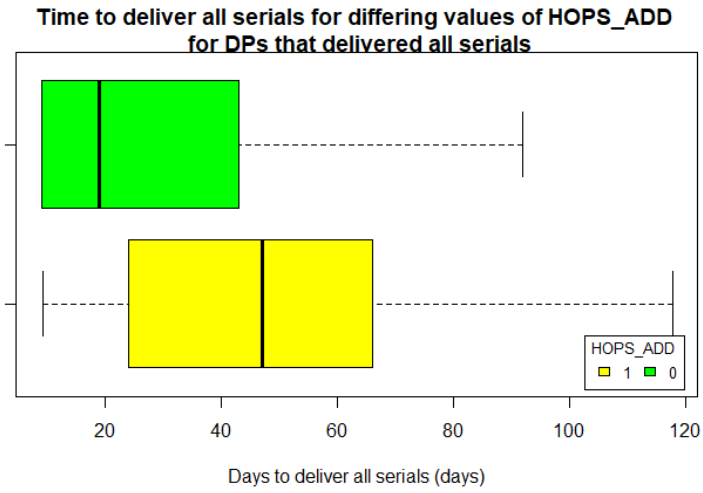

Figure 21. Boxplot depicting time to deliver all serial by HOPS_ADD = 0 or 1 for DPs that delivered all serials.

Table 15. Summary statistics for time to deliver all serials (in days) by HOPS_ADD value.

HOPS_ADD	Min.	1st. Quart.	Median	Mean	3rd Quart.	Max.
0	9.0	9.0	19.0	30.1	43.0	91.9
1	9.3	26.8	47.10	129.04	65.89	1377.45
*DP 46 returned an outlier of 1377.45 days						

The root cause of why PE-MIPv2 returns lower quality solutions for DPs with more computational difficulty is the same as was discussed in the previous section concerning MAXK. DPs using HOPS_ADD = 1 spend more time solving for more variables compared to those using HOPS_ADD = 0 then subsequently terminate upon reaching their time limit. One must also note as shown in Table 7, every DP that failed to deliver all serials used HOPS_ADD = 1. Therefore, the difficulty correlated with HOPS_ADD = 1 also manifests itself in partial solutions like those associated with larger MAXK values.

3. FASTPATHS

The input factor FASTPATHS dictates how many candidate serial paths PE-MIPv2 should consider for a serial after filtering by serial path length has already occurred. The corresponding filter selects serial paths based on each serial's transit time across a path

while embarked on the fastest connector with sufficient capacity capable of traversing each arc of the path. The transit time filter implemented via FASTPATHS ensures that slow serial paths in terms of real transit time are not considered by PE-MIPv2. The full factorial DOE for the final experiment included FASTPATHS = 1 or 2, meaning that only the fastest serial path, or first and second fastest serial paths, were included for consideration by PE-MIPv2. The value used for FASTPATH affects the quality of the solution generated by PE-MIPv2 and the model's computational complexity, though not to the degree as MAXK and HOPS_ADD as demonstrated in the previous subchapters and in Figure 22 and Table 16.

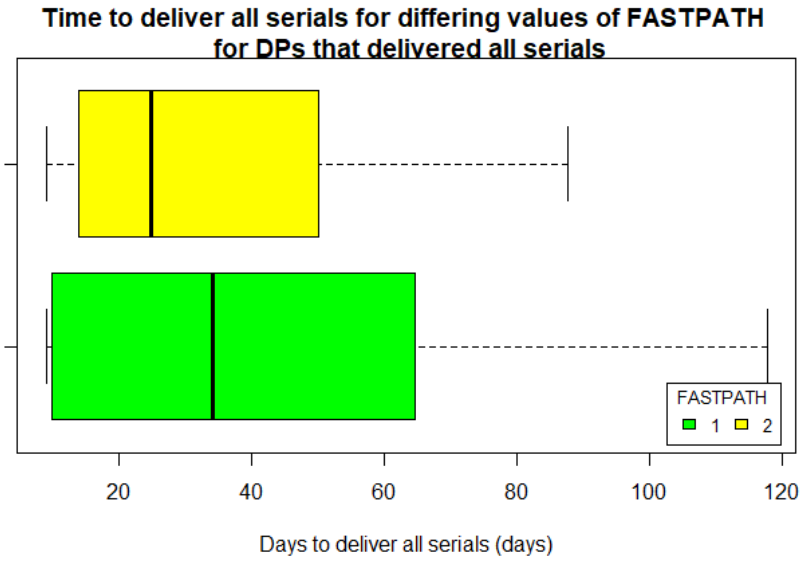

Figure 22. Boxplot depicting time to deliver all serials by FASTPATH value.

Table 16. Summary statistics for time to deliver all serials (in days) by FASTPATH value.

FASTPATH	Min.	1st Quart.	Median	Mean	3rd Quart.	Max.
1	9.00	9.90	34.20	39.11	64.65	177.75
2	9.00	16.02	24.80	116.09	48.94	1377.45*
*DP 46 returned an outlier of 1377.45 days						

Figure 22 and Table 16 show that differing values of FASTPATH can return comparable results in terms of the time to deliver all serials. The real insight concerning FASTPATH is its interaction with the factors MAXK and HOPS_ADD. DPs containing some combination of MAXK = 10 or 15, HOPS_ADD = 1, and FASTPATH = 2 are most likely to fail to deliver all serials before reaching their time limit. 11 of 14 DPs that could not deliver all serials within their assigned time limit used FASTPATH = 2. This result is even more noteworthy because DPs with FASTHPATH = 2 account for every DP using the absolute optimality gap termination criteria that failed to deliver all serials. The full descriptions of all DPs that failed to deliver all serials is found in Table 17.

Table 17. DOE summary for DPs that failed to deliver all serials using FASTPATH = 2.

DP	MAXK	FASTPATHS	HOPS_ADD	USE_ABSMIP GAP	Time limit (hours)	Prop. serials delivered
11	10	2	1	1	24	0.00
12	15	2	1	1	24	0.32
23	10	2	1	1	48	0.83
24	15	2	1	1	48	0.43
35	10	2	1	1	72	0.61
36	15	2	1	1	72	0.32
47	10	2	1	0	24	0.00
48	15	2	1	0	24	0.32
60	15	2	1	0	48	0.43
71	10	2	1	0	72	0.84
72	15	2	1	0	72	0.43

C. OTHER MODEL ATTRIBUTES

1. The Serial Builder Heuristic Algorithm

Joint Publication 3-02 Amphibious Operations defines a serial as "a group of landing force units and their equipment that originate from the same ship and that, for tactical or logistical reason, will land on a specified beach or landing zone at the same time" (Department of Defense [DOD] 2019). The current serial builder heuristic algorithm used by both PE-MIP versions and S-MIP does not employ any tactical or logistical considerations in how it builds serials. This means that equipment is packaged together without any real-world consideration for how it is used. As discussed in Chapter III, certain pieces of equipment must travel with other associated pieces of equipment and qualified operators to facilitate operational employment of that equipment. For example, a trailer needs a truck to pull it on and off a connector and then tow it to where it is needed. The serial builder heuristic algorithm has no means of accounting for this consideration and could build an entire serial of towed equipment without prime movers. Such serials in an operational context could degrade the efficiency of the network by jamming connectors and StS nodes with immovable equipment or worse.

In addition to this operational consideration, the serial builder heuristic algorithm is designed in such a way that it prioritizes making many small serials which can be loaded on any type of connector. This aspect of the heuristic serial builder algorithm results in many more serial paths enumerated by PE-MIP's path enumeration algorithm. These additional serial paths in large scale problem instances can dramatically increase the size and complexity of the optimization problem causing much longer model runtimes.

2. Model Time Considerations

Both versions of PE-MIP are formulated as a continuous-time model. However, serial transit times and key output metrics such as the time to deliver all serials are subdivide and rounded into discrete increments of tenths of days. This aspect of PE-MIP was inherited from the antecedent model S-MIP. This discretization of time and its associated rounding discards or degrades important data in support of overall model simplicity. Rounded and discretized model times also make trouble shooting PE-MIPv2

and debugging its code more difficult than necessary. Future iterations of PE-MIPv2 could be improved using continuous times with an option to convert times into different units for key output metrics such as the time to deliver all serials and other outputs of interest.

D. RECOMMENDATIONS FOR FUTURE WORK

1. Route Circulation Models

PE-MIP and S-MIP model the movement of Marine forces from their source to destination. The type of movement modeled by PE-MIP and S-MIP represents the initial deployment of Marine forces as they first approach the locations from which they will operate during EABO. However, Marine forces will not remain static after their initial deployment to the operating area. For various tactical and logistical reasons Marine forces will continuously move throughout the operational area after reaching their initial operating positions. For example, Marine forces will need to habitually resupply EABs with consumable supplies and rotate personnel such as maintenance teams or evacuate personnel with combat and non-combat related injuries. These movements throughout the operating area will also require optimal use of limited numbers of connectors.

A model that considers optimized route planning for connectors in circulating routes could significantly improve the performance of sustainment networks established for supporting distributed EABs. This model would create routes throughout dispersed EABs like those used by city buses to support municipal public transportation needs. Solutions from such a model could provide valuable insights on sustainment operations in an EABO context. Furthermore, such a model could help force planners determine the appropriate quantity and mixture of different types of connectors needed for sustaining an established network of EABs.

2. Models with Adversarial Interdiction

Adversarial interdiction of logistics platforms is an omnipresent consideration in the operational plans of any professional military. PE-MIP and S-MIP were designed for military logistics applications in an EABO setting, but neither model in its current form has a way of accounting for or considering the effects of enemy actions against the

solutions it generates. Future research should explore the effects of enemy interdiction on the solutions created by PE-MIPv2 then improving those solutions to account for enemy actions. Every connector transit represents an opportunity for the adversary to interdict a connector or discover the location of Marines force by tracking a connector from point to point. PE-MIPv2 could be improved in the short term by minimizing the number of transits made by empty connectors thus reducing unnecessary exposure to enemy interference. More elaborate future models could include some type of stochastic simulation of enemy actions against the connectors and EABs within the model's network using a second arc cost related to risk.

3. Improving the Serial Builder Heuristic Algorithm

As discussed in the previous section, PE-MIPv2's current serial builder heuristic algorithm has several deficiencies. The first deficiency is that the serial builder algorithm builds many small serials to maximize the flexibility of serial assignment to different types of connectors i.e., the algorithm builds small serials that can fit on any sized connector. The second order effect of constructing many small serials is added computational difficulty since with more serials PE-MIPv2 must now filter and consider more candidate serial paths. The second deficiency of PE-MIPv2's serial building algorithm is that it creates nonsensical serials with equipment mixes that at best have no operational benefit of travelling together. PE-MIPv2's serial building algorithm could be improved by turning it into a decision support tool that helps users construct efficient serials based on relational rules for equipment pairing. Such a decision support tool could provide users with different options for serial construction to maximize efficient use of connector capacity or for maximizing the operational effectiveness of serials once they are delivered to their initial operating position. Serial construction is driven by operational employment considerations; therefore, it is neither realistic nor desirable to entirely automate serial construction without human input for insight on how the serials are operationally employed. After human review and adjustment input serials constructed via a serial construction decision support tool can be uploaded via the InputSerial tab as inputs for PE-MIPv2 or a descendant model.

APPENDIX A. S-MIP MODEL FORMULATION

The model formulation contained in Appendix A represents Lieutenant Colonel Freeman's S-MIP as it existed in August 2021 after revisions by Dr. Craparo. Lieutenant Colonel Freeman developed his initial formulation of S-MIP in 2019 while serving as an analyst assigned to OAD. Dr. Craparo and associates modified S-MIP while conducting research at NPS in support of an OAD-sponsored project.

A. SETS AND INDICES

$v \in CXRS$	connectors ("cxrs")
$s \in EQUIP$	serials
$i, j \in NODES$	nodes/locations
$(v, i, j) \in TRIPS$	arcs/transits
$k \in LEGS$	legs executed by the connectors (first leg, second leg, etc.)
$(v, k) \in CLEGS$	connector v executes at least k legs
$(v, k, i, j) \in S\_TRIPS$	cxr transits
$(s, v, k, i, j) \in LOADS$	cxr-serial transits

B. DATA [UNITS]

h_v	usable deck area for cxr v [sq.ft]
v_s	value/priority weight for serial s
a_s	deck footprint size of serial s [sq.ft]
b_s	initial/starting locn (node) for serial s
e_s	terminal/ending locn (node) for serial s
$t_{v,i,j}$	time req'd for cxr v to make transit (i, j) [days]
LAYOVER	time cost for cxrs to enter/leave nodes [days]
BIG	"infinite" = |Legs| * (max(t [v,i,j]) + LAYOVER) * 10
STEP	=1,2,...,MAXLOOPS; the current successive time window

C. DECISION VARIABLES [UNITS]

$W_{v,k}$	time>=0 at which cxr v completes leg k [days]
$X_{v,k,i,j}$	=1 if cxr v makes transit (i, j) on leg k; else 0

$Y_{s,v,k,i,j}$	=1 if serial *s* loaded for transit (*v*,*k*,*i*,*j*); else 0
$Z_{s,i}$	time>=0 at which serial *s* arrives at node *i* [days]
P_s	=1 if serial s not delivered to destination *e* [*s*]; else 0
U_s	shortest-path distance>=0 remaining for serial *s* to *e* [*s*] [miles]

D. FORMULATION

$$\min_{W,X,Y,Z,P,U} \quad z = \sum_{s \in EQUIP} \left(STEP v_s U_s + v_s Z_{s,e_s} + v_s P_S BIG \right)$$

s.t.

$$\sum_{(v,k,j):(s,v,k,i,j) \in LOADS} Y_{s,v,k,i,j} \le 1 \qquad \forall (s,i) \in PUTS \qquad (C1)$$

$$t_{v,i,j} X_{v,k,i,j} + W_{v,k-1} + LAYOVER \le W_{v,k} \qquad \forall (v,k,i,j) \in S\_TRIPS : k > 1 \qquad (C2)$$

$$\sum_{(i,j):(v,k,i,j) \in S\_TRIPS} X_{v,k,i,j} \le 1 \qquad \forall (v,k) \in CLEGS \qquad (C3)$$

$$\sum_{s:(s,v,k,i,j) \in LOADS} a_s Y_{s,v,k,i,j} \le h_v X_{v,k,i,j} \qquad \forall (v,k,i,j) \in S\_TRIPS \qquad (C4)$$

$$\sum_{j':(v,j',i) \in TRIPS} X_{v,k-1,j',i} \ge X_{v,k,i,j} \qquad \forall (v,k,i,j) \in S\_TRIPS : k > 1 \qquad (C5)$$

$$Y_{s,v,k,i,j} \le \sum_{(v',k',j'):(s,v',k',j',i) \in LOADS} Y_{s,v',k',j',i} \qquad \forall (s,v,k,i,j) \in LOADS : i \ne b_s \qquad (C6)$$

$$W_{v,k} + \left(Y_{s,v,k,i,j} - 1 \right) BIG + LAYOVER \le Z_{s,j} \qquad \forall (s,v,k,i,j) \in LOADS \qquad (C7)$$

$$P_s \ge 1 - \sum_{(v,k,i,j):(s,v,k,i,j) \in LOADS,\, j=e_s,\, (v,k,i,j) \in S\_TRIPS} Y_{s,v,k,i,j} \qquad \forall s \in EQUIP \qquad (C8)$$

$$spd_{b_s} - \sum_{(v,k,i,j) \in S\_TRIPS} Y_{s,v,k,i,j} \left(spd_i - spd_j \right) \le U_s \qquad \forall s \in EQUIP \qquad (C9)$$

$$Z_{s,i} + t_{v,i,j} + \left(Y_{s,v,k,i,j} - 1 \right) BIG + LAYOVER \le W_{v,k} \qquad \forall (s,v,k,i,j) \in LOADS \qquad (C10)$$

$$W_{v,k}, Z_{s,i}, U_s \ge 0 \qquad \forall v,k,s,i \qquad (C11)$$

$$X_{v,k,i,j}, Y_{s,v,k,i,j}, P_s \in \{0,1\} \qquad \forall v,k,s,i,j \qquad (C12)$$

Constraint (C1) states that a serial *s* may only be loaded on one connector *v* at a time.

Constraint (C2) states that the time *W*[*v*,*k*] at which vessel *v* completes leg *k*,*i*,*j* is not less than the time at which leg *k*-1 is completed, plus transit time *T*[*v*,*i*,*j*] + LAYOVER.

Constraint (C3) states that vessel *v* may only make one transit (*i*,*j*) per leg *k*.

Constraint (C4) states that the total deck space of serials being delivered by vessel *v* on leg *k* from *i* to *j* cannot exceed *H*[*v*].

Constraint (C5) states that vessel v can only make transit (i,j) on leg k if it also made a transit to i on leg k-1.

Constraint (C6) states that serial s can only be delivered by vessel v on leg-transit (k,i,j) if that serial arrived at i on a different leg-transit (same or other vessel).
Constraint (C7) states that the arrival time of serial s at j $Z[s,j]$ is not less than the vessel v leg-transit (k,i,j) completion time $W[v,k]$ + LAYOVER, or zero for all trips (v,k,i,j) on which serial s does not arrive.

Constraint (C8) records that serial s that did not reach its destination e_s, if applicable.

Constraint (C9) states that the shortest-path distance (spd) $U[s]$ from serial s final location to its intended destination $E[s]$ is not less than the spd from its source $B[s]$ minus the sum of the differences (spd [i] - spd [j]) for all load transits (i,j) conducted.

Constraint (C10) states that the k leg-completion time of vessel v $W[v,k]$ is not less than the transit time t [v,i,j] plus the arrival time $Z[s,i]$ for any cargo loaded from i to j, or zero for cargo not loaded.

Constraints (C11) and (C12) define decision variable domains.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. PE-MIP DEVELOPMENT

A. INITIAL EXPERIMENTS WITH PE-MIPv1 AND S-MIP

After building a script for implementing the PE-MIPv1 reformulation of S-MIP the research team began testing PE-MIPv1 with small problem instances to evaluate and debug PE-MIP's code. These early PE-MIPv1 model runs indicated that PE-MIP performed about as well the current version of S-MIP. Both models returned similar objective values after comparable runtimes on the research team's personal computers. These initial tests motivated the research team to move on to larger scale problem instances on NPS's Hamming high-performance computing cluster. The research team decided that initial experiments with PE-MIPv1 should directly compare its performance with S-MIP to gain insights on any immediate performance differences between the two models. Therefore, initial experiments featured simultaneous runs of PE-MIPv1 and S-MIP with the same input data.

The first experiment directly comparing PE-MIPv1 and S-MIP was conducted from 29 Nov 21 – 07 Dec 21. This experiment used an input file supplied by the SEED Center. The inputs for this experiment represent the deployment of a single MLR from two source nodes to an initial operating position serviced by a single StS node. This experiment used the model's heuristic serial builder and data provided by the WhatWhere worksheet for building the MLR's serials rather than predefined user input serials. For simplicity, a single DP was used during this experiment. Two expeditionary fast transports (TEPF) and four light amphibious warships (LAW) were available as surface connectors with no air connectors available in this DP. The Sea State was set at Sea State 4, and the desired (mixed integer program) MIP tolerance was 0.2.

Both PE-MIPv1 and S-MIP produced integer feasible solutions but failed to find optimal solutions within the specified MIP tolerance in the 192-hour time limit placed on the experiment. PE-MIPv1 returned an objective function value of 56613.6 with approximately 96% of serials delivered in 45.8 days and a MIP gap of 97.3. The S-MIP experiment produced similar results with an objective function value of 46925.3 with

approximately 93% of serials delivered in 47.8 days and a MIP gap of 96.75. The results of these experiments are summarized in Table 18.

Table 18. Summary of key experiment output metrics comparing S-MIP and PE-MIPv1 performance for a single MLR deployment problem.

Model	Serial input mode	Proportion of serials delivered	Last serial delivered time (days)	Objective function value	Relative MIP gap
PE-MIPv1	Heuristic builder: WhatWhere	~0.96	45.8	5139.6	97.3%
S-MIP	Heuristic builder: WhatWhere	~0.93	47.8	46925.3	96.75%

The results of this initial experiment motivated the design of a subsequent experiment. During this second experiment four total model runs would help the research team determine the effects of using the heuristic serial builder algorithm versus user-defined input serials. For the second experiment both S-MIP and PE-MIPv1 would run using one input file featuring user-defined input serials and another with an input file requiring the model's serial building heuristic to build serials, i.e., with equipment input via a WhatWhere worksheet. The second experiment would also simulate deploying a single MLR through the same network as the first experiment with all DP factors carried over from previous experiment

The research team executed the second experiment from 15 Dec 21 – 04 Jan 22 using NPS's Hamming high performance computing cluster. All model runs returned integer feasible solutions before terminating upon reaching their 192-hour time limit. Table 3 summarizes the results of this experiment.

Table 19. Summary of key experiment output metrics comparing S-MIP and PE-MIPv1 performance for a single MLR deployment problem using both user-defined input serials and serials built using the model's heuristic algorithm.

Model	Serial input mode	Proportion of serials delivered	Last serial delivered time (days)	Objective function value	Relative MIP gap
PE-MIPv1	Heuristic builder: WhatWhere	~0.98	16.2	5139.6	92.7%
PE-MIPv1	User input: InputSerials	1	14.5	90.1	49.83%
S-MIP	Heuristic builder: WhatWhere	1	14.5	4722.1	92.08%
S-MIP	User input: InputSerials	1	14.5	90.1	74.95%

The first insight gleaned from this experiment was the effect of using user-defined input serials versus serials built by PE-MIPv1 and S-MIP's serial builder heuristic algorithm. As shown in Table 19 in both instances where the serial builder heuristic algorithm was used the models returned much larger objective function and MIP gap values. PE-MIPv1 run using the serial builder heuristic algorithm also failed to deliver all serials and returned a higher serial delivery time. These outputs were the research team's first indication that the serial builder heuristic algorithm could be impeding the performance of both PE-MIPv1 and S-MIP.

Besides the insight concerning the serial builder heuristic algorithm, these first two experiments demonstrated that in large problems PE-MIPv1 did not outperform S-MIP in the key metrics of objective function value, serial delivery time, MIP gap, or model runtime. These results motivated the research team to modify the path enumeration code to further limit the number of candidate serial paths constructed for PE-MIPv1. The research team determined that implementing a modified version of BFS within the path enumeration code was a viable method of limiting the number of nodes visited in candidate serial paths.

The modified BFS algorithm implemented within the path enumeration returns the shortest path length from the source to StS node in terms of the number of nodes visited along the path. In the real world this means that the number "hops" that a connector would need to make between StS sites in the network is limited to only the minimum number of hops ("MINHOPS") required to reach the destination StS plus a user-defined tolerance; this value is stored as a variable called "MAXHOPS." The number of candidate serial paths generated for the model is then filtered to only those paths with the capacity for the serial to traverse the path and a path length less than or equal to "MAXHOPS."

B. EXPERIMENTS USING ADDITIONAL SERIAL PATH FILTERS

1. Experiments Using Breadth-First Search Path Length Filter

With the BFS filter in place, the research team engaged in an experiment testing PE-MIP's efficiency and effectiveness on the deployment of a single MLR using different inputs for PE-MIP. For one of the inputs the maximum path lengths in hops was MAXHOPS = MINHOPS + 1 and in the other version MAXHOPS = MINHOPS. This experiment used the same input file for both inputs for all other factors with six DPs. The targeted relative MIP gap for all experiment DPs was 20% and the only factor of the DPs that varied across the DPs was the model run time which increased from eight hours to 240 hours for . Table 20 summarizes the input for all 24 DPs used in this experiment.

Table 20. Summary of factor inputs for initial experiment using path length filters.

DP	MAXK	HOPS_ADD	Time Limit (hours)	Serial Input Method
1	6	1	8	User input serials
2	6	1	24	User input serials
3	6	1	48	User input serials
4	6	1	72	User input serials
5	6	1	96	User input serials

DP	MAXK	HOPS_ADD	Time Limit (hours)	Serial Input Method
6	6	1	240	User input serials
7	6	1	8	Heuristic serial builder algorithm
8	6	1	24	Heuristic serial builder algorithm
9	6	1	48	Heuristic serial builder algorithm
10	6	1	72	Heuristic serial builder algorithm
11	6	1	96	Heuristic serial builder algorithm
12	6	1	240	Heuristic serial builder algorithm
13	6	0	8	User input serials
14	6	0	24	User input serials
15	6	0	48	User input serials
16	6	0	72	User input serials
17	6	0	96	User input serials
18	6	0	240	User input serials
19	6	0	8	Heuristic serial builder algorithm
20	6	0	24	Heuristic serial builder algorithm
21	6	0	48	Heuristic serial builder algorithm
22	6	0	72	Heuristic serial builder algorithm
23	6	0	96	Heuristic serial builder algorithm
24	6	0	120	Heuristic serial builder algorithm

Table 21 summarizes the results of this experiment and its key output metrics for PE-MIPv1 using MAXHOPS = MINHOPS + 1 and user input serials.

Table 21. Summary of key experiment output metrics for PE-MIPv1 using MAXHOPS = MINHOPS + 1 with user input serials

DP	Prop. of serials delivered	Last serial delivered time (days)	Objective function value	Relative MIP gap	Model runtime (hours)
1	1	14.5	90.1	84.98%	8
2	1	14.5	90.1	75.36%	24
3	1	14.5	90.1	79.94%	48
4	1	14.5	90.1	74.25%	72
5	1	14.5	90.1	69.74%	96
6	1	14.5	90.1	50.97%	240

As shown by Table 21, although each DP achieved an integer feasible solution, not a single DP in this trial produced a solution within the specified MIP gap tolerance. When the serial builder heuristic algorithm was used to build the serials PE-MIPv1 performed worse in terms of relative MIP gap and objective function value. However, it did return slightly better time of last serial delivery as depicted by Table 22 and Table 23.

Table 22. Summary of key experiment output metrics for a PE-MIP using MAXHOPS = MINHOPS + 1 with serials built by the serial builder heuristic algorithm.

DP	Prop. of serials delivered	Last serial delivered time (days)	Objective function value	Relative MIP gap	Model runtime (hours)
7	1	13.9	5944.2	93.69%	8
8	~0.98	13.9	5932.3	93.68%	24
9	1	13.9	5924.9	93.67%	48
10	~0.85	13.9	5894.6	93.64%	72
11	~0.85	Error	5928.1	93.67%	96
12	Memory error caused CPLEX to abort DP 12				

Table 23. Summary of key experiment output metrics for a PE-MIPv1 using MAXHOPS = MINHOPS with serials built by the serial builder heuristic algorithm

DP	Prop. of serials delivered	Last serial delivered time (days)	Objective function value	Relative MIP gap	Model runtime (hours)
13	1	14.5	4629.9	91.9%	8
14	1	14.5	4629.9	91.9%	24
15	1	14.5	4629.9	91.9%	48
16	1	14.5	4596.8	91.84%	72
17	1	14.5	4629.9	91.9%	96
18	1	14.5	4596.8	91.84%	120

Table 24 displays a significant improvement in model runtime over the results achieved in any of the other trials. All design points for this trial found a solution within the desired optimality gap in less than 35 minutes. This result stands in stark contrast to other trials which could only produce integer feasible solutions in 192 hours. The variance is model runtimes seen in Table 23 occurs because none of the trials in this experiment had exclusive use of its respective computing cluster. Therefore, if other tasks were running in tandem with the model, this could cause variations in the model's total runtime as seen by runtimes ranging from 11 to 34 minutes. SEED Center personnel did not attribute any of the trials reaching their time limit without an optimal solution to this phenomenon.

Table 24. Summary of key experiment output metrics for PE-MIPv1 using MAXHOPS = MINHOPS with user-defined input serials.

DP	Prop. of serials delivered	Last serial delivered time (days)	Objective function value	Relative MIP gap	Model runtime (minutes)
19	1	14.5	90.1	19.99%	~27
20	1	14.5	90.1	19.99%	~34
21	1	14.5	90.1	19.99%	~34
22	1	14.5	90.1	19.99%	~28
23	1	14.5	90.1	19.99%	~11
24	1	14.5	90.1	19.99%	~12

Analyzing the number of candidate serial paths available for consideration by the model before and after applying BFS path length and path capacity filter explains the dramatic differences in model runtime. Table 24 illustrates how the number of serial paths considered by the model dramatically decreases when the set of candidate paths is filtered for capacity and path length. A larger set of candidate serial paths results in longer model runtimes because the model assesses the suitability of each serial path for inclusion in the optimal solution. Therefore, using a more restrictive filter for serial path length (i.e. MAXHOPS = MINHOPS) which further limits the number of possible serial paths for consideration significantly improves model runtime. The quality of the model's solution as defined by the objective function value and the time to deliver serials was not adversely affected by filtering using the more restrictive serial path length filter MAXHOPS = MINHOPS. Table 25 depicts the relationship between the path length filter and the number of paths enumerated before and after filtering.

Table 25. Serial paths generated by path enumeration code before and after BFS path length and path capacity filters are applied.

Serial input method	MAXHOPS value	Unique unfiltered serial paths	Serial paths accepted after filtering
User defined	MAXHOPS= MINHOPS +1	3,913	393
Heuristic algorithm	MAXHOPS= MINHOPS +1	20,242	577
User defined	MAXHOPS= MINHOPS	3,913	360
Heuristic algorithm	MAXHOPS = MINHOPS	20,242	406

As demonstrated by these results captured in Tables 21–24 all DPs returned larger objective function values and MIP gaps when using the serial builder heuristic algorithm instead of user-defined serials. Table 25 shows that these results are attributable to the fact that using the serial builder heuristic algorithm results in larger sets of serial paths before and after filtering. This understanding led to the determination by the research team to

forgo future trials using the serial builder heuristic algorithm with the intent of improving PE-MIP's runtime and instead use only user-defined input serials loaded via the InputSerials worksheet. This insight also inspired a spirited discussion within the research team about the operational usefulness and practicality of the heuristic serial builder which is covered in Chapter V.

2. Excursions in Support of the Research Sponsor and Subsequent Experimentation

In early February 2022 OAD contacted SEED Center personnel requesting to revisit a notional two MLR deployment problem that had been previously tested with S-MIP. This problem featured the deployment of MLR 12 and MLR 4 from their home stations to a single initial operating position via two StS nodes and a network containing 15 nodes total. The serials for this deployment problem were user-defined and the type and number of available connectors approximated the resources expected to be available in theater for this type of operation. With 72.25 hours of runtime S-MIP achieved an objective value of 1,627.9 with all serials delivered in 54.3 days. The research sponsor was interested in what type of results PE-MIP could return with a very brief runtime limited to eight hours. Using the same inputs as the previous S-MIP excursion PE-MIPv1 returned an objective function value of 45,550.05 with a proportion of ~0.92 serial delivered in 63.1 days.

The research team decided to embark on a larger experiment after the excursion into the two MLR deployment problem with abbreviated model runtime. Since PE-MIP returned promising results for a single MLR deployment problem, it was reasoned that with additional time PE-MIPv1 could achieve similar results. This experiment would include eight total DPs each containing the same inputs regarding the number of connectors available, sea state, and MIP tolerance. The DPs varied the number of legs (MAXK) allowed per connector and the model runtime. All DPs were executed using a model in which paths were filtered for length using MAXHOPS = MINHOPS. Table 25 summarizes the relevant factor varied per DP.

Table 26. Experiment factors varied by DP for the initial experiment with a two MLR deployment problem.

DP	MAXK (connector legs)	Time limit (hours)
1	5	24
2	10	24
3	5	48
4	10	48
5	5	72
6	10	72
7	5	96
8	10	96

The research team redundantly executed these experimental trials on both NPS's Hamming high performance computing cluster and NPS's Reaper high performance computing cluster. Performing this experiment redundantly on two different systems was a fortuitous decision because of unexpected difficulties encountered during the experiment. The trials run on the Hamming high performance computing cluster experienced several CPLEX errors causing DPs 3, 5, 6 to terminate while running. DPs 7 and 8 did not run because the experiment reached its overall time limit. The results from DPs 1, 2 and 4 are summarized in Table 27. None of these DPs achieved an optimal solution within the allotted time limit.

Table 27. Summary of key experiment output metrics for the trials successfully conducted on Hamming during initial two MLR deployment experimentation.

DP	Proportion of Serials delivered	Last serial delivered time (days)	Objective function value	Relative MIP gap
1	~0.506	~48.64	1519.5	84.47%
2	0	N/A	N/A	99.95%
4	~0.3	~67.35	2190.3	89.23%

The trials run on the Reaper high performance computing cluster did not experience any errors and returned a full set of results for each design point. Table 27 summarizes the

key output metrics from the trials of the experiment run om the Reaper high performance computing cluster. As with the trails run on the Hamming high performance computing cluster none of the DPs in Table 28 achieved an optimal solution.

Table 28. Summary of key experiment output metrics for the trials conducted on Reaper during two MLR deployment experimentation.

DP	Proportion of serials delivered	Last serial delivered time (days)	Objective function value	Relative MIP GAP
1	~0.33	54.1	1471.45	83.96%
2	~0.92	63.1	45550.05	99.48%
3	1	52.85	1594.15	85.2%
4	~0.92	63.1	45550.05	99.48%
5	1	45.4	1385.5	82.97%
6	~0.99	86.2	3748.2	93.7%
7	1	40.7	1417.8	83.35%
8	~0.23	86.2	1920.2	87.71%

This experiment yielded two very important insights concerning the performance of PE-MIP. The first insight was that as the size of the problem increases in terms of number of serials then the number of serial paths enumerated increases dramatically. All DPs run on the Hamming and Reaper high performance computing clusters enumerated 1,815,545 possible serial paths and then filtered this set of serial paths down to 2,316 candidate serial paths via BFS and capacity filters. The 2,316 filtered serial paths returned during this experiment represent a nearly sixfold increase in the number of serial paths considered by PE-MIP compared to previous experiments. Therefore, the underwhelming output metrics from this experiment can be at least partially attributed to the increased scope and complexity of this problem relative to previous experiments.

The second important insight produced by this experiment was that the number of legs allotted to each connector significantly affects the complexity of the model and its performance. As shown in Tables 27 and 28 the odd-numbered DPs where MAXK = 5 performed better than the even-numbered DPs where MAXK = 10. This indicates that MAXK = 10 could be a too permissive value of this factor which results in more complexity for little return in key performance metrics.

3. Experiments Using Transit Time Filters and Reformulation to an Absolute Optimality Gap

In the interest of further decreasing the model's computational difficulty the research team decided to explore additional filters on candidate serial paths. The research team determined that an additional filter considering the transit time of each serial across different paths could improve the model's performance. This filter calculates the transit times of each serial across a candidate path then selects a quantity of candidate paths determined by the user input variable FASTPATH. This method did not result in any appreciable improvements in the model's runtime or the quality of the model's solutions. These results motivated a reformulation of PE-MIPv1 to feature both an absolute optimality gap rather and a relative optimality gap. This reformulation, known as PE-MIPv2, is discussed in detail in Chapter III. PE-MIPv2 immediately produced promising results during initial experiment comparing PE-MIPv2 with PE-MIPv1. As shown in Table 29, a cursory experiment with PE-MIPv2 using the absolute optimality gap termination criterion returned similar results in terms of proportion of serials delivered and last serial delivery time for the two MLR deployment problem. PE-MIPv2 returned these outputs in a fraction of the time required by PE-MIPv1 using a relative optimality gap.

Table 29. Summary of key experiment output metrics for an exploratory trial conducted on Hamming evaluating the utility of a reformulation using an absolute optimality gap.

Proportion of serials delivered	Last serial delivered time (days)	Objective function value	Solution time (hours)
1	43	11.32	1.96

Based on the results of this experiment using an absolute optimality gap the research team decided to design a final set of experiments evaluating both the relative and absolute optimality gap termination criterion as inputs for of PE-MIPv2. This final experiment is described in Chapter IV, Section D.

APPENDIX C. CONNECTOR FARMER USER'S MANUAL (V0.17) (UPTON 2021)

The Connector Farmer User's Manual (V0.17) is used with permission from Mr. Stephen Upton of the SEED Center. Connector Farmer User's Manual (V0.17) supersedes all previous versions of the Connector Farmer User's Manual due to incremental improvements made in this data farming technique. This manual is distributed amongst NPS students, staff, and research associates experimenting with S-MIP and PE-MIP. The text of Connector Farmer User's Manual (V0.17) contained in Appendix C is reproduced verbatim with minor formatting changes from a digital copy of Connector Farmer User's Manual (V0.17).

A. CHANGE NOTES

The main changes that NPS made to the S-MIP model and code during the FY21 work effort are:

- added Initial Operational Positions (IOP) and alternate Point of Debarkations (APOD). This included:
  - adding a column to the WhatWhere sheet, as well as new syntax to indicate alternate PODS
  - adding a HowFast sheet, to define how long a serial will take to get from POD to IOP
- model changes (in the ConnectorModelSEED_EMC.py file) made by Prof Craparo/Steve Upton to improve the rolling horizon implementation and add VirtualConnectors to facilitate the IOP/APOD changes.
- added an InputSerials sheet to allow the user to define their own serials (loads), thereby bypassing the internal heuristic that maps sticks (rows on the WhatWhere sheet) to serials.

See the appendix contained within this manual for more details.

B. INTRODUCTION

ConnectorFarmer is a data farming wrapper around OAD's ConnectorModel, written in python, as a collection of python scripts. It uses the multicores on a single machine to run multiple instances of the ConnectorModel concurrently. ConnectorMiner is the post-processor that extracts and computes various summary metrics from the output of a ConnectorFarmer run, and also creates an Excel output spreadsheet with intermediate

results for more detailed analysis. It also can use multiple cores on a single machine to post-process individual run output concurrently.

1. Overview

ConnectorFarmer is composed of ConnectorRunner and ConnectorModelSEED_EMC (ConnectorModelSEED_EMC is a modification to ConnectorModelSEED, which is a modification of the original ConnectorModel -see Appendix for more details). ConnectorFarmer takes as input an Excel spreadsheet formatted similar to what is used by the original ConnectorModel, where we added the capability to read in new experiment design factors from the ExperimentDesign sheet of the Excel input, and two new sheets to facilitate the modeling of Initial Operational Positions (IOP) and alternate Point of Debarkations (APOD).

The added new factors, that can now be used on the ExperimentDesign sheet, are:

- the Broken Stow Factor and the Sustained Speed in the Sea State parameters of a connector;
- a set of Experiment-level parameters - VARLIM, MAXLOOPS, SERIALSIZE, OVERLAP, and LAYOVER;
- and a set of CPLEX parameters (described in the 'Preparing to run a designed experiment' section).

The two new sheets added are:

- a HowFast sheet, to define how long a serial will take to travel from an APOD to its IOP;
- an (optional) InputSerials sheet, to allow the user to define their own set of serials (loads), thereby bypassing the internal heuristic that maps sticks to serials.

In addition, we added an IOP column to the WhatWhere sheet, and modified the syntax allowed in the "StS Site" to support the modeling of APODs. To indicate alternate PODs for a stick/serial, the user lists the alternate PODs, separated by a ":." For example, if a stick/serial can debark at either POD C or POD D, then the user would indicate this by putting the string "C:D" in the 'StS Site" column entry (NB: internal/external spaces are stripped in that string, so you can add spaces for readability; this means, however, that there can not be any spaces in the PODs name -this mainly affects the definition of the PODs in the HowFar sheet).

2. How ConnectorFarmer works

When started, ConnectorFarmer:

- creates a pool of workers (using python's multiprocessing library),
- reads in the experimental design from the ExperimentDesign sheet of the Excel input file,
- creates a run directory for each design point (DP),
- creates a DP-specific Excel spreadsheet in the run directory, and
- calls ConnectorRunner to run the ConnectorModelSEED for an individual DP instance, for each DP.

After ConnectorFarmer completes, each of the run directories will contain:

- a DP*.cplex.log file, containing the CPLEX log output for the DP,
- a DP*.log file, containing log entries from running ConnectorRunner and ConnectorModelSEED_EMC,
- a DP*.pkl file, containing the pickled Experiment object,
- a farmer-DP*.log file, containing information on what factors were changed and their settings, and
- a DP*-input.xlsx file, a revised Excel input file specific for this DP.

ConnectorMiner post-processes the output from a ConnectorFarmer run. It relies on data from the pickled Experiment object output and the log files to compute a set of summary metrics and creates an Excel workbook with several sheets containing more detailed output. After running ConnectorMiner, each of the run directories will have two added files:

- a DP*-output.csv file, which holds the set of summary metrics,
- a DP*-raw-output.xlsx file, which holds several sheets containing more detailed output.

After completing post-processing of each individual DP, ConnectorMiner then concatenates the data from all the DP*-output.csv files into a single file, merges that with the experimental design data, and then places that file into the study directory. By convention, this is named "<nameofyourstudydirectory>-alloutput.csv".

C. PREREQUISITES

Prior to runing ConnectorFarmer and ConnectorMiner, the following software packages must be installed first. We use the Anaconda distribution (version 4.8.3), which includes the numpy/scipy, and pandas packages.

1. python (we use, and tested with, version 3.7.6)
    a. pyomo (we use, and tested with version 5.7)
    b. pandas (we use, and tested with version 1.1.0)
    c. openpyxl (we use, and tested with version 3.0.3;note this comes with the Anaconda distribution but not with the generic installers downloaded from python.org)
    d. multiprocessing (part of the python standard library)
2. CPLEX (we use, and tested with version 12.10). If CPLEX is on your PATH, then ConnectorFarmer will use that version; if you want to use a different version, then you can add a line to the cplex.config file to indicate the path to your CPLEX executable. The format is: `cplex_path:<the/path/to/your/cplex>`.

If you use the Anaconda distribution, version 4.8.3, you will need to install version 1.1.0 of pandas as we use several features that are not present in earlier versions of pandas (version 4.8.3 of Anaconda comes with pandas 1.0.1)

To do so, at the command prompt, type:

```
pip install -Iv pandas==1.1.0
```

D. CONNECTORFARMER INSTALLATION

To install ConnectorFarmer, just unzip the ConnectorFarmer.zip file to a suitable location on your machine, such as "c:\ConnectorFarmer." Then create a 'studies' folder where you put individual ConnectorModel Excel input files for your separate experiments. This 'studies' folder can be anywhere on your machine (preferably in a path that contains no spaces). More detail will follow in the Running ConnectorFarmer section.

After unzipping, you should see these files in the directory:

- all-metrics.txt
- ConnectorFarmer.py
- ConnectorFarmerFuns.py
- ConnectorMiner.py
- ConnectorMinerFuns.py
- ConnectorModelSEED_EMC.py
- ConnectorRunner.py
- test_inputs.xlsx
- cplex.config

The Connector* files are the python scripts that are needed to run ConnectorFarmer and ConnectorMiner. The all-metrics.txt file is used by ConnectorMiner and is a line-separated list of functions to compute metrics and generate the intermediate output results

in the Excel output workbooks. The test_inputs.xlsx file is an Excel input file that can be used for test running of ConnectorFarmer and ConnectorMiner. The cplex.config file holds the set of CPLEX parameters that can be modified.

E. PREPARING TO RUN A DESIGNED EXPERIMENT

Prior to running a designed experiment with ConnectorFarmer, using the underlying ConnectorModelSEED_EMC Mixed Integer Program (MIP), you must have an input file. This input file is in the form of an Excel workbook with five (5) required sheets: HowFar, HowFast, ConnectorData, ExperimentDesign, and either a WhatWhere sheet or an InputSerials sheet. If the user wants to define their own set of serials, thereby bypassing the internal heuristic that maps sticks to serials, they put that data into the InputSerials sheet, and place that sheet as the first sheet in the workbook. If instead they want ConnectorFarmer to assign sticks to serials, then they use the WhatWhere sheet to define their sticks, making sure the InputSerials sheet, if it exists, is not the first sheet in the workbook. Both sheets can be in the workbook simultaneously; this allows the user to easily switch inputs, if desired. However, the InputSerials sheet MUST be the first sheet in the workbook if the user desires to use that set of input; the WhatWhere sheet can be anywhere in the workbook. Also, other sheets can be added to the workbook without affecting the model or the run, e.g., supporting data, etc..

There are a number of requirements that the Excel workbook, and the corresponding sheets, must satisfy in order to start a ConnectorFarmer/ConnectorModel run. ConnectorFarmer does pre-validation of the Excel workbook to assist the user, as well as imposes additional requirements of its own to handle the additional factors permitted in the ExperimentDesign sheet.

The requirements are:

1. Each of the five (5) sheets MUST be named as indicated, i.e., spelled exactly as above, including matching case.
2. The InputSerials sheet (if used) MUST be the first sheet in the workbook; if it is placed anywhere else, its input will NOT be used. If used, the InputSerials sheet MUST have columns (named exactly as is, with matching spaces and case; order of the columns does not matter): ‘StS Site’,’IOP’,’Sqft’,and ‘Source’. You can have an optional “Value” column to place weight/value on each of the serials. All

other columns are ignored, so the user can place other columns with additional information useful to the user.

3. The WhatWhere sheet (if used) MUST have columns (named exactly as is, with matching spaces and case; order of the columns does not matter): 'Site','StS Site','IOP','Length','Width',and 'Source'. You can have an optional "Value" column to place weight/value on each of the sticks. All other columns are ignored, so the user can place other columns with additional information useful to the user, e.g., TAMCN. "Length" and "Width" columns expect units in inches. The columns are then multiplied and divided by 144 to arrive at a square footage for the stick.
4. For the HowFar sheet, the first column MUST contain the node names (the name of the column is not used, and can therefore be anything - "Loc" is the default). There then MUST be a column for each node name, using the node name as the column name, and the node name MUST NOT contain spaces. For example, if you have a node "A" in the "Loc" (first) column, then you MUST have a corresponding "A" column. Additionally, there MUST be an "Access" column for each Connector used in the experiment. The column MUST be named "Access - CnxrX," where CnxrX is the name of the connector. Note the space before and after the hyphen.
5. For the HowFast column, you MUST have columns named (named exactly as is, with matching spaces and case; order of the columns does not matter): 'From','To','Time(hrs)'.
6. For every IOP:APOD connection in the WhatWhere/InputSerials (depending on which is used) sheets, there MUST be a complete entry (row) in the HowFast sheet. For example, if you have an IOP of "P1" and an APODs entry of "C:D:E," then you must have "C/P1/<t>," "D/P1/<t>," and "E/P1/<t>" rows in the HowFast sheet, where <t> is some time, specified in hours.
7. For the ConnectorData sheet, you MUST have columns named: "Type," "Square Footage," "Broken Stow Factor," "Usable Square Footage," "Sust. Speed - Sea State 1," "Sust. Speed - Sea State 2","Sust. Speed - Sea State 3," and "Sust. Speed - Sea State 4".
8. For the ExperimentDesign sheet, you must have columns named: "Design Point" and "SeaState." In addition, for each Connector in the experiment, you MUST have a column named "CnxrX Qty," where CnxrX is the name of the connector (no space allowed, but hyphens are permitted).
9. If "CnxrX Qty" is in ExperimentDesign, then there MUST be a CnxrX row in ConnectorData
10. If "CnxrX Qty" is in ExperimentDesign, then there MUST be an Access column for CnxrX in HowFar, i.e., "Access- CnxrX"
11. If there are other CnxrX factors (Broken Stow, Sust. Spd) in ExperimentDesign, then there MUST be a "CnxrX Qty" in ExperimentDesign

The above requirements are the minimal that MUST be satisfied in order to start a ConnectorFarmer/ConnectorModel run. Except for the last requirement, they are implictly documented (by the code) in the original ConnectorModel.py file, and continued in the ConnectorModelSEED_EMC.py file. ConnectorFarmer does some pre-validation and checks for most of the above requirements at present, and reports failure if any of the pre-validation checks fail.

In addition to the Connector parameters that are available as part of a design, the user can also add CPLEX factors and Experiment-level factors to the design. These are added to the ExperimentDesign sheet. See the accompanying test_inputs.xlsx Excel workbook for examples.

Here is a list of the CPLEX factors, with their default values, that can be used in a designed experiment:

- 'threads' : 1
- 'workmem' : 16000
- 'mip_strategy_file' : 3
- 'timelimit' : 28800
- 'dettimelimit' : 1e+70
- 'mip_tolerances_mipgap' : 0.1
- 'mip_strategy_variableselect' : 0
- 'emphasis_memory' : yes
- 'mip_strategy_nodeselect' : 1
- 'emphasis_mip' : 1
- 'mip_strategy_branch' : 1
- 'parallel' : 0

To use in a designed experiment, create a column with the name of the factor, spelled exactly as above (without the single quotes), and place their values in that column. If a CPLEX factor is not specified in the ExperimentDesign sheet, ConnectorFarmer will look for the factor value in a cplex.config file placed in the study directory (see the example cplex.config that comes with the ConnectorFarmer distribution). If the factor value is not found there, then it will use the default as indicated above.

You can also include five Experiment-level factors in the design. They are, with their default values:

- ‘VARLIM’ : 10000
- ‘LAYOVER’ : 0.5
- ‘MAXLOOPS’ : 10
- ‘OVERLAP’ : 1
- ‘SERIALSIZE’ : None

Note that SERIALSIZE of “None” means that you are not supplying an explicit max serial size, therefore the code’s SerialSizeHeuristic function will be used to determine SERIALSIZE for that run, based on the connectors available.

As with the CPLEX factors, to use in a designed experiment, create a column with the name of the factor, spelled exactly as above (without the single quotes), and place their values in that column.

For both the CPLEX and Experiment-level factors, if you misspell the name of the factor, then that factor will not change in the experiment, i.e., it will be ignored. ConnectorFarmer documents what factors were used in the individual DP*.log files in the corresponding run directory. Here is an example from the first few lines of a DP*.log file:

```
09/23/2021 10:33:47:   using CPLEX options: {‘threads’: 1, ‘workmem’: 16000.0, ‘workdir’: Wi
ndowsPath(‘C:/ConnectorFarmer/studies/test1/run_14’), ‘logfile’: WindowsPath(‘C:/ConnectorF
armer/studies/test1/run_14/DP14.cplex.log’), ‘mip_strategy_file’: 3, ‘timelimit’: 1800.0, ‘mip_tol
erances_mipgap’: 0.1, ‘mip_strategy_variableselect’: 0, ‘emphasis_memory’: ‘yes’, ‘mip_strateg
y_nodeselect’: 1, ‘emphasis_mip’: 1, ‘mip_strategy_branch’: 1, ‘dettimelimit’: 1e+70, ‘parallel’:
0}
09/23/2021 10:33:47:   processing Experiment top_level factors from ExperimentDesign: C:\Con
nectorFarmer\studies\test1\run_14\DP14-input.xlsx
09/23/2021 10:33:47:   found these top levels factors in the ExperimentDesign: {‘OVERLAP’, ‘V
ARLIM’, ‘MAXLOOPS’}
09/23/2021 10:33:47:   using top level parameters: {‘VARLIM’: 10000, ‘LAYOVER’: 0.5, ‘MA
XLOOPS’: 1, ‘OVERLAP’: 1, ‘SERIALSIZE’: None}
```

Once you have an Excel input file properly constructed, you are ready to run ConnectorFarmer!

F. RUNNING

As mentioned previously, the ConnectorFarmer software consists of two components: ConnectorFarmer itself, and ConnectorMiner. Both are run individually, i.e., ConnectorFarmer doesn’t depend on ConnectorMiner, and vice versa. However, ConnectorMiner does rely on certain conventions used in ConnectorFarmer, e.g., the

naming of the run directories, and the pickled output files rely on code in the ConnectorFarmerSEED_EMC file to establish structure of the pickled Experiment objects. Both ConnectorFarmer and ConnectorMiner are designed to be run from the command line.

1. Running ConnectorFarmer

ConnectorFarmer is run from the command line and takes two required arguments:

- the absolute or relative path to the Excel spreadsheet input file
- the number of processors to use for this run

If the user only supplies the two required arguments, ConnectorFarmer will run all design points in the ExperimentDesign sheet of the Excel input file. If the user only wants to run specific design points, the user can supply a space-separated list of the design points they wish to run after the first two required arguments. The design points can be names (no spaces or hyphens) in addition to numbers; e.g., "scen14".

First, to run ConnectorFarmer, you need to change directories to where you installed ConnectorFarmer, using your preferred shell, e.g., DOS, powershell, or the anaconda python prompt. In the examples below, we assume you are using the anaconda python prompt and that you installed ConnectorFarmer to 'C:\ConnectorFarmer' - adjust your path according to where you installed it.

Here is the general command line invocation of ConnectorFarmer (from the directory where the ConnectorFarmer.py file resides):

```
prompt>python ConnectorFarmer.py <pathtoExcelInputfile> <numprocessors> [DPs*]
```

where:

- <pathtoExcelInputfile> is the absolute or relative path to the Excel spreadsheet input file (we assume no spaces in input file name),
- <numprocessors> is the number of processors to use for this run,
- [DPs*] is an optional space-separated list of design points (DPs) (as listed in the Design Point column of the ExperimentDesign sheet).

We recommend creating separate "study" folders, one for each designed experiment. ConnectorFarmer uses the path of the Excel input file as the study directory, and will overwrite any existing output.

(1) Test Running ConnectorFarmer

To test, we will use the test_inputs.xlsx file that accompanies this distribution as a sample input file.

- create a 'C:\connectorFarmer\studies\test1' directory,
- copy test_inputs.xlsx to that directory (no spaces in file name),
- (optional) copy the cplex.config file from the ConnectorFarmer directory to your study directory and make any desired changes,
- change directories to C:\ConnectorFarmer (or whereever you placed the set of Connector* files),
- start ConnectorFarmer:

```
(base) PS c:\ConnectorFarmer> python ConnectorFarmer.py c:\ConnectorFarmer\studies\test1\test_inputs.xlsx 1
```

Once started, in the console/command prompt window, you should start to see scrolling lines of text. The logger in ConnectorFarmer dumps its output to the console in addition to a "farmer.log" file in the study directory. There are also individual loggers for each DP and they each dump their output to their individual run directories (by convention, these run directories are created as "run_X" where X is the DP name in the study directory).

After ConnectorFarmer completes successfully, you should have two "run_X" directories in c:\ConnectorFarmer\studies\test1\, where X runs is 14 and 15 (these are read in from the Design Point column of the Excel input file). In each of those folders, you should have five (5) files as described above, i.e., three DP* log files, a DP*.pkl file, and an Excel input file specific for that DP.

2. Running ConnectorMiner

As mentioned above, ConnectorMiner is run from the command line. ConnectorMiner takes three required arguments:

- the absolute or relative path to the Excel spreadsheet input file (no spaces in file name),
- the absolute or relative path to the metrics file,
- the number of processors to use for this run

If the user only supplies the three required arguments, ConnectorMiner will post-process all design points in the ExperimentDesign sheet of the Excel input file. If the user

only wants to post-process specific design points, the user can supply a space-separated list of the design points they wish to post-process after the first two required arguments.

First, to run ConnectorMiner, you need to change directories to where you installed ConnectorFarmer, using your preferred shell, e.g., DOS, powershell, or the anaconda python prompt. In the examples below, we assume you are using the anaconda python prompt and that you installed ConnectorFarmer to 'C:\ConnectorFarmer' - adjust your path accordingly.

Here is the general command line invocation of ConnectorMiner (from the directory where the ConnectorFarmer.py and ConnectorMiner.py files reside):

```
prompt>python ConnectorMiner.py <pathtoExcelInputfile> <pathtoMetricsfile> <numprocessors > [DPs*]
```

where:

- <pathtoExcelInputfile> is the absolute or relative path to the Excel spreadsheet input file (no spaces in file name),
- <pathtoMetricsfile> is the absolute or relative path to the metrics file,
- <numprocessors> is the number of processors to use for this run,
- [DPs*] is an optional space-separated list of design points (DPs).
-

For the <pathtoMetricsfile>, you can edit the provided "all-metrics.txt" file and put in your study directory, or leave as is. It currently lists all the available metrics and output sheets that can be generated by ConnectorMiner. Metrics that are outputted to the DP*-output.csv file all start with the word "compute." Metrics that create sheets in the DP*-raw-output.xlsx Excel workbook all start with the word "sheet." The code for both the 'compute_' and the 'sheet_' functions can be found in the ConnectorMinerFuns.py file. Lines that do not start with either "compute" or "sheet" are ignored. You can intersperse comments in the metrics file, if desired (conventionally, these start with "#"). Also, the order of the "compute" metrics in the metrics file defines the order they appear in the DP*-output.csv file (duplicate entries are ignored; the first metric encountered defines the order of appearance).

(1) Test Running ConnectorMiner

To test running ConnectorMiner, you will need some output. We will assume you tested running ConnectorFarmer as above and have a 'C:\connectorFarmer\studies\test1' study directory with all the "run_X" sub-directories, and the 'test_inputs.xlsx' is in that study directory.

Start ConnectorMiner with the following command in the command window, from your ConnectorFarmer install directory:

```
(base) PS c:\ConnectorFarmer> python ConnectorMiner.py  c:\ConnectorFarmer\studies\test1\test_inputs.xlsx all-metrics.txt 1
```

Once started, in the console/command prompt window, you should start to see scrolling lines of text. The logger in ConnectorMiner dumps its output to the console. If you use more than one processor, the output lines of text will intermix with output from processing another DP.

After ConnectorMiner completes successfully, in those two "run_X" directories in c:\ConnectorFarmer\studies\test1\, you should now have two additional output files as described above, i.e., a csv output file, and an Excel output file. In the test1 folder you should also have a "test1-alloutput.csv" file with the concatenated results from the individual DP*-output.csv files.

That concludes this User's Manual If there are any questions, or if you run into issues running either ConnectorFarmer or ConnectorMiner, please send an email to Stephen Upton at scupton@nps.edu.

3. Appendix

This section contains information on changes made to the model by the SEED Center and Prof. Emily Craparo at NPS. They are listed in reverse chronological order, i.e., the most recent changes are listed first.

(1) Additional changes made to the S-MIP model (7/16/2021)

Prof Emily Craparo, of the NPS OR department, made additional recommendations/modifications to the S-MIP model. These changes are described below, and are copied from the document "Connector Model – further ideas for IOP

modifications.docx" by Prof Emily Craparo (undated). Option 1 changes listed below were implemented by Prof Craparo and Stephen Upton.

The current S-MIP formulation aims to minimize the time required for a set of connectors to move each serial from its Port of Embarkation (POE) to the Port of Debarkation (POD). In the real world, each serial would then need to move a certain distance from its POD to its Initial Operational Position (IOP). This move is generally accomplished by the serial itself (e.g., a truck on a road network). Given the quality of the road network and distance between the POD and the IOP, it might take the serial several days to complete this move. OAD would like the S-MIP to consider the time required to complete this ground leg, rather than only the time required to complete the at-sea legs. Additionally, OAD would like to specify a set of candidate PODs for each serial rather than only a single POD. For example:

- Assume it will take a particular serial 11 hours to move from the Aerial POE (APOE) to the Aerial POD (APOD) and 48 hours to move from the APOD to IOP, for a total movement time of 59 hours.
- Assume it will take a particular serial 36 hours to move from the Sea POE (SPOE) to the Sea POD (SPOD) and 10 hours to move from the SPOD to the IOP, for a total movement time of 46 hours.

Currently, the model considers only a single POD e_s for each serial. It seems straightforward to let e_s represent a set of possible PODs for each serial rather than only a single POD, and I had initially envisioned a potential model change using this approach. However, it is less straightforward to model the total time to transit to the IOP and the tradeoffs between using different PODs. I therefore propose we initially try modifying the input data to account for the POD-IOP transit (Option 1 below), rather than modifying the model (more details available upon request).

Option 1: change the data, not the model.

Augment the network with all relevant IOP nodes; let e_s represent the IOP for each serial. At each POD, create a set of "virtual connectors" that are able to travel between the POD and the relevant IOPs for the serials that may use that POD. In order to avoid artificially restricting the problem at this point, create one "virtual connector" for each serial that could conceivably use that POD. Let $t_{v,i,j}$ reflect the time required for the serials

to make the final transit from the POD to the IOP. (Is it the same for each type of serial? If not, we'll have to be careful how we model the eligibility of serials to use connectors.) Run the model as usual; the last segment from POD to IOP will now be completed just like any other.

- Benefits of Option 1: No model changes required, only data changes (unless different serials have different transit times between the same POD and IOP; we may need a small model change to address that). Model accurately captures (and minimizes) the total time to reach the IOP.
- Drawback of Option 1: Bigger instances; more computation time required.
- Data modifications required for Option 1:
  - Include IOPs in set NODES.
  - Create "virtual connectors" at POD nodes and include them in set CXRS.
  - Use set TRIPS to account for the fact that virtual connectors can only transit between PODs and IOPs.
  - For set CLEGS, I believe we can let each "virtual connector" only use a single leg, to reduce model size. (Recall that we will create a "virtual connector" for each serial that may use that POD.)
  - S_TRIPS and LOADS should be modified in a manner consistent with the above (each serial can only transit between its designated PODs and IOP; "virtual" connectors have similar restrictions).

I believe all of these modifications can be accomplished using the following pieces of basic information:

- For each serial, the possible PODs that serial may use.
- For each serial, the desired final IOP.
- For each (POD, IOP) pair, the transit time required for a serial to go from POD to IOP.

Option 2: see the document, "Connector Model – further ideas for IOP modifications.docx," for further details.

(2) Changes made to ConnectorModelSEED.py, incorporated into ConnectorModelSEED_EMC.py (3/21/2021)

Prof Emily Craparo, of the NPS OR department, made several modifications to the ConnectorModelSEED file. These changes are described below.

Notes from Prof Emily Craparo (3/21/2021) Here are the main highlights:

- New version significantly streamlines the sets s_Trips and s_Loads. This results in more legs being generated for a given VARLIM, which in turn should increase solution quality, although computation time may increase along with it (I expect it would). For a given VARLIM, you should see fewer rolling horizon iterations with the new model than with the old, with each iteration taking longer. You may need to play with the VARLIM values to find the "sweet spot." You may also be able to get away with a higher mipgap with the new model, since it is taking a longer view and gaining some solution quality that way. Again, something to experiment with.
- New version chooses the largest possible OVERLAP each iteration as discussed in our emails with Matt and Chris. It does this in an iterative fashion each rolling horizon iteration: first it nominates an OVERLAP value, then builds out the appropriate sets (we need to know OVERLAP to do this, so we know which positions the ships reached in the prior rolling horizon loop), counts the new variables, and adjusts OVERLAP down if necessary, again building out the sets and counting variables. The old code actually set the OVERLAP value for the next iteration while building out sets, which I guess it could usually get away with because it was not building out s_Trips and s_Loads very efficiently (and anyway, there are no real consequences to OVERLAP being "wrong"). It makes much more sense to do these things jointly.
- New version handles initial positions of serials correctly. The old version had a bug where if a serial was delivered during the OVERLAP window, it was marked as delivered and not handled in the next rolling horizon iteration (or ever again). It also calculated the initial positions of the serials in a manner similar to how it calculated the OVERLAP window in the prior bullet point; again, you need to know OVERLAP in order to know what "initial" even means.

- New version reverts the DistanceLeft constraint to the expression shown in the formulation document. The original version was using the initial positions of the serials, which were buggy, and was expressed as an inequality. It was giving some truly wacky behavior that stumped me for a long time. I think(???) someone probably noticed that the model was infeasible when that constraint was expressed as in the formulation document, and they "fixed" the problem by modifying it as we saw in the original version of the code…but the real issue was with the G[s] values and the Y variable values prior to the OVERLAP window (there was a little bug with those too); the constraint was just the canary in the coal mine.

(3) Changes made to original ConnectorModel (8/4/2020)

We made several modifications to the original ConnectorModel code. These are contained in the ConnectorModelSEED file, many of them to handle logging and running a specific design point, or DP. Six changes were made to the actual model code, however:

1. the addition of reading in Stick Values to be used when determining the value of a composed serial (if used, the column must be named ‘Stick Value’; if not present, the Stick Value is set to 1, to mimic the original model);
2. sorting by Site, in descending order, (the first column on the WhatWhere sheet of the Excel spreadsheet input), after sorting by Source and StS site when composing serials in the “MakeSerials” function;
3. adding a SerialGroup column to the Serials data structure which indicates what serial an individual stick belongs to;
4. fixed bug on line 673 of the original code, replacing second M.Equip with M.Ships;
5. fixed bug on line 1352 of the original code, which had misplaced parentheses; and
6. changed LAYOVER to LAYOVER/2 in two constraints.

None of the preceding changes affect the optimization formulation:

1. adds the capability to set the stick value or ‘weight’ in the input (in the original file, the stick values are all set to 1, but the value/weight is part of the original model formulation);
2. allows the user more control over how serials are constructed, and is performed before model formulation;
3. is a bookkeeping addition;
4. fixed double counting of Equipment;
5. fixed bug preventing running of the NetworkMetrics function; and
6. to adjust for double counting of the LAYOVER value in two constraints.

LIST OF REFERENCES

Beech H, Siqi Y (2016) South China Sea: Just where exactly did China get the South China Sea nine-dashed line from? (July 19). https://time.com/4412191/nine-dash-line-9-south-china-sea/.

Brondorfer R, Klug T, Lamorgese L, Manning C, Reuther M, Schlechete T (2015) Recent success stories on optimization of railway systems. Technical report ZR14-47, Zuse Institute Berlin, https://www.zib.de/optimization#opus-year2015

Brown G, Carlyle M, Dell R, Brau J (2013) Optimizing intratheater military airlift in Iraq and Afghanistan, *Military Operations Research*, 18(3), https://www.mors.org/Publications/MOR-Journal.

Christafore RM (2017) Generating ship-to-shore bulk fuel delivery schedules for the Marine expeditionary unit. Master's thesis, Graduate School Operations Research, Naval Postgraduate School, Monterey, CA

Danielson ME (2018) Scheduling amphibious connectors to deliver multiple commodities. Master's thesis, Graduate School Operations Research, Naval Postgraduate School, Monterey, CA

DeMarco JFC (2021) Map of South China Sea economic exclusion zones diagram provided to CPT Forest Sentinella via personal communication, April 6, 2021, for use in her thesis work. Extracted from CPT Sentinella's thesis for reuse by the author January 26, 2022.

Department of Defense (2019) Amphibious Operations. JP 3-02 Washington, DC. https://www.jcs.mil/Portals/36/Documents/Doctrine/pubs/jp3_02.pdf.

Department of the Navy (2021) A Concept for Stand-in Forces, Washington, DC. https://www.marines.mil/News/Press-Releases/Press-Release Display/Article/2858309/marine-corps-publishes-new-document-titled-a-concept-for stand-in-forces/

Department of the Navy (2019) Commandant's Planning Guidance 38th Commandant of the Marine Corps, Washington, DC. https://www.marines.mil/News/Publications/MCPEL/Electronic-Library/Display/Article/1907265/38th-commandants-planning-guidance/cpg/.

Headquarters, U.S. Marine Corps. (2021) Tentative Manual for Expeditionary Advanced Base Operations. (Washington, DC). https://www.mcwl.marines.mil/TMEABO/.

Department of the Navy (2018) Marine Corps Doctrinal Publication (MCDP) 3, Expeditionary Operations. Washington, DC, https://www.marines.mil/News/Publications/MCPEL/Electronic-Library Display/Article/899839/mcdp-3/

Estrada E, Knight P (2015) Introduction to Network Theory, Euler and the Konigsberg Bridge, *A First Course in Network Theory* (Oxford University Press, New York), 1–7

Ford L, Fulkerson D (1956) Maximal flow through a network, Canadian Journal of Mathematics (8), https://www.cambridge.org/core/journals/canadianjournal-of mathematics/article/maximal-flow-through-a-network

Freeman N (2019) Schedule Mixed Integer Program formulation, Python code, and associated Excel worksheets provided to the SEED Center via personal communication and shared with author July 2021.

Harris T, Ross F (1955) Fundamentals of a method for evaluating rail net capacities. Technical report RM-1573, RAND Corporation, Santa Monica, CA. Unclassified 1999. https://apps.dtic.mil/sti/pdfs/AD0093458.pdf

Holland C, Levis J, Nuggehalli R, Santilli B, Winters J (2017) UPS Optimizes Delivery Routes. INFORMS Journal on Applied Analytics 47(1), INFORMS PubsOnLine

Laporte G (1992) The Vehicle Routing Problem: An overview of exact and approximate algorithms. European Journal of Operational Research (59)

Lucas T, Kelton S, Sanchez S, Sanchez P, Anderson B, "Changing the Paradigm: Simulation, Often the Method of First Resort," Naval Research Logistics, 62 (4), 2015, 293–303.

Middlebrook M (2001) May the 25th, The Falklands War, (Pen and Sword, United Kingdom), 239–248

Schrijver A (2002) On the history of the transportation and maximum flow problems. Mathematical Programing (91), https://link.springer.com/article/10.1007%2Fs101070100259

Sentinella F (2021) Using data farming and optimization to enable analysis of concepts of employment for surface connectors. Master's thesis, Graduate School Operations Research, Naval Postgraduate School, Monterey, CA

Think Defence (2021) The Atlantic Conveyor, Accessed January 26, 2022, https://www.thinkdefence.co.uk/the-atlantic-conveyor/

Treves T (2008) United Nations Convention on the of the Sea. United Nations Audiovisual Library of International Law, https://legal.un.org/avl/pdf/ha/uncls/uncls_e.pdf

United Nations (2022), United Nations Treaties, Chapter XXI, Law of the Sea. Accessed April 26, 2022, https://treaties.un.org/pages/ViewDetailsIII.aspx?src=TREATY&mtdsg_no=XXI-6&chapter=21&Temp=mtdsg3&clang=_en

Upton S (2021) ConnectorFarmer User's Manual version 0.14, Unpublished user's manual, SEED Center for Data Farming, Monterey CA.

Ward PW (2008) Optimizing ship-to-shore movement for Hospital Ship humanitarian assistance operations. Master's Thesis, Department of OperationsResearch, Naval Postgraduate School, Monterey, CA.

Yuen L, Ismail W, Omar K, Zirour M (2008) Vehicle routing problem: models and solutions. Journal of Quality Measurement and Analysis 4 (1), https://core.ac.uk/

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California