NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

GENERATING SHIP-TO-SHORE BULK FUEL DELIVERY SCHEDULES FOR THE MARINE EXPEDITIONARY UNIT

by

Robert M. Christafore Jr.

June 2017

Thesis Co-Advisors:	Michael Atkinson
	Kyle Lin
Second Reader:	Emily Craparo

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704–0188

1. AGENCY USE ONLY *(Leave Blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 2017	Master's Thesis August 2016 to June 2017

4. TITLE AND SUBTITLE

GENERATING SHIP-TO-SHORE BULK FUEL DELIVERY SCHEDULES FOR THE MARINE EXPEDITIONARY UNIT

5. FUNDING NUMBERS

6. AUTHOR(S)

Robert M. Christafore Jr.

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

Naval Postgraduate School
Monterey, CA 93943

8. PERFORMING ORGANIZATION REPORT NUMBER

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

USMC Expeditionary Energy Office

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

The views expressed in this document are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol Number: N/A.

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release. Distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

Resupplying Marine Corps units ashore from a seabase presents a unique challenge for amphibious planners. In particular, it is a laborious process to generate the schedules for the ship-to-shore assets that deliver supplies ashore. In this thesis, we focus specifically on the delivery of bulk fuel for a Marine Expeditionary Unit (MEU). We introduce the MEU Amphibious Connector Scheduler (MACS) tool to quickly provide amphibious planners with optimized and executable ship-to-shore delivery schedules of bulk fuel to multiple locations ashore. MACS consists of three main models. The first is a dynamic network flow model to compute the optimal number of runs (i.e., round-trips) for each delivery asset to meet the demand for fuel on shore as quickly as possible. The second model is an assignment heuristic that orders the runs for each delivery asset. This assignment heuristic allows us to bypass a slow mixed integer linear program. The final model is a linear program that takes the output from the first two models and creates a minute-by-minute schedule that minimizes the average completion time over the delivery assets. We analyze several different scenarios, and MACS generates schedules in less than one minute.

14. SUBJECT TERMS

MEU, scheduling, schedule, amphibious, connectors, fuel, expeditionary, ARG

15. NUMBER OF PAGES 97

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

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2–89)
Prescribed by ANSI Std. 239–18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

GENERATING SHIP-TO-SHORE BULK FUEL DELIVERY SCHEDULES FOR THE MARINE EXPEDITIONARY UNIT

Robert M. Christafore Jr.
Major, United States Marine Corps
M.S.B.A., Boston University, 2009
B.S., Virginia Military Institute, 2002

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
June 2017

Approved by: Michael Atkinson
Thesis Co-Advisor

Kyle Lin
Thesis Co-Advisor

Emily Craparo
Second Reader

Patricia Jacobs
Chair, Department of Operations Research

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Resupplying Marine Corps units ashore from a seabase presents a unique challenge for amphibious planners. In particular, it is a laborious process to generate the schedules for the ship-to-shore assets that deliver supplies ashore. In this thesis, we focus specifically on the delivery of bulk fuel for a Marine Expeditionary Unit (MEU). We introduce the MEU Amphibious Connector Scheduler (MACS) tool to quickly provide amphibious planners with optimized and executable ship-to-shore delivery schedules of bulk fuel to multiple locations ashore. MACS consists of three main models. The first is a dynamic network flow model to compute the optimal number of runs (i.e., round-trips) for each delivery asset to meet the demand for fuel on shore as quickly as possible. The second model is an assignment heuristic that orders the runs for each delivery asset. This assignment heuristic allows us to bypass a slow mixed integer linear program. The final model is a linear program that takes the output from the first two models and creates a minute-by-minute schedule that minimizes the average completion time over the delivery assets. We analyze several different scenarios, and MACS generates schedules in less than one minute.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

List of Figures

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

List of Acronyms and Abbreviations

ARG	Amphibious Ready Group
BLT	Battalion Landing Team
COMPHIBRON	Commander, Amphibious Squadron
CSV	Comma Separated Values
LCAC	Landing Craft Air Cushion
LCU	Landing Craft Utility
LHD	Landing Helicopter Dock
LPD	Landing Platform Dock
LSD	Landing Ship Dock
LZ	Landing Zone
MACS	MEU Amphibious Connector Scheduler
MAGTF	Marine Air Ground Task Force
MEB	Marine Expeditionary Brigade
MEU	Marine Expeditionary Unit
MILP	Mixed Integer Linear Program
MOC	Marine Corps Operating Construct
MTVR	Modular Tactical Vehicle Recovery
MPF	Maritime Prepositioning Force
PHIBRON	Amphibious Squadron

R2P2 Rapid Response Planning Process

SIXCON Six Container

TBFDS Tactical Bulk Fuel Delivery System

Executive Summary

Planning for amphibious operations is a time-consuming and arduous process due to the complexity of tasks, elevated risks, and constraints of ship-to-shore movement. Specifically, current daily planning efforts for sustainment operations of Marine Expeditionary Unit (MEU) units ashore are inefficient and overburdened by other concurrent planning requirements. This results in inefficient use of air and surface delivery assets, not meeting operational needs, and staff exhaustion. A combination of competing requirements, insufficient fuel transport containers, and time-constrained planning factors create a sizable problem for planners that must be addressed each day that units are conducting operations ashore.

Scheduling the ship-to-shore movement of people, assets, and resources within a time-dependent and constrained environment is frequently the most tedious, time-consuming, and complex aspects of the overall plan. The coordination of loading, movement, unloading, and docking of multiple, large, powerful ship-to-shore delivery assets (called connectors) over the open ocean, between multiple ships and beaches, requires extreme detail and diligence.

This thesis develops the MEU Amphibious Connector Scheduler (MACS) planning tool using a multi-model approach to quickly and efficiently develop feasible ship-to-shore amphibious schedules to deliver bulk fuel from a seabase. This tool uses reasonable planning inputs to develop minute-by-minute schedules of both surface and air amphibious connectors. We define amphibious schedules as a collection of connector runs (i.e., round-trips) comprising the following six time points:

1. Load at ship
2. Depart ship
3. Arrive at beach/landing zone
4. Unload at beach/landing zone
5. Depart beach/landing zone
6. Arrive back to the ship

MACS easily solves basic and complex MEU resupply scenarios, integrating both surface

and air connectors to satisfy fuel demand ashore as quickly as possible. Analytical planning tools are increasingly important to amphibious planners because the Marine Corps and Navy must continue to progress not only their technologies and tactics, but also their staff planning processes to operate in today's complex and distributed operating environments. MACS can significantly reduce the amount of time and staff man-hours necessary to plan bulk fuel resupply operations from a seabase. Planning aids such as MACS are critical if the Marine Corps wants to remain the premier amphibious force in readiness.

MACS integrates three separate models using realistic planning inputs. The first model, called the Quickest Flow model, is a dynamic network flow model formulated as a linear program. The Quickest Flow's objective is to satisfy demand for fuel ashore as quickly as possible. The primary output of the Quickest Flow model is the number of runs for each connector type from the seabase to each land node. This information alone is of immense value to amphibious planners as they attempt to allocate relatively few connectors across multiple different missions to include required maintenance.

The output from the Quickest Flow model is used by the second model, the Assignment Heuristic, to create a "first cut" of the schedule through the use of different assignment policies. An example of an assignment policy is the policy that sends the next available connector to the land node that has the largest number of scheduled runs remaining. The Assignment Heuristic is critical to the practical usability of the overall model. Without the Assignment Heuristic, we would need to use a binary mixed integer linear program to transform the Quickest Flow model output to the final schedule, which would make it difficult to impossible to solve in a reasonable amount of time for many scenarios.

The final model, the Scheduler Linear Program, is a linear program that takes the output from the first two models and creates a minute-by-minute schedule that minimizes the average completion time for each connector type. The Scheduler Linear Program accounts for potential congestion and smooths out the schedule from the Assignment Heuristic to develop the final amphibious schedule.

We analyze several different MEU-size scenarios, and MACS generates schedules in less than one minute. Future work could study larger examples, such as Marine Expeditionary Brigade (MEB) scenarios. This thesis only focuses on the delivery of fuel, but the machinery developed has the potential to be the foundation for a much more comprehensive amphibious

planning tool that incorporates personnel, vehicles, and pallets of equipment.

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgments

Where and how to begin...

To my incredible wife, **Heather**, you truly are amazing with all that you do. Thank you for all your love, support, and humor. The kids and I are truly the luckiest people in the world.

To my children, **Michael** and **Maddie**, I love you guys to death and you are always in my thoughts.

To Professors **Michael Atkinson** and **Kyle Lin**...all I can say is that I really lucked out. I couldn't have asked for more patient, brilliant, and engaging thesis advisors. Even though I will probably never use blue or green font ever again, I can say that I don't think I would be here if it were not for you gentlemen.

To the rest of my family, thank you for your unwavering love and support. You have always been there for me.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

The Marine Expeditionary Unit (MEU) remains America's preeminent force in readiness for conflicts virtually anywhere in the world. The MEU and the Amphibious Ready Group (ARG) create a highly capable amphibious force able to strike and conduct operations from the sea without a land-staging base. The Marine Corps and U.S. Navy call this seabasing. Operations conducted from a seabase (ships) to the shore via transport assets—called connectors—are known as amphibious operations and are among the most complex and challenging within the Department of Defense.

Planning for amphibious operations is a time-consuming and arduous process due to the complexity of tasks, elevated risks, and constraints of ship-to-shore movement. The Marine Corps has earned the reputation of world experts in amphibious operations, but the MEU and ARG planners can be quickly overwhelmed by the inherent difficulties and staff synergies that must be coordinated for even the most basic operations—moving personnel, material, and equipment back and forth between the seabase and the shore. This is especially prevalent given the current highly complex and demanding distributed operating environments, when dealing with sustainment operations for forces ashore.

Current daily planning efforts for sustainment operations of MEU units ashore are inefficient and overburdened by concurrent planning requirements. This results in inefficient use of air and surface delivery assets, not meeting operational needs, and staff exhaustion. A combination of competing requirements, insufficient fuel transport containers, and time constrained planning factors creates a sizable problem for planners that must be addressed each day that units are conducting operations ashore.

Scheduling the ship-to-shore movement of people, assets, and resources within a time dependent and constrained environment is frequently the most tedious, time consuming, and complex aspects of the overall plan. The coordination of loading, movement, unloading, and docking of multiple, large, powerful ship-to-shore connectors over the open ocean, between multiple ships and beaches requires extreme detail and diligence regardless of the scale or complexity of the overall mission or campaign.

The purpose of this thesis is to design a tool to streamline the planning cycle of amphibious operations. Specifically, we formulate mathematical models and develop algorithms to schedule connectors that transport bulk fuel from ships to shore. We call this tool MEU Amphibious Connector Scheduler (MACS), which takes less than one minute to run on a personal computer, and creates an effective delivery schedule. MACS dramatically reduces the staff planning man-hours and frees up personnel to focus on the next major phase of the operation.

1.1 Motivation of Research

As a Battalion Landing Team (BLT) Operations Officer, personal experience purports that (1) the current planning methods for sustainment operations bogs down a MEU/ARG staff within two to three days; (2) efficient use of delivery assets (connectors) quickly no longer becomes a requirement; (3) MEU/ARG staffs have very few analytic planning tools to streamline the process of moving bulk sustainment assets ashore. This last point is important as resupply operations are inherently more fixed and require less "operational art" compared to more kinetic operations such as raids or assaults. Therefore, a quantitative tool would have the greatest positive effect on optimizing the planning process. The motivation of this research is furthered discussed in three aspects below.

1.1.1 Operational Tempo

The Marine Corps prides itself on maintaining an operational tempo unmatched by other services or adversaries. As seen during Operation Iraqi Freedom, a land-based Marine Corps' operational tempo overwhelmed the Iraqi Army and pushed far ahead of the United States Army despite being in the most contested and populated areas. This ability to constantly out-cycle adversaries is one of the main ingredients to the success of the Marine Corps. Operational tempo becomes incredibly difficult to maintain in an amphibious operating environment.

The complexity, increased risk, finite resources, and command and control challenges of amphibious operations can drastically degrade the operational tempo of even the most proficient units. Very often the time required to develop an executable ship-to-shore movement plan affects the operational timeline negatively due to the complexity and completeness

required of amphibious operations' plans. Additionally, the MEU/ARG Team must be able to simultaneously plan and execute multiple missions across the range of military operations furthering the necessity for comprehensive quantitative planning tools. Tempo is one of the main tenets of maneuver warfare that allows a force to take advantage of enemy weaknesses vice striking enemy strengths. It is imperative that forces ashore maintain their operational speed and momentum by a responsive and efficient seabase. This is especially applicable when it comes to fuel sustainment. A seabase that is constantly playing catchup with key logistics will inevitably have an adverse effect on overall mission success. Streamlining and finding efficiencies within the daily planning cycle in the form of reliable, prudent, and efficient analytical planning tools will directly allow the Marine Corps to maintain its operational edge and free up planners to focus on planning more decisive actions or missions.

1.1.2 Efficient Use of Assets

The efficient use of people, assets and resources is a major planning consideration within amphibious operations. A MEU/ARG is isolated, serving as a mobile seabase with no ground lines of communication for resupply or maintenance capability. Everything used for a particular mission originates from the ships in the seabase and must be exercised and used in an efficient manner. The only way for amphibious ships to be resupplied while at sea is from other ships, which further reinforces the importance of efficiently using resources. Waste of fuel and asset usage must be minimized as ships are not privy to ease of resupply or almost indefinite storage capability inherent of land bases. Additionally, in an era of constrained resources and budgets, the Navy and Marine Corps must continue to find ways to reduce waste and find efficiencies.

1.1.3 Marine Corps Operating Construct

The Marine Corps Operating Construct (MOC): *How an Expeditionary Force Operates in the 21st Century* published in September 2016 outlines how the Marine Corps will operate, fight, and win in 2025 and beyond, as well as shapes the Corps' actions in designing and developing future capabilities and capacity of the future force. It is essentially a document to guide the collective efforts of all Marines and Marine Corps equities to ensure the Marine

Corps' future readiness and relevancy. The MOC explains that the 21st century Marine Air Ground Task Force (MAGTF) must be able to operate and fight at sea, from the sea, and ashore as an integrated part of the larger Naval force as well as Combined/Joint force. This guiding document promotes the requirement for the Marine Corps to be the premier amphibious force within the Department of Defense. Maximizing the Navy's ability to create rapidly deployable seabases throughout the world is critical to the long-term prospects of the Marine Corps [1].

This document stresses the Marine Corps' dedication to be America's premier amphibious force in readiness. This requires the Marine Corps to continue to innovate and develop new best practices to maintain its operational edge in not only an amphibious operating environment, but in a distributed environment as well. These requirements further reinforce the argument for analytical operational planning and scheduling tools to support sustained, distributed operations from the sea.

1.2 Mathematical Approach

To develop an effective schedule for connectors to ship bulk fuel from ships to shore, we break up the process into three separate models.

1. The first model formulates a dynamic network flow model to compute the optimal number of runs (i.e., round-trips) for each connector type in order to meet the demand on shore as quickly as possible. This model is formulated as a linear program.
2. The second model treats each connector type separately and uses a heuristic to produce an ordering of runs for each connector type, so as to meet the constraints due to ship space and landing spots on shore.
3. The third model uses the output from the second model to create a minute-by-minute schedule that minimizes the mission completion time. This problem is also formulated as a linear program.

Since it is straightforward to enter data to run the model, the end product is an easy-to-use tool for any amphibious planner. By formulating the optimization models as linear programs, the whole procedure typically takes less than one minute to produce an effective schedule on a personal computer, which dramatically reduces the manual planning that

often takes up several hours.

1.3 Related Work

In order to generate the connector schedule, we first need to determine the number of runs (or round-trips) by connector type. To do so, we formulate a dynamic network flow model. Most standard network flow models maximize the flow from a source to a sink without a time component [2]. Dynamic network flow models add a time component by expanding the network and essentially recreating a copy of the network for each time period to examine how the flow moves in space and time (see for example [3]).

Hamacher and Tjandra's [3] develop a dynamic network flow approach to model large-scale evacuation problems. Of particular relevance to our work is the Maximum Dynamic Flow problems described in Sections 3–6 in Hamacher and Tjandra [3] that maximizes the dynamic flow that reaches a sink over a given time horizon T. We use this model as a starting point for determining the number of runs by connector type. Chapter 3 describes this effort in detail.

Dynamic network flow models are popular with evacuation problems due to the emphasis on moving quickly through a network. In his thesis, Langford [4] uses a space-time network flow representation to examine the evacuation of individuals from a neighborhood in case of an emergency. This work solves for evacuation routes and neighborhood clearing times if a central authority can effectively dictate the evacuation plan. His thesis focuses on minimizing the time required to evacuate neighborhoods via a proposed road network. The outputs of this thesis quantify clearing times of the neighborhoods in question. The dynamic network flow component of our MACS tool uses a similar methodology to quickly move flow through a network. One modeling difference is that our approach puts a flow constraint on the nodes, whereas Langford's work puts a flow constraint on the arcs. Malveo [5] examines a similar evacuation setup as Langford [4] and primarily focuses on the impact of congestion. That is during a mass evacuation, if the roads are packed with vehicles, then the flow rate along the roads will decrease. Malveo [5] handles this by discretizing the nonlinear relationship between the number of vehicles on the road and the velocity. Our application does not have congestion along arcs during connector transit. However, we do account for congestion at nodes when connectors may need to wait for a spot to open before

unloading.

There exists a limited number of articles on mathematical modeling of ship-to-shore optimization in the literature. We next describe the three works most similar to ours. Viado [6] develops a mixed integer program that determines the fuel inventory and delivery requirements for a MEU. His thesis focuses on minimizing the initial fuel supply that must be staged ashore prior to the initiation of operations and a general just-in-time fuel delivery schedule. The outputs of this thesis provide an hourly schedule of delivery times to meet the forecasted fuel requirements of a large amphibious operation. Our model produces a minute-by-minute plan, rather than an hourly plan, and explicitly accounts for potential congestion at beaches, landing zones, and ships when multiple connectors are there at the same time.

Ward [7] formulates a mixed integer optimization model to generate schedules of ship-to-shore movements from hospital ships during Humanitarian Assistance operations. His model has a similar network structure to ours including: dynamic network using discretized time periods, transit time on arcs for different connector types, and number of connector spots at each node. He approximately accounts for loading and unloading times, but not with the fidelity that we do. He focuses on getting personnel ashore (e.g., doctors) to perform particular tasks (e.g., operations) and defines cost penalties for missing deadlines and costs for using connectors. His objective is to determine a schedule to minimize a weighted cost function. The crucial difference between Ward's work [7] and ours is that we incorporate multiple round-trips for each connector; the connectors in Ward's model are used either once or not at all. Determining the total number of round-trips is nontrivial and one of the key pieces of our analysis.

Reitter [8] develops a seabase logistics planning tool. This tool takes in very detailed information and primarily focuses on determining sustainment requirements of deployed forces ashore. The inputs include which types of units are ashore and what operations these units are performing. The sustainment requirement outputs from Reitter's tool would be very useful as an input to our model. We need the fuel demand ashore at various nodes and Reitter's model [8] could provide this. Reitter's formulation does provide an aircraft scheduling algorithm that is similar to an assignment heuristic we develop (the second model in Section 1.2). However, our model incorporates both air and surface connectors,

accounts for potential congestion during unloading, and models the network structure on land. Reitter does not include these aspects in his planning tool.

Our MACS tool generates a schedule, and there are many algorithms that schedule many different types of tasks. Hartman [9] develops a Passengers and Cargo Route Optimization Program (PROP) that serves to provide an optimized plan to transport personnel and cargo by air for non-combat or non-operational purposes. The PROP is a mixed integer program that generates PMC planning solutions while minimizing operating costs, reducing planning time, and satisfying all PMC-unique constraints. The output provides takeoff and landing times, routes, and the cargo and personnel transported. This model is much different from our work in that it accounts only for the assignment of persons to assigned aircraft for administrative transport to a network of land nodes without the development of a comprehensive amphibious schedule using multiple connectors across a dispersed network of multiple ships and beaches/LZs.

Jacobs [10] develops a tool to assist in building daily flight schedules for training. The tool ensures students perform their required training modules and are matched with appropriately qualified instructors. The tool is driven by a mixed integer linear program that allows users to adjust weights on a value-oriented objective function to help increase the throughput of students. The Jacobs tool and our work both have a matching aspect: we match connector round-trips to beaches and Jacobs matches students to instructors. However, the context and underlying mathematical machinery are much different.

1.4 Thesis Organization

Chapter 2 provides background information on amphibious operations. In Chapter 3 we describe the three models that construct the MACS tool. Chapter 4 demonstrates the MACS tool via a scenario and presents results. We conclude the thesis and suggest future research opportunities in Chapter 5.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 2:
Background on Amphibious Operations

In order to appreciate how this thesis can impact amphibious operations planning, it is essential to have a general understanding of amphibious planning requirements and equipment capabilities of the overall amphibious task force. The amphibious task force comprises two coequal commands: the Marine Expeditionary Unit (MEU) and the Amphibious Ready Group (ARG). The MEU is commanded by a Marine Colonel who is known as the MEU Commander, while the ARG is commanded by a Navy Captain who is known as the Commander, Amphibious Squadron (COMPHIBRON). The MEU provides the landing force (Marines, equipment, trucks, etc.) and the squadron of aircraft. The ARG is comprised of Navy amphibious ships and surface connectors to move MEU assets over the water to conduct operations ashore. This chapter supplies the readers with a baseline knowledge of amphibious planning and assets, most notably what we call connectors, that are covered in this thesis.

2.1 Amphibious Ready Group

The seabase structure used by the MEU is the Amphibious Ready Group (ARG). The ARG consists of three amphibious ships, each with a welldeck capable of housing and launching surface connectors and storing the MEU's equipment. The three classes of ships are the Landing Helicopter Dock (LHD) class, Landing Ship Dock (LSD) class, and Landing Platform Dock (LPD). The ships also serve as seaborne command and control platforms until control can be transferred ashore to Marine Units. [11].

2.1.1 ARG Ships

Landing Helicopter Dock (LHD) Class Ship

The LHD, seen in Figure 2.1, serves as the MEU/ARG command ship as well as the main rotary wing and fixed wing seabasing platform with over 90% of MEU aircraft capability. For our model we assume that all air connectors are based on the LHD. The LHD also has

a very large well-deck that houses several surface connectors. The LHD provides the bulk of the amphibious force organic to the MEU [11].

Figure 2.1. Landing Helicopter Dock (LHD) Class Ship. Source: [12].

Landing Platform Dock (LPD) Class Ship

The LPD, seen in Figure 2.2 is the second largest ship in the ARG behind the LHD and has the capability to provide an alternate command and control capability. The LPD has a much smaller flight deck that is used mainly for air refueling and casualty evacuation operations. The LPD has a large welldeck that contains several surface connectors [11].

Figure 2.2. Landing Platform Dock (LPD) Class Ship. Source: [13].

Landing Ship Dock (LSD) Class Ship

The LSD, seen in Figure 2.3 is oldest of the three ARG ship classes. It has the least air capability in the ARG and houses some surface connectors. The LSD generally contains mainly MEU logistics equipment [11].

Figure 2.3. Landing Ship Dock (LSD) Class Ship. Source: [14].

2.2 Amphibious Connectors

Connectors are the vehicles the Navy and Marines use to move assets, personnel, and equipment, from ships to various locations on land. They are generally divided into two subgroups: surface and air connectors. The sequencing of the connectors is key to an efficient amphibious schedule that satisfies the MEU and ARG Commanders' daily objectives. Most of the planning requirements' rigor mentioned in Chapter 1 deals with developing the offload schedule for all connectors across all ships being used for that particular mission.

Both air and surface connectors require the same general time events on an amphibious schedule. These time points are load at ship, depart ship, arrive at beach (or landing zone), unload at beach, depart beach, and arrive back at ship.

2.2.1 Surface Connectors

The two surface connectors used in this thesis are the Landing Craft Air Cushioned vehicle (LCAC) and the Landing Craft Utility (LCU). These are Navy platforms and under the command of the Amphibious Squadron (COMPHIBRON). Both platforms provide the MEU and ARG with the ability to move vehicles, supplies, and people from ship to shore

via movement over the water. While the LCU has the ability to transport twice as much as an LCAC, the LCU is much slower and has more stringent landing constraints. The LCAC carries less, but is substantially faster and is able to hover over the ground, which reduces the landing constraints.

Landing Craft Air Cushioned (LCAC)

The LCAC is the mainstay of the Navy's surface connector fleet, and is the main connector-type used for moving vehicles and equipment ashore due to its speed and hovercraft capability. The hovercraft capability enables it to land on a much wider array of beaches than the LCU as it is able to hover over the land and is not powered by a traditional subsurface propeller [11]. See Figure 2.4:

(a) LCAC Transporting Supplies. Source: [15].

(b) LCAC Unloading at Beach. Source: [16].

Figure 2.4. LCACs Transporting Supplies and Equipment

As displayed in Figure 2.4a, LCACs can travel over the water at speeds up to 50 knots while transporting 60 tons of equipment, but these planning factors are subject to sea state, beach slope, and the LCACs fuel requirements [17]. Due to the high rate of speed and ability to traverse higher waves, equipment must be firmly secured to the deck. This process requires time and we account for this loading and securing time in our model. For most of the scenarios we use a planning factor of 20-30 minutes for loading at the ship. Each ARG generally deploys with 5 LCACs divided over two ships, the LHD (3 LCACs) and LPD (2 LCACs) [11].

Landing Craft Utility (LCU)

The LCU, seen in Figure 2.5, is a larger, but less capable landing craft. It can transport virtually double the weight of the LCAC per run, but is substantially slower and less capable in rougher seas. Additionally, because it operates more like a traditional vessel it is much more restricted in the beaches that it is able to land. The slope, grade, and composition of the beach, as well as the tides have more profound impacts to the usability of the LCU versus the LCAC. The LCU can carry up to 120 tons and has a larger area to transport equipment. The LCU is also more capable at transporting personnel [11].

(a) LCU Transporting Supplies. Source: [18].

(b) LCU Unloading at Beach. Source: [19].

Figure 2.5. LCUs Transporting Supplies and Equipment

The MEU/ARG generally deploys with two LCUs aboard one of the ships, usually the LSD. The LCUs are substantially slower with a cruising speed at between 6-9 knots. Even with the increased lift capacity of the LCU, for missions requiring multiple runs, LCACs generally provide a greater amount of assets delivered than the LCU over the course of a day. Specifically, for missions requiring multiple ship-to-shore trips, the LCAC speed dominates the increased capacity of the LCU. Additionally, the ship must ballast lower for an LCU vice an LCAC because the LCU operates more like a vessel itself. Due to the substantial differences in speed, beach reachability and time to offload given capacity, LCACs and LCUs will generally be assigned different beaches. If LCACs and LCUs do unload at the same beach, then they will be assigned separate landing spots.

2.2.2 Air Connectors

The Marine Corps squadron supplies the assault support aircraft used to move personnel and equipment from ship to locations further inland. The two air connectors are the CH-53E

Super Stallion and MV-22 Osprey. The CH-53E has superior lift capability with the MV-22 being much faster. For purposes of fuel delivery, both have the same capability as their maximum lift is dictated by the fuel transport container that both use, not the total weight of the fuel. The system emplaced in the aircraft to transport bulk fuel is called the Tactical Bulk Fuel Delivery System (TBFDS). These are fuel containers placed inside the aircraft that have hoses that deploy once the aircraft lands to refuel vehicles. The aircraft has to be configured for this mission and thus can only be used for fuel delivery operations for that day. The CH-53s and MV-22s are usually all concentrated on the LHD.

CH-53E Super Stallion

The CH-53E, seen in Figure 2.6 is the Marine Corps' heavy lift helicopter with the capability to lift in excess of 30,000 pounds. It has a top speed of 170 knots, a cruising speed of 150 knots, and range of 540 nautical miles [20]. The MEU's composite squadron deploys with four CH-53E's and are generally all concentrated on one ship (usually LHD). The CH-53E is the aircraft of choice for the TBFDS described in Section 2.3.

(a) CH-53E in Flight. Source: [21].

(b) CH-53E's Departing an LHD. Source: [22].

Figure 2.6. CH-53E Super Stallion

MV-22 Osprey

The MV-22 Osprey is the cornerstone of the Marine Corps assault support aircraft. It is technically a tilt-rotor aircraft, not a traditional helicopter, and provides significantly enhanced operational range and speed. The MV-22 Osprey can reach speeds of up to 320 knots, generally operates at between 240 to 280 knots, and has a range of up to 1050 miles. The Osprey has a max payload capacity of 20,000 pounds compared to more than 30,000 pounds for the CH-53E [20]. The MEU deploys with twelve ospreys concentrated on one ship.

(a) MV-22 in Flight. Source: [23].

(b) MV-22's departing an LHD. Source: [24].

Figure 2.7. MV-22 Osprey

2.3 Fuel Containers for Surface and Air Connectors

Surprisingly, one of the main constraints associated with fuel delivery operations in an amphibious operating environment is the lack of bulk fuel delivery storage on both surface and air connectors. Surface and air connectors each have specific fuel storage platforms. The main platform for surface connectors (LCAC and LCU) is the Six Container (SIXCON) fuel storage tank seen in Figures 2.8c and 2.8d. These tanks are emplaced on the Marines' primary supply truck called the Modular Tactical Vehicle Recovery (MTVR). The MTVR can carry up to three tanks and fuel pump (similar to gas station nozzle and hose), but generally is only outfitted with two for amphibious operations. Each SIXCON contains up to 900 gallons of fuel, and the number available for fuel transport on a MEU varies [25].

Air connectors (CH-53E and MV-22s) use the TBFDS, seen in Figures 2.8a and 2.8b. These are internally stored 800 gallon tanks that come equipped with their own hoses to provide bulk fuel to forward units once on the ground. This long-range bulk fuel resupply capability enables the amphibious task force to expand the operating area ashore and support distributed operations. Both the CH-53E and the MV-22 are capable of transporting up to three TBFDS tanks. However, due to extremely constrained space aboard amphibious ships, squadrons generally only embark 2-6 total systems. This constraint was built into the model for realism, but can be scaled up to more systems if required.

(a) Single Tactical Bulk Fuel Delivery System unit. Source: [26].

(b) TBFDS in a CH-53E. Source: [27].

(c) USMC SIXCON fuel container. Source: [28].

(d) SIXCON on an MTVR. Source: [29].

Figure 2.8. Primary USMC Fuel Containers Used on Amphibious Connectors

2.4 Staff of the Amphibious Task Force

As previously mentioned the Navy's amphibious planning entity is the Amphibious Squadron (Amphibious Squadron (PHIBRON)) staff. The PHIBRON staff serves as a coequal command to the MEU staff and provides the tasking and coordination for each of the ARG ships as well as the surface connectors and air command and control that supports the Marine Corps' plan. The lack of a unified commander physically aboard the ARG of all personnel and equipment adds a layer of complexity that does not exist outside of

amphibious operations.

The ARG staff is not as large as the MEU staff, and that sometimes serves as a constraint in the overall amphibious planning process. The PHIBRON staff and MEU staff are responsible for several critical planning documents to include an amphibious schedule that is fully nested with the Marines' landing plan. MEU and ARG planners act as one nested planning team for each mission, but much of the planning is conducted in a "stovepipe" manner with sync meetings to ensure all facets of the plan are aligned and synchronized. This allows planners to focus on their products, but also leads to much longer planning time frames [11].

2.5 Planning Requirements and Products

MEU and ARG planners execute two general planning methods–deliberate planning and crisis action planning. The latter is also called the Rapid Response Planning Process (R2P2) [11]. For the purpose of this thesis we will focus on the deliberate planning process even though this model has the potential to significantly add value and efficiency to crisis action planning. This section serves primarily to introduce the various planning products that can either be replaced or significantly streamlined by the model in this thesis. Both the MEU and ARG have similar and supporting planning products and outline the execution of a particular mission.

Joint Publication (JP) 3-02 [30] details the steps that amphibious planners need to take prior to developing a landing plan as listed below:

1. As the landing force, the Marines develop a ground scheme of maneuver to execute once ashore
2. Commander, Landing Force (generally the MEU Commander) identifies support requirements and Navy assets needed to flow forces ashore and accomplish the mission.
3. Commander, Amphibious Task Force (ARG Commander) assesses the MEU Commander's requests and identifies additional Navy assets needed.
4. Commander, Amphibious Task Force requests additional assets from higher authority if necessary.
5. Plans are adjusted to match assets available.
6. Commander, Amphibious Task Force and Commander, Landing Force agree on the

final allocation of means (air and surface connectors, additional aircraft, etc.).

7. Detailed Landing Plan is the developed.

This is a very involved and systematic process given the coordination, communication, and detailed planning required to execute amphibious operations. This thesis provides a means to significantly streamline, with a high degree of fidelity, steps 2 through 7. Providing the user a final optimized amphibious delivery schedule of fuel given assets available that servers as the landing plan [30].

Specifically, this thesis presents a tool that will significantly reduce the amount of time planners will need to develop operational and executable plans. More importantly this tool possesses the ability to provide MEU and ARG planners with a feasible and executable ship-to-shore plan in minutes. Allowing planners to quickly transition to refine the plan, not spending more time and effort in determining whether or not a plan to move fuel ashore is viable. Below we list several specific amphibious planning products outlined in NTTP 3-02.1M/MCWP 3-31.5 [11] that could be either streamlined or completely replaced by our MACS tool. Please see NTTP 3-02.1M/MCWP 3-31.5 [11] for a more detailed description of each of these products.

Amphibious Task Force Commander products (ARG)

- Landing Craft Employment Plan
- Debarkation Schedule
- Approach Schedule

Landing Force Commander Landing Plan products (MEU)

- Landing Force Landing Plan
- Landing Craft and Amphibious Vehicle Assignment Table
- Landing Diagram
- Landing Force Sequence Table
- Assault Schedule
- Heliteam Wave and Serial Assignment Table
- Assault Support Employment and Assault Landing Table

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 3:
Mathematical Formulation

In amphibious operations, a *run* refers to a round trip by a connector to either a beach or Landing Zone (LZ), and then back to the ship. The goal of this thesis is to produce a minute-by-minute schedule for connector runs. This section presents the mathematical formulation to achieve this goal. Table 3.1 gives a sample of the output, which specifies a connector run by 6 time points.

Ship	Run	Destination	Start Load	Depart Ship	Arrive at Beach	Start Unload	Depart Beach	Arrive at Ship
LHD	1	B1	0.0	30.0	100.0	100.0	125.0	195.0
LHD	2	B1	30.0	60.0	130.0	130.0	155.0	225.0
LHD	3	B2	60.0	90.0	180.0	180.0	205.0	295.0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table 3.1. Example of an amphibious delivery schedule for LCAC

We next turn to the inputs required to generate such a schedule. The inputs for this model are reasonable, easy to obtain by the planner, and require no pre-calculating or configuring on behalf of the planner. The specific planning inputs are covered extensively in Chapter 4, but for context we list a few broad examples of the planning inputs below:

- number of ships in seabase, number of connectors on each ship, number of docking spots for each connector type by ship;
- connector fuel capacity and speed;
- land network layout (locations of units needing resupply) and distances from the ships, number of landing spots at each land node;
- fuel demand at each location.

To create an optimized amphibious schedule requires a series of different models. The first model, called the *Quickest Flow model*, is a linear program that produces the overall number of runs for each connector type from the seabase to each land node to satisfy the demand. For example the Quickest Flow model might specify that we need six LCAC runs to Beach A, four LCAC runs to Beach B, and ten MV22 runs to LZ1. The output from the Quickest Flow model is used by the second model, the *Assignment Heuristic*, to create a "first cut"

of the schedule through the use of different assignment policies. The final model, the *Scheduler Linear Program*, accounts for potential congestion and smooths out the schedule from the Assignment Heuristic to develop the final minute-by-minute amphibious schedule. The Assignment Heuristic is critical to the practical usability of the overall model. Without the Assignment Heuristic, we would need to use a binary mixed integer linear program to transform the Quickest Flow model output to the final schedule, which would make it difficult to impossible to solve in a reasonable amount of time for many scenarios.

This chapter covers the formulation of the three main models comprising the MACS tool. Figure 3.1 below provides an overview of the overall model in order to build context for the reader as they examine the formulation of the main models throughout the rest of this chapter. Each section in this chapter treats its given model as a stand-alone entity. Chapter 4 provides a more detailed explanation of the implementation of the overall program that connects all three models, with detailed examples of the inputs and outputs of each step.

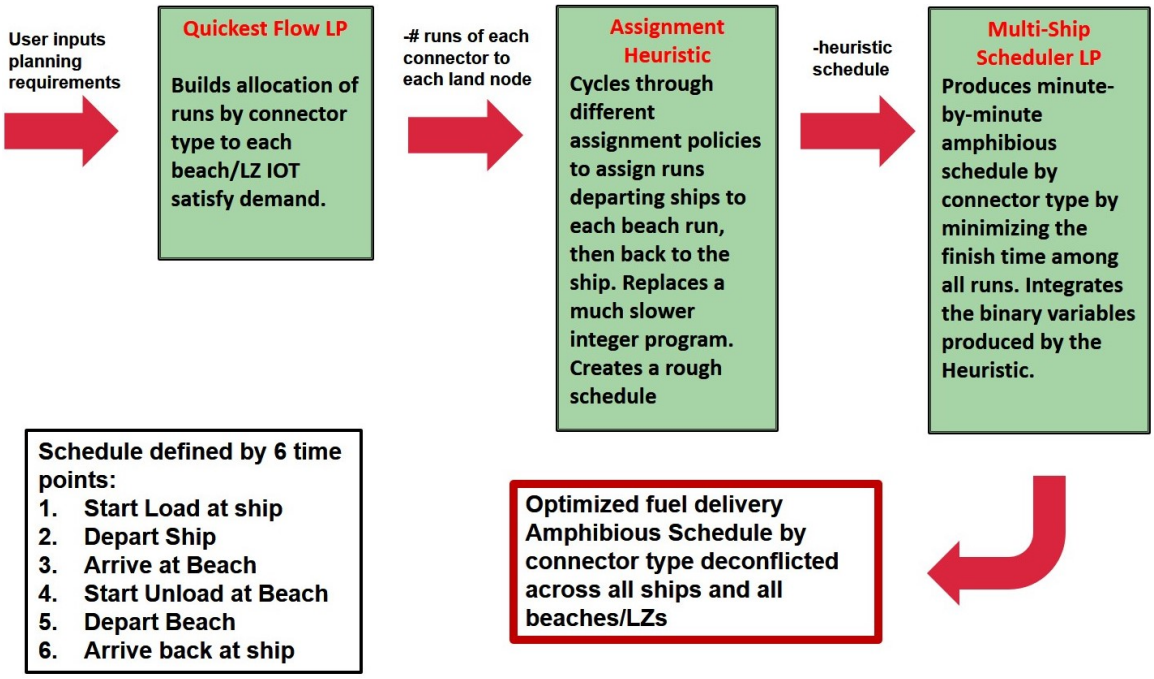


Figure 3.1. MACS Tool Models Overview

3.1 Quickest Flow Formulation

This section presents an optimization model that determines the number of runs by connector type needed to accomplish the fuel delivery mission. We focus on minimizing the time to deliver all fuel demanded by the land nodes. At a high level this problem relates to classic maximum network flow problems [2] as we flow fuel to the land nodes. However, there are two key differences: First, we include a time dimension, which standard network flow models do not account for. Second, the flow actually happens in discrete runs moving back-and-forth between the ship and the land nodes not as a continuous flow. We address these differences through the implementation of a dynamic network flow model and an approximation of discrete runs via continuous flow.

We start with a total amount of supply for each connector type at the seabase (ships). The model then pushes the flow from the seabase to beaches and landing zones, and then further inland via a network flow framework. The main difference between this model and standard network flow models is that we add a time dimension. The model specifies the flow on an arc (i, j) leaving node i at time period t, $X_{i,j,t}$, vice simply $X_{i,j}$. Figure 3.2 illustrates a simple network representing these types of problems.

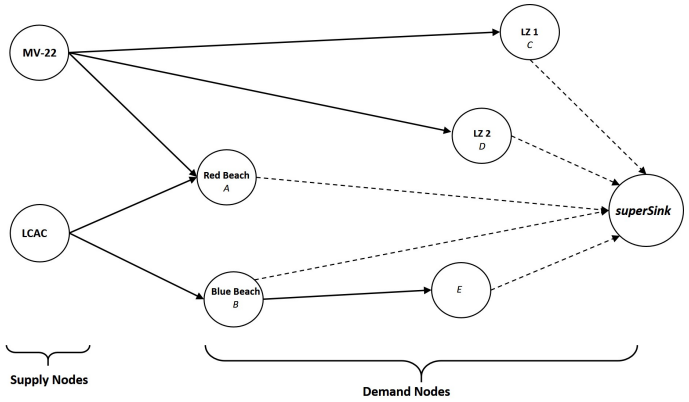


Figure 3.2. Simple Network with Two Supply Nodes and Five Demand Nodes.

With respect to the specific network architecture, there are two types of nodes: supply nodes and demand nodes. Each supply node represents one type of connector, and is indexed by

$s \in S$, where S is the set of different connector types. A flow from supply node s represents fuel shipped via connector type s, which in this model can be an LCAC, LCU, MV-22, or CH-53. We approximate the flow of fuel by aggregating supply over all ships in the seabase and all connectors. Each demand node represents one physical location on the shore, and is indexed by $l \in L$, where L is the set of demand nodes ashore. A flow into demand node l represents fuel delivered to that location. Additionally, like many network flow problems, we define a *supersink* node that accumulates all satisfied demand across all demand nodes.

This model contains two sets of decision variables. The decision variable $X_{i,j,t}$ represents fuel flow leaving node i to node j at time t. The output we need for our overall analysis is the sea-to-land flow: $X_{s,j,t}$ for $s \in S$. In order to determine the best sea-to-land flow we also need to include land-to-land movement ($X_{i,j,t}$ for $i, j \in L$) in the Quickest Flow model because we want to flow fuel to all demand points as fast as possible. For the simple network shown in Figure 3.2 our analysis must account for the land-to-land flow between Blue Beach and Node E in order to determine the total flow originating from the LCAC supply node. The second decision variable $Y_{n,t}$ represents the amount of fuel stored at node n at time t before it is shipped somewhere else at a later time period.

We next introduce the model parameters. The fuel supply available from supply node s is written by $supply_s$, for $s \in S$. For example, if we have 5 LCACs across 2 ships then this supply represents the total amount of fuel we can push ashore with all 5 LCACs during the course of a welldeck crew day. The total demand at demand node l is written by $demand_l$, for $l \in L$. The storage capacity of demand node l is written by $storage_l$, for $l \in L$. The amount of fuel/flow we can push out of a given node each time period is limited by the connectors we have available at the node, and is denoted by u_n. The parameter u_n is the maximum amount of flow we can push out of a node each time period and applies to all nodes: supply and demand. Whereas $supply_s$ only applies to supply nodes and specifies the total fuel we can push ashore in a day by connector type. The time to traverse an arc from i to j is defined by $tau_{i,j}$. The variable $X_{i,j,t}$ is the flow leaving node i at time t headed to node j, and it will arrive at node j at time $t + tau_{i,j}$.

Our goal is to flow the fuel to minimize the time to satisfy all demand. To accomplish this, we fix the total number of time periods and maximize the amount of fuel demand satisfied across all demand points during this time window. If the amount of demand satisfied is

the total demand, we are done. Otherwise we increase the number of time periods and repeat the process. Thus we solve the problem many times, increasing the time window until all demand is satisfied. The formulation we present below is for only one fixed time window. This model is an adaptation of the evacuation models that appear in Sections 3-6 of Hamacher and Tjandra [3], with a few modifications that we discuss when defining each element of the formulation.

We present the linear program below, and then explain its constraints.

Indices and Sets

$t \in T$	time periods
$s \in S$	supply nodes
$l \in L$	land nodes that have a fuel demand
$ss \in superSink$	single $ss \in superSink$
$allSinks$	All sink nodes $allSinks = L \cup superSink$
$i, j, n \in N$	$n \in N$ all nodes in network: $N = S \cup allSinks$
$(i, j) \in A$	arcs directed from node i to node j in N

Data [units]

$maxTime$	total time available for operations [minutes]
u_n	capacity on flow leaving node n, each time period [gallons]
$tau_{i,j}$	time to travel on each arc (i,j) [minutes]
$nodeCap_l$	storage capacity at node l [gallons]
$supply_s$	total supply of fuel node s [gallons]
$demand_l$	fuel demand at node l [gallons]

Decision Variables [units]

$X_{i,j,t}$	Flow of fuel from i to j starting at time t [gallons]
$Y_{n,t}$	Fuel stored at node n at time period t [gallons]

Formulation

$$\max_{X,Y} \quad \sum_{t \in T} \sum_{(i,ss) \in A} X_{i,ss,t} \tag{3.1}$$

$$\text{s.t.} \quad Y_{s,0} = supply_s \qquad \forall s \in S \tag{3.2}$$

$$Y_{l,0} = 0 \qquad \forall l \in allSinks \tag{3.3}$$

$$Y_{l,t} \leq nodeCap_l \qquad \forall l \in L, \forall t \in T \tag{3.4}$$

$$\sum_{\substack{j \in N \setminus \{ss\}: \\ (n,j) \in A}} X_{n,j,t} \leq u_n \qquad \forall n \in N, \forall t \in T \tag{3.5}$$

$$\sum_{t \in T} X_{l,ss,t} \leq demand_l \qquad \forall l \in L \tag{3.6}$$

$$Y_{n,t+1} = Y_{n,t} + \sum_{\substack{i \in N: \\ (i,n) \in A}} X_{i,n,t-tau_{i,n}} - \sum_{\substack{j \in N: \\ (n,j) \in A}} X_{n,j,t} \qquad \forall n \in N, \forall t \in T \tag{3.7}$$

$$Y_{l,maxTime} = 0 \qquad \forall l \in L \tag{3.8}$$

$$\sum_{t \in T} \sum_{s \in S} \sum_{\substack{j \in N: \\ (s,j) \in A}} X_{s,j,t} \leq \sum_{l \in L} demand_l \tag{3.9}$$

$$X_{i,j,t} \geq 0 \qquad \forall i \in N, \forall j \in N, \forall t \in T \tag{3.10}$$

$$Y_{n,t} \geq 0 \qquad \forall n \in N, \forall t \in T \tag{3.11}$$

The objective for this model is to maximize the amount of demand satisfied by the end of all time periods. Therefore, we track the flow of fuel to the supersink node ss by the end of all time periods T. The constraints associated with the quickest flow formulation primarily deal with maintaining the balance of flow between nodes over time. Specifically, these constraints ensure that for each time point the flow into demand nodes does not exceed capacity.

1. Constraint (3.2) ensures all supply originates at the source nodes $s \in S$ at time 0.
2. Constraint (3.3) initializes the starting supply at the land nodes to zero.

3. Constraint (3.4) ensures that the on-hand supply amount stored at each node and at each time period is less than the node's storage capacity.
4. Constraint (3.5) creates an upperbound for the amount of flow emanating from each node at each time period. This constraint ensures that the total outgoing flow from each node does not exceed the node's ability to push fuel out in one time period.
5. Constraint (3.6) ensures the correct amount of demand is satisfied at each land node. This is accomplished through $superSink$ bookkeeping that requires that the total flow going to the $superSink$ node from each land node does not exceed the node's demand.
6. Constraint (3.7) defines the amount of storage at a node n in a given period in terms of the previous storage value and the flow into and out of the node.
7. Constraint (3.8) ensures that there is not excess fuel stored at any land node at the end of the time period of interest. All fuel delivered ashore should be consumed and all storage of fuel resides at the seabase via the connector nodes s.
8. Constraint (3.9) has a similar aim to constraint (3.8) in that it ensures that the total fuel sent ashore across all supply nodes does not exceed the requirement.
9. Constraints (3.10) and (3.11) ensure the decision variable values are nonnegative.

The output from this model that we concern ourselves with is the sea-to-land flow: $X_{s,j,t}$ for $s \in S$. Specifically, we want the fuel flow from each connector type s to each land node j over the course of the day $\sum_{t \in T} X_{s,j,t}$. We translate this value into number of runs by dividing this total flow over the capacity of a connector type. Thus we now have the total number of runs required by connector type that we can assign to specific ships and beaches or LZs in the form of a schedule.

Our model is based on the evacuation models of Hamacher and Tjandra [3]. There are two main distinctions with our approach. In the Hamacher and Tjandra model the goal is to maximize the number of people evacuated; it does not matter where the evacuees end up. In our model this is equivalent to maximizing the total flow of fuel ashore without regard to how the fuel is allocated across the demand nodes. In our model we want individual demand nodes to receive exactly their fuel demand; no more, no less. We built Constraint (3.6) to ensure demand is properly satisfied at each node. Second, we have both a supply and a demand that will not match up (demand must not exceed supply), whereas Hamacher and Tjandra simply consider the number of people to evacuate, so the supply implicitly equals demand. This results in extra bookkeeping constraints such as Constraint (3.9) to

ensure the supply sent ashore does not exceed the demand ashore.

A downside of this formulation is we have to solve the model repeatedly until all demand is satisfied. We could choose a large time window and maximize a discounted sum of demand satisfied by time period. An issue with this formulation is we need to choose a large enough time window such that all the flow originating from the supply nodes leaves the seabase during this time window. We leave this for future work.

3.2 Assignment Heuristic

The Assignment Heuristic described in this section and the Scheduler Linear Program described in Section 3.3 only apply to one connector type at a time. We have to re-run these two models separately for each connector type to produce the final schedule for each connector. The Quickest Flow model in Section 3.1 provides the total number of runs from the seabase to each land node by connector type. The next step is to allocate those runs across the ships to create a schedule. This requires us to expand our notion of runs. We can define runs in terms of the order in which connectors *depart* the ship or *return* to the ship. For example, the first run from the LHD might be the third run to *return* to the LHD. We can also define runs from the perspective of arrivals to a beach or LZ. For multiple ships and multiple beaches and/or LZs we need to associate ship runs with beach runs. For example, if the first run from the LHD and the first run from the LSD both are going to Beach A, then which arrives to Beach A first? Does the first run of Beach A correspond to the first LHD run or the first LSD run? This situation requires an assignment to create the proper sequencing of connectors to beach or land nodes. To fully formulate this assignment problem of ship runs to beach runs would require assignment variables and integer programing. For completeness we present the integer program formulation in the Appendix. Even for small examples, the integer program can take hours to solve (if it solves at all). Possible follow-on research could focus on improving the integer program formulation and performance.

To solve the task of assigning each departing ship run to each beach run, and then assigning each beach run back to a return ship run, we develop a heuristic vice a binary integer program. The algorithm essentially tracks what each connector is doing at any given moment. The algorithm focuses on each connector and tracks it as it moves back and forth between the

ship and land location. The algorithm processes events chronologically associated with the connectors. These events include loading at the ship, transiting to land, unloading on land, and transiting back to the ship. After processing a specific event for a connector, the connector is associated with an updated time and event, and the algorithm continues by determining the connector with the next event to process. The algorithm continues until all runs at all beaches/LZs have been assigned. We describe the algorithm in this section and provide pseudocode in the Appendix.

The algorithm also tracks pertinent information with respect to each beach/LZ and ship in addition to the connector and stores the information in objects. The algorithm then uses this information to identify and assign which "event" should be scheduled next with respect to the connector. The three variables—namely, connector, beach, and ship—are discussed below.

Connector. The heuristic tracks each connector's current status. The statuses are generally related to components of the current runs, such as loading the ship, unloading at the beach/LZ, and transiting. For each connector the algorithm tracks its current status, the time associated with the status, the connector ID and its associated ship ID, the current departure run, the beach ID and the beach run associated with the current run, and the return ship run. There are 4 statuses, described in detail below.

1. Waiting to load at ship: The time associated with this status is the time the connector returns to the ship. At this point the connector is ready and waiting to load. The connector is able to load whenever the next loading spot on that ship is available, which is tracked via the Ship object. We define a "spot" as a position where the connector can load or unload. If the number of spots is less than the number of connectors, then not all connectors can load simultaneously. To transition to Status 2, the connector's time updates to when the connector finishes loading.
2. Ready for assignment from ship to beach: The time associated with this status is when the connector has finished loading and is prepared to transit to the beach or LZ (essentially the result from Status 1). At this point the assignment of the connector to a beach/LZ takes place. This requires an assignment policy that sends a connector to a beach/LZ, given the current situation. For example, one assignment policy might send the connector to the beach/LZ that has the largest number of remaining runs.

The assignment policy is a critical part of this algorithm, and we defer additional discussion to (Section 3.3.1). For now we take the assignment as given, and we transition from Task 2 to Task 3 by updating the time to arrival of the connector to the assigned beach.

3. Waiting to unload at the Beach/LZ: This is the time the connector arrives at the beach from Status 2. At that moment, the connector is waiting to unload its cargo. The connector will unload once a spot is available. We track when the next spot to unload becomes available in the Beach/LZ object. After unloading, the connector transitions to Status 4, with an updated time at the completion of unloading.
4. Return to ship: The time associated with this status is when the connector departs the beach immediately after unloading. At this point the connector cycles back to Status 1, with the corresponding time being the time when the connector arrives back to the ship. As the connector finishes Status 4 and transitions to Status 1, it has completed an entire run. The algorithm saves the information about this just-completed run before updating the connector information for the next run starting with Status 1.

While the algorithm primarily tracks the connectors making runs back-and-forth between ships and beaches, it also tracks some auxiliary information about each ship and beach. This auxiliary information is useful in tracking the total number of runs and assigning connectors to beaches. A description of the Beach and Ship objects is described as follows:

Beach. For each beach, we track the total number of runs initiated and completed. The difference between the two provides a measure of congestion at the beach and can be used in the assignment policy. The algorithm also tracks when each beach spot is next available for a new connector to begin unloading.

Ship. For each ship, we tabulate the number of runs initiated and completed. We also track when each spot for loading on the ship is next available for a returning connector. This is important in determining when the connectors can begin reloading. For this model, connectors always return to the same ship, but can be scheduled to different beaches or LZs over multiple runs.

3.2.1 Assignment Policies

The beach assignment component introduced in Status 2 is the key aspect of the heuristic and the overall program. The algorithm uses the following assignment policies:

1. Largest remaining run deficit: This policy tracks the run deficit of each land node and assigns the connector to the node with the largest deficit. For example, this policy would assign the next LCAC run to the beach node with the largest remaining number of connector runs required.
2. Shortest round trip: This policy focuses on the time it takes based on speed and distance for a connector to complete a run. This assigns runs based on how quickly the run can be accomplished.
3. Minimize congestion at each land node: This policy tracks how much congestion is currently at each beach/LZ and then assigns the next connector to the beach/LZ with the smallest congestion.
4. Random policy: The policy chooses uniformly at random among the beach nodes that still have remaining runs.

The default policy is the shortest-round trip. However, the heuristic can run each of these policies separately and then choose the best one. The shortest round trip policy has the added benefit of accounting for potential waiting due to congestion by considering the number of connectors already at the beach or in transit, which is tracked by the Beach object. The algorithm saves information about each run and builds a preliminary schedule once all the runs have been assigned. This preliminary schedule is then used by the Scheduling Linear Program (see Section 3.3) to produce the final minute-by-minute schedule.

Each run can be defined in one of three ways: departure ship run, beach run, and returning ship run. The algorithm tracks these three definitions for each run. We next walk through a simple example to demonstrate our heuristic. For more details, see the pseudocode in the Appendix. For this example, we assume the following:

- Two ships (A and B) with 1 available spot to load on each ship
- 3 connectors all of the same type
- 2 beaches (X and Y) with 1 available spot to unload on each beach

Ship	Connector ID	Status	Time	Departure Run	Return Run	Beach ID	Beach Run
A	A1	1	0	–	–	–	–
A	A2	1	0	–	–	–	–
B	B1	1	0	–	–	–	–

Table 3.2. Starting Conditions

A1 and B1 are loaded to start with moving to Status 2, A2 has to wait for A1 to depart so a spot opens for A2 to start loading. We assume 30 minutes to load on Ship A, 50 minutes on ship B.

Ship	Connector ID	Status	Time	Departure Run	Return Run	Beach ID	Beach Run
A	A1	2	30	–	–	–	–
A	A2	1	0	–	–	–	–
B	B1	2	50	–	–	–	–

Table 3.3. After first connector has loaded on each ship

A2 next loads (starting at time 30)

Ship	Connector ID	Status	Time	Departure Run	Return Run	Beach ID	Beach Run
A	A1	2	30	–	–	–	–
A	A2	2	60	–	–	–	–
B	B1	2	50	–	–	–	–

Table 3.4. All connectors now loaded on ships

A1 is assigned to beach X, then B1 assigned beach Y, then A2 assigned beach Y. Time updates to arrival time to beach. It takes 50 minutes from A to X, 40 minutes from A to Y, and 60 minutes from B to Y.

Ship	Connector ID	Status	Time	Departure Run	Return Run	Beach ID	Beach Run
A	A1	3	80	1	–	X	–
A	A2	3	100	2	–	Y	–
B	B1	3	110	1	–	Y	–

Table 3.5. All connectors arrive at beaches

A1 arrives first to X and A2 arrives first to Y (even though A2 left after B1). A1 and A2 unload and then are ready to head back (Status 4). B1 has to wait for A2 to unload. We assume 35 minutes to unload at X and 20 minutes to unload at Y.

Ship	Connector ID	Task	Time	Departure Run	Return Run	Beach ID	Beach Run
A	A1	4	115	1	–	X	1
A	A2	4	120	2	–	Y	1
B	B1	3	110	1	–	Y	–

Table 3.6. First connectors unload at beaches

B1 unloads (120-140) and A1 and A2 arrive back to the seabase.

Ship	Connector ID	Task	Time	Departure Run	Return Run	Beach ID	Beach Run
A	A1	1	165	1	–	X	1
A	A2	1	160	2	–	Y	1
B	B1	4	140	1	–	Y	2

Table 3.7. A1 and A2 finish runs, B1 finish unloading at beach

B1 returns and all three have finished one run. The heuristic saves this information and then processes the next run for each connector. Note that none of the three types of runs (ship departure run, ship return run, beach run) in Table 3.8 are the same for any connector.

Ship	Connector ID	Task	Time	Departure Run	Return Run	Beach ID	Beach Run
A	A1	1	165	1	2	X	1
A	A2	1	160	2	1	Y	1
B	B1	1	200	1	1	Y	2

Table 3.8. All three connectors complete first run

3.3 Scheduler Linear Program

This section covers the third and final model used to generate the amphibious schedule. As with the heuristic in Section 3.2, we run this scheduler separately for each connector type. This scheduling program uses parameter inputs provided by the planner and the preliminary schedule assignment produced by the assignment heuristic in Section 3.2. This program produces an optimized, minute-by-minute schedule of the runs from the ships to each land node (beach or landing zone), by connector type. Specifically, the model minimizes the average completion time among all runs incorporating the six time points listed below:

1. Load at ship
2. Depart ship
3. Arrive at beach/LZ

4. Unload at beach/LZ
5. Depart beach/LZ
6. Arrive back to the ship

These six time points are related by the following rules:

1. Connector cannot start loading until spot is available.
2. Connector cannot depart ship until it is loaded.
3. Connector cannot arrive at the land node until after it departs ship and transits.
4. Connector cannot unload its cargo until after it arrives at the land node and there is a spot available.
5. Connector cannot depart land node until after it has unloaded its cargo.
6. A connector cannot return to the ship until after it has completed its transit back to the ship.

The linear program uses four main parameters produced by the two previous models in addition to parameters supplied by the planner.

- $runsBeach$: The number of runs to each beach by connector type.
- $runsShip$: For each ship, the number of runs sent out for each connector type.
- a: A binary parameter that associates a departure run of a ship with a beach run. This can be written as $a_{i,j,k,m} = 1$ if the jth departure run of ship i corresponds to the mth run of beach k. Otherwise $a_{i,j,k,m} = 0$.
- c: A binary parameter that associates a beach run with a returning run of a ship. Written as $c_{k,m,i,j} = 1$ if the mth run of beach k corresponds to the jth returning run of ship i. Otherwise $c_{k,m,i,j} = 0$.

The first parameter $runsBeach$ comes directly from the Quickest Flow linear program, while the other three are products of the Assignment Heuristic. The parameters a and c are critical components of the overall model as they serve as functional replacements for assignment decision variables of a binary mixed integer linear program. To illustrate a and c, consider the example in Section 3.2.1:

$$a_{A,1,X,1} = a_{A,2,Y,1} = a_{B,1,Y,2} = 1, \qquad c_{X,1,A,2} = c_{Y,1,A,1} = c_{Y,2,B,1} = 1.$$

We provide additional examples of these binary variables in Chapter 4.

The six time points define the critical events for an amphibious offload schedule and are key components to the applicability of this program. The overall goal of this model is to minimize the average finish time across all runs of all ships. The formulation for the Scheduler LP is as follows.

Indices and Sets

$s \in S$	ships
$b \in B$	land nodes that can be directly reached by this connector type
$i \in I$	6 time points for each run
$rs \in Rx_s$	set of runs from each ship
$rb \in Ry_b$	set of runs from each land node

Data [units]

$numConn_s$	number of connectors per ship s
$runsShip_s$	total number of runs from each ship s
$runsBeach_b$	total number of runs to each land node b
$spotsShip_s$	total number of welldeck or fligthdeck spots for each ship s
$spotsBeach_b$	total number of landing spots at each land node b
$transTime_{s,b}$	transit time from ship s to land node b [minutes]
$loadTime_s$	time to load connector on ship s [minuets]
$unloadTime_b$	time to unload connector at land node b [minutes]
$dayLength$	day length of welldeck/flight crew day [minutes]
$oWSlack$	(on water slack) slack time value to limit time connectors are vulnerable waiting to unload.
$a_{s,rs,b,rb}$	Binary parameter that associate ship departure runs with beach runs: $a_{s,rs,b,rb} = 1$ if the rs departure run of ship s corresponds to the rb run of land node b
$c_{b,rb,s,rs}$	Binary parameter that associate beach runs with ship arrival runs : $c_{b,rb,s,rs} = 1$ if the rb run of land node b corresponds to the rs return run of ship s

Decision Variables [units]

$X_{i,s,rs}$	the time of the ith event of the rs departing run from ship s
$Y_{i,b,rb}$	the time of the ith event of the rb run of beach b
$Z_{i,s,rs}$	the time of the ith event of the rs returning run to ship s
$welldeckStart_s$	time to start welldeck/flight deck on ship s
$welldeckEnd_s$	time to end welldeck/flight deck on ship s

Formulation

$$\min_{\substack{X,Y,Z,\\ welldeckStart,\\ welldeckEnd}} \quad \sum_{s\in S}\sum_{rs\in Rx_s} X_{6,s,rs} \tag{3.12}$$

s.t.
$$welldeckStart_s \le X_{i,s,rs} \le welldeckEnd_s \qquad \forall i \in I, \forall s \in S, \forall rs \in Rx_s \tag{3.13}$$
$$welldeckEnd_s = welldeckStart_s + dayLength \qquad \forall s \in S \tag{3.14}$$
$$X_{2,s,rs} \ge X_{1,s,rs} + loadTime_s \qquad \forall s \in S, \forall rs \in Rx_s \tag{3.15}$$
$$X_{3,s,rs} \ge X_{2,s,rs} + \sum_{b\in B}\sum_{rb\in Ry_b} a_{s,rs,b,rb} transTime_{s,b} \qquad \forall s \in S, \forall rs \in Rx_s \tag{3.16}$$
$$Y_{4,b,rb} \ge Y_{3,b,rb} \qquad \forall b \in B, \forall rb \in Ry_b \tag{3.17}$$
$$Y_{5,b,rb} \ge Y_{4,b,rb} + unloadTime_b \qquad \forall b \in B, \forall rb \in Ry_b \tag{3.18}$$
$$Y_{6,b,rb} \ge Y_{5,b,rb} + \sum_{s\in S}\sum_{rb\in Ry_b} a_{s,rs,b,rb} transTime_{s,b} \qquad \forall b \in B, \forall rb \in Ry_b \tag{3.19}$$
$$X_{4,s,rs} - X_{2,s,rs} \le onWSlack \times \sum_{b\in B}\sum_{rb\in Ry_b} a_{s,rs,b,rb} transTime_{s,b} \qquad \forall s \in S, \forall rs \in Rx_s \tag{3.20}$$
$$X_{i,s,rs} \ge X_{i-1,s,rs} \qquad \forall i \in I, \forall s \in S, \forall rs \in Rx_s \tag{3.21}$$
$$Z_{i,s,rs} \ge Z_{i-1,s,rs} \qquad \forall i \in I, \forall s \in S, \forall rs \in Rx_s \tag{3.22}$$
$$Y_{i,b,rb} \ge Y_{i-1,b,rb} \qquad \forall i \in I, \forall b \in B, \forall rb \in Ry_b \tag{3.23}$$
$$X_{1,s,rs} \ge X_{2,s,rs-spotsShip_s} \qquad \forall s \in S, \forall rs \in Rx_s \tag{3.24}$$
$$Y_{4,b,rb} \ge Y_{5,b,rb-spotsBeach_b} \qquad \forall b \in B, \forall rb \in Ry_b \tag{3.25}$$
$$X_{1,s,rs} \ge Z_{6,s,rs-numConn_s} \qquad \forall s \in S, \forall rs \in Rx_s \tag{3.26}$$
$$X_{2,s,rs} \ge X_{2,s,rs-1} \qquad \forall s \in S, \forall rs \in Rx_s \tag{3.27}$$
$$Z_{6,s,rs} \ge Z_{6,s,rs-1} \qquad \forall s \in S, \forall rs \in Rx_s \tag{3.28}$$
$$Y_{4,b,rb} \ge Y_{4,b,rb-1} \qquad \forall b \in B, \forall rb \in Ry_b \tag{3.29}$$
$$Y_{i,b,rb} = \sum_{s\in S}\sum_{rs\in Rx_s} a_{s,rs,b,rb} X_{i,s,rs} \qquad \forall i \in I, \forall b \in B, \forall rb \in Ry_b \tag{3.30}$$

$$Z_{i,s,rs} = \sum_{b \in B} \sum_{b \in Ry_b} c_{b,rb,s,rs} Y_{i,b,rb} \qquad \forall i \in I, \forall s \in S, \forall rs \in Rx_s \quad (3.31)$$

$$X_{i,s,rs} \geq 0 \qquad \forall i \in I, \forall s \in S, \forall rs \in Rx_s \quad (3.32)$$

$$Y_{i,b,rb} \geq 0 \qquad \forall i \in I, \forall b \in B, \forall rb \in Ry_b \quad (3.33)$$

$$Z_{i,s,rs} \geq 0 \qquad \forall i \in I, \forall s \in S, \forall rs \in Rx_s \quad (3.34)$$

$$welldeckStart_s \geq 0 \qquad \forall s \in S \quad (3.35)$$

$$welldeckEnd_s \geq 0 \qquad \forall s \in S \quad (3.36)$$

The objective function for this model is a minimization of the average time required to complete all runs across all ships, as $X_{6,s,rs}$ specifies the completion time of the run. It is straightforward to consider other modifications, such as minimizing the final completion run. Most of the constraints associated with the Scheduler Linear Program ensure the correct sequencing of ship-to-shore events with respect to the six time points of each connector for each run. We explain these constraints below:

1. Constraints (3.13) and (3.14) ensures that all runs occur within the defined welldeck or flightdeck time window.
2. The next set of constraints ensure the proper ordering of the six time points that define the sequence of events for each unique connector run to the land node.
   (a) Constraint (3.15) ensures that each connector does not depart the ship until after it is loaded.
   (b) Constraint (3.16) ensures that each connector does not arrive at its assigned destination until after it has departed the ship and transited.
   (c) Constraint (3.17) ensures that each connector does not unload its cargo until after it has arrived at its destination.
   (d) Constraint (3.18) ensures that each connector does not depart the beach/land node until after it has offloaded its cargo.
   (e) Constraint (3.19) ensures that each connector cannot return to the ship until after it has departed the beach and transited.
3. Constraint (3.20) limits the amount of risk associated with connectors sitting idle off of the coast, and thus minimizes inefficient congestion. If the slack parameter is 1, then that means the commander will not tolerate any idling; increasing it beyond 1 allows some tolerance for idling and congestion off the beach as the connectors wait

to unload.

4. Constraints (3.21), (3.22), and (3.23) ensure that all six time points maintain the proper ordering for each of the decision variables X, Y, Z.
5. The next subset of constraints connects the various runs to create a fluid and deconflicted schedule.
   (a) Constraint (3.24) dictates that a connector cannot load on a ship until there is a spot available on the ship.
   (b) Constraint (3.25) is similar to Constraint (3.24), but with respect to beach or landing zone spots. This constraint ensures that connectors cannot occupy beach spots to unload until one is available.
   (c) Finally, Constraint (3.26) connects specific-type connectors to each other to ensure proper flow and deconfliction of schedule events. More specifically, the constraint mandates that a connector cannot start loading until the connector has returned from the previous run.
6. Constraints (3.27), (3.28), and (3.29) mandate that runs need to go in order.
7. Constraint (3.30) ensures that each departure run from the ship corresponds with exactly one run from the beach. Each Y connects with exactly one X.
8. Finally, Constraint (3.31) associates runs on each beach with the return runs to the ships. Each Z connects to exactly one Y.
9. Constraints (3.32), (3.33), (3.34), (3.35), and (3.36) ensure the decision variables are nonnegative.

In Appendix 1 we formulate the integer program version of this model. The key difference is the binary assignment variables a and c are decision variables in the integer program, not fixed parameters as they are for the above formulation. In Chapter 4 we provide some comparisons of the optimal schedule produced by the integer program and our heuristics described in Section 3.2–3.3. However, we primarily focus on the heuristic approach presented in this chapter because the integer program becomes computationally intractable for larger problems.

CHAPTER 4:
Demonstration and Results

We use MACS to solve several realistic scenarios. For each scenario, we found an operationally feasible, executable, and complete amphibious schedule to satisfy the fuel demand ashore. We implement MACS in Python. The implementation includes the three models presented in Chapter 3, handling of input and output, and data parsing and massaging. We use Pyomo with a CPLEX solver for the two linear programs: Quickest Flow and Scheduler. Pyomo is an optimization modeling language based in Python [31], [32]. The initial planning parameters are inputted into MACS via six Comma Separated Values (CSV) files. Figure 4.1 graphically depicts the general progression of the model.

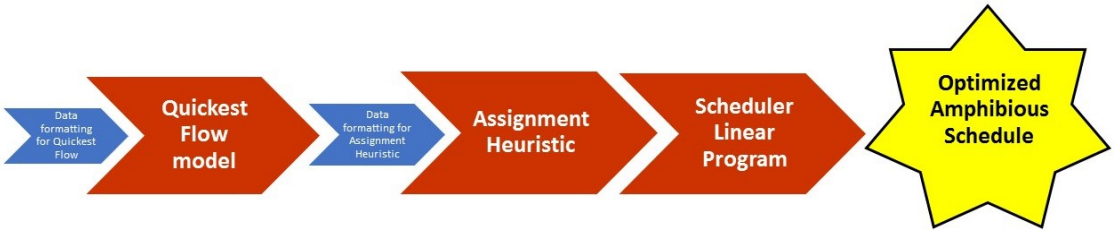


Figure 4.1. Overall Model Components

In the next several sections, we will describe how the three models described in Chapter 3 connect from initial inputs to final schedule output. To aid in this explanation, we will refer to a concrete model, which is illustrated in Figure 4.2 .

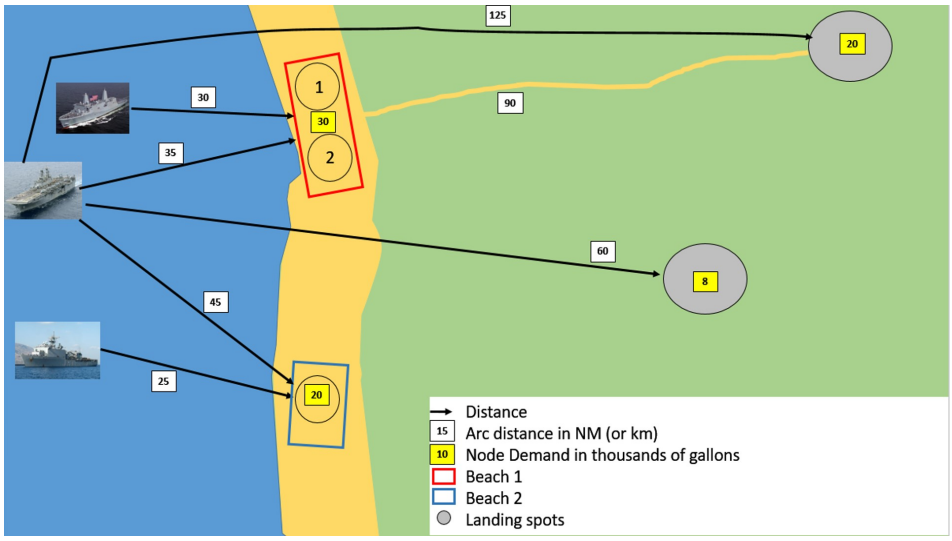


Figure 4.2. Scenario Used for Demonstration and Results

This scenario has the three standard ships of the ARG (reference Section 2.1.1); Two beaches: Beach 1 and Beach 2 with Beach 1 having two unloading spots; Two landing zones represented by the gray circles: LZ1 and LZ2 with a road that connects Beach 1 to LZ1, LZ2 is only accessible via aircraft.

4.1 Initial Planning inputs

The user inputs required by the model are reasonable planning metrics that any amphibious planner would use to formulate a landing plan. These parameters mainly relate to distances, fuel demand, supply, and connectors available for the mission. A detailed explanation of the input parameters (contained within CSV files) appears below.

4.1.1 Connector Information

The connector information required includes the connector type, capacity to carry fuel as cargo, classification, and average transit speed. These factors can be changed based on factors such as operational necessity, weather, etc. It is important to reiterate that even though we are solely concerned about surface/air connectors from the seabase to the shore, we need land connector information (e.g., trucks such as MTVRs) for the Quickest Flow model because the land connectors impact how quickly we can push fuel to the more inland locations. See Figure 4.3 for an example.

	A	B	C	D
1	type	category	capacity	velocity
2	LCAC	surface	4	30
3	LCU	surface	8	10
4	CH53	air	2	120
5	MV22	air	2	200
6	MTVR1	land	2	20
7	MTVR2	land	2	20

Figure 4.3. Example of Connector-specific Inputs.

4.1.2 Seabase Information

Figure 4.4 contains the supply side planning parameters. Each row in the CSV corresponds to a different ship in the seabase. The CSV contains the number of connectors by type, the number of loading spots for each connector type, and the time it takes to load each connector type with fuel. We assume the seabase has an unlimited amount of fuel available to transport ashore. In reality the ships in the seabase will themselves need to be replenished with fuel periodically.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	id	LCAC	LCU	CH53	MV22	LCACSpots	LCUSpots	CH53Spots	MV22Spots	LCACLoad	LCULoad	CH53Load	MV22Load
2	LHD	3	0	2	2	1	0	2	2	30	80	20	20
3	LPD	2	0	0	0	1	0	0	0	30	80	20	20
4	LSD	0	2	0	0	0	1	0	0	40	40	20	20

Figure 4.4. Seabase Planning Inputs

4.1.3 Land Node Inputs

The planning inputs in Figures 4.5 and 4.6 define the land network of the scenario. Figure 4.5 presents information about each node in the land network. This node information includes the fuel demand, capacity to store fuel, and number of land connectors available. This CSV also contains the number of unloading spots and time to unload for each type of surface and air connector. This information specifies whether the particular land node is accessible to a type of surface or air connector.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	id	demand	capacity	MTVR1	MTVR2	LCACSpots	LCUSpots	CH53Spots	MV22Spots	LCACUnload	LCUUnload	CH53Unload	MV22Unload
2	B1	30	100	2	2	2	0	0	0	25	0	25	25
3	B2	20	100	0	0	1	1	0	0	25	40	25	25
4	LZ1	20	100	2	2	0	0	2	2	0	0	25	25
5	LZ2	8	100	0	0	0	0	2	2	0	0	25	25

Figure 4.5. Land Node Planning Requirements

4.1.4 Land Arc Inputs

Figure 4.6 contains the arc information for the land network; we assume the network is a directed graph. The land connectors travel on these land arcs delivering fuel between land nodes.

	A	B	C
1	start	end	distance
2	B1	LZ1	90
3	LZ1	B1	90

Figure 4.6. Inputs for Ground Transport Capacity Ashore

4.1.5 Distance Planning Inputs

To connect the supply nodes to the demand nodes, we need the distances from each of the ships to each of the demand nodes. Figure 4.7 presents this information.

	A	B	C	D	E
1		B1	B2	LZ1	LZ2
2	LHD	35	45	125	60
3	LPD	30	40	120	55
4	LSD	40	25	110	45

Figure 4.7. Distance Planning Inputs

4.1.6 General Parameters

The final input CSV contains a few general parameters that do not naturally fit within the previous categories. The *dayLength* parameter defines the length of time during the day

that the welldeck or flightdeck can operate. The Quickest Flow model discretizes time, and *timeStepsPerHour* specifies the number of time periods in an hour; the default is a 15 minute time period. Finally, in Section 3.2 we list several possible policies that could be used in the Assignment Heuristic. In MACS we try several policies and choose the best one. The *numSchedulesToTest* parameter specifies how many policies to consider.

	A	B
1	dayLength	10
2	timeStepsPerHour	4
3	numSchedulesToTest	6

Figure 4.8. Time-specific Inputs.

4.2 Quickest Flow Implementation

MACS first executes the Quickest Flow model described in Chapter 3. This model provides us with the number of runs by connector type (e.g., 5 runs of LCACs to Beach 1 and 10 runs of MV-22 to LZ1). The main inputs to this model are the planning parameters outlined in Section 4.1.

Preprocessing Inputs required by the Quickest Flow Algorithm

To run Quickest Flow we need all the data defined in the formulation in Chapter 3.1. Some of that data comes directly from the CSV inputs presented in Section 4.1. For other parameters, we need to perform some additional preprocessing. We describe below how we determine each of the Quickest Flow parameters.

- $maxTime$: We vary this parameter iteratively until the objective function equals total demand: $\sum_{l \in L} demand_l$
- u_n: This is the amount of fuel we can push out of a node each time period. We define u_n for each node $n \in N$, but recall that N is the union of all supply nodes S and land nodes L. We define u_n separately based on whether n is in S or L. For each supply node $n \in S$, $u_n = \frac{supply_s}{maxTime}$. For each demand node $n \in L$, we compute the round-trip time along each outgoing arc for each land connector type using the information in

Figures 4.3 and 4.6. Combining this round-trip time with land connector capacity in Figure 4.3 and number of land connectors in Figure 4.5 produces a rate per time-period that each land connector type can push fuel along one particular arc. Summing over all types of land connectors produces the rate at which we can push fuel along one particular arc, if all land connectors at the node of interest just travel on that one arc. We then average over all outgoing arcs to generate the final value of u_n for $n \in L$.

- $tau_{i,j}$: the time to travel along an arc. For a land-to-land arc, the distance along an arc comes directly from Figure 4.6. Combining the distance with velocity for each type of land connector (see Figure 4.3) produces a travel time along the arc by land connector type. We weight these travel times by the number of land connectors of each type at the originating node (see Figure 4.5) to determine the final value of $tau_{i,j}$. In general $tau_{i,j} \neq tau_{j,i}$ because $tau_{i,j}$ depends upon the allocation of land connectors at node i and $tau_{j,i}$ depends upon the allocation of land connectors at node j
  For a supply-to-land arc, we have the distance from each ship to the land node from Figure 4.7, and we have the velocity for the particular supply-type (i.e., connector) from Figure 4.3. This information produces the travel time from each ship to the land node. We then weight these travel times by the number of connectors of interest at each ship (Figure 4.4) to compute the final value of $tau_{i,j}$.
- $supply_s$: This is an estimate of total supply we can push ashore during the day by connector type. For each ship-to-land connection, we have an estimate of the total round trip time for the particular connector type using the information from Figures 4.3, 4.4, 4.5, and 4.7. The round-trip time includes transit, loading and unloading. Combining round-trip time with connector capacity (Figure 4.3) and number of connectors per ship (Figure 4.4) produces an estimate for the total amount of fuel we can push to one land node during the day if all connectors run non-stop from the seabase to that one land node. The final value of $supply_s$ is the maximum over all accessible land nodes.
- $nodeCap_l$: This parameter comes directly from Figure 4.5
- $demand_l$: This parameter comes directly from Figure 4.5

The main outputs from this model are the amount of flow by connector to each location, and more importantly, the number of runs by connector type to satisfy the flow demand. MACS executes the Quickest Flow model repeatedly, changing the $maxTime$ parameter, until all

demand ashore is satisfied. Table 4.1 presents the results of the Quickest Flow model for our baseline scenario presented in Figure 4.2:

Line	Connector Type	Land Node	Number of Runs
1	MV-22	LZ1	7
2	MV-22	LZ2	3
3	CH-53	LZ1	4
4	CH-53	LZ2	2
5	LCAC	B1	8
6	LCAC	B2	4
7	LCU	B2	1

Table 4.1. Output from Quickest Flow Model for Scenario outlined in Figure 4.2

As an example we inspect Line 5 from Table 4.1 and see that the model indicates that we need eight LCAC runs to B1 (Beach 1) as part of the optimized fuel flow plan. The results in Table 4.1 are critical inputs to the Assignment Heuristic.

4.3 Assignment Heuristic Implementation and Key Outputs

As indicated in Chapter 3 the Assignment Heuristic allocates the runs across the ships to create an initial schedule. More specifically, it assigns each departing ship run to a beach run, and then assigns each beach run back to a returning ship run . The heuristic produces the critical assignment parameters $a_{s,rs,b,rb}$ and $c_{b,rb,s,rs}$ that replace binary decision variables of a mixed integer linear program. Recall that $a_{s,rs,b,rb} = 1$ if the rs departure run of ship s corresponds to the rb run of beach b, and $c_{b,rb,s,rs} = 1$ if the rb run of beach b corresponds to the rs returning run of ship s. For more details on these binary parameters refer to the Scheduler Linear Program model in Chapter 3. Below we present the assignment parameters produced by the heuristic for the LCAC for the scenario in Figure 4.2.

The table below shows the non-zero values of $a_{s,rs,b,rb}$ for the LCAC requirements for this scenario.

For example, Table 4.2 indicates that the 5th departure run from Ship 1 corresponds to the 1st beach run from Beach 2. In this particular case both the LHD and LPD send their first

Ship ID	Ship Run	Beach ID	Beach Run
1	1	1	2
1	2	1	4
1	3	1	5
1	4	1	8
1	5	2	1
1	6	2	2
2	1	1	1
2	2	1	3
2	3	1	6
2	4	1	7
2	5	2	3
2	6	2	4

Table 4.2. Assignment Parameter $a_{s,rs,b,rb}$ that tracks the departure run of a ship-to-beach run for the LCAC for Scenario 4.2

four runs to Beach 1 and their last two runs to Beach 2. We next present the non-zero values of $c_{b,rb,s,rs}$ for the LCAC.

Beach ID	Beach Run	Ship ID	Ship Run
1	1	2	1
1	2	1	1
1	3	2	2
1	4	1	2
1	5	1	3
1	6	2	3
1	7	2	4
1	8	1	4
2	1	1	5
2	2	1	6
2	3	2	5
2	4	2	6

Table 4.3. Assignment Parameter $c_{b,rb,s,rs}$ that tracks the departure run of a beach-to-ship run for the LCAC for Scenario 4.2

We see in the above two tables that the heuristic assigns the two variables so that each ship departure and return run maps to only one beach run.

4.4 Scheduler Linear Program and Final Schedule

The final stage of MACS is the Scheduler Linear Program. Recall that this model produces the optimized amphibious schedule incorporating the six time points outlined in Section 3.3. The output will be an amphibious delivery schedule for each connector deconflicted across all ships, beaches, and connectors. Before presenting the final schedule, we discuss the inputs to the Scheduler Linear Program.

Preprocessing Inputs required by the Scheduler Linear Program Algorithm

To run the Scheduler Linear Program requires specifying the data in the formulation in Chapter 3.3. Most of those parameters come directly from the CSV inputs in Figures 4.3–4.8. However, the following four parameters require some additional preprocessing:

- $runsShip_s$: The number of runs for each connector type for each ship in the ARG. This is a direct output of the Assignment Heuristic. This information can be obtained from Table 4.2.
- $runsBeach_b$: The number of runs for each connector type to each beach/landing zone. This is a direct output of the Quickest Flow model. See Table 4.1.
- The variable a: Assignment parameter generated by the Assignment Heuristic. See Table 4.2.
- The variable c: Assignment parameter generated by the Assignment Heuristic. See Table 4.3.

We now present the final schedule. For the scenario in Figure 4.2, MACS took less than 40 seconds to run all three models and generate the schedule for all four connector types. We ran MACS on a Lenovo Core i5 laptop with 8 gigabytes of RAM. The schedules produces by this model are feasible, executable, and efficient given the initial planning parameters.

4.4.1 LCAC Schedule

Ship	Run	Destination	Start Load	Depart Ship	Arrive at Beach	Start Unload	Depart Beach	Arrive at Ship
LHD	1	B1	0.0	30.0	100.0	100.0	125.0	195.0
LHD	2	B1	30.0	60.0	130.0	130.0	155.0	225.0
LHD	3	B1	60.0	90.0	160.0	160.0	185.0	255.0
LHD	4	B1	195.0	225.0	295.0	295.0	320.0	390.0
LHD	5	B2	225.0	255.0	345.0	345.0	370.0	460.0
LHD	6	B2	255.0	285.0	375.0	375.0	400.0	490.0
LPD	1	B1	0.0	30.0	90.0	90.0	115.0	175.0
LPD	2	B1	30.0	60.0	120.0	120.0	145.0	205.0
LPD	3	B1	175.0	205.0	265.0	265.0	290.0	350.0
LPD	4	B1	205.0	235.0	295.0	295.0	320.0	380.0
LPD	5	B2	350.0	380.0	460.0	460.0	485.0	565.0
LPD	6	B2	380.0	410.0	490.0	490.0	515.0	595.0

Table 4.4. Final LCAC Amphibious Fuel Delivery Schedule

For this scenario we used a total of five LCACs (three on the LHD and 2 on the LPD). With respect to the LCAC schedule there are a few points to highlight: First, the *Arrive at Beach* and *Start Unload* times are identical meaning that there is no waiting for a landing spot. Second, when comparing the *Arrive at Beach* and the *Depart Beach* times, we see that there will never be more than two connectors (number of spots) on Beach 1 at one time. Similarly there will never be more than one connector at Beach 2 at one time. Finally, when comparing the *Arrive at Ship* column and *Start Load*, we see that a connector cannot start loading for its next run until it returns to the ship. This schedule is unique from the other three in that it includes multiple ships to multiple beaches. Due to the common disposition of connector types to different ARG ships (e.g. MV-22s and CH-53s on LHD), they represent schedules comprised of one ship and multiple beaches/LZs. In this example both ships send all LCACs only to Beach 1 until all eight of its runs are satisfied, and then the operation concludes by sending all runs to Beach 2.

4.4.2 LCU Schedule

Ship	Run	Destination	Start Load	Depart Ship	Arrive at Beach	Start Unload	Depart Beach	Arrive at ship
LSD	1	B2	0.0	40.0	190.0	190.0	230.0	380.0

Table 4.5. Final LCU Amphibious Fuel Delivery Schedule

4.4.3 MV-22 Schedule

Ship	Run	Destination	Start Load	Depart Ship	Arrive at LZ	Start Unload	Depart LZ	Arrive at Ship
LHD	1	LZ1	0.0	20.0	57.5	57.5	82.5	120.0
LHD	2	LZ2	0.0	20.0	38.0	38.0	63.0	81.0
LHD	3	LZ2	81.0	101.0	119.0	119.0	144.0	162.0
LHD	4	LZ2	120.0	140.0	158.0	158.0	183.0	201.0
LHD	5	LZ1	162.0	182.0	219.5	219.5	244.5	282.0
LHD	6	LZ1	201.0	221.0	258.5	258.5	283.5	321.0
LHD	7	LZ1	282.0	302.0	339.5	339.5	364.5	402.0
LHD	8	LZ1	321.0	341.0	378.5	378.5	403.5	441.0
LHD	9	LZ1	402.0	422.0	459.5	459.5	484.5	522.0
LHD	10	LZ1	441.0	461.0	498.5	498.5	523.5	561.0

Table 4.6. Final MV-22 Amphibious Fuel Delivery Schedule

This scenario calls for two MV-22s from one ship (LHD) delivering fuel to two different LZs. When inspecting the *Arrive at Ship* and *Start Load* columns of Lines 1–4 we see that one MV-22 performs runs 1 and 4 and the other MV-22 does runs 2 and 3. This occurs because LZ2 is much closer than LZ1.

4.4.4 CH-53 Schedule

Ship	Run	Destination	Start Load	Depart Ship	Arrive at LZ	Start Unload	Depart LZ	Arrive at Ship
LHD	1	LZ1	0.0	20.0	82.5	82.5	107.5	170.0
LHD	2	LZ2	0.0	20.0	50.0	50.0	75.0	105.0
LHD	3	LZ1	105.0	125.0	187.5	187.5	212.5	275.0
LHD	4	LZ2	170.0	190.0	220.0	220.0	245.0	275.0
LHD	5	LZ1	275.0	295.0	357.5	357.5	382.5	445.0
LHD	6	LZ1	275.0	295.0	357.5	357.5	382.5	445.0

Table 4.7. Final CH-53E Amphibious Fuel Delivery Schedule

The first two runs of both CH-53s follow the same pattern: one run to LZ1 and one run to LZ2. One CH-53 does runs 1 and 4, and the other does runs 2 and 3. They both arrive back to the seabase at the same time after these two runs. They then head to LZ1 in tandem (Lines 5 and 6).

4.5 Objective Value Comparison

Our objective value in the Scheduler Linear Program is the average completion time across all runs. It is straightforward to modify the objective function to handle the maximum

completion time. We refer to these measures of effectiveness as *mission* completion time to avoid confusion with the time it takes to run the algorithms. Table 4.8 presents both the average and maximum mission completion time for all four types of connectors. For the LCACs, the average and maximum are over all runs for both the Landing Helicopter Dock (LHD) and Landing Platform Dock (LPD).

We formulate a Mixed Integer Linear Program (MILP) that treats the binary variables $a_{s,rs,b,rb}$ and $c_{b,rb,s,rs}$ as decision variables and hence bypasses the assignment heuristic. We present the full MILP formulation in the Appendix. Table 4.8 also lists the MILP results, when the MILP solves within 12 hours.

	Average		Maximum	
Connector	MACS	MILP	MACS	MILP
MV-22	309.3	309.3	561.0	561.0
CH-53	285.83	285.83	445.0	445.0
LCAC	357.08	—	595.0	—
LCU	380.0	380.0	380.0	380.0

Table 4.8. Average and Maximum Mission Completion Time (minutes) Comparison for the MACS Model vs. MILP Model

From Table 4.8 we see that the MACS model performs extremely well when compared to the MILP. Furthermore, the MACS model creates an executable schedule for all four connector types in less than a 40 seconds. The MACS results for the LCU, MV-22, and CH-53 match the MILP exactly. The MILP solves for the LCU and CH-53 schedule within seconds, but it takes 20 minutes to solve for the MV-22. We provide the MILP with the MACS solution as a warm-start. The biggest discrepancy with performance occurs with the LCAC. The MILP was unable to solve for the LCAC schedule after 12 hours. The optimality gap at that point was 43% and the best integer solution found up to that point matched the MACS solution. This illustrates the utility of MACS. The LCAC requires twelve runs from two different ships to two different beaches; very common for a full day of amphibious operations for a MEU/ARG.

When comparing the run-time performance of the MACS versus the MILP it should be noted that the MACS executes Quickest Flow, Assignment Heuristic, and the Scheduler Linear Program for all four connector types in one execution of the MACS tool (all in less than 40 seconds). The MILP run-times quoted above only generate the schedule for one connector type at a time. To run the MILP (that replaces the Assignment Heuristic and

Scheduler Linear Program) we still need to first run the Quickest Flow model to determine the runs allocation. The MILP times presented above do not account for the additional run-time to execute the Quickest Flow. For this scenario the Quickest Flow model runs in approximately 10 seconds.

Both the MACS and MILP model take the number of runs to each beach by connector type as input from the Quickest Flow model. Future work could modify Quickest Flow, or the Quickest Flow output, to consider other combinations of runs in an effort to generate more effective schedules.

4.6 Connector Usage Analysis

We now illustrate the utility of using MACS to perform what-if analysis. In particular we adjust the number of MV-22s available to perform the mission. This is useful in practice because we can see the impact changing connector availability has on the operational requirements and feasibility of the other connectors to accomplish the fuel delivery mission. This allows MEU and ARG Commanders to better balance the allocation of assets to fuel delivery missions versus other missions to support the overall operation. Figure 4.9 shows how connector usage varies as we change the number of available MV-22s

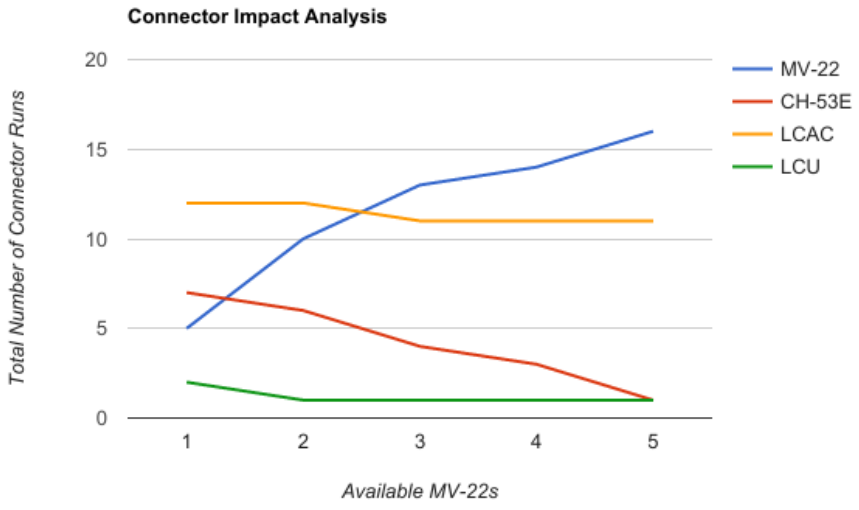


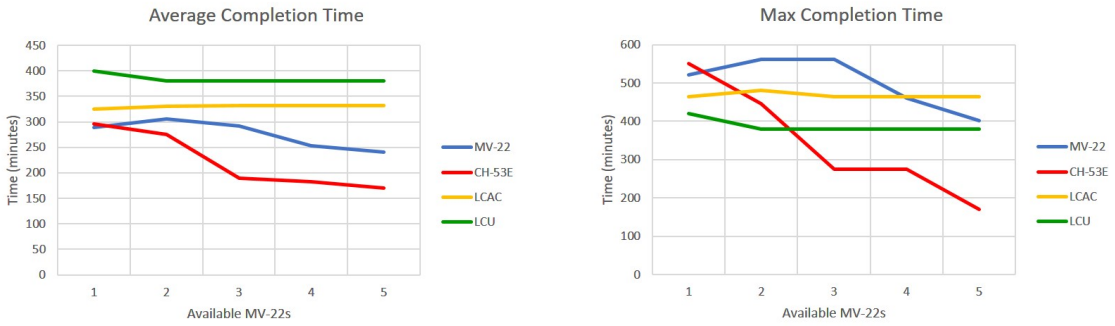
Figure 4.9. Impact on Connector Run Requirements when Adjusting the MV-22 Allocation

Figure 4.9 provides the following insight:

- The mission is infeasible when no MV-22s are allocated to the mission. However if we increase the number of CH-53s from 2 to 3 (keeping MV-22s at 0), the mission is feasible.
- As more MV-22s are allocated to the mission, CH-53E required runs decrease significantly.
- When we increase the allocation of MV-22s to more than two, the number of LCAC runs decreases from 12 runs to 11 runs. This occurs because it is faster to deliver more fuel to LZ1 via MV-22s and then transport fuel to Beach 1 via ground assets. LZ1 has four trucks available to transport fuel (See Figure 4.5).

Objective Value Impact

We next analyze the impact on the average and maximum mission completion time as we vary the MV-22 allocation. The two graphs comprising Figure 4.10 illustrate the effects.



(a) Average Mission Completion Time (b) Maximum Mission Completion Time

Figure 4.10. Average and Maximum Mission Completion Times when varying allocation of MV-22s

We see in Figure 4.10a that the average mission completion time for CH-53Es decreases significantly due the allocation of additional faster aircraft (MV-22) and less required runs from the CH-53E. The worst-case average mission completion times corresponds to the LCU in all cases.

In Figure 4.10b we witness a significant reduction in the maximum mission completion time of the CH-53E for the same reasons discussed above. The maximum time for the MV-22 increases as we move from one to three MV-22s, but then decreases significantly thereafter. This illustrates how the speed of the MV-22 can impact the overall mission timeline. Eventually, with enough MV-22s we can deliver the fuel very quickly.

In both figures we see that when the MV-22 allocation is reduced to one aircraft it has the most impact on CH-53s and the LCU mission completion times, especially the maximum mission completion time for the CH-53s.

Placing information such as this in the hands of amphibious planners and decision makers is incredibly valuable as there is a constant struggle for the allocation of connector assets in an amphibious Marine Air Ground Task Force (MAGTF). Decision makers can see the impact one or two more connectors can have on the mission.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 5:
Conclusions

5.1 Summary

This thesis develops a suite of interconnected models called MACS that produces a feasible and executable ship-to-shore amphibious schedule for fuel delivery operations. MACS integrates both air and surface connectors and accounts for potential congestion at loading and unloading sites. In the scenarios we consider, the entire process runs in under 40 seconds. Experience shows that this tool has the potential to significantly improve the planning process to efficiently deliver bulk fuel to units ashore to maintain operational tempo.

Through the use of a dynamic network flow model based on mission specific planner inputs, the Quickest Flow model effectively provides planners with the number of required runs by connector type that satisfy demand within the time allotted. This information by itself provides significant support to decision makers. The Assignment Heuristic uses assignment policies to quickly produce assignment parameters that allow us to bypass the use of a slow mixed integer linear program. Finally, the Scheduler Linear Program integrates the assignment parameters to incorporate the six time points that define any amphibious schedule.

To date, this model has easily constructed schedules for several different MEU fuel resupply scenarios that includes preliminary testing of a larger scenario (6 ships in ARG) with a much more diverse and complex network of resupply locations ashore. In future work, we plan to perform more testing on larger scenarios (see Section 5.2). When compared to the MILP formulation in Appendix 1, the MACS tool is able to solve all scenarios quickly with the MILP unable to fully develop connector schedules for scenarios requiring more than approximately 10 runs in a reasonable amount of time.

5.2 Future Research

5.2.1 Model Improvements

Exploring algorithmic improvements to the three underlying models comprising the MACS tool and/or the MILP could further enhance the speed and performance of the tool. This is especially important as we use MACS to execute more complex and demanding scenarios. It may be possible to improve the MILP performance by incorporating additional constraints on the binary assignment variables a and c. While it is doubtful that the MILP would be able to scale to very large scenarios, it is important to compare our heuristic performance to the optimal when possible.

Currently we consider four assignment policies in the Assignment Heuristic (see Chapter 3.2.1). One could define additional policies or tweak the existing policies to improve the Assignment Heuristic. The Quickest Flow model is arguably the most important component of our algorithm, so improving that model has potential to make a significant impact. We could consider slight variants of the current output from Quickest Flow. For example we could increase or decrease the number of runs for each connector by one. Another option would modify the objective function in Quickest Flow. Currently we run Quickest Flow many times to determine the minimum time to deliver all the fuel. We could instead use a weighted discounted average as our objective function to reduce the number of iterations while still ensuring fuel is delivered quickly. For the Scheduler Linear Program, we could add additional constraints to make the schedule more operationally relevance. For example aircraft almost always travel to and from land nodes in tandem, and thus we should have a constraint to force this into the schedule.

5.2.2 Apply to All Personnel and Equipment in a MAGTF

This thesis presents the tool for scheduling bulk fuel resupply to units ashore, but possesses the speed and functionality to be the foundation for a much more comprehensive amphibious planning tool. This tool would be able to optimize and schedule ship-to-shore movement of all equipment and personnel organic to a MEU. This would revolutionize how the Marines and Navy plan for amphibious operations, and significantly reduce the planning cycle required to provide key decision makers with viable amphibious plans from which to

choose. To start with, we could divide connector cargo into a limited number of categories such as fuel, personnel, vehicles, and pallets of equipment. It no longer suffices to merely specify when each connector leaves the seabase. We now need to also determine what material is on each connector. One possibility is to expand the Quickest Flow model to a multi-commodity flow variant. Efforts are currently underway to expand the capabilities of the model to develop a comprehensive amphibious planning tool.

5.2.3 Stress MACS Against Larger and More Complex Scenarios

We have conducted very preliminary testing of the MACS tool with a larger amphibious force (essentially a scenario that includes 2 MEUs). The next step after codifying the tool's performance against the 2-MEU model would be to apply this model to a Marine Expeditionary Brigade (MEB) sized scenario. The MEB and associated amphibious task force is significantly larger than a MEU, and would be a good test for the model. There are several Marine Corps schools and courses that have well thought-out and comprehensive scenarios that take planners days to plan bulk fuel resupply. We believe the MACS tool would demonstrate its full potential as a highly effective and efficient planning tool for larger amphibious forces. Additionally, integrating the Maritime Prepositioning Force (MPF) assets into the tool would further display the flexibility of the MACS.

5.2.4 Integrate Future Combat Ship-to-Shore Systems

The Marine Corps is currently experimenting with new technologies that could be integrated into the MACS model. Unmanned logistics systems such as the K-MAX vertical lift platform could immediately be introduced into the tool to determine its overall impact on fuel delivery operations in an amphibious environment. Additionally, the CH-53K would be another interesting platform to introduce into the MACS tool to see how its added lift capacity, speed, and capability will enhance amphibious operations as it begins to replace older CH-53E models.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A:
Mixed Integer Linear Program

In this appendix, we present the formulation of a mixed integer linear program (MILP) that we use as a comparative tool to assess the optimality performance of the Assignment Heuristic and Scheduler Linear Program from Chapters 3.2–3.3. The key difference between the MILP and the Scheduler Linear Program is that in the MILP $a_{s,rs,b,rb}$ and $c_{b,rb,s,rs}$ are binary assignment decision variables, whereas in the Scheduler Linear Program they are fixed parameters. The Assignment Heuristic generates $a_{s,rs,b,rb}$ and $c_{b,rb,s,rs}$ for the Scheduler Linear Program. Recall that $a_{s,rs,b,rb}$ assigns departure ship runs to beach runs and $c_{b,rb,s,rs}$ assigns beach runs to returning ship runs. The MILP can generate the optimal schedule very quickly for a small number of total runs (~ 5). However, even for small scenarios (like the one presented in Chapter 4), the number of runs can easily exceed 20, and the MILP cannot solve in a reasonable amount of time.

The MILP formulation is nearly identical to the Scheduler Linear Program in Chapter 3 with the exception of the added binary decision variables. We also need to add a few other parameters and variables for bookkeeping purposes and because we need to linearize constraints.

Indices and Sets

$s \in S$	ships
$b \in B$	land nodes that can be directly reached by this connector type
$i \in I$	6 time points for each run
$rs \in Rx_s$	set of runs from each ship, cardinality specified by $maxRuns$
$rb \in Ry_b$	set of runs from each land node

Data [units]

$numConn_s$	number of connectors per ship s
$maxRuns$	maximum runs from each ship: $\sum_{b \in B} runsBeach_b$
$runsBeach_b$	total number of runs to each land node b
$spotsShip_s$	total number of welldeck or flightdeck spots for each ship s

$spotsBeach_b$	total number of landing spots at each land node b
$transTime_{s,b}$	transit time from ship s to land node b [minutes]
$loadTime_s$	time to load connector on ship s [minutes]
$unloadTime_b$	time to unload connector at land node l [minutes]
$dayLength$	length of welldeck/flight crew day [minutes]
$onWSlack$	slack time value to limit time connectors are vulnerable waiting to unload
M	Bookkeeping variable to linearize constraints: $dayLength \times (numShips + 1)$

Binary Integer Decision Variables

$a_{s,rs,b,rb}$	Binary variable that associate ship departure runs with beach runs: $a_{s,rs,b,rb} = 1$ if the rs departure run of ship s corresponds to the rb run of land node b
$c_{b,rb,s,rs}$	Binary variable that associate beach runs with ship arrival runs : $c_{b,rb,s,rs} = 1$ if the rb run of land node b corresponds to the rs return run of ship s

Continuous Decision Variables [units]

$X_{i,s,rs}$	the time of the ith event of the rs departing run from ship s [minutes]
$Y_{i,b,rb}$	the time of the ith event of the rb run of beach b [minutes]
$Z_{i,s,rs}$	the time of the ith event of the rs returning run to ship s [minutes]
$welldeckStart_s$	time to start welldeck/flight deck on ship s [minutes]
$welldeckEnd_s$	time to end welldeck/flight deck on ship s [minutes]
$Xa_{i,s,rs,b,rb}$	New variable to linearize problem defined as $X_{i,s,rs} \times a_{s,rs,b,rb}$ [minutes]
$Yc_{i,b,rb,s,rs}$	New variable to linearize problem defined as $Y_{i,b,rb} \times c_{b,rb,s,rs}$ [minutes]

Formulation

$$\min_{\substack{X,Y,Z,\\ welldeckStart,\\ welldeckEnd,\\ Xa,Yc\\ a,c}} \quad \sum_{s\in S}\sum_{rs\in Rx_s} X_{6,s,rs} \tag{A.1}$$

$$\text{s.t.} \quad welldeckStart_s \le X_{i,s,rs} \le welldeckEnd_s \qquad \forall i \in I, \forall s \in S, \forall rs \in Rx_s \tag{A.2}$$

$$welldeckEnd_s = welldeckStart_s + dayLength \qquad \forall s \in S \tag{A.3}$$

$$X_{2,s,rs} \ge X_{1,s,rs} + loadTime_s \times \sum_{b\in B}\sum_{rb\in Ry_b} a_{s,rs,b,rb} \qquad \forall s \in S, \forall rs \in Rx_s \tag{A.4}$$

$$X_{3,s,rs} \geq X_{2,s,rs} + \sum_{b \in B} \sum_{rb \in Ry_b} a_{s,rs,b,rb} \times transTime_{s,b} \quad \forall s \in S, \forall rs \in Rx_s \tag{A.5}$$

$$Y_{4,b,rb} \geq Y_{3,b,rb} \quad \forall b \in B, \forall rb \in Ry_b \tag{A.6}$$

$$Y_{5,b,rb} \geq Y_{4,b,rb} + unloadTime_b \quad \forall b \in B, \forall rb \in Ry_b \tag{A.7}$$

$$Y_{6,b,rb} \geq Y_{5,b,rb} + \sum_{s \in S} \sum_{rs \in Rx_s} a_{s,rs,b,rb} \times transTime_{s,b} \quad \forall b \in B, \forall rb \in Ry_b \tag{A.8}$$

$$X_{4,s,rs} - X_{2,s,rs} \leq onWSlack \times \sum_{b \in B} \sum_{rb \in Ry_b} a_{s,rs,b,rb} transTime_{s,b} \quad \forall s \in S, \forall rs \in Rx_s \tag{A.9}$$

$$X_{i,s,rs} \geq X_{i-1,s,rs} \quad \forall i \in I, \forall s \in S, \forall rs \in Rx_s \tag{A.10}$$

$$Z_{i,s,rs} \geq Z_{i-1,s,rs} \quad \forall i \in I, \forall s \in S, \forall rs \in Rx_s \tag{A.11}$$

$$Y_{i,b,rb} \geq Y_{i-1,b,rb} \quad \forall i \in I, \forall b \in B, \forall rb \in Ry_b \tag{A.12}$$

$$X_{1,s,rs} \geq X_{2,s,rs-spotsShip_s} \quad \forall s \in S, \forall rs \in Rx_s \tag{A.13}$$

$$Y_{4,b,rb} \geq Y_{5,b,rb-spotsBeach_b} \quad \forall b \in B, \forall rb \in Ry_b \tag{A.14}$$

$$X_{1,s,rs} \geq Z_{6,s,rs-numConn_s} \quad \forall s \in S, \forall rs \in Rx_s \tag{A.15}$$

$$X_{2,s,rs} \geq X_{2,s,rs-1} \quad \forall s \in S, \forall rs \in Rx_s \tag{A.16}$$

$$Z_{6,s,rs} \geq Z_{6,s,rs-1} \quad \forall s \in S, \forall rs \in Rx_s \tag{A.17}$$

$$Y_{4,b,rb} \geq Y_{4,b,rb-1} \quad \forall b \in B, \forall rb \in Ry_b \tag{A.18}$$

$$Y_{i,b,rb} = \sum_{s \in S} \sum_{rs \in Rx_s} Xa_{i,s,rs,b,rb} \quad \forall i \in I, \forall b \in B, \forall rb \in Ry_b \tag{A.19}$$

$$Xa_{i,s,rs,b,rb} \leq a_{s,rs,b,rb} \times M \quad \forall i \in I, \forall s \in S, \forall rs \in Rx_s, \forall b \in B, \forall rb \in Ry_b \tag{A.20}$$

$$Xa_{i,s,rs,b,rb} \geq X_{i,s,rs} - (1 - a_{s,rs,b,rb}) \times M \quad \forall i \in I, \forall s \in S, \forall rs \in Rx_s, \forall b \in B, \forall rb \in Ry_b \tag{A.21}$$

$$Xa_{i,s,rs,b,rb} \leq X_{i,s,rs} \quad \forall i \in I, \forall s \in S, \forall rs \in Rx_s, \forall b \in B, \forall rb \in Ry_b \tag{A.22}$$

$$\sum_{b \in B} \sum_{rb \in Ry_b} a_{s,rs,b,rb} \leq 1 \quad \forall s \in S, \forall rs \in Rx_s \tag{A.23}$$

$$\sum_{s \in S} \sum_{rs \in Rx_s} a_{s,rs,b,rb} = 1 \quad \forall b \in B, \forall rb \in Ry_b \tag{A.24}$$

$$\sum_{b \in B} \sum_{rb \in Ry_b} a_{s,rs,b,rb} \geq \sum_{b \in B} \sum_{rb \in Ry_b} a_{s,rs-1,b,rb} \quad \forall s \in S, \forall rs \in Rx_s \tag{A.25}$$

$$Z_{i,s,rs} = \sum_{b \in B} \sum_{rb \in Ry_b} Yc_{i,b,rb,s,rs} \quad \forall i \in I, \forall s \in S, \forall rs \in Rx_s \tag{A.26}$$

$$Yc_{i,b,rb,s,rs} \leq c_{b,rb,s,rs} \times M \quad \forall i \in I, \forall b \in B, \forall rb \in Ry_b, \forall s \in S, \forall rs \in Rx_s \tag{A.27}$$

$$Yc_{i,b,rb,s,rs} \geq Y_{i,b,rb} - (1 - c_{b,rb,s,rs}) \times M \qquad \forall i \in I, \forall b \in B, \forall rb \in Ry_b, \forall s \in S, \forall rs \in Rx_s \quad \text{(A.28)}$$

$$Yc_{i,b,rb,s,rs} \leq Y_{i,b,rb} \qquad \forall i \in I, \forall b \in B, \forall rb \in Ry_b, \forall s \in S, \forall rs \in Rx_s \quad \text{(A.29)}$$

$$\sum_{s \in S} \sum_{rs \in Rx_s} c_{b,rb,s,rs} = 1 \qquad \forall b \in B, \forall rb \in Ry_b \quad \text{(A.30)}$$

$$\sum_{b \in B} \sum_{rb \in Ry_b} c_{b,rb,s,rs} \leq 1 \qquad \forall s \in S, \forall rs \in Rx_s \quad \text{(A.31)}$$

$$\sum_{rs \in Rx_s} c_{b,rb,s,rs} = \sum_{rs \in Rx_s} a_{s,rs,b,rb} \qquad \forall s \in S, \forall b \in B, \forall rb \in Ry_b \quad \text{(A.32)}$$

$$\sum_{b \in B} \sum_{rb \in Ry_b} c_{b,rb,s,rs} \geq \sum_{b \in B} \sum_{rb \in Ry_b} c_{b,rb,s,rs-1} \qquad \forall s \in S, \forall b \in B, \forall rb \in Ry_b \quad \text{(A.33)}$$

$$X_{i,s,rs} \geq 0 \qquad \forall i \in I, \forall s \in S, \forall rs \in Rx_s \quad \text{(A.34)}$$

$$Y_{i,b,rb} \geq 0 \qquad \forall i \in I, \forall b \in B, \forall rb \in Ry_b \quad \text{(A.35)}$$

$$Z_{i,s,rs} \geq 0 \qquad \forall i \in I, \forall s \in S, \forall rs \in Rx_s \quad \text{(A.36)}$$

$$welldeckStart_s \geq 0 \qquad \forall s \in S \quad \text{(A.37)}$$

$$welldeckEnd_s \geq 0 \qquad \forall s \in S \quad \text{(A.38)}$$

$$Xa_{i,s,rs,b,rb} \geq 0 \qquad \forall i \in I, \forall s \in S, \forall rs \in Rx_s, \forall b \in B, \forall rb \in Ry_b \quad \text{(A.39)}$$

$$Yc_{i,b,rb,s,rs} \geq 0 \qquad \forall i \in I, \forall b \in B, \forall rb \in Ry_b, \forall s \in S, \forall rs \in Rx_s \quad \text{(A.40)}$$

$$a_{s,rs,b,rb} \in \{0, 1\} \qquad \forall s \in S, \forall rs \in Rx_s, \forall b \in B, \forall rb \in Ry_b \quad \text{(A.41)}$$

$$c_{b,rb,s,rs} \in \{0, 1\} \qquad \forall b \in B, \forall rb \in Ry_b, \forall s \in S, \forall rs \in Rx_s \quad \text{(A.42)}$$

In the Scheduler Linear Program in Section 3.3 we know how many runs originate from each ship, and therefore Rx_s is an input. In the MILP, the algorithm determines the number of runs by ship, and thus we must be careful in how we define Rx_s. We do know the total number of runs that must originate from the seabase, $maxRuns$, and thus in the MILP we schedule $maxRuns$ runs for each ship: $|Rx_s| = maxRuns$ for all s. This results in many more runs than are actually required, but provides no restrictions on the number of runs from any given ship. This approach produces *phantom* runs. For example if there are two

ships and $maxRuns = 10$, then the MILP schedules 20 total runs (10 for each ship), even though only 10 runs are needed. The extra 10 runs are phantom runs. The MILP might determine to send 4 (actual) runs from Ship 1, which results in 6 phantom runs from Ship 1. We define the MILP such that $X_{i,s,rs} = 0$ for all rs corresponding to phantom runs.

The objective function and Constraints (A.2) through (A.18) are identical to the formulation in Section 3.3. The added constraints to this model mainly deal with linearizing the new decision variables $Xa_{i,s,rs,b,rb}$ and $Yc_{i,b,rb,s,rs}$. We also have some additional bookkeeping constraints for the binary decision variables. We explain the new constraints below:

1. Constraint (A.19) corresponds to Constraint (3.30), but uses the linearized variable $Xa_{i,s,rs,b,rb}$. The constraint connects the X variables with the Y variables.
2. With the MILP we use $Xa_{i,s,rs,b,rb}$ to avoid having a product of a binary variable and a continuous variable. To eliminate this product requires a linearization via a set of constraints. Constraints (A.20), (A.21), and (A.22) perform this linearization using the standard approach.
3. Constraint (A.23) ensures that each departure ship run goes to at most one beach run. This is an inequality because phantom runs are not assigned to a beach run.
4. Constraint (A.24) ensures that each run on the beach has to align with exactly one departure ship run. This is an equality because each beach run must be satisfied by a ship run.
5. Constraint (A.25) handles the phantom run issue discussed earlier. All phantom runs are assigned to smaller indices for rs, thus ensuring the X and Z variables are 0 for phantom runs. The later rs indices correspond to actual runs.
6. Constraint (A.26) corresponds to Constraint (3.31) , but uses the linearized variable $Yc_{i,b,rb,s,rs}$. The constraint connects the Y variables with the Z variables.
7. Constraints (A.27), (A.28), and (A.29) perform the same linearization operation as Constraints (A.20), (A.21), and (A.22) to linearize the $Yc_{i,b,rb,s,rs}$ variables.
8. Constraint (A.30) ensures that exactly one beach run associates with exactly one returning ship run.
9. Constraint (A.31) ensures that each returning ship run can be assigned at most one beach run. The inequality results from the phantom run approach.
10. Constraint (A.32) connects a to c in that the number of departure runs assigned from ship to beach must match the number of return runs–that is X to Y via a must match

Y to Z via c.

11. Constraint (A.33) is similar to Constraint (A.25) in that the phantom runs returning occur for the first indices.
12. Constraints (A.34), (A.35), (A.36), (A.37), (A.38), (A.39), and (A.40) ensure the decision variables are nonnegative.
13. Constraints (A.41) and (A.42) mandate that a and c are binary variables.

We use the values of $a_{s,rs,b,rb}$ and $c_{b,rb,s,rs}$ generated by the Assignment Heuristic to warm-start the MILP. The warm-start quickly produces a feasible solution that often is near-optimal for the scenarios we have studied.

APPENDIX B:
Assignment Heuristic Pseudocode

In this appendix, we provide the pseudocode for the assignment heuristic. We only present the assignment policy for the largest runs deficit. That is we compare the required number of runs to each beach/LZ and the number already assigned and then choose the land node with the largest difference between the two. For a copy of the actual code, please contact Michael Atkinson (`mpatkins@nps.edu`) or Kyle Lin (`kylin@nps.edu`).

```
1:  function RUN_ASSIGNMENT_HEURISTIC
2:      b_runs ← GET_BEACH_RUNS                          ▷ From Quickest Flow
3:      num_con ← GET_NUM_CONNECTORS                     ▷ From Input CSVs
4:      sb_load ← GET_SEABASE_LOADTIME                   ▷ From Input CSVs
5:      sb_spots ← GET_SEABASE_SPOTS                     ▷ From Input CSVs
6:      b_unload ← GET_BEACH_UNLOADTIME                  ▷ From Input CSVs
7:      b_spots ← GET_BEACH_SPOTS                        ▷ From Input CSVs
8:      trav_time ← GET_TRAVEL_TIME                      ▷ From Input CSVs

9:      sb_obj ← INIT_SEABASEOBJ(sb_spots, num_con)      ▷ Tracks info about each ship
10:     beach_obj ← INIT_BEACHOBJ(b_spots)               ▷ Tracks info about each beach
11:     conn_pq ← INIT_CONNECTORS(num_con)               ▷ Tracks connector info in Priority Q

12:
13:     while runs remain do
14:         next_event = conn_pq.get()                   ▷ Pull next connector event from Pri Q
15:         proc_next_event(b_runs,next_event,sb_load ,b_unload,b_spots,
              trav_time,beach_objs,sb_obj)
16:         conn_pq.get().put(next_event)                ▷ Put updated connector info back to Pri Q
17:     end while
18: end function
```

19: **function** INIT_SEABASEOBJ(sb_spots, num_con)
20: sb_obj = [] ▷ Store info on each ship
21:
22: **for** i in 1:num_ships **do**
▷ First element ship id
▷ 2nd element is next run to depart
▷ 3rd element is next run to arrive
▷ 4th element is the next time a spot is available to load
23: $sb\_obj[i] \leftarrow [i, 1, (1 - num\_con[i]), 0]$
24: **end for**
25: **return** sb_obj
26: **end function**

27: **function** INIT_BEACHOBJ(b_spots)
28: $beach\_obj$ = [] ▷ Store info on each beach
29:
30: **for** i in 1:num_beaches **do**
▷ First element beach id
▷ 2nd element is next run to arrive to beach
▷ 3rd element is the next time a spot is available to unload
▷ 4th element is number connectors in transit or offloading at beach
▷ 5th element is number connectors already assigned to beach
31: $beach\_obj[i] \leftarrow [i, 1, 0, 0, 0]$
32: **end for**
33: **return** $beach\_obj$
34: **end function**

```
35: function INIT_CONNECTORS(num_con)
36:     conn_pq = INIT_PRIORITYQUEUE          ▷ Store connector info in Priority Queue
37:
38:     for i in 1:num_ships do
39:         cur_num ← num_con[i]                    ▷ Number of connectors on Ship i
40:
41:         for j in 1:cur_num do
                                          ▷ 1st element: Time associated with current task
                                                      ▷ 2nd element: current task
                                                           ▷ 3rd element: ship ID
                                                      ▷ 4th element: connector ID
                                                 ▷ 5th element: departure ship run
                                            ▷ 6th element: beach ID for this run
                                                      ▷ 7th element: beach run
                                                 ▷ 8th element: return ship run
42:             con_info ← [0, 1, i, j, −1, −1, −1, −1]
43:             conn_pq.put(con_info)          ▷ Put connector info into Priority Queue
44:         end for
45:     end for
46:     return conn_pq
47: end function
```

```
48: function PROC_NEXT_EVENT(b_runs, next_event, sb_load, b_unload, b_spots, trav_time)
        beach_obj, sb_obj                                   ▷ Additional function inputs

49:     cur_task ← next_event[2]                            ▷ 2nd element in connector info list
50:
51:     if cur_task == 1 then
52:         process_sb_event(next_event, sb_load, sb_obj)
53:     else if cur_task == 2 then
54:         process_b_assignment(b_runs, next_event, b_unload, b_spots, trav_time,
              sb_obj, beach_obj)
55:     else if cur_task == 3 then
56:         process_b_event(next_event, b_unload, beach_obj)
57:     else
58:         process_return_to_sb(next_event, trav_time, beach_obj)
59:     end if
60: end function

61: function PROCESS_SB_EVENT(next_event, sb_load, sb_obj)
62:     SAVE_COMPLETED_RUN                  ▷ Before initializing next run, save previous run info
63:     arrival_time ← next_event[1]
64:     ship_id ← next_event[3]
65:     next_spot ← get_next_free_sbspot(sb_obj, ship_id)          ▷ Which load spot next free
66:     time_spot_avail ← get_time_next_free_sbspot(sb_obj, ship_id)   ▷ Time next spot free
67:     start_time ← max(arrival_time, time_spot_avail)        ▷ Time connector starts loading
68:     splash_time ← start_time + sb_load[ship_id]              ▷ Time connector leaves ship
69:     next_event[2] ← 2                                  ▷ New Connector task: leaving ship
70:     next_event[1] ← splash_time                ▷ New Connector time: departure from ship
71:     UPDATE_SB_OBJECT(sb_obj, ship_id, splash_time)     ▷ Spot availability, number of runs
72: end function
```

```
73: function PROCESS_B_ASSIGNMENT(b_runs, next_event, b_unload, b_spots, trav_time)
        sb_obj, beach_obj                                   ▷ Additional function inputs

74:     assign_time ← next_event[1]
75:     ship_id ← next_event[3]
76:     next_beach ← ASSIGN_NEXT_BEACH(b_runs, beach_obj)          ▷ Which beach to go to
77:     beach_time ← assign_time + trav_time[ship_id][next_beach]  ▷ Time arrive at beach
78:     next_event[2] ← 3                          ▷ New Connector task: ready to unload at beach
79:     next_event[1] ← beach_time                 ▷ New Connector time: arrive at beach
80:     next_event[6] ← next_beach                 ▷ Store beach assigned to run
81:     next_event[5] ← SHIP_DEPART_RUN(sb_obj, ship_id)           ▷ Get run number
82:     UPDATE_SB_OBJECT(sb_obj, ship_id)                          ▷ Number of runs
83:     UPDATE_BEACH_OBJECT(beach_obj, beach_id)   ▷ Number of runs assigned to beach
84: end function

85: function PROCESS_B_EVENT(next_event, b_unload, beach_obj)
86:     arrival_time ← next_event[1]
87:     beach_id ← next_event[6]
88:     next_spot ← get_next_free_bspot(beach_obj, beach_id)       ▷ Which load spot next free
89:     time_avail ← get_time_next_free_bspot(beach_obj, beach_id) ▷ Time next spot free
90:     start_time ← max(arrival_time, time_avail)                 ▷ Time connector starts unloading
91:     splash_time ← start_time + beach_unload[beach_id]          ▷ Time connector leaves beach
92:     next_event[2] ← 4                          ▷ New Connector task: returning to ship
93:     next_event[1] ← splash_time                ▷ New Connector time: departure from beach
94:     next_event[7] ← BEACH_RUN_NUM(beach_obj, beach_id)         ▷ Beach run number
95:     UPDATE_BEACH_OBJECT(beach_obj, beach_id, splash_time)      ▷ Spot availability,
                                                                     number of runs
96: end function
```

```
97: function PROCESS_RETURN_TO_SB(next_event, trav_time, beach_obj)
98:     splash_time ← next_event[1]
99:     ship_id ← next_event[3]
100:    beach_id ← next_event[6]
101:    return_time ← splash_time + trav_time[ship_id][beach_id]      ▷ Time back at ship
102:    next_event[2] ← 1                          ▷ New Connector task: ready to load at ship
103:    next_event[1] ← return_time                ▷ New Connector time: arrival back to ship
104:    UPDATE_BEACH_OBJECT(beach_obj, beach_id)   ▷ One fewer connector at beach
105: end function

106: function ASSIGN_NEXT_BEACH(b_runs, beach_obj)
107:    max_deficit ← 0          ▷ assign next run to beach with largest run deficit
108:    next_beach ← −1                                              ▷ ID of next beach
109:
110:    for i in 1:num_beaches do
111:        tot_runs ← b_runs[i]                                     ▷ Total runs required
112:        runs_assign ← beach_obj[i][5]                            ▷ Runs assigned to beach
113:        cur_deficit ← tot_runs − runs_assign                     ▷ Remaining runs
114:
115:        if cur_deficit > max_deficit then                        ▷ Found new best candidate
116:            max_deficit ← cur_deficit
117:            next_beach ← i
118:        end if
119:    end for
120:    return best_beach
121: end function
```

List of References

[1] *The Marine Corps Operating Concept: How an Expeditionary Force Operates in the 21st Century*, 1st ed., Headquarters United States Marine Corps, Washington, D. C., 2016, pp. 4–15, 23–23.

[2] R. K. Ahujah, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*, 1st ed. Upper Saddle River, NJ: Prentice Hall, 1993.

[3] H. W. Hamacher and S. A. Tjandra, "Mathematical modelling of evacuation problems: A state of art," Institut Techno- und Wirtschaftsmathematik, Los Angeles, CA, Tech. Rep. TR-2000 (4230-46)-3, Nov. 1988.

[4] W. P. Langford, "A space-time flow optimization model for neighborhood evacuation," M.S. thesis, Naval Postgraduate School, Monterey, California USA, 2010.

[5] A. E. Malveo, "Assessing the impact of congestion during a multi-county evacuation," M.S. thesis, Naval Postgraduate School, Monterey, California USA, 2013.

[6] H. A. Viado, "An optimization model for sea-based supply of bulk fuel for a deployed marine expeditionary unit," M.S. thesis, Naval Postgraduate School, Monterey, California USA, 1999.

[7] P. W. Ward, "Optimizing ship-to-shore movement for hospital ship humanitarian assistance operations," M.S. thesis, Naval Postgraduate School, Monterey, California USA, 2008.

[8] N. L. Reitter, "A decision support system for sea-based sustainment operations," M.S. thesis, Naval Postgraduate School, Monterey, California USA, 1999.

[9] T. A. Hartman, "Rapid airlift planning for amphibious-ready groups," M.S. thesis, Naval Postgraduate School, Monterey, California USA, 2015.

[10] R. S. Jacobs, "Optimization of daily flight training schedules," M.S. thesis, Naval Postgraduate School, Monterey, California USA, 2014.

[11] *Ship-to-Shore Movement*, NTTP 3-02.1/MCWP 3-31.5, U.S. Dept. of the Navy, Washington, DC, 2007, a.2–A.3, J.2–J.5, J.19–J.20, J.24, J.26.

[12] File: The amphibious assault ship USS Kearsarge (LHD 3) conducts operations at sea May 30, 2013, in the Gulf of Aden 130530-N-SB587-594.jpg. (2013). *Wikimedia Commons*. [Online]. Available: https://commons.wikimedia.org/wiki/File:The_amphibious_assault_ship_USS_Kearsarge_(LHD_3)_conducts_operations_at_

sea_May_30,_2013,_in_the_Gulf_of_Aden_130530-N-SB587-594.jpg. Accessed Apr. 15, 2017.

[13] USS New York (LPD-21). (n.d.). NavSource Online: Amphibious Photo Archive. [Online]. Available: http://www.navsource.org/archives/10/09/10092123.jpg. Accessed Apr. 15, 2017.

[14] LSD Whidbey Island / Harpers Ferry Class Dock Landing Ships, United States of America. (n.d.). naval-technology.com. [Online]. Available: http://www.naval-technology.com/projects/harpers/. Accessed Apr. 15, 2017.

[15] US Navy's LCAC Hovercraft Can Go from Land to Sea in Seconds, Here are 5 More Cool Facts. (2015). TECHEBLOG. [Online]. Available: http://www.techeblog.com/index.php/tech-gadget/us-navy-s-lcac-hovercraft-can-go-from-land-to-sea-in-seconds-here-are-5-more-cool-facts. Accessed Apr. 15, 2017.

[16] AiirSource Military. (2013, May. 5). LCAC Air-Cushion Vehicle Landing & Unloading LAV-25 Amphibious Vehicle. [YouTube video]. Available: https://www.youtube.com/watch?v=MklnI1bPGcM. Accessed Apr. 15, 2017.

[17] *Employment of Landing Craft Air Cushion (LCAC)*, NWP 3-02.12/MCRP 3-31.1A, U.S. Dept. of the Navy, Washington, DC, 1997.

[18] Landing Craft Utility (LCU). (n.d.). GlobalSecurity.org. [Online]. Available: http://www.globalsecurity.org/military/systems/ship/images/lcu-lha-2-5.jpg. Accessed Apr. 15, 2017.

[19] File:US Navy 040720-N-4304S-132 Landing Craft Utilities (LCU) assigned to the Surf Riders of Assault Craft Unit One (ACU-1) offload equipment on the beach during amphibious assault training in support of exercise Rim of the Pacific.jpg. (2004). *Wikimedia Commons*. [Online]. Available: https://commons.wikimedia.org/wiki/File:US_Navy_040720-N-4304S-132_Landing_Craft_Utilities_(LCU)_assigned_to_the_Surf_Riders_of_Assault_Craft_Unit_One_(ACU-1)_offload_equipment_on_the_beach_during_amphibious_assault_training_in_support_of_exercise_Rim_of_the_Pacific.jpg. Accessed Apr. 15, 2017.

[20] *HELICOPTER CAPABILITIES / OPERATIONS*, B2C3197, U.S. Marine Corps, The Basic School, Marine Corps Training Command, Camp Barrett, VA, 2014, 27–31.

[21] Sikorsky CH-53E Super Stallion (S-65E/80) - USA - Marines. (2011). AIRLINERS. [Online]. Available: http://www.airliners.net/photo/USA---Marines/Sikorsky-CH-53E-Super/1914599/L. Accessed Apr. 15, 2017.

[22] CH-53E Super Stallion. (n.d.). Military.com. [Online]. Available: http://www.military.com/equipment/ch-53e-super-stallion. Accessed Apr. 15, 2017.

[23] Bell Boeing V-22 Osprey. (n.d.). *Wikipedia*. [Online]. Available: https://en.wikipedia.org/wiki/Bell_Boeing_V-22_Osprey. Accessed Apr. 15, 2017.

[24] This Time Lapse Shows Literally A Boat Load Of MV-22 Osprey Departures. (2015). FOXTROT ALPHA. [Online]. Available: http://foxtrotalpha.jalopnik.com/this-time-lapse-shows-literally-a-boat-load-of-mv-22-os-1706674695. Accessed Apr. 15, 2017.

[25] *Bulk Liquids Operations*, MCWP 4-11.6, U.S. Dept. of the Navy, Washington, DC, 1996, 3-2–3-3.

[26] Tactical Bulk Fuel Delivery System (TBFDS). (n.d.). Robertson Fuel Systems. [Online]. Available: http://www.robertsonfuelsystems.com/defense/tactical-bulk-fuel-delivery-system-tbfds/. Accessed Apr. 17, 2017.

[27] Tactical Bulk Fuel Delivery System, CH-53E (TBFDS, CH-53E). (1995). Biggerhammer.net. [Online]. Available: http://biggerhammer.net/factfile.nsf/ffiles/1ce24aa4f7e1ad7485256289005a0dd8.html. Accessed Apr. 17, 2017.

[28] SIXCON FUEL TANK ASSEMBLY. (n.d.). GSA Auctions. [Online]. Available: https://gsaauctions.gov/gsaauctions/aucdsclnk?sl=A1QSCI16128702. Accessed Apr. 17, 2017.

[29] C. Shepherd. (2012, Feb. 3). US Marine Corps AMK23 Cargo Truck with SIXCON Fuel Modules. [Online]. Available: https://www.flickr.com/photos/rcsadvmedia/8377677801

[30] *Amphibious Operations*, JP 3-02, U. S. Joint Staff, Washington, DC, 2014.

[31] W. E. Hart, C. Laird, J. P. Watson, and D. L. Woodruff, *Pyomo–optimization modeling in python*. New York, NY: Springer Science & Business Media, 2012, vol. 67.

[32] W. E. Hart, J. P. Watson, and D. L. Woodruff, "Pyomo: modeling and solving mathematical programs in python," *Mathematical Programming Computation*, vol. 3, no. 3, pp. 219–260, 2011.

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