

Strategies for Launch and Assembly of Modular Spacecraft

by

Erica Lynn Gralla

B.S.E., Mechanical & Aerospace Engineering (2004)

Princeton University

Submitted to the Department of Aeronautics & Astronautics
in Partial Fulfillment of the Requirements for the Degree of

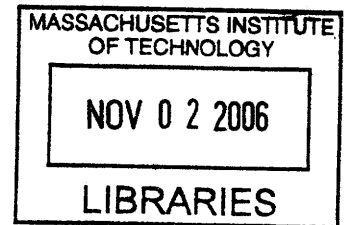
Master of Science in Aeronautics & Astronautics

at the

Massachusetts Institute of Technology

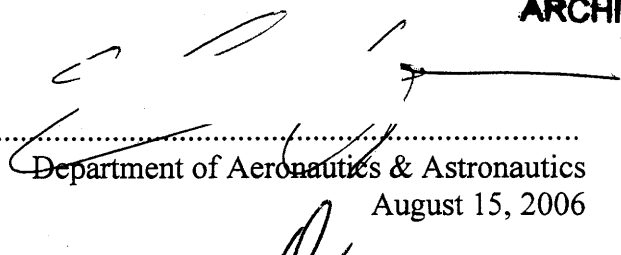
September 2006

© 2006 Massachusetts Institute of Technology
All rights reserved

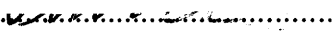


ARCHIVES

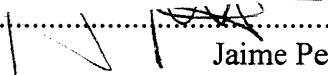
Signature of Author.....


Department of Aeronautics & Astronautics
August 15, 2006

Certified by


Olivier L. de Weck
Associate Professor of Aeronautics and Astronautics
and Engineering Systems
Thesis Supervisor

Accepted by


Jaime Peraire
Professor of Aeronautics and Astronautics
Chair, Committee on Graduate Students

Strategies for Launch and Assembly of Modular Spacecraft

by

Erica Lynn Gralla

Submitted to the Department of Aeronautics and Astronautics
on August 15, 2006, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

NASA's human lunar and Mars exploration program requires a new transportation system between Earth and the Moon or Mars. In recent years, unfortunately, human space exploration programs have faced myriad political, technical, and financial difficulties. In order to avoid such problems, future human space exploration programs should be designed from the start for affordability. This thesis addresses one aspect of affordable exploration programs by tackling the issue of high costs for access to space. While launch vehicle trades for exploration programs are relatively well understood, on-orbit assembly has been given much less attention, but is an equally important component of the infrastructure enabling human access to space. Two separate but related perspectives on in-space assembly of modular spacecraft are provided: first, the coupling between launch vehicle selection, vehicle design, and on-orbit assembly is explored to provide a quantitative understanding of this combined tradespace; and second, a number of on-orbit assembly methods are analyzed in order to understand the potential value of a reusable assembly support infrastructure. Within the first topic, a quantitative enumeration of the launcher-assembly tradespace (in terms of both cost and risk) is provided based on a generalizable process for generating spacecraft modules and launch manifests from a transportation architecture. An optimal module size and launcher capability is found for a sample architecture at 82 metric tons; a 28-mt EELV emerges as another good option. The results show that the spacecraft design, assembly planning, and launcher selection are highly coupled and should be considered together, rather than separately. Within the second topic, four separate assembly strategies involving module self-assembly, tug-based assembly, and in-space refueling are modeled and compared in terms of mass-to-orbit requirements for various on-orbit assembly tasks. Results show that the assembly strategy has a significant impact on overall launch mass, and reusable space tugs with in-space refueling can significantly reduce the required launch mass for on-orbit assembly. This thesis thus examines a broad but focused set of issues associated with on-orbit assembly of next-generation modular spacecraft.

Thesis supervisor: Olivier L. de Weck

Title: Associate Professor of Aeronautics and Astronautics and Engineering Systems

Acknowledgments

My first and greatest thanks go to my advisor Professor Olivier de Weck, for his guidance, support, and inspiration. Despite an amazing number of demands on his time, he seemed always ready with an answer, some advice, or a word of encouragement – or a new and interesting research topic to distract me from my classes. Oli deserves many thanks for supporting my varied research directions, propensity for choosing conferences in far-off locations, and abrupt changes in doctoral research aspirations. And of course, for introducing me to the wonders of the Arctic and the Swiss mountains. For all that and more, thank you.

I would also like to acknowledge the support of my colleagues and friends on the two major projects I worked on at MIT: the CE&R study and the Space Logistics project. Without the 8am meetings over IAP and the late-night Powerpoint engineering sessions, my life as a masters student would not have been complete. In all seriousness, I could not have accomplished nearly as much research without the dedication of both these teams.

Finally, I want to thank my professors and friends at MIT for their support over the past two years, especially in the Space Systems Lab and the space architecture research group. The 37 cluster has been a second home to me (perhaps more often than I would like), with too many silly moments in our office to count (dangling coconut heads, distorting cameras, ice axes appearing on my desk...). Thanks to the SSL, the '06 and '07 girls, and especially Sam. And my parents of course for their frequent messages and phone calls. You all made the past two years a great experience.

Erica Gralla
Cambridge, Massachusetts
August 2006

Contents

Contents	7
<i>List of Figures</i>	<i>9</i>
<i>List of Tables</i>	<i>11</i>
1. Introduction	13
<i>1.1 Motivation</i>	<i>15</i>
<i>1.2 Background.....</i>	<i>16</i>
1.2.1 On-Orbit Assembly Basics	16
1.2.2 History of On-Orbit Assembly	18
<i>1.3 Literature Review.....</i>	<i>23</i>
1.3.1 Assembly Literature	23
1.3.2 On-Orbit Servicing Literature.....	28
1.3.3 Modularity Literature	30
1.3.4 Literature Summary.....	30
<i>1.4 Research Goals.....</i>	<i>31</i>
2. Launching Assemble-able Architectures	33
<i>2.1 Designing for Assembly.....</i>	<i>34</i>
<i>2.2 Chunking and Manifesting</i>	<i>35</i>
2.2.1 Sample Transportation Architecture	35
2.2.2 Launch ‘Chunking’	37
<i>2.3 Launch Vehicle Sizing Model.....</i>	<i>40</i>
2.3.1 Full Factorial Search	41
2.3.2 Integer Optimization	42
<i>2.4 Launch Vehicle Sizing Results.....</i>	<i>45</i>
2.4.1 Optimal Launch Vehicle Size Selection.....	46
2.4.2 Integer Optimization Results	49
2.4.3 Risk Analysis: Payload Sparing.....	51
<i>2.5 Conclusions and Design Recommendations.....</i>	<i>54</i>
3. Assembly Strategies	57
<i>3.1 Assembly Techniques and Challenges</i>	<i>58</i>
3.1.1 Assembly Challenges	58
3.1.2 Assembly Techniques.....	59
<i>3.2 Assembly Strategies</i>	<i>61</i>
3.2.1 Basic Assembly Concepts	62

3.2.2 Assembly Strategies	63
3.3 <i>Assembly Trades Model</i>	65
3.3.1 Assembly Model Overview	65
3.3.2 Spacecraft Models	67
3.3.3 Propellant Requirements Model	69
3.3.4 Overhead Mass Metric	71
3.3.5 Baseline Parameters and Assumptions	72
3.4 <i>Trade Study Results</i>	74
3.4.1 Vehicle Scenario Parameters	74
3.4.2 Orbit Design Parameters	78
3.4.3 Tradespace Exploration	79
3.4.4 Sensitivity Analysis	80
3.5 <i>Assembly Strategy Selection</i>	84
3.5.1 Trade Study Conclusions	85
3.5.2 Future Work	87
3.5.3 Summary of Conclusions	88
4. Conclusions	89
4.1 <i>Design Strategies for Assembly</i>	90
4.2 <i>Additional Architecture Considerations</i>	92
4.3 <i>Summary & Conclusions</i>	93
4.4 <i>Recommendations for Future Work</i>	95
References	97
A. Acronyms & Abbreviations	103
B. Code for Launch Analysis	105
C. Code for Assembly Model	119
D. Assembly Trade Study Data	131

List of Figures

Figure 1.1	14
<i>Astronauts Herrington and Lopez-Alegria of STS-113 work on the P1 truss of the International Space Station.</i>	
Figure 1.2	19
<i>Apollo probe-and-drogue docking system.</i>	
Figure 1.3	20
<i>The Space Shuttle docking mechanism is an androgynous design.</i>	
Figure 1.4	21
<i>International Space Station configuration.</i>	
Figure 2.1	36
<i>Operations concepts for lunar and Mars missions.</i>	
Figure 2.2	37
<i>Vehicles for lunar and Mars missions.</i>	
Figure 2.3	38
<i>An overview of the launch manifesting process.</i>	
Figure 2.4	46
<i>Surplus mass and number of launches are shown for various launch vehicle capacities.</i>	
Figure 2.5	47
<i>Lunar CTS launch mass surplus for various launch vehicle and TMI staging strategies.</i>	
Figure 2.6	47
<i>Lunar habitat launch mass surplus for various launch vehicle sizes and TMI staging strategies.</i>	
Figure 2.7	48
<i>The metrics number of launches and mass surplus are plotted for complete lunar missions, including the CTS and Habitat vehicle stacks.</i>	
Figure 2.8	50
<i>The number of launches and cost for the lunar CTS are shown.</i>	
Figure 2.9	52
<i>Curves show the overall mission launch success probability for various sparing strategies.</i>	
Figure 2.10	53
<i>Mission risk for various payload types for a lunar mission.</i>	
Figure 2.11	53
<i>Mission risk for Mars mission.</i>	
Figure 3.1	60
<i>Options tree for uncrewed assembly.</i>	
Figure 3.2	64
<i>The four assembly strategies illustrated.</i>	
Figure 3.3	66
<i>On-orbit assembly model block diagram.</i>	
Figure 3.4	68
<i>Notional vehicle models of the space tug and self-assembled module.</i>	
Figure 3.5	76
<i>Results showing the change in overhead mass as the number of modules is varied.</i>	
Figure 3.6	77
<i>Results showing the change in overhead mass as the module mass is varied.</i>	
Figure 3.7	78

Results for varying assembly orbit altitude.

Figure 3.8	79
<i>Minimum overhead mass as a function of module mass and number of modules.</i>	
Figure 3.9	81
<i>Number of modules comparison for various engine masses.</i>	
Figure 3.10	83
<i>Number of modules comparison for various Isp values.</i>	

List of Tables

Table 1.1.....28
Overview of major docking systems.

Table 2.1.....50
Sample launch vehicle data.

Table 3.1.....72
On-orbit assembly model baseline values.

Table 3.2.....73
Typical engine characteristics.

Table D.1.....132
Overhead mass variation with number of modules.

Table D.2.....132
Overhead mass variation with module mass.

Table D.3.....132
Overhead mass variation with orbit altitude.

1

Introduction

The term ‘on-orbit assembly’ usually conjures up images of astronauts on spacewalks putting together complex trusses for the International Space Station, much like the scene pictured in Figure 1.1. Such feats are almost a part of everyday life at NASA these days (at least before the Columbia tragedy); this in itself is a tremendous technical achievement. About 160 spacewalks will be required to complete the assembly of the International Space Station, totaling more than double the number of extra-vehicular activity (EVA) hours NASA had previously completed [NASA 1999]. The planned 1-million pound international research facility in orbit will showcase the work of many partner nations cooperating in the largest space construction project in the history of mankind.

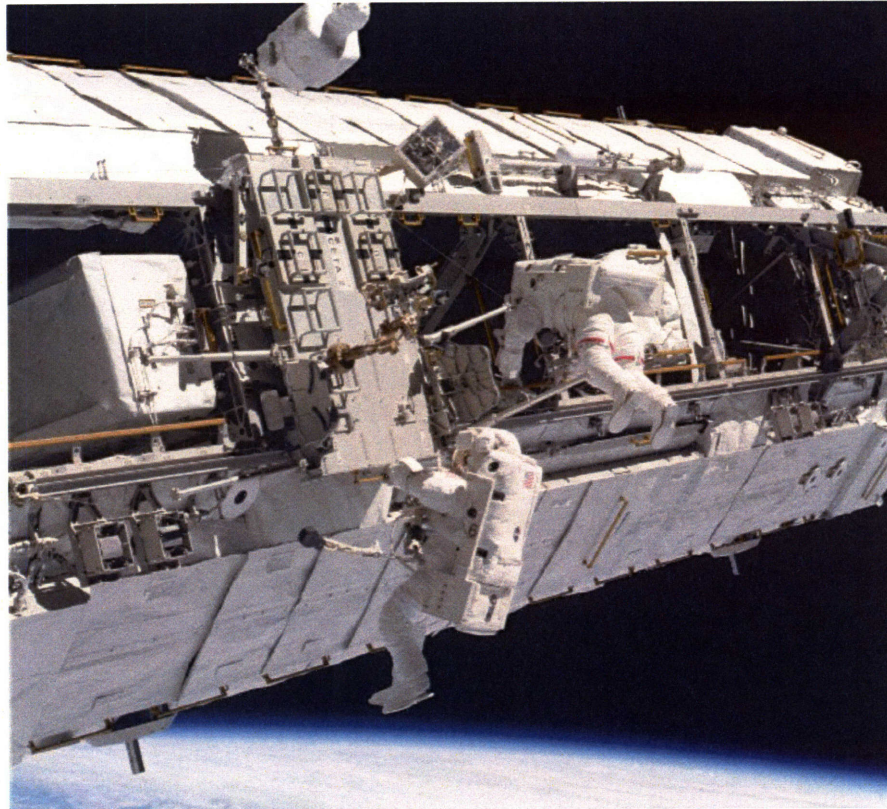


Figure 1.1: Astronauts Herrington and Lopez-Alegria of STS-113 work on the P1 truss of the International Space Station. [Space.com]

On the way to that goal, however, NASA has seen many dark days. Tragically, the Space Shuttle Columbia was lost because a piece of insulating foam damaged its leading edge [NASA 2003]. Delays caused construction dates to be pushed back ever farther. And massive cost overruns turned public and political opinions against NASA and its financial management. Despite all this, the space construction project itself has gone on with hardly a hitch.

Since President George W. Bush introduced his Vision for Space Exploration on January 14, 2004, NASA has been scrambling to figure out how to send humans back to the Moon and then on to Mars [NASA 2004]. Among the questions needing an urgent answer is, ‘How can we build on the technical success of the International Space Station and other human spaceflight programs, without repeating their financial and political problems?’ The answer, we might argue, lies in dissecting the root causes of these problems and working *now*, in the early design stages, to reduce the chance of spiraling development and operations costs in the future. The political problems stem from

NASA's financial issues, so we might start by trying to increase the long-term affordability of future human spaceflight programs.

One way to ensure greater affordability is to examine the complex human spaceflight system-of-systems, and focus on the parts of that system likely to drive up costs. The Earth-to-orbit architecture – all parts of a space program that transport vehicles from the ground into low Earth orbit, mission-ready – is one of the most costly parts of space programs today. The assembly of the international space station has cost NASA and the United States billions of dollars; could these costs have been reduced through the utilization of different methods and technologies for launch or on-orbit assembly? Perhaps a reduction of the complexity of the parts to be assembled, or a decrease in the number of EVA hours required, could have made the project more affordable. In an attempt to answer this question for the benefit of future human and unmanned spaceflight missions, this thesis examines Earth-to-orbit architectures, with an eye toward designing for affordability.

1.1 Motivation

In recent years, human space exploration programs such as the Shuttle and the International Space Station have been plagued by political and technical problems as well as soaring costs. In order to avoid such difficulties, next-generation human space exploration programs should be designed for affordability. By viewing exploration programs as 'systems-of-systems', we can focus on reducing costs through the use of flexible, reusable infrastructures to support various aspects of manned spaceflight.

One of the most difficult pieces of this system-of-systems architecture is the issue of access to space. Current evolved expendable launch vehicles (EELV's) can loft only about 25 metric tons into low Earth orbit (LEO) [Isakowitz 2004]; however, major human exploration ventures such as lunar or Mars exploration will require spacecraft many times that size. Even with a heavy-lift launch vehicle (HLLV), on-orbit assembly is required for short lunar missions (one launch for the crew, one for the lunar lander stack) [NASA 2005a]. For Mars missions, significantly more launches will be required to hoist the large exploration spacecraft into orbit. Whether the cheaper EELV's or the larger, more expensive HLLV's are employed, significant on-orbit assembly will be required.

Definition [On-Orbit Assembly]: In this research, on-orbit assembly is understood as the process of carrying out rendezvous and hard docking for a set of N modules in Low Earth Orbit, whereby the modules may be brought together using their own power and propellant or may be assembled by a separate spacecraft.

While the launch vehicle tradespace is relatively well understood, the other key piece of the puzzle has been given much less attention. On-orbit assembly of separately launched components is an equally important component of the infrastructure enabling human access to space. Reducing launch costs by using inexpensive EELV's is pointless if a complex and costly on-orbit assembly process is thereby necessitated. However, if the cost and risk of on-orbit assembly can be reduced, the launch tradespace could become more flexible, and the entire Earth-to-orbit architecture could be streamlined for affordability.

The Earth-to-orbit architecture encompasses all processes required to transport a spacecraft into LEO in its final configuration for transit to its destination. Thus, for conventional missions, the Earth-to-orbit architecture includes the launch and assembly processes, along with any other supporting processes such as orbit phasing, rendezvous, orbit loiters, etc. The focus of this thesis is on the on-orbit assembly portion, but because the launch architecture is closely linked to assembly, it is also studied in the context of its impact on assembly. In this thesis, we look at the entire Earth-to-orbit architecture and investigate the combined launch and assembly tradespace, with the goal of increasing affordability for large (usually manned) space missions. More detailed research goals are provided in Section 1.4 below.

1.2 Background

This section provides background for the study of on-orbit assembly. We first introduce some of the basics of on-orbit assembly, then provide historical background.

1.2.1 On-Orbit Assembly Basics

The ultimate goal of on-orbit assembly is to physically join two or more spacecraft or modules such that they function as a single spacecraft subsequent to the

assembly. On-orbit assembly is a relatively complex process, depending on several component processes to function correctly in sequence: the two (or more) spacecraft must rendezvous in space, match their orbits and orientations, then physically join through some mechanism.

Assuming both spacecraft modules are in orbit around the Earth (or the same planetary body), a rendezvous must be performed. The rendezvous process ensures that the two modules to be assembled are within some fixed distance of each other, moving at the same velocity relative to Earth and near-zero velocity relative to each other. Often this means they are in the same or very near orbits with one leading (target) and one trailing (chaser). Rendezvous is usually a complex task requiring significant efforts by ground planners and sophisticated hardware to measure spacecraft locations and ephemeris. The rendezvous trajectory must be planned carefully to ensure that collisions do not occur, and that propellant usage is kept within allowable limits. More information on this topic can be found in the literature, including [Fehse 2003].

The second task is to maneuver the vehicles into position for physical attachment. The key requirements here are to measure the relative states of the vehicles (such as orientation, range, angle, and speed) and to perform maneuvers to match the states. The measurement of spacecraft states depends on sensors with inherent errors, generating uncertainty in the spacecraft state measurements. This of course complicates the task of matching the spacecraft states, so the selection of onboard sensors is key to successful on-orbit assembly.

The third and final task is the physical joining of the spacecraft modules. This can be accomplished via several different methods; [AIAA 1992] defines each of these options. *Berthing* describes the process of using a grapple interface (such as a robot arm) to bring two modules together. *Docking*, on the other hand, refers to the joining of two spacecraft by “actively commanding the translational and/or rotational maneuvers necessary to bring them together and latch.” Generally, one spacecraft, declared the *active* spacecraft, performs these maneuvers, while the other spacecraft remains *passive* until docking is accomplished.

Clearly, docking and berthing impose very different requirements on the relative speeds and positions of the two spacecraft just before joining. For berthing, the spacecraft

must be maneuvered very carefully into position (zero relative velocity) so that the robot arm can grasp the passive spacecraft. On the other hand, docking mechanisms are generally designed to absorb some amount of error in the relative position of the modules, making the requirements on trajectory design and sensor measurements slightly less stringent.

The best examples of berthing mechanisms are the robotic ‘arms’ (or remote manipulator systems) of the Space Shuttle and the International Space Station. Many types of docking mechanisms exist. The earliest and simplest type is called a ‘probe-and-drogue’ system, shown in Figure 1.2, in which a probe on one spacecraft (the active spacecraft) is directed into the drogue (cone) on another spacecraft (the passive spacecraft). The disadvantages of such designs are that the probe-drogue assembly prevents human passage between the docked modules, and that the spacecraft cannot switch roles (one is male, one is female). In response to these difficulties, androgynous docking systems were developed, of which one example is the orbiter docking port (see Figure 1.3). Androgynous systems are those in which either port can function as the active or the passive side; this increases reliability (through redundancy) but also leads to additional mass and complexity.

In this thesis, we focus mainly on docking rather than berthing systems. Berthing is less forgiving in terms of trajectory, measurement, and control, and therefore necessarily requires more human involvement than docking. In the future, we are looking to reduce the costs and complexity of on-orbit assembly; the relative simplicity of docking seems the easiest route to this reduction.

1.2.2 History of On-Orbit Assembly

Earth-to-orbit architectures have been studied since the dawn of the space age. On-orbit assembly has been a key component of the most exciting manned space missions, including trips to the Moon and the creation of an orbiting research station. The component capabilities of on-orbit assembly – rendezvous and docking (or berthing) – have been a focus of the program almost since day one. Many of these component technologies are relatively mature as a result. This section provides an overview of historical experience in rendezvous, docking, and on-orbit assembly, informed in part by

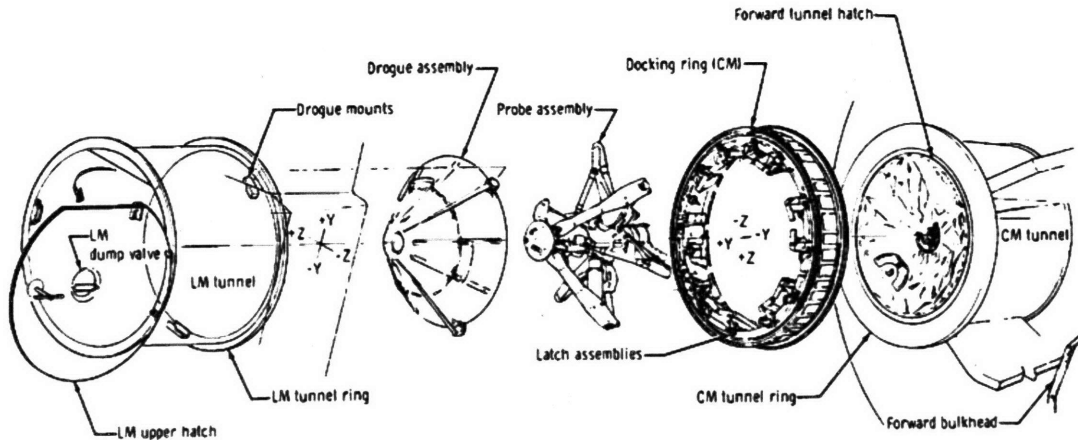


Figure 1.2: Apollo probe-and-drogue docking system. [Langley 1972]

the excellent perspectives provided by [Zimpfer 2005] and [Fehse 2003]. The goal is to determine the state of the art in *operational* on-orbit assembly, thereby establishing a basis for this study's look at the future of the technology.

1.2.2.1 Apollo

The American space program began its life in an effort to catch up to the Russians, who had stunned the world by launching Sputnik. It soon became clear that the best way to beat the Russians was to send men to the Moon, and thoughts quickly turned to the technologies that would be required to enable such a mission. One of those technologies was on-orbit assembly (on a small scale), and it was in fact the Earth-to-orbit architecture that drove this requirement for on-orbit assembly. Von Braun – the designer of the giant Earth-to-orbit Saturn rockets¹ – originally envisioned a ‘direct’ architecture, in which one huge rocket blasted a single spacecraft towards the Moon; the spacecraft would land, ascend, and return to Earth. However, the amount of mass required for such an architecture was virtually impossible to launch. Therefore, alternative architectures were studied, including Earth orbit rendezvous and lunar orbit rendezvous (which was ultimately selected). Both options required some form of in-space assembly. With the lunar orbit rendezvous architecture, the lunar module ascends from the lunar surface to rendezvous and dock with the command module in lunar orbit. Many

¹ The Saturn V rocket had a payload capacity of 118 metric tons to Low Earth Orbit in its 3-stage configuration.

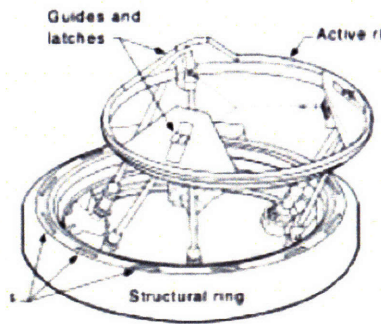


Figure 1.3: *The Space Shuttle docking mechanism is an androgynous design. [Zimpfer 2005]*

at NASA were wary of performing complex assembly tasks in distant lunar orbit, but the Earth-to-orbit launch constraints made such an architecture necessary.

This first attempt at in-space assembly relied heavily on human involvement. The ground crew did extensive planning for the rendezvous maneuvers, and the capture and docking were executed by the crew onboard the two spacecraft. The docking mechanism was a probe-and-drogue design, shown in Figure 1.2. The crew controlled the active spacecraft (probe) manually during the docking maneuver.

1.2.2.2 Shuttle

The Shuttle does not specifically perform on-orbit assembly as such, but it has performed several berthing/docking maneuvers, including the rendezvous and capture of the Hubble Space Telescope, and docking with space stations Mir and ISS. As in Apollo, the ground plans trajectories and the crew performs the final docking maneuvers, but more sophisticated automation and tools are employed on the shuttle than on the Apollo spacecraft. The orbiter's docking mechanism is also more sophisticated than that of Apollo, employing an androgynous design (shown in Figure 1.3).

1.2.2.3 International Space Station

The ISS is the grandest example of on-orbit assembly to-date (see Figure 1.4). [Goetz 2003] describes the complexity of the assembly task. In 2002, more than two million parts were on-orbit; when completed, the station should weigh almost one million pounds. The extreme complexity of assembling multiple parts built in various locations, many of which had never been connected on the ground, accounts for a large part of the

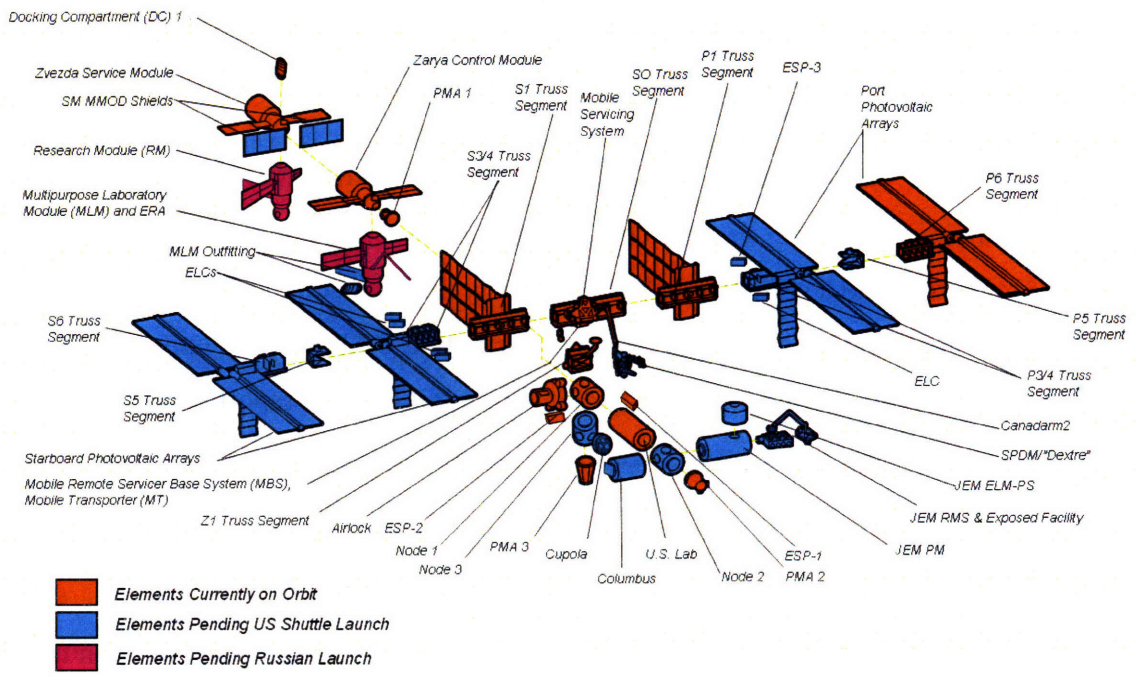


Figure 1.4: International Space Station configuration (courtesy NASA)

high cost of assembling ISS. At least five different types of attachment mechanisms are used on the station, various utility connections are required across most attachment points, and extensive testing is required for each connection. In addition, costs are driven up by the large number of EVA and IVA man-hours (extra-vehicular and intra-vehicular activities, respectively) required to put various pieces together. Nevertheless, assembly so far has been a technical success, despite being significantly over budget. The achievement shows that on-orbit assembly on a large scale is indeed technically feasible; the hurdle for next-generation programs is to make assembly *financially* feasible.

1.2.2.4 DART

The DART mission (Demonstration of Autonomous Rendezvous Technology) was intended to demonstrate autonomous rendezvous capability and to perform a series of close-range proximity maneuvers around a target spacecraft. When NASA's vision for space exploration was announced, DART became a high-profile mission because in-space assembly appeared to be a critical component of manned lunar or Mars missions. DART was expected to rendezvous autonomously with its target MUBLCOM, a retired military satellite outfitted years ago with special reflectors. Using an advanced suite of sensors designed to work with MUBLCOM's reflectors, DART would perform a series of proximity maneuvers around the spacecraft, designed to demonstrate the abilities of

the sensors and the navigation system for autonomous rendezvous and docking operations.

DART was ultimately unsuccessful in its demonstration of proximity maneuvers around the target MUBLCOM spacecraft, and in fact collided with MUBLCOM while attempting to avoid a collision. NASA's publicly released summary [NASA 2006] describes the causes of the mishap, largely attributing DART's problems to faulty navigation system software design. Despite these failures, the report emphasizes, the technology of autonomous rendezvous remains critical for NASA's long-term vision for space exploration.

1.2.2.5 Orbital Express

The Orbital Express mission is a more in-depth technology test mission than DART. Orbital Express aims to test the feasibility of on-orbit servicing by demonstrating the capabilities for autonomous rendezvous and docking, spacecraft refueling, and servicing through the attachment of 'plug and stay' ORU boxes. The project is funded by the Defense Advanced Research Projects Agency (DARPA) to prove at least the technical feasibility of on-orbit servicing.

The larger vision is an on-orbit servicing architecture in the post-2010 timeframe. The concept calls for low-cost launchers to fill on-orbit depots with propellant and other ORU boxes, such as avionics upgrades. A servicer spacecraft would load equipment from the depot for a particular target spacecraft, rendezvous and dock with its target, and perform refueling and servicing maneuvers, then return to the depot for its next mission. Orbital Express will demonstrate the feasibility of the idea.

The Orbital Express project is quite relevant for on-orbit assembly as well, because a number of the technologies to be demonstrated (autonomous rendezvous, docking, and refueling) could be key elements of an on-orbit assembly infrastructure [Dornheim 2006].

1.2.2.6 ETS-VII

In preparation for development of the H-II Transfer Vehicle (HTV) for ISS logistics support, the Japanese space agency (then NASDA, now JAXA) developed a spacecraft for rendezvous and docking tests. ETS-VII in 1998 performed the first

autonomous rendezvous and docking between two unmanned spacecraft. The target and chaser were launched together, then separated to test autonomous docking from a 2-m hold point. After resolving some anomalies with attitude control jets on the target spacecraft, a second test was completed with a rendezvous and docking from a 12-km range. Japanese technologies for autonomous rendezvous and docking in both the relative approach and docking phases were validated [Kawano 1999].

Other on-orbit assembly operations have been accomplished, but the overview provided here is sufficient to establish the state of the art in operational on-orbit assembly to date. (More detailed information on each of these past missions can be found in sources highlighted in the literature review below).

1.3 Literature Review

This thesis touches a broad range of issues dealing with Earth-to-orbit architectures, on-orbit assembly, and on-orbit servicing (related to assembly). Because the work focuses on the impact of on-orbit assembly, this literature review also focuses on on-orbit assembly literature, especially aspects dealing with methods and technologies for on-orbit assembly. In addition, due to the extensive study of space tugs for on-orbit assembly, we provide a short review of literature dealing with space tugs and on-orbit servicing architectures. A brief note on modularity is also included.

1.3.1 Assembly Literature

1.3.1.1 Missions Past, Present, and Future

The limited on-orbit assembly performed during the Apollo program is discussed by [Zimpfer 2005], which also discusses rendezvous and capture operations of the Shuttle.

The only major ongoing effort involving on-orbit assembly is the construction of the International Space Station. There is a large body of literature dealing with both the planning and operations of on-orbit assembly for ISS and for its previous incarnation, the Space Station Freedom. In an enlightening paper from the Freedom era, [Brand 1990] describes the issues and challenges facing NASA as the Space Station Freedom is

designed. Many of these issues remain to this day. Brand insightfully mentions that “factors that determine the difficulty of construction on orbit include the configuration of the station, capabilities of the transportation system that will carry components to orbit, and the actual magnitude of the assembly work required by extravehicular (EV) crewmen or robots.”

One of the best recent discussions of ISS assembly is provided by [Goetz 2003]. He provides an overview of systems engineering and management practices for the space station program, in the process providing an enlightening overview of the extreme complexity of the project. (Recall the discussion on ISS in Section 1.2.2, which is based on this article.) The successful assembly so far is attributed by Goetz to “sound systems engineering practices” and “ground test and verification programs.” A goal for future assembly programs could be to reduce the dependency on ground support and automate some of these functions. Detailed information on ISS assembly is also provided by [Covault 1997].

[Rumford 2003] summarizes the DART mission in great detail, focusing on the spacecraft design. A less technical summary is provided by [Iannotta 2005], which also describes more carefully the context and motivation for the project. Both were written before the mission. [NASA 2006] provides a post-mission report on the mission failures, summarizing the reasons for DART’s problems on-orbit. The complete NASA report is not available to the public.

[Dornheim 2006] provides a high-level overview of Orbital Express, including some discussion of the business case for on-orbit servicing. In addition, he discusses the context of the mission, including history, technology, etc. [Whelan 2000] describes the goals of the Orbital Express mission in detail, focusing on how the project benefits the Department of Defense and civil space programs by proving the feasibility of on-orbit servicing. He describes the vision of an on-orbit servicing architecture based on the technologies demonstrated by Orbital Express.

A short summary of the ETS-VII mission is provided by [AW&ST 1998], which gives relevant parameters for the successful tests completed by the satellite. A more in-depth summary of the mission and its objectives is given in [Kawano 1999].

1.3.1.2 Assembly Methods

One of the biggest questions discussed in the literature is the best method for on-orbit assembly; the major options are crewed assembly, crew-operated robotic assembly, automated robotic assembly, and autonomous assembly (and combinations of these four ideas).

[Purves 2002] looks at the cost-effectiveness of various assembly strategies, weighing the benefits of astronaut-assisted assembly against tele-operated or autonomous robotic assembly. His results do not show significant difference in cost-effectiveness between astronaut- and robot-assisted assembly efforts, although he does note that astronauts are expensive and must be used sparingly. However, he assumes that a facility which supports humans in the assembly orbit (such as ISS) is available. If the cost of creating and maintaining such a facility were added in, robotic assembly would most likely appear to great advantage.

[Muller 2002] also looks at the astronaut-robot tradeoff as part of his study of assembling a large telescope using ISS. He concludes that astronaut EVAs are too expensive and complex for the telescope, and that stringent requirements (such as avoidance of contamination) would make this method difficult to implement. Astronauts would supervise the complex task of assembling the telescope parts, while a robot carried out the assembly based on a pre-programmed, ground-tested sequence of maneuvers.

Like [Muller 2002], much of the remainder of the literature dealing with on-orbit assembly looks specifically at problems related to assembling complex (non-modular) structures in orbit, or assumes that humans are required for assembly. [Hand 2002] and [Weater 1987] do not even examine options that do not require a human in the loop; [Doggett 2002] discusses the design of truss structures for assembly in space; [Ayer 2001] also discusses assembly of a large complex structure, although it is labeled 'modular'; and [Senda 2002] looks at robotic autonomous assembly, but still focuses on complex truss structures.

[Akin 2002] describes a large database of work on human and robotic assembly of large space structures, concluding that humans and robots working together is the most effective assembly method. Again, the structure to be assembled is a truss, rather than a

series of modules that can be simply docked together. Most likely in this latter case astronauts would no longer be cost-effective.

In the past, on-orbit assembly has also been examined from the systems perspective, as we propose to do in this thesis. Most of the work is at least a decade old, however, and therefore less applicable to the problems of today.

[Morgenthaler 1990] and [Morgenthaler 1991] date from before the International Space Station assembly, but tackle many of the same issues we face today. The former discusses relevant concerns for on-orbit assembly of Mars missions, and the latter addresses the problem of the launch/assembly tradeoff for large space systems.

[Morgenthaler 1991] compares the cost of assembly based on cost models for the launch vehicle, spacecraft, docking, crew transportation, and facilities. These cost models are generally functions of the mass and/or risk associated with each component. His main purpose is to suggest that this type of model can assist with the choice of launch vehicle size for future Mars missions, and he draws conclusions for a sample Mars mission. He suggests that the optimal size for heavy-lift launch vehicles lies in the range between 100 and 200 tonnes. He also concludes (as we do) that smaller launch vehicles incur a greater risk of delays in assembly, while larger launch vehicles incur a greater risk of losing an expensive, important payload.

Perhaps the most relevant work is [Moses 2005]. He discusses plans to develop a model that compares life cycle costs for modular systems requiring in-space assembly. The goal is an understanding of how to score competing designs implementing different types of modularity. In February 2005, this study was in the planning stages only, so no results were available.

Finally, NASA's Exploration Systems Architecture Study (ESAS) team recently looked at on-orbit assembly in the context of lunar and Mars missions. Their report [NASA 2005a] concludes that assembly should be avoided to the extent possible, based in part on a requirement that "no more than four launches [shall be used] for a single human lunar mission." As a result, launch vehicles are limited to a "minimum payload lift class" of 70 mt, eliminating a large swath of the trade space. The original four-launch limitation is not discussed in detail, but we can infer that the idea of a greater number of

launches for one mission appeared too risky. In Chapter 2, we provide an analysis that shows this may not always be the case.

1.3.1.3 Assembly Technologies

Finally, we look at the literature on various technologies essential to on-orbit assembly. Successful assembly depends on a combination of many well-studied technologies including guidance, navigation (including sensing), and autonomy. These topics are entire fields unto themselves and the literature is therefore not reviewed here. However, we discuss one paper on the historical context for in-space assembly, and several others on docking system technologies.

As mentioned previously, [Zimpfer 2005] provides a very good overview of historical progress in rendezvous, docking, and in-space assembly (see Section 1.2).

For information specifically on docking systems, [AIAA 1992] and [Gonzalez-Vallejo 1993] provide good, detailed overviews of various types of systems. The European docking port designs are described in [Tobias 1989]. More recent articles include [Zimpfer 2005] and [Wertz 2003]. It is unfortunately difficult to get detailed information on the ADBS docking system currently under development at NASA, but [Lewis 1999] and [Fehse 2003] provide brief overviews; [NASA 2005] provided further information which cannot be published.

These papers paint the history of docking mechanism design, which is heavily weighted toward complex systems for manned spaceflight. The first American and Russian docking mechanisms – developed for the Moon programs – were both probe-and-drogue designs; subsequently, the Apollo-Soyuz program sparked the development of the Androgynous Peripheral Assembling System, the ancestor of all androgynous docking systems. Improvements to this system resulted in the Androgynous Peripheral Docking System (APDS), currently used to dock the Shuttle to the ISS. The system weighs 330 kg and measures 1.5 m in diameter. A couple of other systems are/were developed for ISS: the European Hermes-Columbus system allows low approach velocities, and the Common Berthing Mechanism mates the large space station modules together on the ISS, but is designed for berthing only. Finally, the Advanced Docking and Berthing System (ADBS; previously called Low Impact Docking System or LIDS) is currently under development at NASA, and is designed for low impact velocities. It

weighs about 350 kg including avionics and a hatch. A comparatively small number of mechanisms have been developed for unmanned missions. The ETS-VII mission, DART, XSS-11, and Orbital Express all included docking ports, but their designs are not discussed extensively.

Table 1.1: Overview of Major Docking Systems

	Androg. (Y/N)	Mass (kg)	Diam. (m)	Vel. (m/s)	Lat. V. (m/s)
Apollo CSM-LM [Langley 1972]	N	140	0.8	0.03-0.3	0.15
Apollo-Soyuz APAS-75	Y	264	0.8	0.2-0.4	0.3
Shuttle-Mir/ISS APDS / APAS-89	Y	330	0.8	0.05-0.15	0.25
ADBS / LIDS	Y	~350			

Relevant parameters for the major docking system designs are summarized in Table 1.1. This summary shows that standardized docking system designs have focused mainly on highly capable mechanisms for complex manned space vehicles (or stations); these systems weigh on the order of 300 kg. For docking unmanned modules, in which a transfer tunnel and perhaps utility connections are not needed, it is likely that a much lighter design could be created, but no standardized systems have been developed. We therefore rely in this thesis on the mechanisms thus far created, but keep in mind that lighter systems could most likely be developed. The sensitivity of on-orbit assembly strategies to docking mechanism mass is explored later on.

1.3.2 On-Orbit Servicing Literature

On-orbit servicing is closely related to assembly, because it depends on many of the same component technologies – rendezvous, docking, and proximity maneuvers between two spacecraft. In addition, because this work looks at the use of space tugs for on-orbit assembly, the literature on on-orbit servicing is relevant. In this section, we provide an overview of papers discussing various aspects of on-orbit servicing.

As mentioned earlier, [Whelan 2000] makes the case for an on-orbit servicing architecture in the post-2010 timeframe; Orbital Express will prove at least the technical feasibility of the idea. The concept builds on the air force’s in-air refueling capability and easy avionics upgrades, suggesting that in-space refueling and ‘plug-in’ avionics box upgrades could make an effective on-orbit servicing infrastructure.

[Moe 2005] notes that robotic servicing in space has been examined in-depth with reference to the Hubble Space Telescope (HST) program. He suggests several operational ideas for demonstrating assembly and servicing using the planned HST servicing architecture.

[Turner 2001] makes a case for an extensive on-orbit servicing architecture in which spacecraft are entirely dependent on servicers for orbit maintenance and other “non-intrusive” tasks (e.g. no equipment upgrades or repairs). Such an architecture would allow more cost-effective spacecraft design by reducing requirements for large propellant tanks.

[Saleh 2002] proposes a new systems-type approach to assessing the value of on-orbit servicing. By taking into account the flexibility provided to spacecraft designers by servicing opportunities, and by studying the value (price) under which servicing would be useful, new conclusions can be reached to guide the future development of on-orbit servicing architectures. A companion paper, [Lamassoure 2002], applies the new flexibility-based valuation framework to two types of space missions: commercial missions with uncertain revenues and military missions with uncertain target locations. The framework is shown to generate new conclusions on the value of on-orbit servicing.

[McManus 2003] looks at on-orbit servicing from the systems perspective, examining a very large tradespace for orbital transfer vehicles (a special type of servicer designed to modify orbits). The study of this large but crudely modeled tradespace of vehicle designs helps to identify families of feasible and cost-effective designs. The major vehicle design types that emerge are: an electric tug that makes a one-way trip from LEO to GEO, a ‘Nuclear Monster’ depending on nuclear thermal propulsion that can make the round trip to GEO and back, and smaller ‘Tenders’ with a storable bi-propellant propulsion system, suitable for missions within LEO. This research focuses on the utilization of this last family of designs.

In a related paper, [Galabova 2003] examines two of these families of tug designs in greater detail. She determines that the design is driven by the mission scenario of the servicer (e.g. GEO retirement or LEO servicing), and therefore the tugs must be designed differently for each mission (one universal tug is not feasible). She looks at the business case and concludes that these two sample missions can be cost-effective, if the tugs are

optimized for each scenario. [Galabova 2003] also provides an excellent literature review of previous work in on-orbit servicing, more comprehensive than that provided here.

1.3.3 Modularity Literature

Because modularity is an enabling concept for the assembly techniques we study in this thesis, it is worth mentioning the spacecraft modularity literature here. [Nadir 2005] defines modularity as “the clustering of the functions of a system into various modules while minimizing the coupling between the modules and maximizing the cohesion among the modules.” He identifies other possible definitions, the most useful of which labels modularity as “the standardization of interfaces between design elements and the reuse of functional units.”² For our purposes, the concept of a modular spacecraft embodies *at least* the idea of standard interfaces, and the division of functionality into smaller units. These two concepts lessen the complexity of both the vehicle design and assembly processes significantly.

[Nadir 2005] provides a good literature survey for modularity, so this can be referred to for additional background on the topic (see Nadir’s Chapter 4).

1.3.4 Literature Summary

In summary, the literature on in-orbit servicing is largely recent, and deals mainly with far-off servicing architecture concepts. The Orbital Express mission is the only current implementation of such capabilities. The same technologies are utilized in on-orbit servicing and in-space assembly, so the literature dealing with the former topic is relevant to this work. In addition, the systems analysis techniques developed for study of servicing can be adapted for use in the study of on-orbit assembly strategies.

A large body of literature exists dealing with on-orbit assembly, but much of it focuses on the assembly of large, complex structures. Very little research has focused on the assembly (or design) of modular spacecraft. Also, most of the assembly literature focuses on specific technical issues; only a few papers view the problem from a systems perspective. This thesis addresses these deficits by looking at how modularity can ease

² This quote is Nadir’s summary of [Enright 1998].

the technical complexity of assembly, and also by employing a systems perspective to analyze the best assembly strategies.

1.4 Research Goals

In this chapter, we have laid out the background for understanding the difficulties involved in assembling large spacecraft in orbit. This problem of assembly is actually part of the larger challenge of transporting large spacecraft from Earth to orbit. An affordable launch architecture is a prerequisite to an affordable on-orbit assembly strategy. Launch and assembly are inextricably linked, and must be considered together, as we will discuss in Chapter 2. Therefore, this thesis looks at entire Earth-to-orbit architectures, and attempts to increase their affordability with better launch and on-orbit assembly strategies.

Based on the background – history and literature review – presented earlier in this chapter, it is clear that on-orbit assembly is technically feasible but generally has proven quite expensive. Similarly, on-orbit servicing seems feasible but has not been conclusively proven cost-effective. The challenge, then, is to develop new methods for on-orbit assembly that build on previous experience but make operations more affordable. Recall that in the past the individual assembly tasks have been quite complex, and have depended on human involvement: building trusses, constructing telescopes, etc. Today, with the flexible architectures provided by more modular spacecraft designs, we can develop new, more affordable strategies for on-orbit assembly.

The goal of this research is to understand the impact of on-orbit assembly on the system-of-systems that makes up the space exploration mission. If a spacecraft must be launched in several pieces, what is the impact on the vehicle's design? How can the costs of launch and on-orbit assembly be reduced? We hypothesize that past experience in assembly and new technologies for on-orbit servicing can be leveraged to find methods for making assembly less costly, especially by utilizing new modular spacecraft designs. In addition, we suggest that an examination of the combined launch-and-assembly tradespace will yield new insights about the impact of assembly requirements on launch and transportation architectures. In short, we will examine how on-orbit assembly affects

space exploration missions, and develop methods for increasing the affordability of such missions by designing systems specifically for on-orbit assembly.

This research is divided into two separate but related parts. First, in Chapter 2, we examine what it takes to make an architecture ‘assemble-able’. In other words, what changes must be made to a transportation architecture in order to make it easily and cheaply assemble-able in Earth orbit? How can large vehicles be modularized for ease of launch and assembly, and how can launch vehicles be selected to minimize the costs of the Earth-to-orbit transportation? These questions are addressed through the modeling of the combined launch-and-assembly tradespace.

Second, in Chapter 3, we look more specifically at strategies for on-orbit assembly of modular spacecraft. We compare various assembly methods quantitatively, in particular focusing on the comparison between self-assembled missions and the utilization of an on-orbit servicer, or space tug, to assist in the assembly task. The goal is to find out the types of assembly tasks for which a space tug is valuable, in order to gain an understanding of the value of such a flexible, reusable on-orbit assembly infrastructure. More specific research goals are provided in the relevant chapters. Chapter 4 summarizes our conclusions and points to directions for future research.

1.4.1 Notes

Because any large space mission undertaking can encounter the same types of problems, we take human space exploration as a representative case study throughout this work. However, the conclusions reached in this thesis apply equally to any large space undertaking, whether its goal is exploration or anything else, and regardless of whether it carries humans.

A review of the acronyms used commonly throughout this thesis is provided in Appendix A.

2

Launching Assemble-able Architectures

What factors contribute to the ease with which a spacecraft can be assembled? In other words, what makes an ‘assemble-able’ architecture? These are the questions we address in this chapter. More specifically, we look at a sample manned lunar/Mars transportation architecture, and examine how it can be modified to make it more easily launched and assembled. The emphasis here is on the launch component; the assembly of similar architectures is addressed in Chapter 3.

First, we give a qualitative overview of the challenges of designing for launch and assembly. The second section introduces the idea of taking an optimized design for a

transportation architecture and breaking it into ‘chunks’ that can be launched and assembled. Third, we build upon this idea to find an optimal launch vehicle size, thereby examining the combined launch and assembly tradespace for the sample transportation architecture. By choosing launch vehicles based on the transportation architecture and changing the architecture to accommodate various launch vehicles, we provide a quantitative enumeration of the launch vehicle trade space. Iteration between in-space architecture design, chunking and launch vehicle selection is necessary to arrive at an optimal solution.

2.1 Designing for Assembly

Designing for assembly is no easy task. More than a decade ago, as the Space Station Freedom was being designed, [Brand 1990] insightfully recognized many of the challenges to be faced in on-orbit assembly, writing, “factors that determine the difficulty of construction on orbit include the configuration of the station, capabilities of the transportation system that will carry components to orbit, and the actual magnitude of the assembly work required by extravehicular (EV) crewmen or robots.” In hindsight, he was entirely correct, and his warnings ring equally true today. The ISS program has grown increasingly expensive in part because of the large amount of assembly work that requires the involvement of humans (either through EVA’s or on-site operation of the robotic arm). The ISS modules are not designed to be easily assembled without human assistance, and additionally, the transportation system expected to loft *all* the large modules to orbit (the Shuttle) is quite expensive and subject to costly problems and delays. All these problems must be surmounted in order to plan affordable Moon and Mars programs.

Other challenges in designing for assembly include timing constraints and launch risk. Certain modules, such as those containing high-performance H₂/LO_x propellants, are subject to boil-off problems and cannot be left waiting in orbit for long periods of time. It is also risky to leave any module loitering in space, as problems can develop over time that the ground cannot fix. Additionally, the need for multiple launches can be said to increase risk, because a number of launches must be successful in order for the mission to succeed.

The biggest challenge in designing an architecture for on-orbit assembly, however, is understanding how to *modularize* a set of vehicles so that they can be both launched and assembled easily. The smaller the module, the easier and cheaper it is to launch, yet small modules make the assembly process more difficult because more rendezvous-and-docking operations are required. Moreover, small modules can increase the ‘mass penalty’ for docking equipment and other additional mass due to low volumetric efficiency. On the other hand, large modules necessitate dependency on large, expensive launch vehicles (like the Shuttle), but are significantly easier to assemble.

The following sections address these challenges, demonstrating a process for finding the optimal balance between ease of assembly and ease of launch; in other words, the best way to design an *assemble-able architecture*.

2.2 Chunking and Manifesting

In this section, we describe a generalizable process for breaking large spacecraft into ‘chunks’ that fit on launch vehicles and can be assembled in orbit. While the process is general, we show it for a representative case study: a transportation architecture designed to send humans to the Moon and Mars [Crawley 2005]. We first describe this sample transportation architecture, then discuss how to break it into launch-able, assemble-able pieces.

2.2.1 Sample Transportation Architecture

A transportation architecture can be defined as a set of vehicles used to transport crews and cargo between Earth and the Moon or Mars. In this chapter, we consider one set of lunar/Mars transportation architectures developed as part of a Concept Exploration and Refinement (CE&R) study at MIT/Draper [Crawley 2005]. These architectures were created using a “Mars-back” approach, considering requirements for missions to the Moon and Mars in parallel and designing common elements (modules) to be used in both types of missions. The resulting architectures consist of sets of modular vehicles that transport crew and cargo between Earth and the Moon or Mars.

Figures 2.1 and 2.2 outline the baseline transportation architecture. Figure 2.1 illustrates the operations concept for lunar and Mars missions, and Figure 2.2 provides

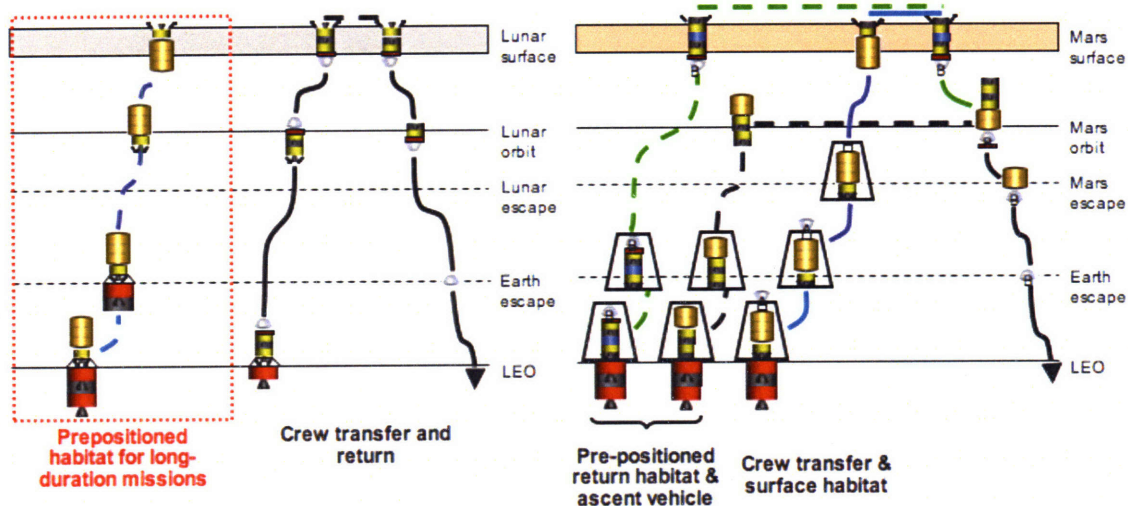


Figure 2.1: Operations concepts for lunar and Mars missions. For lunar missions (left), the crew transfers to the surface and returns to Earth in a single vehicle. For long-duration missions, a surface habitat can be pre-positioned on the surface. For Mars missions (right), the crew lands in the surface habitat. An ascent vehicle is pre-positioned on the surface, and a return habitat is pre-positioned in Mars orbit.

mass breakdowns for each of the vehicles used in these missions. The study envisions three distinct types of missions. First, a series of lunar ‘sorties’ of short duration – approximately 7 to 10 days – could be sent to various landing locations on the Moon, in the same manner as the Apollo missions. Second, a lunar base could be established and crewed during long-duration lunar missions. Third, a (necessarily long-duration) mission would be sent to Mars, building on the experience of long-duration exploration on the Moon.

For short lunar missions, a single ‘vehicle’ (stack of modules) ferries the crew to the lunar surface and back. This is the so-called ‘direct’ lunar architecture; its counterpart in Apollo was called ‘lunar orbit rendezvous’ (the ‘direct’ architecture could not be accomplished easily with 1960’s technology). The crew compartment is called the Crew Exploration Vehicle (CEV). The stack also includes a small cargo module, a CH₄/LOx lunar descent/ascent module for landing on and leaving the lunar surface, and a H₂/LOx Earth departure stage (EDS) for the trans-Moon injection (TMI) and lunar orbit insertion (LOI) burns. This stack is called the lunar Crew Transfer System (CTS). For the long lunar missions, a similar stack could be used to pre-position a lunar habitat (with the uncrewed habitat replacing the CEV, and two EDS stages).

Mars missions utilize the same set of hardware in different configurations, with an added heat shield for aerocapture in the Martian atmosphere. First, an ascent vehicle is

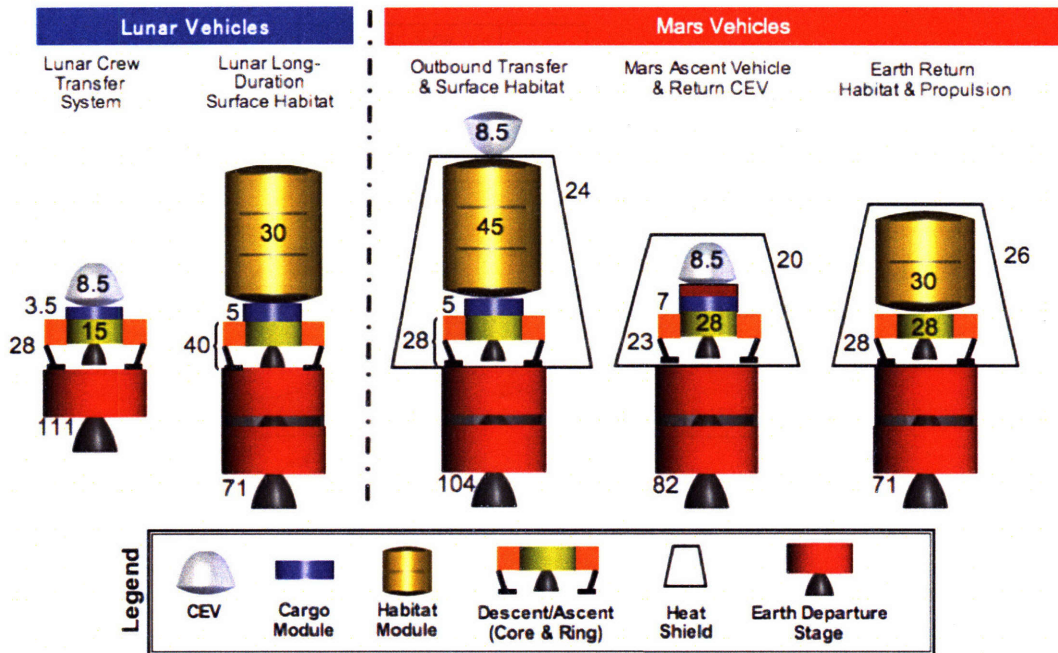


Figure 2.2: Vehicles for lunar and Mars missions are shown. Mass breakdowns (metric tons) are provided for each of the vehicles shown in Figure 2.1.

pre-positioned on the Martian surface, with an uncrewed CEV ready for the return flight. At the same time, the Earth return habitat (along with its propulsion stages for the return flight) is pre-positioned in Martian orbit. Finally, the crew transfers in an ‘outbound and surface habitat’ to the Martian surface and performs its mission, then transfers to the ascent vehicle for launch into Mars orbit and lastly into the Earth return habitat for the long journey back to Earth.

This transportation architecture was selected after a survey of over 1100 possible architectures. The best of these were chosen for further refinement with an eye toward designing vehicles with common hardware across lunar and Mars missions. As a result, the architecture shown here includes a high degree of modularity. Still, the habitat, landing, and propulsion stages are sized *optimally* for each mission. Little thought has been given (so far) as to how these vehicles could be launched into orbit.

2.2.2 Launch ‘Chunking’

With a baseline transportation architecture defined, the next step is to figure out how to get the required vehicles into low Earth orbit (LEO). None of the currently available launch vehicles can launch the stacks entirely, and even the planned Ares

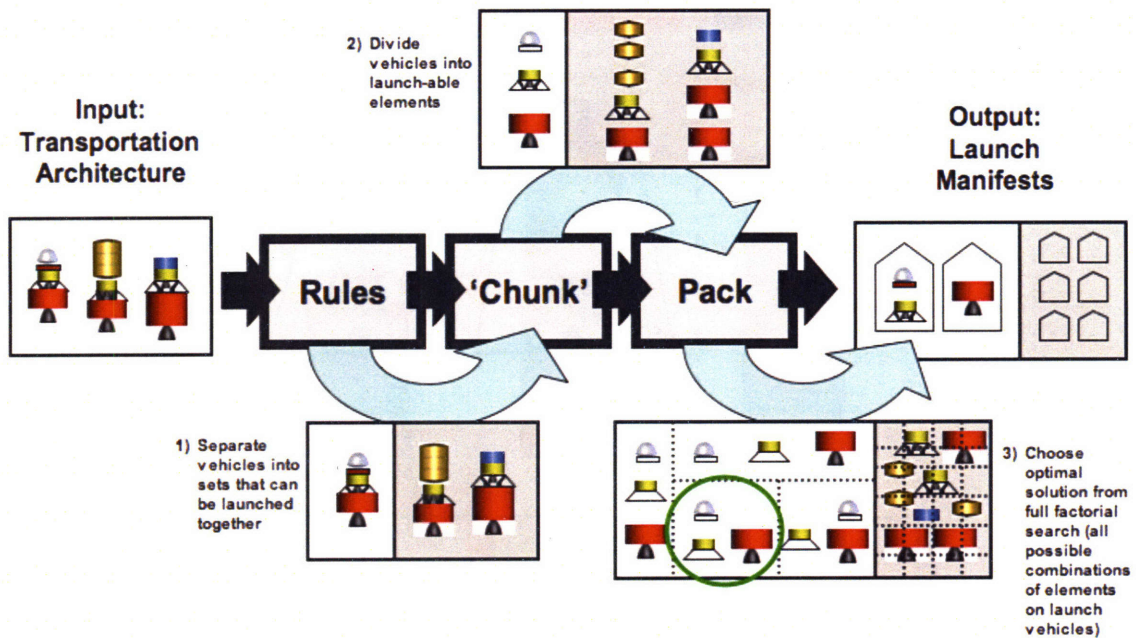


Figure 2.3: An overview of the launch manifesting process is shown. The input is a transportation architecture, which is divided according to rules into combinations of modules that can be launched together. The modules are divided into launch-able ‘chunks’, and the optimal packing arrangement is chosen from among all possible combinations to yield the fewest launches. The designer examines the results, tweaks the rules and chunking strategy, and repeats to converge on an optimal design.

heavy-lifters [NASA 2006a] are not equal to the task (except possibly for the lunar architecture). Clearly, the stacks must be launched in smaller ‘chunks’ or modules, which can then be assembled in orbit. The question remains: how large should the modules and the launch vehicle be? The remainder of this chapter attempts to answer this question.

Based on the transportation architecture defined above, we must determine how many launches are required, and what elements are launched on each vehicle. This is a three-step process, consisting of:

1. Logical rules governing allowable combinations of modules on launch vehicles.
2. Division of large modules into elements that fit on smaller launch vehicles.
3. Packing elements efficiently into launch vehicles.

Figure 2.3 shows an overview of this process.

The first step – defining rules – is relatively simple; our baseline analysis utilizes only very basic rules. Each vehicle stack is launched separately from all others. In some cases, we further assume that crewed modules (the Crew Exploration Vehicle) are launched on a separate human-rated launch vehicle (not considered in this analysis). Other rules can potentially be added to support trade studies; for example, in order to assess the value of a low-cost, low-reliability launcher, we could further require that consumables (e.g. propellant) not be launched with any other type of cargo.

The second step is to divide large vehicles into ‘chunks’ that fit on smaller launch vehicles. While the modular vehicle design provides natural breakpoints, dividing vehicles into their component modules does not always generate elements that can be launched on small (e.g. 28 mt) launch vehicles. For example, the lunar CTS (see Figure 2.2) could be divided into a CEV weighing around 9 metric tons, cargo totaling about 4 metric tons, a lander-and-ascent module at 43 metric tons, and an EDS with a wet mass of 111 metric tons. The latter two modules would not fit on current expendable launch vehicles (with maximum capabilities to LEO of about 28 metric tons). The Ares I (formerly referred to as the Crew Launch Vehicle CLV) has about the same capability (25 mt). While the planned Ares V heavy-lifter (130 mt) might be able to loft nearly an entire stack to low Earth orbit, Mars-bound vehicles would certainly exceed its capability and we would encounter the same problem again [NASA 2006a]. These ‘natural’ or atomic elements must be subdivided further to fit on launch vehicles.

Unfortunately, simply dividing an element’s total mass into launch-able ‘chunks’ does not generate an accurate model for the launch strategy, because it does not take into account the extra mass required to create separate modules from a single monolithic element. Creating two habitat modules from one large habitat would require additional structure and docking ports, at a minimum. Therefore, a ‘mass penalty’ can be imposed on any elements divided in this manner to account for this extra mass. However, we employ more accurate methods of modeling this modularization for specific types of modules.

In the case of our baseline architecture, two types of modules require further division into launch-sized chunks: the trans-Moon/Mars-injection (TMI) stages and the habitats. The TMI modules are relatively simple (tanks, propellant, and engines) and can

be modularized into launch-sized elements by staging the TMI burn. The rocket equation (3.6) is used to model the mass of each stage based on a maximum allowed stage mass, the required delta-V, and a mass fraction of 0.11 (based on [Wertz 1999]). By sequentially burning and dropping the TMI modules, this staging process can be advantageous until the mass of each stage becomes so small that the added mass of an additional set of engines outweighs the benefit of dropping the module when its propellant is spent. By staging the burn, the TMI module can be broken down into any number of stages in order to generate modules that fit on virtually any launch vehicle.

The habitats are more difficult to divide. The CE&R project developed a model to size full (un-modularized) habitats, and modified it to generate habitat ‘plugs’ – sections of the habitat that can be plugged together with end-caps to create a single pressure vessel [Crawley 2005]. Alternative modularization options include launching habitats without their internal subsystems and outfitting them separately [NASA 2005a], or designing more modular vehicles based on a concept such as truncated octahedral [Nadir 2005]. Any other modules for which specialized models are unavailable can be broken down – modularized – by dividing the mass of the full element into launch-sized modules and adding a ‘mass penalty’ for the extra structure and other hardware required. The mass penalty is not easy to estimate, and depends strongly on the type of vehicle. A simple and generalizable estimation method is to find the mass of the docking port to be added (about 300 kg – see Section 1.3.1.3), and apply a structures mass fraction to estimate the additional structure required. This method will generate rough estimates, but should be sufficiently accurate for trade studies.

2.3 Launch Vehicle Sizing Model

The process outlined in Section 2.2 creates a series of launch-able modules that must be packed into launch vehicles. For each launch vehicle size, an optimal packing solution must be found that minimizes the number of launches required for the overall architecture. This is a nontrivial task, because the problem grows rapidly with an increasing number of modules.

2.3.1 Full Factorial Search

The most straightforward method of solving this problem is a full factorial search. A full factorial search is performed by generating all possible combinations of modules on launch vehicles. The optimal solutions are those with the lowest number of launches. When several different optimal solutions exist, one can be chosen arbitrarily, or other screening criteria can be included here (e.g., give preference to solutions that launch elements in their final assembled configurations). The solution is found using a model developed in Matlab; the code for this model is given in Appendix B. Invalid launch solutions (sets of modules that the vehicle cannot launch) are screened out, then the number of launches is compared across all valid solutions. Only the optimal solution is reported for each launch vehicle size. With this full factorial search process, optimal launch manifests can be generated for a wide range of module sets and launch vehicle sizes.

Unfortunately, the time required to solve the full factorial search problem grows rapidly with the number of modules because all possible combinations of modules on launch vehicles must be computed. For example, the lunar vehicles outlined in this chapter can be packed into launch vehicles using this method, but the time required to find optimal launch manifests for Mars vehicles is prohibitively long. We can calculate how the problem grows as both the number of modules and types of launchers are increased. The number of launch packing solutions p with *only one launch vehicle type* can be found for a set of n modules using Eq. 2.1.

$$p = \sum_{k=1}^n \frac{n!}{k!(n-k)!} \quad (2.1)$$

The term inside the summation in Eq. 2.1 describes the number of combinations of n modules taken k at a time. To describe all possible (not necessarily feasible) launch manifest configurations, we must sum over all values of k . Clearly, many of these possible solutions are infeasible (e.g. launching all modules on one launch vehicle – n vehicles taken n at a time, or $k = n$ – is infeasible in most cases). In addition, we have a requirement to launch all the modules, which requires combining some of the solutions

counted in Eq. 2.1 (i.e. it finds the 1-2 and 3-4 module stacks as separate solutions; they must be combined for our purposes into one solution dictating two launches). However, the search space is *bounded* by the number of possible solutions p found by Eq. 2.1. If we now consider multiple *types* of launch vehicles the search space increases further. For each configuration found by Eq. 2.1, there are now m possible launch vehicles. The search space grows as a tree with branching factor m and depth p , wherein the number of leaves in the tree q represents the number of possible solutions. Therefore, the number of solutions is bounded by

$$q = pm^p \tag{2.2}$$

Clearly, the problem grows large quite quickly. For a simple 5-module scenario, there are 31 possible launch configurations if only one launch vehicle type is available; with two types, the problem grows exponentially, and we must consider on the order of 2^{31} solutions, or over 66 billion possibilities. As mentioned earlier, many of these are infeasible or must be combined to create feasible solutions; nevertheless, this analysis provides an idea of the difficulty of the problem.

2.3.2 Integer Optimization

Because of the increasing size of the full factorial search, we attempt to formulate the problem so that we can take advantage of existing optimization methods. To that end, we define the problem more formally, incorporating cost into the objective function (rather than simply minimizing the number of launches).³

2.3.2.1 Formal Problem Statement

There is a set of n items (cargo), each with a mass of m_i , ($\forall i = 1, \dots, n$). There is a set of M launch vehicles each having a mass capacity of v_j , ($\forall j = 1, \dots, M$). Each vehicle has a fixed launch cost c_j and a variable cost per unit of mass surplus (i.e. unused mass-to-orbit capability) denoted by α_j . The objective is to find the best way to manifest

³ This integer optimization formulation was created by Hamed Mamani with my assistance. We worked together to find a better way to solve the problem but the integer optimization itself was performed by Hamed. The results analysis is my own.

all the items on some set of vehicles such that the launch and mass surplus cost are minimized. Below, a mathematical formulation for the problem is given, and then we look at some of the implementation aspects of this formulation.

2.3.2.2 Problem Formulation

The variables are defined as follows:

- n number of items (cargo) = number of modules
- M number of launch vehicles
- m_i mass of item i , ($i = 1, \dots, n$) [mt]
- v_j capacity of vehicle j , ($j = 1, \dots, M$) [mt]
- α_j variable cost per each unit of mass for vehicle j , ($j = 1, \dots, M$) [\$/kg]
- c_j launch cost of vehicle j , ($j = 1, \dots, M$) [\\$]

The decision variables are:

- $x_{ij} = \begin{cases} 1 & \text{if item } i \text{ goes into vehicle } j \\ 0 & \text{otherwise} \end{cases}$
- $y_j = \begin{cases} 1 & \text{if vehicle } j \text{ is not empty (i.e. used)} \\ 0 & \text{if vehicle } j \text{ is empty (i.e. not used)} \end{cases}$

With this notation, the objective function stated in 2.3.2.1 can be written

$$Z = \sum_{j=1}^M c_j y_j + \sum_{j=1}^M \alpha_j \left(v_j - \sum_{i=1}^n m_i x_{ij} \right) y_j \quad (2.1)$$

The first term in Equation 2.1 describes the fixed cost of all launch vehicles, and the term in parentheses describes the unused space on all vehicles; this is multiplied by the cost per unit mass of unused space. This effectively mimics the opportunity cost of launching only partially filled launch vehicles. There is a nonlinear term in this objective, but it can be ignored in the case that all values of α_j are equal (i.e. $\alpha_j = \alpha_k = \alpha$, $\forall j, k$), as

shown in Eqs. 2.2. (All values of α_j are clearly equal if only one type of launch vehicle is available, or if the cost per unit mass is the same for all available launch vehicles).

$$\begin{aligned}
Z &= \sum_{j=1}^M c_j y_j + \sum_{j=1}^M \alpha_j \left(v_j - \sum_{i=1}^n m_i x_{ij} \right) y_j \\
&= \sum_{j=1}^M c_j y_j + \alpha \sum_{j=1}^M v_j y_j - \alpha \sum_{j=1}^M \sum_{i=1}^n m_i x_{ij} y_j \\
&= \sum_{j=1}^M c_j y_j + \alpha \sum_{j=1}^M v_j y_j - \alpha \sum_{i=1}^n m_i
\end{aligned} \tag{2.2}$$

Note that the last equality is obtained because in any feasible assignment $\sum_j x_{ij} y_j = 1$, and since $\alpha \sum_i m_i$ is a constant, we can simply ignore it in the optimization objective function. As a result, the overall optimization problem is as follows:

$$\min \sum_{j=1}^M (c_j + \alpha v_j) y_j \tag{2.3}$$

subject to

$$\begin{aligned}
\sum_{j=1}^M x_{ij} &= 1, \quad \forall i = 1, \dots, n \\
\sum_{i=1}^n m_i x_{ij} &\leq v_j y_j, \quad \forall j = 1, \dots, M \\
x_{ij} &\in \{0,1\}, \quad \forall i = 1, \dots, n, \quad \forall j = 1, \dots, M \\
y_j &\in \{0,1\}, \quad \forall j = 1, \dots, M
\end{aligned} \tag{2.4}$$

The first constraint ensures that every item will be assigned to a vehicle. The second constraint ensures that if a vehicle is used (i.e. $y_j = 1$), it satisfies the capacity constraint, i.e. it cannot be overloaded. Moreover, if a vehicle is not used (i.e. $y_j = 0$), the second equation forces $x_{ij} = 0$ for all i (i.e. no items can be assigned to this vehicle).

In this problem, since we want to find the optimal launch solution, we do not have the exact value for M , the number of vehicles. However, since each item would be assigned to at most one vehicle, the number of vehicles required is bounded by the

number of items. As a result, we need to consider at most n vehicles of each type; thus, M is the product of n and the number of types of vehicles.

Before discussing the implementation, it is worthwhile to comment briefly on this formulation. The above formulation is an *integer optimization* problem, since x_{ij} and y_j are integer variables (restricted to be either '0' or '1'). These types of problems are known to be *hard* problems in the context of mathematical programming. In fact, there is not a known algorithm to solve these types of problems efficiently (i.e. in polynomial time as a function of the number of variables). More specifically, the problem posed here is known to be hard even when there is only one type of vehicle, since it reduces to the classical bin packing problem [Coffman 1997].

One of the most effective methods of solving these types of problems is the OPLstudio software package. It attempts to first solve the same model without the integrality constraints, called the linear relaxation model. It then searches for feasible and optimal solutions using the solution to the relaxation, and implementing branch and bound and sometimes some heuristic methods. To implement this formulation in the software, we also added a set of inequalities known as *valid inequalities* to strengthen the formulation and improve the solving time of the problem [Gralla 2005].

Initial attempts to solve the launch packing problem have been successful for significantly larger problems (i.e. more modules to be packed) than the full factorial method (results provided in Section 2.4.2), but an effective solution method for all problem sizes has not been found. However, this formulation shows potential for efficiently solving the launch packing problem, and incorporating the essential cost metric into the objective function. Moreover, the stated formulation and solution methods are very flexible in terms of adding more constraints to the model (e.g. incorporating volume constraints), unlike the classical packing algorithms, which break when the formulation is changed.

2.4 Launch Vehicle Sizing Results

The final step is to determine what launch vehicle is the best choice. The answer depends on the standard metrics of cost and risk. The integer optimization formulation above incorporates cost in the metric, but results have not been consistently produced

using this method for the full range of launchers and modules desired. Therefore, we turn first to the full factorial search method outlined in Section 2.3.1, in which the ‘surrogate’ metrics of number of launches and launch mass surplus are substituted for cost in the objective function. The number of launches affects the total launch cost as well as the mission risk (both in terms of launch reliability and required on-orbit assembly operations). A low launch mass surplus also leads to reduced launch costs (less ‘wasted’ launch capacity). Therefore, the optimal launch vehicle choice should have a low mass surplus and a relatively low number of launches (to minimize risk). The first section below summarizes the results from the full factorial solution method; subsequently, we provide results from the integer optimization problem.

2.4.1 Optimal Launch Vehicle Size Selection

Results for the mass surplus and number of launches metrics for both lunar (long-duration) and Mars missions, across a series of launch vehicle sizes, are shown in Figure

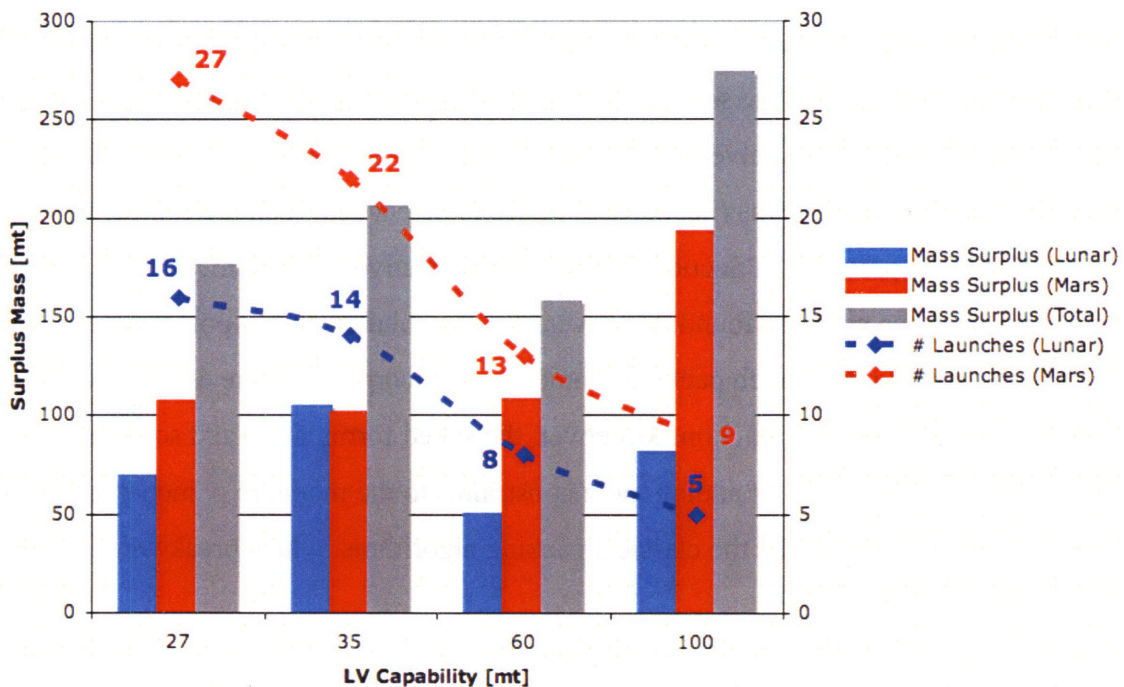


Figure 2.4: Surplus mass and number of launches are shown for various launch vehicle capacities (27, 35, 60, and 100 mt). Both lunar and Mars missions are shown. Note that the lunar mission shown here includes the launch of a long-duration habitat and other equipment for a 180-day stay on the lunar surface.

2.4.

The results indicate that certain launch vehicle sizes are significantly more efficient (less mass surplus) than others; in the case of our baseline architecture, the 60-

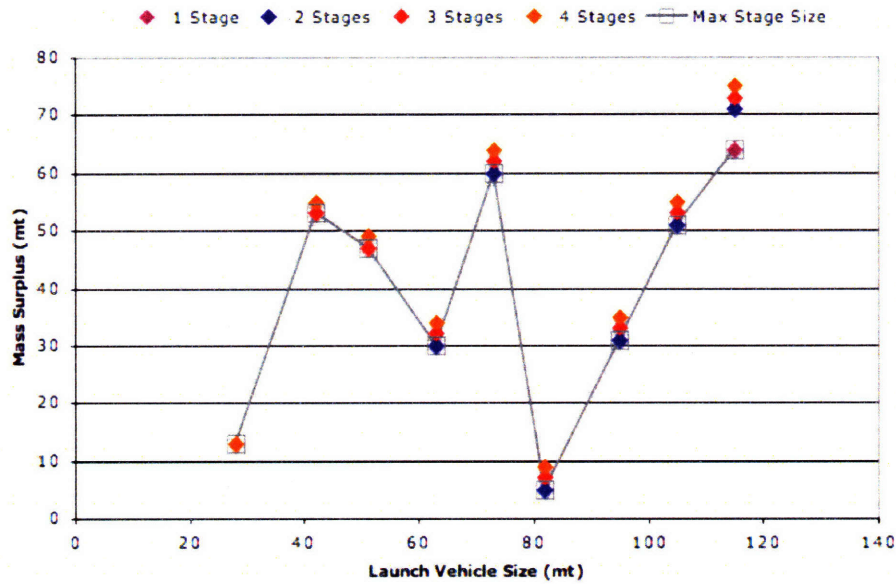


Figure 2.5: Lunar CTS launch mass surplus for various launch vehicle and TMI staging strategies is shown. The colors of each point indicate varying numbers of TMI stages for each launch vehicle size. The gray boxes highlight the largest possible TMI stage size for each launch vehicle.

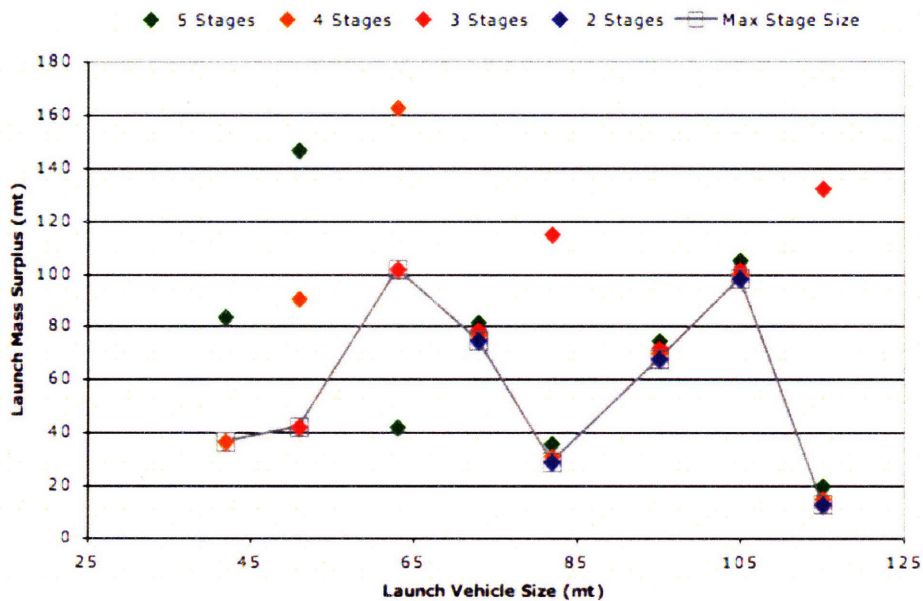


Figure 2.6: Lunar habitat launch mass surplus for various launch vehicle sizes and TMI staging strategies is shown. See above caption for more details.

mt launch vehicle is the best choice because it yields the smallest amount of surplus mass (i.e. wasted launch capacity). However, this analysis is limited in that the launch vehicle sizes are chosen somewhat arbitrarily and do not cover the entire range of possible choices. As discussed above, the current methods of solving this problem require significant amounts of time, so results could not be obtained for a continuous distribution of launch vehicle sizes. The main conclusion to be drawn from Figure 2.4 is that an optimal (or at least a *better*) launch vehicle size *does* exist for a given transportation architecture and module size. It may lie at the 60-mt mark, or it may lie somewhere in between the four discrete launch vehicle sizes modeled here.

Due to the smaller size of the problem, more detailed results could be computed for the lunar missions only (temporarily ignoring the Mars vehicles). In this case we use a larger set of possible launch vehicles (115, 105, 95, 82, 73, 63, 51, 42 mt) based on existing or projected launch vehicle designs (see Table 2.1). The data from this more detailed analysis can be used to study trends within the problem. One of the major questions arising from the launch vehicle sizing discussion is what size modules, or ‘chunks’, to create from large vehicles such as the TMI stages. Figures 2.5 and 2.6 plot,

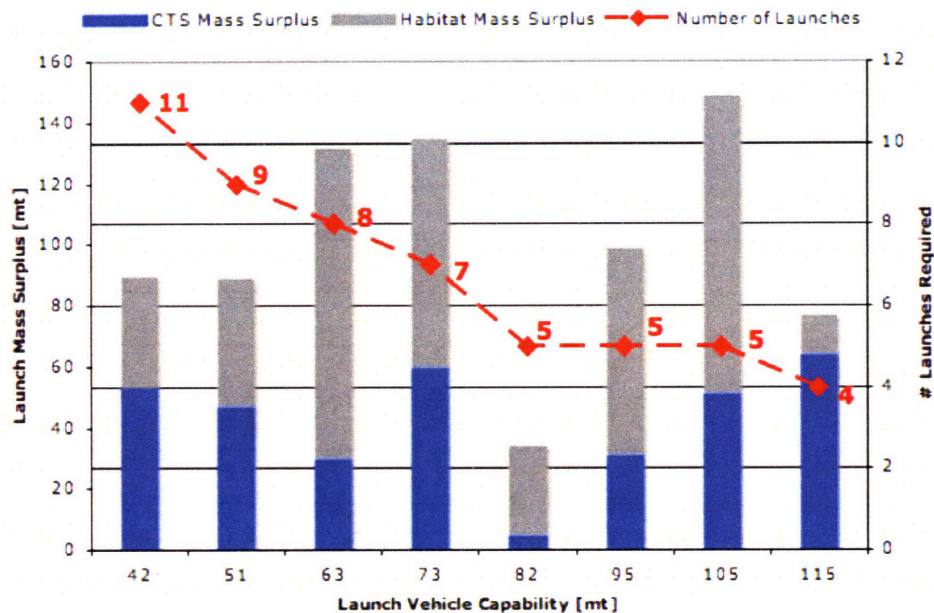


Figure 2.7: The metrics number of launches and mass surplus are plotted for complete lunar missions, including the CTS and Habitat vehicle stacks. The bars show the mass surplus contribution from each of the lunar vehicles, while the number of launches required for each launcher size is plotted in red.

for each lunar vehicle, the launch mass surplus for various launch vehicle sizes and TMI staging strategies. Each color indicates that the TMI stage has been divided into a different number of ‘chunks’, i.e. the Earth Departure Stage (EDS) is assumed to be decomposed into smaller chunks which we call TMI stages. The data indicate that in almost every case, the most efficient solution utilizes the largest possible TMI chunk size that fits in the launch vehicle.

Assuming that this trend holds for the data given here, Figure 2.7 plots the mass surplus and number of launches required for this more comprehensive set of launch vehicles. Note that the minimum launch surplus is no longer at 60 mt, although the CTS stack exhibits a local minimum at that point. For this transportation architecture, the optimal launch vehicle size is 82 metric tons, a launcher size not modeled in the previous analysis. Note that at this launcher size, there is also a ‘knee’ in the curve showing the number of launches required: increasing the launcher size to 95 or even 105 mt does not decrease the number of launches required, but the additional launch capability is not really used efficiently but mainly goes to launch surplus. Only once the launch capacity is increased to 115 metric tons does the number of launches decrease again. These minima in the number of launches and mass surplus suggest that the cost of launching this set of modules would be minimized by using a launch vehicle with a capacity near 82 mt. The data clearly show the existence of an optimal launch vehicle size for this transportation architecture.

2.4.2 Integer Optimization Results

While the full factorial search provides a method for finding an optimal launch vehicle size based on the launch mass surplus, the integer optimization formulation shows greater potential in terms of flexibility and also incorporates the cost metric directly into the objective function. Therefore, the variation in cost for various types of launch vehicles can be accounted for. A sample set of launch vehicles with associated costs is provided in Table 2.1. The set spans the range of possible launch vehicle capacities, but exhibits wide variation in terms of cost. (Note that it is difficult to estimate costs for various types of launch vehicles at this stage; these numbers are based on

various sources, and should be considered *only* as a sample dataset for this problem. Development costs are not included. See [Isakowitz 2004] and [MIT/NASA 2005].)

Table 2.1: Sample launch vehicle data.

Sample Vehicles	Capacity [mt]	Launch Cost [\$M]	Unit Cost [\$/kg]
EELV	28	170	6320
Clean Sheet 42	42	480	12069
Clean Sheet 51	51	590	12217
Clean Sheet 63	63	640	10728
Clean Sheet 73	73	720	10416
Shuttle-Derived (SDV) Sidemount 82	82	600	7727
Shuttle-Derived (SDV) Sidemount 95	95	980	10894
Clean Sheet 105	105	1300	11162
Shuttle-Derived (SDV) Inline 115	115	1390	12765

As in the full factorial case, the results show that the best solution in nearly every case is to choose the largest possible TMI stage size that fits in a given launch vehicle. With this assumption, the cost and number of launches required to launch the Lunar Crew Transportation System are plotted in Figure 2.8.

The plot shows that the objective function is low for the 28-mt vehicle (EELV, e.g. Atlas V-HLV or Delta IV-Heavy) due to its low costs and also for the 82-mt vehicle due in part to its low cost-to-capacity ratio, and in part to its low mass surplus (as shown

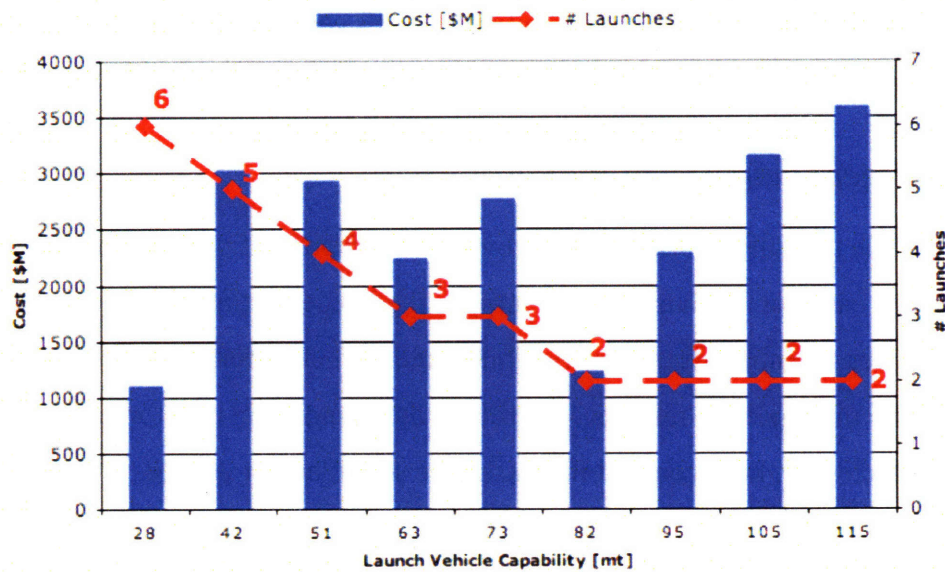


Figure 2.8: The number of launches and cost for the lunar CTS are shown. The bars indicate the cost for each launch vehicle, while the number of launches required for each launcher size is plotted in red.

in Figure 2.7, above). Recall that the full factorial results (Figure 2.7) also showed the 82-mt vehicle as the best choice, confirming that the mass surplus is a reasonable surrogate metric for cost. The 28-mt EELV did not show particular advantages in Figure 2.7, however, because its low cost was not taken into account. However, in this formulation, the EELV is the optimal solution even when other launch vehicles are available at the same time. For example, if the elements could be launched on either an Atlas V or a SDV, the optimal solution places all elements on Atlas V launch vehicles. This analysis thus indicates that the optimal solution is to split the TMI into four stages and use six Atlas V-HLV launch vehicles. Note that in this problem formulation, the optimal solution is driven in large part by the estimated cost of the launch vehicle, and that cost and risk of assembling the smaller modules with the 28 mt launch vehicle are not yet included.

2.4.3 Risk Analysis: Payload Sparing

The above analysis optimized the launch vehicle choice for reduced launch cost (driven by mass surplus and number of launches). The final step is to address the second major metric: risk (or reliability). The mission risk associated with a particular launch vehicle choice depends on the number of launches (with more launches, the risk of a launch failure increases) and on the *payload* of each launch. For example, the TMI stages are relatively simple and inexpensive to build, so if a launch containing a TMI stage is lost, it is easier to replace than, for example, the more complex and expensive habitats. To simplify the analysis, we classify the TMI stages as ‘low-value’ payloads, and all other items as ‘high-value’ payloads. Thus, mission risk can be analyzed in terms of the payload sparing requirements.⁴

Assuming a launch success rate of 0.98, the total probability of achieving delivery of all desired payloads to low Earth orbit (LEO) is determined for various quantities of payloads and available spares. In this analysis, available spares are equivalent to launch failures. This is the case because we are assuming that a spare is successfully launched if a primary launch fails. The overall launch sequence reliability is found from

⁴ This work on payload sparing and risk analysis was completed by William Nadir and myself. See [Gralla 2005].

$$P = \sum_{i=0}^{N_S} \binom{N_L}{i} p^{(N_L-i)} (1-p)^i \quad (2.5)$$

The variable P represents the total probability of success of the set of required payload launches, N_L is the number of launches required for the payload assuming no launch failures, N_S is the number of spares needed, and p is the probability of successfully launching each individual launch vehicle. The first term indicates the number of combinations of sparing payloads within the total number of payloads.

The results for a varying number of launches and spares are shown in Figure 2.9. The black line indicates the probability of launching *all* payloads successfully for a given number of launches. The red line indicates the chance of launching *all but one* payload successfully, and so on. It is apparent that even for relatively small numbers of launches, the risk of losing a single launch is fairly significant; for example, for the optimal 82-mt launcher found above, 5 launches are required, and the chance that all would be successful is only 90%. On the other hand, if two spares are available, the probability of launching even twenty payloads successfully is nearly 100%. Note that we do not distinguish between launching humans and unmanned payloads in this analysis.

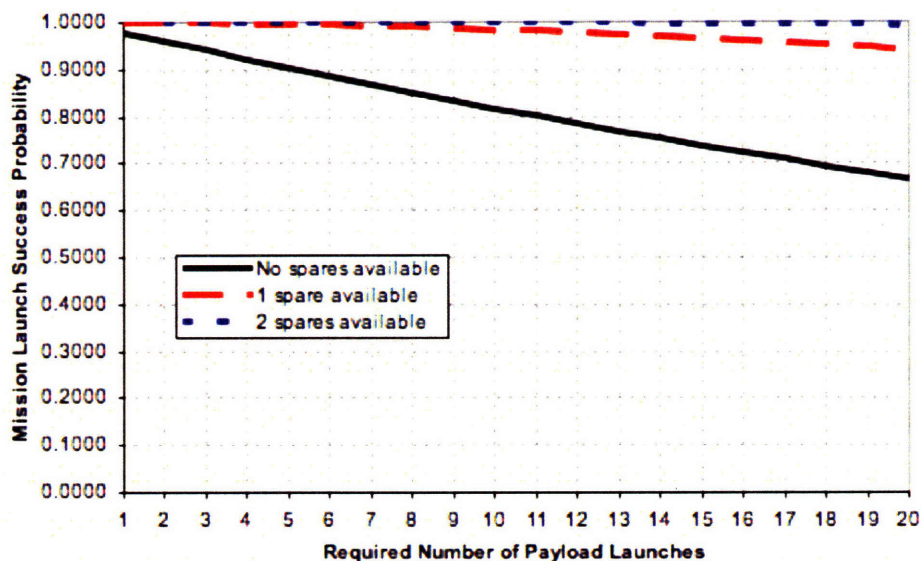


Figure 2.9: Curves show the overall mission launch success probability for various sparing strategies. Each line shows how the mission launch success probability varies based on the required number of payload launches for each sparing strategy.

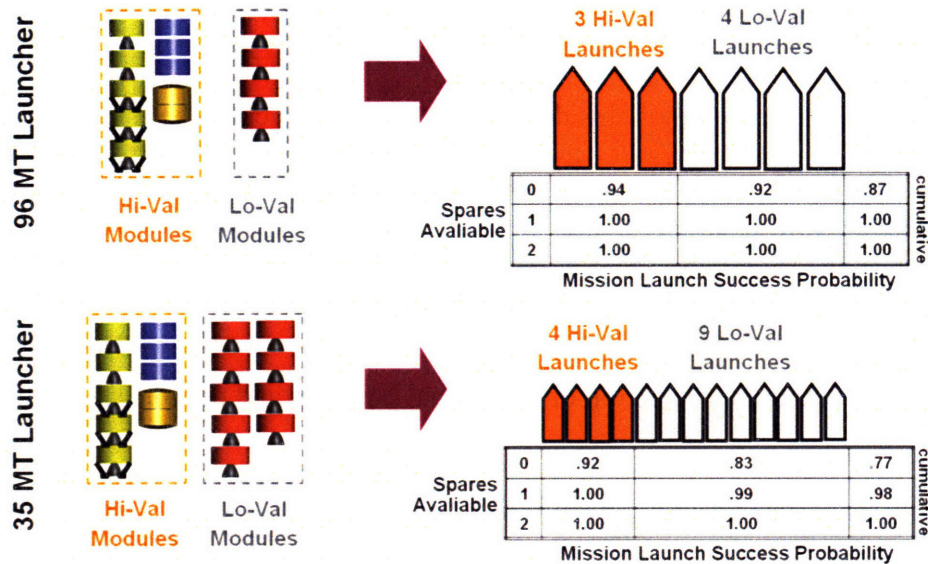


Figure 2.10: Mission risk for various payload types is shown for a lunar mission. The high value and low value modules are separated on the left. More launches are required for a smaller (35-mt) launcher. The probability of success given various sparing strategies is given for each type of payload in the table at the right.

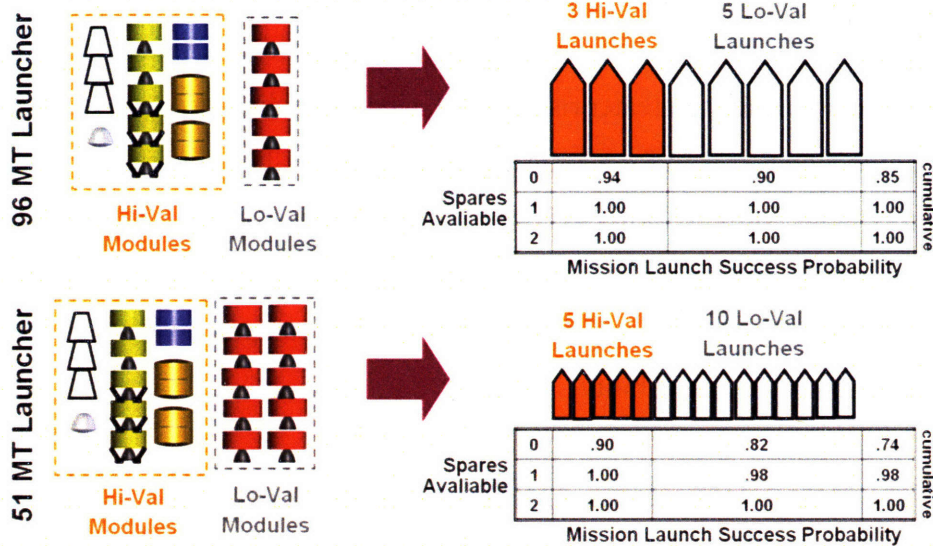


Figure 2.11: Mission risk for Mars mission. See above caption for details.

This analysis is applied to two example sets of payloads for the Moon and Mars; results are shown in Figures 2.10 and 2.11, respectively. These figures break out the payloads into high-value and low-value modules and show the number of launches of each type required. The probability of completing all launches successfully assuming the availability of 0, 1, or 2 spares of each type is shown in the tables. The data show that without spares, the chance of launching all high-value payloads successfully is fairly high

even for the smaller launch vehicles, while the chance of successfully launching all the low value payloads decreases to 83% and 82% for the smaller 35-mt and 51-mt launch vehicles, respectively. Based on this data, a sparing strategy could be formulated to keep several spares for the low-value payloads, but fewer for the high-value modules. This type of analysis shows that modularity, sparing strategies, and risk analysis can be used to strategically lower mission risk without incurring unnecessary costs. In addition, this method can be used to quantify the risk associated with the choice of a particular launch vehicle, aiding in the final selection of an optimal launcher and module size in terms of both cost and risk.

This represents a paradigm shift in the sense that some launch failures of unmanned payloads are expected and that mission and campaign planning takes these into account through contingency planning ahead of time. Obviously, failures of manned launches will always remain unacceptable and will have to be mitigated through launch escape systems to help preserve the lives of the human crew. Finally, note that this type of analysis provides a counter-argument to the generally held assumption⁵ that a higher number of launches leads to greater risk. In fact, since it is cheaper to provide a spare for a small module/launcher than for a large module and heavy-lifter, a greater number of launches may in fact provide equivalent or greater robustness to the risks (since launch reliability is significantly improved when a spare is available). Spreading out low-value modules (e.g. propellant) among smaller launch vehicles can potentially increase overall program affordability and reliability.

2.5 Conclusions and Design Recommendations

For the Draper/MIT lunar architecture taken here as a case study, a clear optimal launch vehicle size emerges at 82 metric tons, requiring five launches to complete one long-duration lunar mission. Even with no spares, the chance of completing all launches successfully is approximately 90%; with one spare, the probability is nearly 100%. Another good choice emerges when the varying launch costs of the vehicles are taken

⁵ [NASA 2005a], for example, imposes a requirement that “no more than four launches will be used to accomplish a single human lunar mission.” (p. 12). The requirement is not justified explicitly in the document, but it seems clear that the risk of multiple launches is the motivation for the requirement.

into account: the same lunar mission can be launched on eleven EELV-type launch vehicles (28-mt capacity) for a slightly lower cost. However, the chance of launching all payloads successfully decreases to 80%. These results provide a solid quantitative basis from which to understand the launch vehicle selection tradespace for this set of lunar/Mars architectures. The data reaffirm the pre-existing supposition that the EELV's advantage lies in cost savings, while the HLLV has the edge in reducing risk by reducing the number of launches required.

The analysis discussed in this paper provides a method for selecting an optimally sized launch vehicle for a given transportation architecture. Perhaps more importantly, it also suggests ways to *optimize the architecture itself* for the selected launch vehicle. Based on the launch vehicle size, an optimal 'chunk' or module size can be found to fit easily divisible modules (such as propulsion stages) onto the selected launch vehicle. Any spacecraft component (lander, etc) can be divided into assemble-able modules by imposing an estimated 'mass penalty' for modularization.⁶ Then, the best (or most efficient) module/launcher size can be found using either a full factorial search to minimize launch surplus or integer optimization to minimize cost.

This type of analysis was applied here to a set of Moon/Mars transportation architectures developed at MIT/Draper, but should be generally applicable to any set of modular vehicles. The results of this analysis provide a much-needed quantitative method for understanding the combined launch-and-assembly tradespace for assemble-able architectures.

⁶ The only spacecraft component that might *not* be divisible is the heat shield or aeroshell for Mars missions. The assembly of such a critical piece of hardware in orbit has not been technically validated. See [Crawley 2005] for more discussion.

3

Assembly Strategies

Access to space is one of the most expensive and difficult portions of manned (and unmanned) spaceflight missions. In Chapter 2, we discussed the issue of launch vehicle design and sizing for assemble-able transportation architectures. On-orbit assembly is another key component of the infrastructure enabling access to space, yet it is not nearly so well-understood nor widely discussed as the launch vehicle selection process. In this chapter, we seek to address that deficiency through a conceptual study of the options for reducing the costs of on-orbit assembly for various types of spaceflight missions.

First, an overview of the operational challenges associated with on-orbit assembly is presented, recapping the body of literature on possible assembly techniques that were

developed largely during planning for the international space station. We thereafter narrow the focus to more affordable techniques for uncrewed assembly with some degree of autonomy, and discuss four basic assembly strategies that will be compared in this study. A model is developed to compare these assembly strategies in terms of the overhead mass required for assembly, and results of this trade study are presented, along with some general conclusions about the value of reusable infrastructures such as space tugs and on-orbit fuel depots.

3.1 Assembly Techniques and Challenges

In this section, we recap a number of relevant conclusions from the background (1.2) and literature review (1.3) sections of Chapter 1.

On-orbit assembly is an extremely challenging undertaking. Mission planners have in the past been reluctant to depend on in-space assembly due to the inherent complexity of the process, both in the design and operation of the mission. The result has been that only the largest space missions, generally those involving humans, have demonstrated on-orbit assembly: the Apollo program and ISS, for example. While these programs have proven the technical feasibility of on-orbit assembly, they have utilized time-consuming and expensive assembly techniques. We discuss the challenges facing assembly planners and some of the techniques that have been developed to overcome these challenges.

3.1.1 Assembly Challenges

As described in Chapter 1, assembly is a complex operation, depending on a number of component processes to function properly and promptly. Two (or more) modules must be launched successfully, enter and maintain their respective orbits, rendezvous in space, match their attitude and position/velocity states, dock or berth, and thereafter function as a single spacecraft. In the inaccessible space environment, with sensor uncertainties and hardware malfunctions, successfully completing all of these sub-processes is by no means easy. Furthermore, other considerations dictate further constraints on assembled missions. Certain modules, especially those carrying hydrogen propellants, cannot loiter long in orbit, so the timing constraints on assembly would

require launches in quick succession (with exact turn-around time depending on boil-off rate and number of modules). Drag could alter the orbit of the spacecraft before all modules are launched or assembled, necessitating propellant usage for orbit maintenance. Controlling a half-assembled spacecraft might also be difficult, and require complex guidance and attitude control capabilities [Clark 1987]. However, as past programs in on-orbit assembly have proven, success is certainly possible.

The major reasons for the success of programs like the ISS include careful planning and human-in-the-loop operations. Myriad hours have been devoted to planning ISS assembly operations, with every small maneuver carefully scripted and reviewed. With humans to guide robotic operations, perform tasks too complex for robots, monitor rendezvous trajectories in real time, flag problems as they occur and intervene when necessary, assembly has proceeded relatively successfully. Unfortunately, this extensive involvement of humans in both planning and operations is extremely expensive. In particular, time spent during EVAs or even IVAs on the space station drives up expenses significantly, if we take into account the costs of maintaining that space station as a base for assembly operations. The ISS is not in a well-situated orbit for lunar or Mars missions, so it is quite probable that it will not be a feasible base for assembly of future large missions. One of the biggest challenges of in-space assembly, it appears, is high cost. Therefore, we must look at all the alternative methods for in-space assembly in order to make it more affordable.

3.1.2 Assembly Techniques

As discussed in Chapter 1, many authors have examined various techniques for on-orbit assembly. The main types of assembly ‘actors’ are crew on-site (EVA or IVA), tele-operated robots, or autonomous robots; options for physical mating include docking and berthing, which depend on various types of docking ports or grappling arms, respectively. Docking port designs can range from simple probe-and-drogue designs with no connecting passage and no utility connections to complex androgynous mechanisms which handle various approach speeds and provide large human passageways and complex fluid or electrical connections. Among this wide range of options, only those

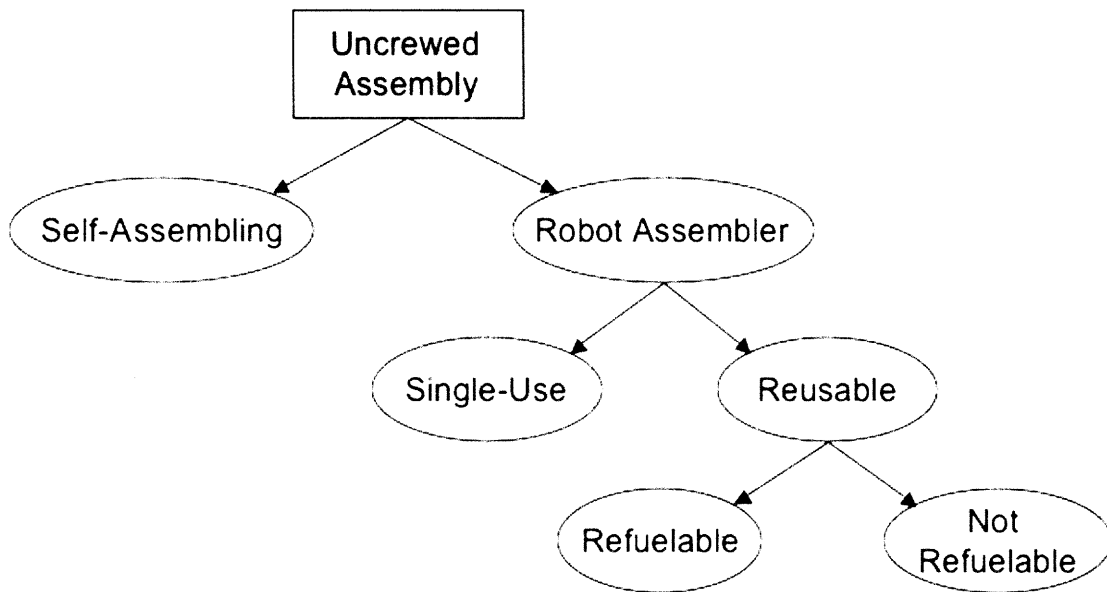


Figure 3.1: Options tree for uncrewed assembly. Modules can either be self-assembling or require a robot assembler, which may (or may not) be reusable and refuelable.

involving humans have been tested extensively in space (the only successful autonomous assembly was the small ETS-VII mission).

In this thesis, we look to the future of on-orbit assembly, and study the options that have not been extensively proven operationally: *assembly by robots with limited human involvement*. We focus mainly on the selection of the robotic ‘actor’ and therefore assume a docking port along the lines of the NASA system currently under development – the ADBS (see Section 1.3.1.3). Thus we can explore specifically the impact of the assembly technique without polluting the results with changes brought on by the docking/berthing technology.

Within the realm of uncrewed assembly, many options remain. Modules could be self-assembling, or a robot assembler could be employed to bring passive modules together. The assembler could be single-use or reusable, could be specific to certain types of modules, or employ a universal design with an arm equipped to grapple any spacecraft. In this thesis, we distill this space of options into the following technological choices: a module can be self-assembling or passive (requiring a robot assembler), and the robot assembler can be either single-use, or reusable. A reusable assembler either must carry all the fuel for all its missions, or it must be capable of refueling on-orbit. These basic

options, shown in Figure 3.1, allow us to study whether it is in fact valuable to create a reusable assembly infrastructure in space.

One possible type of reusable assembler is the so-called ‘space tug,’ an idea borrowed from the literature of on-orbit servicing. Space tugs are multi-use spacecraft that attach to and propel other spacecraft, modifying their orbits. Tugs have the potential to support a wide variety of space mission types, such as retiring geostationary communication satellites or cleaning up space debris (see Section 1.3 and [Galabova 2003]). In this case, we focus on their potential applications to on-orbit assembly tasks.

In summary, we learn the following lesson from the history of on-orbit assembly, especially that of ISS: on the one hand, the technology and experience gained by assembling such a complex station will be invaluable in the next generation of space exploration programs; on the other hand, the rising costs of assembly point to the need to reduce assembly complexity and simplify the assembly process in order to create a sustainable, affordable program. Therefore, we turn to uncrewed assembly with some degree of autonomy, and examine the best options within this tradespace.

3.2 Assembly Strategies

It is clear from Chapter 2 that human exploration of the Moon and Mars will require on-orbit assembly of large spacecraft. In order for any such exploration program to be sustainable, it must avoid the difficulties encountered by the ISS program (as discussed in Section 3.1 above), and develop a more affordable assembly strategy. Astronaut participation and extensive, unique planning for each mission cannot be the norm for next-generation on-orbit assembly. The life cycle costs of assembly could be reduced through the development of a flexible, reusable infrastructure to assist in the assembly task.

The question still remains: what form should this reusable infrastructure take? Affordable on-orbit assembly could be as simple as developing a common docking port and propulsion system, or as complex as a fleet of space tugs equipped to capture any spacecraft and transport it anywhere, anytime. This study focuses on the potential benefits of two key technologies discussed in the preceding section: space tugs and on-

orbit refueling. In order to quantify the benefits of each of these technologies, we must distill them into well-defined assembly strategies, including vehicle designs and an operations concept.

3.2.1 Basic Assembly Concepts

We can discover three basic assembly concepts for large space missions by focusing on the key elements of in-space assembly. Propulsion and guidance/navigation are the two essential elements of successful assembly, so we distill three basic concepts in which the propulsion for rendezvous and docking are provided in different ways.

- 1. Self-Assembly:** Each module performs its own rendezvous and docking operations.
- 2. Module as Tug:** A single module collects and assembles all other modules.
- 3. Space Tug:** A dedicated, reusable space tug module collects and assembles the modules.

The first strategy has one major disadvantage relative to the other two: each to-be-assembled module requires its own propulsion and guidance capabilities in order to perform the orbit transfer and rendezvous maneuvers. In the latter two strategies, only one module (or a separate tug) must have propulsion hardware, even though the “passive” modules may still require a basic attitude control system for attitude stabilization to prevent tumbling. These two strategies appear quite similar; the major difference lies in the reusability of the propulsive spacecraft. With a ‘module-as-tug’ strategy, the tug would be useful for only one mission, whereas a dedicated space tug could be reused for several assembly tasks. For the purposes of this study, that difference is irrelevant, since we focus on generic case studies rather than particular missions. Therefore, for clarity, we treat the latter two strategies as a single ‘tug-based assembly’ strategy for the purposes of this analysis.

3.2.2 Assembly Strategies

These two basic assembly strategies do not capture all the potential variations of space tug use. We initially explored several possibilities and settled on the addition of two variations: the use of multiple tugs for a single stack assembly task and the addition of in-space refueling for a single tug. Other variations are certainly possible but the following four strategies are representative of the on-orbit assembly tradespace:

1. **Self-Assembly:** Each module performs its own rendezvous and docking operations.
2. **Single Tug:** A single dedicated, reusable space tug performs all assembly operations, including shuttling modules from the parking to the assembly orbit.
3. **Multiple Tugs:** Each tug module performs only a certain number of assembly transfers; therefore, multiple tugs are required to complete the assembly task.
4. **In-Space Refueling:** A single tug spacecraft performs all assembly operations, but it is refueled after a certain number of transfers (new propellant tanks are launched or the tug is refueled from an orbiting depot).

The operations concepts for each of these strategies are illustrated in Figure 3.2. The sequence of launched elements is indicated at the bottom of each illustration.

The sequence of events in the self-assembly case (1) is straightforward: each module is launched into a parking orbit, then transfers under its own power and propellant to an assembly orbit to rendezvous and dock with the other modules. A propulsion system must be present on each module. In the tug case (2), each module is launched into a parking orbit. At that point, the tug docks with the module and transfers it to the assembly orbit to rendezvous and dock with the pre-assembled stack. The tug then separates from the module stack and returns to the parking orbit to retrieve the next module. Both processes repeat until assembly is complete.

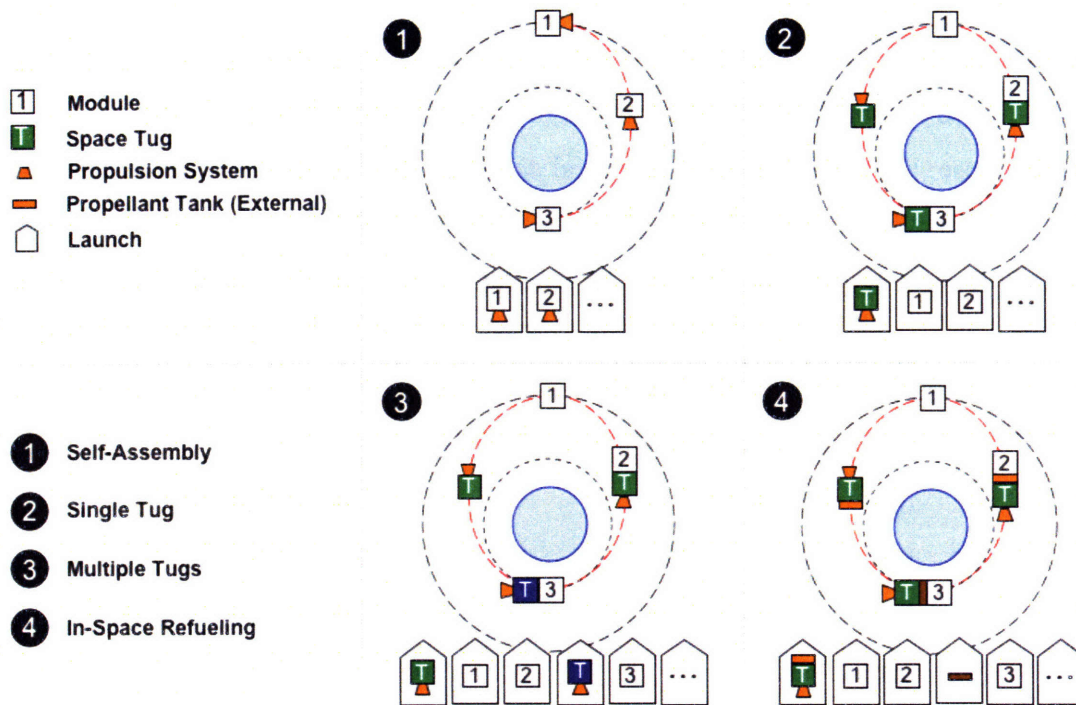


Figure 3.2: The four assembly strategies are illustrated. The inner circular orbit is the parking orbit, while the outer is the assembly orbit. Red dashed lines indicate the outbound and return transfers.

The latter strategy (2) has the disadvantage that the tug must carry *all* the propellant for assembling all modules back and forth many times. While only one module needs a full propulsion system (the space tug itself), some inefficiency is incurred by having to shuttle propellant back and forth. The use of multiple tugs (3) alleviates this difficulty, by launching a new tug after a certain number of modules have been assembled. The in-space refueling option (4) also addresses this difficulty, this time by allowing the launch of fresh propellant tanks (as modeled here) after a certain number of modules have been assembled. The choice of the number of modules per tug spacecraft (or fresh tank) drives the performance of both strategies.

The launch and assembly sequences as illustrated in Figure 3.2 are modeled in some detail; the mathematics is described in Section 3.3. These four scenarios capture most of the possible concepts for tug-based on-orbit assembly.

3.3 Assembly Trades Model

In evaluating the potential of these four basic assembly strategies, the key question is whether the benefits of space tug deployment outweigh the costs of designing, launching, and operating an entirely separate spacecraft to provide propulsion. We expect that some on-orbit assembly tasks are more easily or cheaply accomplished with the support of a reusable space tug.

An *assembly task* can be characterized by a set of attributes: the vehicle design (e.g. number and mass of modules to be assembled, tug mass, etc.), and the orbit design (e.g. altitude and inclination of parking and assembly orbits). In order to investigate the benefits of the space tug assembly infrastructure, we must understand what kinds of assembly tasks are best accomplished using a space tug.

We must therefore understand how changes in the assembly strategy (among the four listed in Section 3.2) impact the overall launch mass. By tracking this metric as both the assembly strategy and assembly task are varied, the circumstances under which space tugs are valuable can be determined. To that end, we perform a trade study that compares the ‘cost’ (defined later) of tug-based on-orbit assembly strategies to that of the same tasks accomplished without the aid of a tug.

The following sections describe how assembly tasks and strategies are modeled, while Section 3.4 describes the results of a trade study based on this model.

3.3.1 Assembly Model Overview

A Matlab-based model has been developed to enable trades between the four strategies described above (the code is given in Appendix C). In addition, the model provides a determination of the kinds of assembly scenarios for which a tug is useful. A diagram of the model inputs and outputs is shown in Figure 3.3.

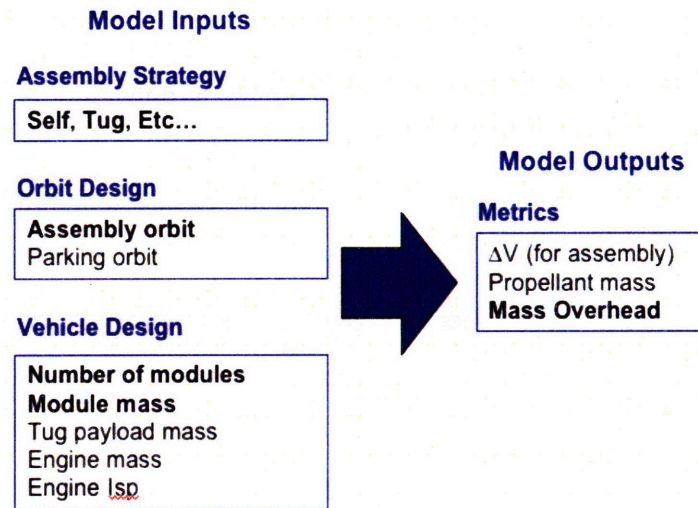


Figure 3.3: On-orbit assembly model block diagram. Input parameters in bold-face type are variables in the study; those in plain type are fixed parameters.

The inputs are grouped into three categories. The assembly strategy indicates the type of strategy being evaluated; the orbit design captures information on the parking and assembly orbits; and the vehicle scenario captures information on the vehicles themselves, such as mass properties, the number of modules to be assembled, the engine specific impulse, etc. The outputs include standard metrics such as time and propellant required, along with a comparative metric called ‘mass overhead’.⁷ In Figure 3.3, input parameters in bold-face type are variables in the study; those in plain type are fixed parameters (sensitivity analysis is performed on the most important of these fixed parameters; see Section 3.4).

The rationale for this model is that it enables comparisons between assembly strategies and allows investigation of the sensitivity of the results to variations in the input scenario, such as changes in the number and size (mass) of modules, or the altitude or inclination of the assembly orbit. The obvious metric is the total launch mass, but this quantity needs to be carefully defined before comparisons can be made; this is the reason for the introduction of the mass overhead output, which captures the *extra mass required*

⁷ This ‘mass overhead’ is different from the concept of ‘mass surplus’ used in Chapter 2, and is described in detail in Section 3.3.4.

for on-orbit assembly beyond the mass of the modules themselves. The various assembly strategies can thereby be directly compared. A detailed description of this metric is included in Section 3.3.4.

The following sections provide detailed descriptions of the implementation of the model sketched out above. We describe the vehicle models, orbital mechanics model, overhead mass metric, and baseline parameters.

3.3.2 Spacecraft Models

Two different vehicles must be modeled (module and tug), along with several variations on each of these vehicles. Because we are drawing comparisons between tug- and self-assembly strategies, it is essential that the schemes used for modeling both vehicles be *consistent* with each other. We cannot model one in great detail and oversimplify the other; the levels of fidelity and the underlying assumptions must match, in order to ensure an accurate comparison.

In this conceptual exploration of the assembly tradespace, it is not necessary to model the vehicles extremely accurately. We therefore simplify the models to the essential elements *affected by the on-orbit assembly strategy*: payload, propulsion system, and support structure. We assume that the remainder of the spacecraft mass would be similar between the various assembly strategies and can therefore be ignored (at this level of detail). This modeling approach is depicted in Figure 3.4.

Figure 3.4 shows a notional model for both vehicles: the space tug (left) and a self-assembly module. A module assembled by a tug would consist simply of the red ‘module’ box on the right-side vehicle, with no extra structure, tank, propellant, or engines. Each vehicle is modeled as a payload (the module, in the self-assembly case), with associated propulsion system and structure. The propulsion system is made up of an engine, a propellant tank, and the propellant itself.

We estimate the mass of each of these components based on simple rule-of-thumb relationships, described in the following paragraphs. The baseline values for the design parameters are provided in Section 3.3.5. The tank mass m_{tank} depends on the amount of propellant required for the trip, but the engine mass m_{eng} is fixed so that it is the same in both tugs and self-assembled modules, regardless of assembly task. The tug payload m_{pld}

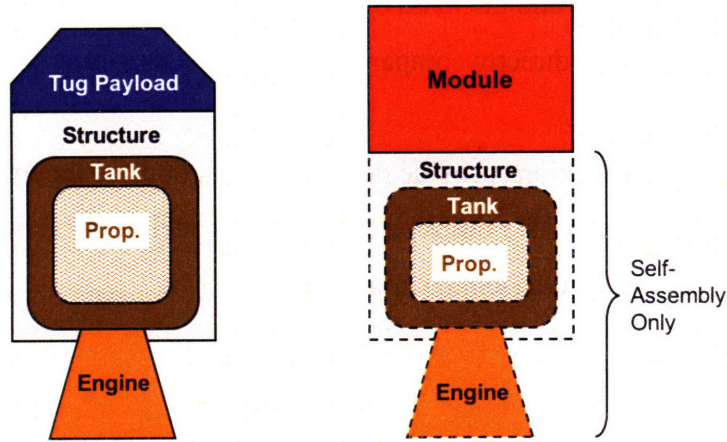


Figure 3.4: Notional vehicle models of the space tug (left) and self-assembled module (right).

(docking port, grappling arm, etc.) is an estimate of the mass of a docking port or grappling arm, and the structure mass m_{str} depends on the mass of the payload and propulsion system combined. The propellant tank and structure masses are calculated based on the mass fractions f_p and f_{str} , respectively. The factors used in this model are shown in Table 3.1.

With this framework, the mass of each vehicle can be calculated based on a given engine mass, payload mass, and propellant requirement. In the tug case, the payload mass is simply the tug payload mass m_{pld} . In the self-assembly case, the payload mass is the mass of the module to be assembled m_{mod} . The method for calculating the mass of each vehicle component is given below. For the space tug, the calculations are given in Eqs. 3.1 below.

$$\begin{aligned}
 m_{tank} &= f_p \cdot m_p \\
 m_{str} &= f_{str} (m_p + m_{tank} + m_{pld} + m_{eng}) \\
 m_{dry} &= m_{pld} + m_{tank} + m_{str} + m_{eng}
 \end{aligned}
 \tag{3.1}$$

For the self-propelled module, the calculations are given in Eqs. 3.2. Note that the structure mass of the self-propelled module does not depend on the module mass; we assume that the module mass already accounts for its structure. Likewise, the module's docking ports are already accounted for in the module mass.

$$\begin{aligned}
m_{tank} &= f_p + m_p \\
m_{str} &= f_{str}(m_p + m_{tank} + m_{eng}) \\
m_{dry} &= m_{mod} + m_{tank} + m_{str} + m_{eng}
\end{aligned} \tag{3.2}$$

With this framework, we can model the space tug and self-propelled module spacecraft at a reasonable degree of accuracy. The mass depends on the size of the required propellant tanks, but a fixed mass ‘penalty’ is also incurred because the engine mass is fixed. Thus we can capture the idea that it is more expensive to outfit many small modules with their own propulsion systems. The assumed values for engine mass and payload masses are shown in Table 3.1.

3.3.3 Propellant Requirements Model

The model calculates propellant requirements by modeling the orbital maneuvers required to perform rendezvous operations for all modules. Docking operations are not modeled (and are not expected to be a major contributor to propellant requirements). Several simplifications are assumed for clarity. First, phasing operations are not implemented. Phasing should contribute very little ‘cost’ in terms of propellant requirements, and since time is not considered as a metric, phasing can be ignored for the purposes of this study. Second, only simple inclination changes and Hohmann transfers are modeled; combined plane changes and altitude changes are not implemented. (These combined maneuvers would affect most strategies equally, so they would not affect this comparative study).

For each transfer from parking to assembly orbit, the payload is calculated based on either the module mass or the mass of the combined tug/module stack. The inclination change is performed first (if necessary), according to

$$v = \sqrt{\frac{\mu}{r}} \tag{3.3}$$

$$\Delta V_i = 2v \sin \frac{\theta}{2} \tag{3.4}$$

In Eqs. 3.3 and 3.4, v represents the circular orbit velocity, r is the orbit radius, μ is the mass parameter (gravitational constant) of the central body (Earth in this case), and θ is the required inclination change. A Hohmann transfer from the parking orbit to the assembly orbit is then performed, and the ΔV is found from Eq. 3.5, where r_1 represents the initial orbit radius, and r_2 is the radius of the final orbit.

$$\begin{aligned}\Delta V_1 &= \left[\mu \left(\frac{2}{r_1} - \frac{2}{r_1 + r_2} \right) \right]^{1/2} - \left[\mu \left(\frac{1}{r_1} \right) \right]^{1/2} \\ \Delta V_2 &= \left[\mu \left(\frac{2}{r_2} - \frac{2}{r_1 + r_2} \right) \right]^{1/2} - \left[\mu \left(\frac{1}{r_2} \right) \right]^{1/2} \\ \Delta V_H &= \Delta V_1 + \Delta V_2\end{aligned}\tag{3.5}$$

Finally, the propellant required to provide the ΔV for each of these maneuvers can be found from the rocket equation, given in two forms in Eq. 3.6.

$$m_p = m_f \left(e^{\frac{\Delta V}{I_{sp} g}} - 1 \right) = m_o \left(1 - e^{-\frac{\Delta V}{I_{sp} g}} \right)\tag{3.6}$$

The propellant mass m_p can be found based on either the initial mass m_o or the final mass m_f of the spacecraft. The propellant mass for each module in the self-assembly case is found by a straightforward calculation using the final mass of the module, but the tug cases are more complex. The single tug, for example, carries enough propellant to transport *all* modules to the assembly orbit, so it pushes its own propellant as payload for many of the transfers. Therefore, the tug propellant mass must be calculated iteratively. Based on an estimate of the tug propellant mass, a value for m_p is found and compared to the initial value. If not within a small tolerance value, the process is repeated, using the calculated m_p as the new guess. In this manner, an accurate value for the tug propellant mass for the entire mission can be calculated. For the other tug-based assembly cases, more complex iteration loops are used to calculate the propellant required for each of multiple tugs, each tank in the refueling case, etc.

With this model, accurate propellant requirements for on-orbit assembly can be generated, based on the assumptions given initially. All results shown below satisfy the rocket equation (assuming chemical propellants and impulsive burns) and mass closure requirements.

3.3.4 Overhead Mass Metric

The model output is technically the total propellant mass required for assembly, but this is only part of the comparison between the assembly strategies. The true metric of comparison is cost, but this is difficult to model at this early conceptual stage of the study. One widely used surrogate metric is launch mass, of which the required propellant mass forms a significant part. We adapt this surrogate metric to capture the comparison between the various strategies.

The comparison between the two basic strategies is driven by the respective advantages of each: the tug case allows for lighter modules without propulsion and navigation capabilities, while the self-assembly case does not require return transfers from assembly to parking orbits, nor transfer of excess propellant between the parking and assembly orbits (because the tug must carry propellant for its entire mission). To capture the true differences between the strategies, we introduce the overhead mass metric m_v . The overhead mass is the total weight of all extra fittings, including propellant, that are required for on-orbit assembly. It is calculated differently for each strategy: details are given in Eqs. 3.7 below.

$$\begin{array}{ll}
 \text{Self:} & m_v = n_{mod}(m_{str} + m_{tank} + m_p + m_{eng}) \\
 \text{Single Tug:} & m_v = m_p + m_{tug} \\
 \text{Multiple Tugs:} & m_v = n_{tug}(m_{p,tug} + m_{tug}) \\
 \text{In-Space Refueling:} & m_v = n_{tanks}(m_{p,tank} + m_{tank}) + m_{tug} - m_{tank}
 \end{array} \tag{3.7}$$

For the self-assembly case, m_v depends on the mass of propellant for each module, plus the mass of all the additional fittings required – engine, propellant tank, and supporting structure. For the single tug scenario, m_v depends only on the mass of the tug propellant m_p and the tug itself m_{tug} . For multiple tugs, the mass of the tug and the propellant carried by each tug ($m_{p,tug}$) is simply multiplied by the number of tugs n_{tug} , assuming all tugs are of identical design. The in-space refueling case, as modeled here,

assumes that new tanks of propellant are launched for each tug refueling (rather than in-space propellant transfer from a depot to previously used tanks). Thus, the overhead mass depends on the mass of each tank m_{tank} and the propellant in each tank $m_{p,tank}$, multiplied by the number of tanks required n_{tanks} . The mass of the tug spacecraft must also be taken into account, but the mass of its included propellant tank has already been accounted for within the first term of the equation, so it is subtracted here. With this overhead mass metric, all four scenarios can be weighed against one another based on the output from the model.

3.3.5 Baseline Parameters and Assumptions

Baseline values are selected for the variables and parameters based on literature searches and the requirements generated by the NASA CE&R study (see Chapter 2, or [Crawley 2005]). Initial research helped to refine these values, shown in Table 3.1.

Table 3.1: On-orbit assembly model baseline values

Variable	Type	Baseline	Range
Assembly strategy	Variable	-	[Self, Single-Tug, Multi-Tug, In-Space Refuel]
Assembly orbit	Variable	400 km, 28.5 deg	200 – 1000 km
Parking orbit	Fixed	185 km, 28.5 deg	-
Phasing strategy	Fixed	Wait in lower orbit	-
Module dry mass	Variable	15 mt	5 – 30 mt
Number of modules	Variable	-	1 – 15
Engine mass	Fixed	200 kg	-
Tug payload mass	Fixed	300 kg	-
Engine Isp	Fixed	310 s	-
Propellant Fraction f_{pp}	Fixed	0.12	-
Structures Fraction f_{str}	Fixed	0.15	-

The rationale varies for the selection of each of these baseline values. The parking orbit is baselined at a standard parking orbit for launch from Kennedy Space Center (KSC). The assembly orbit’s altitude and inclination are varied using the parking orbit parameters as minimum values because drag perturbations make orbits lower than 200 km infeasible, and inclination changes have the same ΔV ‘cost’ whether they increase or decrease inclination; therefore, for the purposes of this study, the direction of inclination change is irrelevant.

The baseline module dry mass was chosen to fit on current launch vehicles (~ 27 mt, see Chapter 2) while reserving a reasonable amount of launch mass for propellant (in the self-assembly case), and varied from the lowest feasible size (based on [Crawley 2005]) to 30 mt. Note that both the upper limit on module dry mass and the range for the number of modules to be assembled is on the low end of the possible requirements spectrum (Figure 2.4 showed up to 27 modules may be required, and module masses may reach 100 mt). Modeling higher values for each of these parameters does not add any value to the study, because the results are simply a continuation of the same trends shown at the ranges modeled here.

The engine Isp is a standard value for bi-propellant engines (see Table 3.2 and [Wertz 1999]), and the phasing strategy is the logical choice (lowest ΔV for this type of mission) among several standard methods (double Hohmann transfer, elliptical phasing loops, sub- and super-orbital drift). (Recall from 3.3.3 that phasing is not explicitly accounted for; this phasing method was initially modeled to ensure that propellant usage for phasing would be negligible.)

The engine mass estimate is intended to capture all the fixed components of the propulsion system, including the engine and all other system hardware, attitude control, etc.. [Wertz 1999] shows that liquid propellant engines weigh on the order of 100 kg; Table 3.2 summarizes typical engine characteristics. We double this number to account for the extra fittings. This is obviously a rough estimate but we perform extensive sensitivity analysis to understand how changing this value affects the results. In addition, we ensure that the resulting tug dry mass estimates match those found in the literature: [McManus 2003] models a bi-propellant GEO tug at 1100-1300 kg, and [Galabova 2003] describes a LEO tug weighing in at around 650 kg. With an engine mass of about 200 kg, the tugs weigh in on the low end of this range of values.

Table 3.2: Typical Engine Characteristics

Engine	Isp (s)	Propellants	Mass (kg)
RL10-A (Pratt & Whitney)	446	LO2/LH2	138.35
Delta-II (Aerojet)	320	N2O4/MMH	99.79
OME/UR (Aerojet)	340	N2O4/MMH	90.72
RS-41 (Rocketdyne)	312	N2O4/MMH	113.40

The tug payload refers to the docking/berthing equipment carried by the tug; this could take the form of a docking port, a robot arm, or something related. The baseline value was estimated based on the mass of modern docking systems and values in the space tug literature. The latest NASA docking port design – the Advanced Docking and Berthing System – weighs in at about 300 kg [NASA 2005]. In addition, [McManus 2003] estimates a reasonable tug payload could weigh about the same amount, based on typical sizes and masses of industrial robots. Again, sensitivity analysis shows the impact of varying this estimate (see Section 3.4).

Finally, the propellant and structures mass fractions are based on relationships given in [McManus 2003], [Lamassoure 2002], and [Wertz 1999].

3.4 Trade Study Results

With the model described in Section 3.3 above, a comprehensive trade study can be carried out to investigate the relative value of the four assembly strategies: self-assembly, single tug, multiple tugs, and in-space refueling. As mentioned above, the on-orbit assembly model is used to explore the design space and to understand the effects of varying several parameters on the overhead mass m_v and on the comparison between the various strategies. We reiterate that the end goal is to understand which assembly strategy is better for various kinds of scenarios.

The study follows a basic structure in which a parameter (or two) is varied within a specific range while the others are held constant at their baseline values. (Recall that the ranges and baselines are summarized in Table 3.1). First, the vehicle scenario parameters are varied, then the orbit design variables; thus an exploration of the tradespace is completed. Finally, sensitivity analysis is conducted to understand the impact of some of the assumed and baseline values.

3.4.1 Vehicle Scenario Parameters

The vehicle scenario is described by both the number of modules and the mass of each module that must be assembled. For clarity it is assumed that all modules are identical.

3.4.1.1 Number of Modules

Figure 3.5 shows the variation in additional mass for each of the tug strategies, as the number of modules is varied from 1 to 15. Each plot is based on a different value for ‘M/T’, defined below.

Definition [M/T Parameter]: The ‘M/T’ parameter signifies either the number of modules transferred per tug or per tank (for the multiple tugs and in-space refueling scenarios, respectively). Modules are transferred sequentially; for example, in a multiple-tug scenario with $M/T = 3$, the tug brings module 1 to the assembly orbit, returns to the parking orbit to retrieve module 2 and assemble it to module 1, repeats for module 3, then is retired. A second tug is launched to assemble the next 3 modules.

Note that, unless otherwise specified, results are calculated from the baseline values given in Table 3.1.

In the first graph, with M/T equal to one (one module per tug/tank), the trends are fairly clear. The self-assembly case shows an essentially linear increase in the metric m_v for increasing numbers of modules. The single tug case, on the other hand, has a lower slope at lower values on the horizontal axis, and a higher slope as the number of modules increases. The reason for this behavior is that in this scenario, the tug is required to begin its life carrying all the propellant required to assemble *all* modules. Therefore, it must push a large amount of propellant back and forth between the parking and assembly orbits in cases with a high number of modules. Thus, the single tug strategy is useful *only* at lower numbers of modules.

The multiple tugs and refueling strategies appear to advantage over the single tug. In this case, with M/T equal to one, the multiple tugs case performs rather poorly, with a higher additional mass metric than all other strategies (except single tug at high x -values). This is due to the requirement for a new tug spacecraft for every module transfer. The multi-tug strategy with a ratio of $M/T=1$ performs worse than self-assembly because tugs have a higher mass overhead than modules with an integrated propulsion system. The use of space tugs for on-orbit assembly appears to make sense only when tugs are reused for more than one module. The in-space refueling scenario, on the other hand, performs

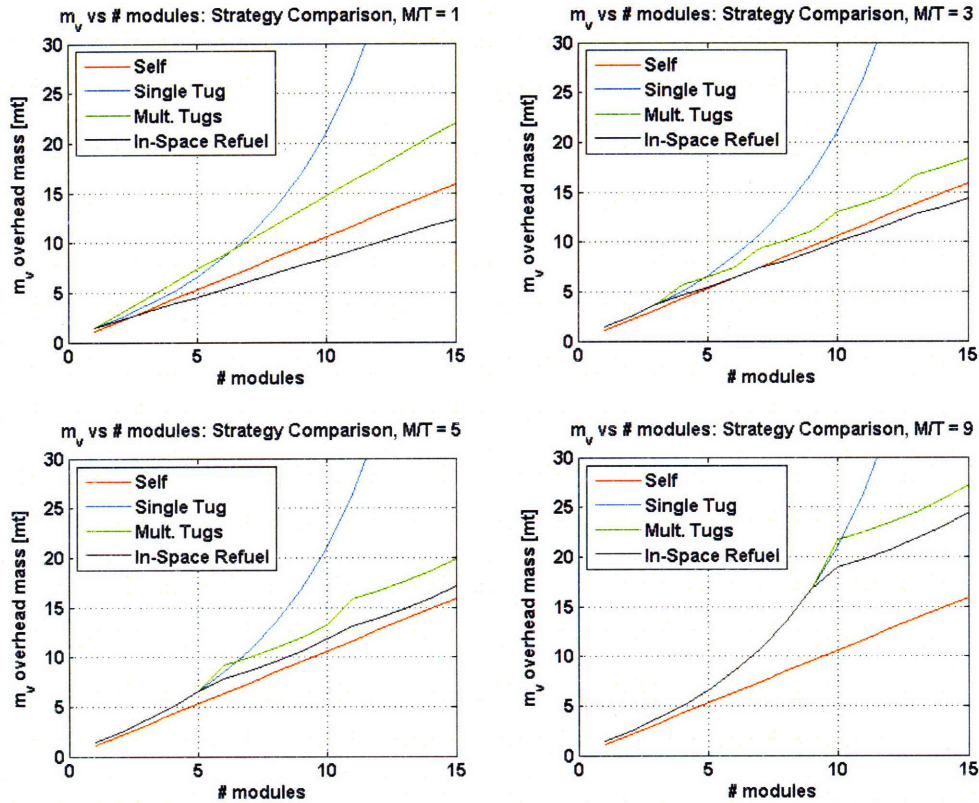


Figure 3.5: Results showing the change in overhead mass as the number of modules is varied. Each plot is based on a different value of ‘M/T’, or modules per tug/tank.

consistently better than any others, showing a linear increase with number of modules at a lower slope than self-assembly (because it requires only a new propellant tank for each module and not an entire propulsion system).

The other three graphs, with M/T values of 3, 5, and 8, also display clear trends. While the self-assembly and single tug scenarios do not change based on M/T, the multiple tugs and in-space refueling scenarios vary. The ‘jagged’ curves are due to uneven divisions of modules into M/T-sized chunks. For example, with an M/T of 5, both scenarios show higher m_v values for a six-module scenario, because an entire tug or tank must be launched for the one remaining module (after the first five have been transferred on the first tug/tank). With an M/T of 3, in-space refueling is advantageous at higher numbers of modules; however, note that the best option (least m_v) overall remains in-space refueling *with an M/T of 1*. Based on the trends visible in this set of graphs, it is clear that while mid-level M/T values (e.g. 3, 5) improve the performance of the multiple tugs strategy (over M/T’s of 1 or 9), the improvement is not sufficient to make the

strategy more attractive than either self-assembly or in-space refueling at $M/T=1$. Clearly, high M/T values, such as 9, do not improve the situation (too many return transfers required).

3.4.1.2 Module Mass

The remaining vehicle scenario parameter is the mass of the individual modules. Results for the overhead metric as the module mass increases from 5 to 30 mt are shown in Figure 3.6. In this case, the number of modules is fixed at 5.

First, note that these graphs can be misleading: the y-intercept of the multiple tugs and in-space refueling lines is highly dependent on the number of modules and M/T (see Figure 3.5). The key point here is the slope of each line. The single tug case has the highest slope; therefore, its overhead mass increases fastest as module mass increases. Self-assembly and in-space refueling (only when M/T is 1) have the lowest slopes, so the increase in m_v as module mass increases is smaller than for the other strategies. This makes sense as the mass for propulsion and attitude control has a fixed component which is independent of module mass. Thus, as modules are increased in mass, the relative percentage of that mass due to propulsion and attitude control gets smaller.

The obvious conclusion here is that in-space refueling provides the best option at an M/T of 1; self-assembly is a close second-best. Note that these results are consistent with the conclusions drawn based on Figure 3.5.

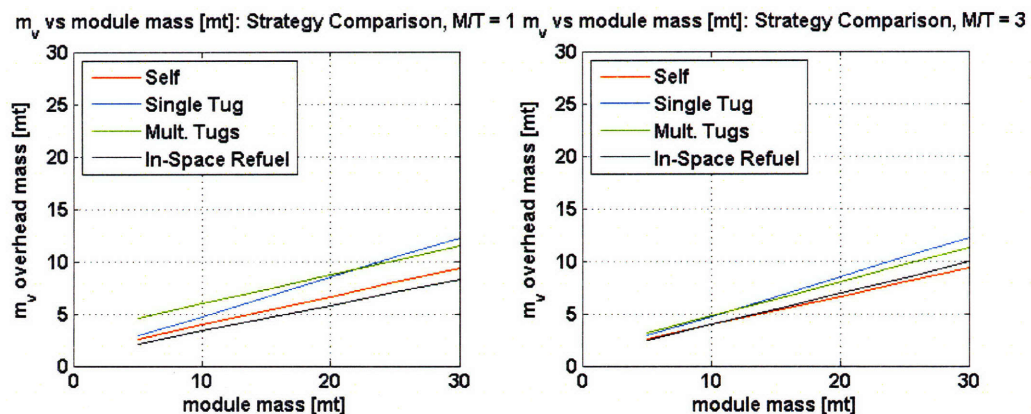


Figure 3.6: Results showing the change in overhead mass as the module mass is varied. Each plot is based on a different value of 'M/T', or modules per tug/tank.

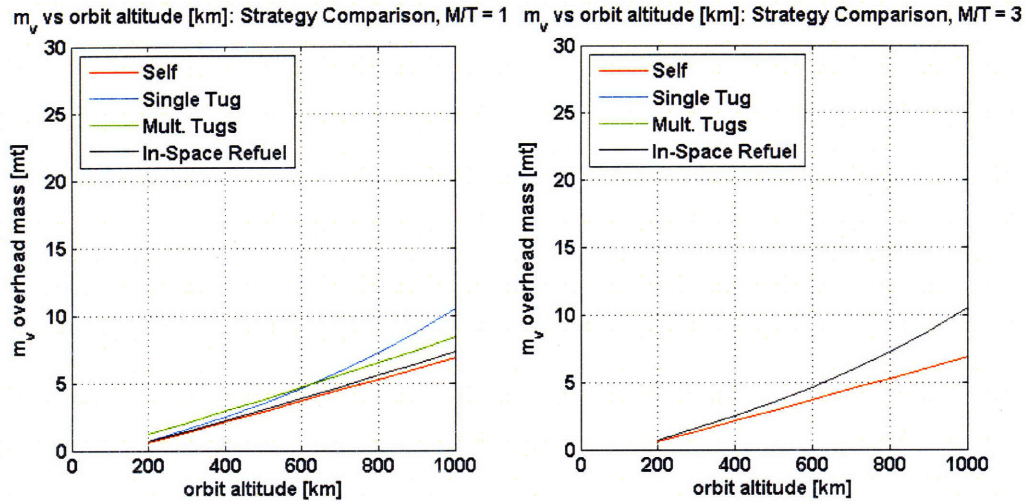


Figure 3.7a: Results for varying assembly orbit altitude, with 2 modules.

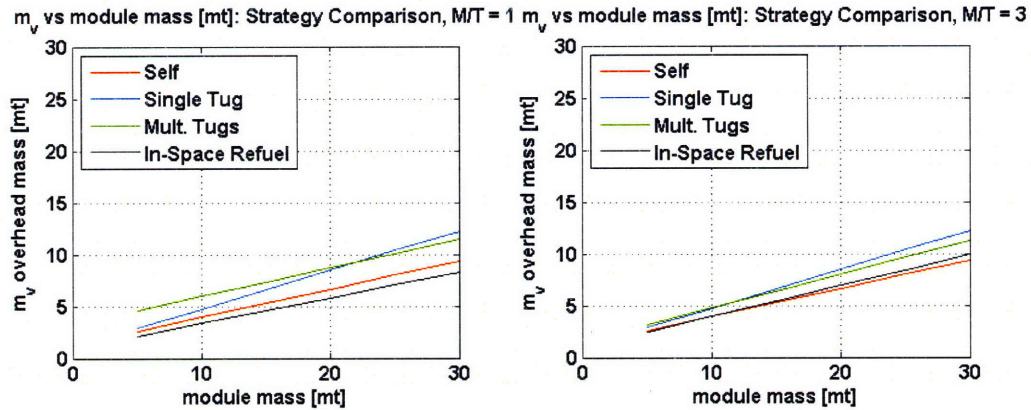


Figure 3.7b: Results for varying assembly orbit altitude, with 5 modules.

3.4.2 Orbit Design Parameters

Figures 3.7a and 3.7b show plots comparing the five assembly strategies as the orbit altitude is varied. The assembly orbit altitude is plotted along the x-axis. In this case, the number of modules in Figure 3.7a is fixed at 2, and in 3.7b at 5. In these plots, no inclination changes are required (based on our analysis, the addition of inclination change simply exacerbates the trends shown here). The required ΔV cost for each scenario is therefore based on the difference between the assembly orbit parameters and the parking orbit, at 185 km.

Based on Figures 3.7a and 3.7b, it is clear that increasing the altitude of the assembly orbit increases the overhead mass for all strategies; again, the slope of the lines indicates the rate at which overhead mass goes up as altitude is increased. The results

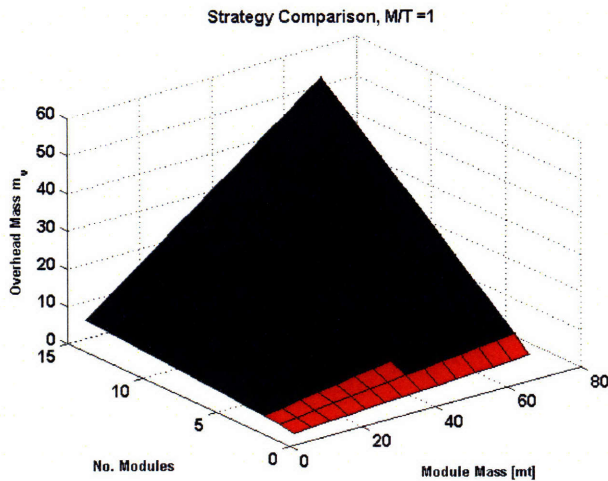


Figure 3.8: Minimum overhead mass as a function of module mass and number of modules. See previous figures for legend..

differ based on the number of modules. In Figure 3.7a, with a 2-module assembly task, the self-assembly strategy is consistently favored, for all orbit altitudes. On the other hand, in Figure 3.7b (5 modules), the self-assembly task has a higher m_v value than in-space refueling when $M/T = 1$. As found in the previous section, self-assembly has the advantage for small numbers of modules.

Interestingly, these plots show the first assembly scenario in which the single tug strategy shows significant advantages. In all cases, for very low assembly orbits (near 200-300 km), the single tug strategy has the lowest overhead mass (along with other strategies). At 400 km, our baseline assembly orbit, the strategy's overhead mass is significantly higher than most of the others, explaining why the single-tug case always appears to poor advantage in the rest of the study.

Similar plots can be generated for changes in orbit inclination, but due to their higher ΔV cost, the trends for these maneuvers are similar but even more pronounced. The data on which these plots are based is presented in tabular form in Appendix D.

3.4.3 Tradespace Exploration

Finally, the results obtained above are combined to create a general idea of the tradespace. With the baseline values for the vehicle and orbit design parameters set (Table 3.1), the overhead mass is plotted as a function of both the module mass and number of modules. Because the lowest overhead mass in Figures 3.5, 3.6, and 3.7 was obtained for $M/T = 1$, we look only at that case here. The surface in Figure 3.8 shows the

minimum additional mass possible at each point in the x - y plane; the color coding shows which strategy provides the minimum mass at that point. (See Figure 3.7 for legend).

For very low numbers of modules, the self-assembly strategy is superior, but the in-space refueling case wins out as the number of modules increases beyond very low values. As the mass increases, in-space refueling becomes valuable at lower numbers of modules. The plot makes a very clear case for in-space refueled space tugs as an assembly strategy.

3.4.4 Sensitivity Analysis

The assembly trades model is based on a number of estimates and assumptions (necessary at this conceptual stage of the study). It is therefore important to understand the sensitivity of the results to changes in these assumptions. The most important assumptions are the fixed engine mass m_{eng} , the tug payload mass m_{pld} and I_{sp} , and the propellant and structures fractions f_{prp} and f_{str} . Recall that the baseline values are given in Table 3.1. In this section, we describe our sensitivity analysis, in which each of these parameters is varied from the baseline, and the results are analyzed to determine the direction and extent of the ensuing change in the results.

We first investigate the sensitivity to the engine mass m_{eng} . Figures 3.9a, 3.9b, and 3.9c show a comparison of all four strategies as the number of modules increases, as in Figure 3.5 above. In this case, however, 3.9a shows the results when the engine mass is 200 kg (the baseline), 3.9b shows the results for an increased mass of 500 kg, and 3.9c shows the results for a decreased mass of 100 kg. We can thereby examine the sensitivity of the results to changes in the engine mass.

In the baseline case (Figure 3.9a), the in-space refueling strategy has the lowest overhead mass in all cases except the one-module and two-module tasks; self-assembly is a close second. Interestingly, neither of the tug strategies appears at all useful due to high overhead mass as the number of modules increases. In the low-mass case (Figure 3.9c), the results change slightly but, significantly, the sorting order of the strategies does not change, indicating that results are relatively insensitive to decreasing engine mass.

In the high-mass case (Figure 3.9b), on the other hand, the results do change somewhat. In-space refueling appears even more valuable as it gains a greater advantage over the other three strategies. However, the single-tug strategy, which looked bad in the

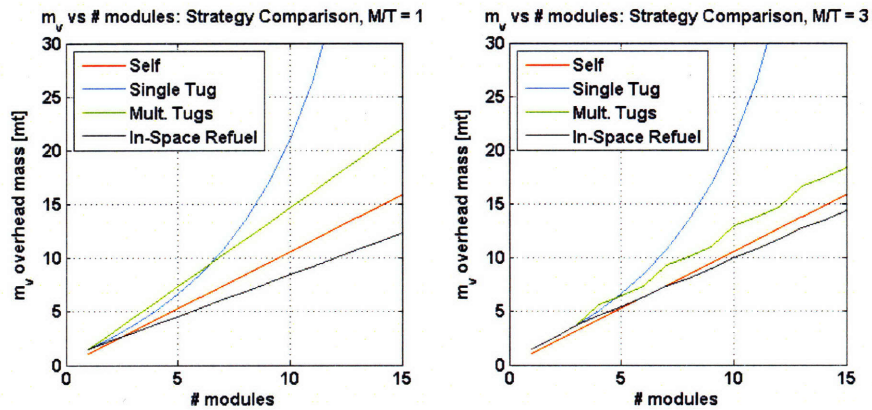


Figure 3.9a: Number of modules comparison, with 200 kg engine mass.

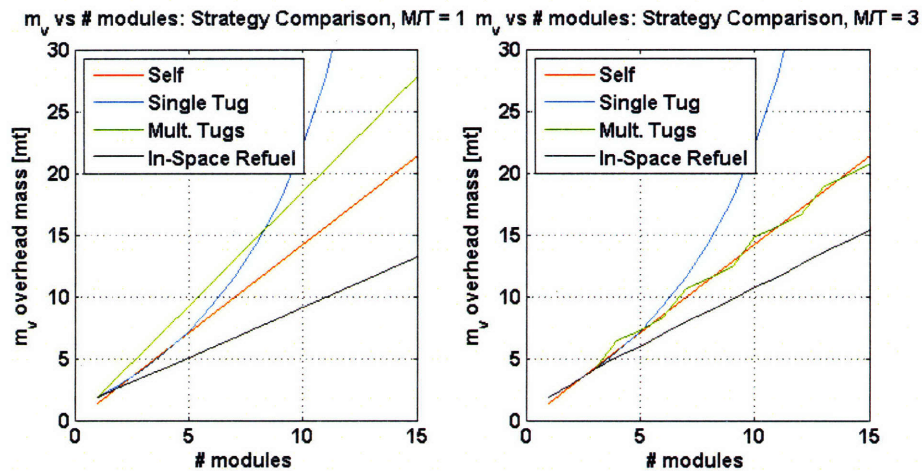


Figure 3.9b: Number of modules comparison, with 500 kg engine mass.

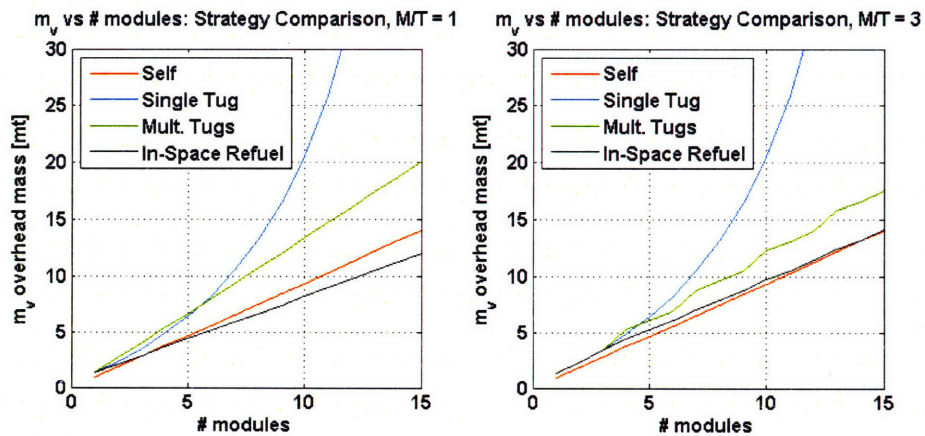


Figure 3.9c: Number of modules comparison, with 100 kg engine mass.

baseline case, is slightly better than self-assembly for smaller assembly tasks. More significantly, the multiple tugs case for $M/T=3$ is an improvement over self-assembly for most assembly tasks (assuming M/T can be adjusted to the task, to remove the ‘jumps’ in m_v). Therefore, we can conclude that the results are indeed sensitive to the engine mass: increasing the fixed engine mass makes the tug cases more attractive, and decreases the relative value of self-assembly.

This sensitivity makes sense because the fixed component of the engine mass is what drives the advantage of space tug-based assembly scenarios. Including this fixed engine mass on every module makes self-assembly less attractive when the engine mass is large.

We expect the choice of propellant type, or Isp, for the space tug to have a similarly significant effect on the results. Recall that the baseline value was 310 s, corresponding to the range of standard bi-propellant propulsion systems. Figures 3.10a, 3.10b, and 3.10c show the results for three other types of propellant: a 200 s Isp for monopropellant, a 420 s Isp for H₂/LO_x, and a 1500 s Isp for electric propulsion, respectively. These figures can be compared to Figure 3.4 (310 s Isp). Based on 3.9a, we can conclude that lowering the Isp makes all four strategies perform less well in terms of overhead mass, but affects them all more or less equally. Raising the Isp slightly to 420 s increases the performance of the single tug strategy slightly, but not enough to surpass in-space refueling or self-assembly. However, giving the Isp a large boost to 1500 s does indeed change the results significantly. Most notably, the single-tug strategy shows very good performance, showing a consistently lower overhead mass than self-assembly. Only the in-space refueling strategy can provide better performance. All four strategies show better performance from the higher Isp, but the single tug strategy is the most sensitive to changes in this parameter. Thus, we can conclude that the results presented in this chapter are only slightly sensitive to small changes in Isp (e.g. from bi-propellant to H₂/LO_x); however, the use of electrical propulsion – or some other high-Isp propellant – could change the study results significantly, making the use of space tugs more attractive.

The same type of study was performed to investigate sensitivity to the tug payload mass. The baseline tug payload mass of 300 kg was both increased and decreased and the results were inspected for changes from the baseline. In this case, however, the results

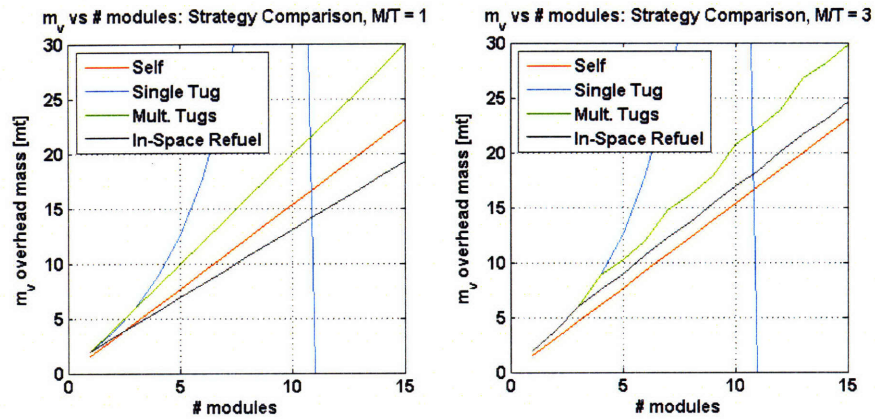


Figure 3.10a: Number of modules comparison, with 200 s Isp.

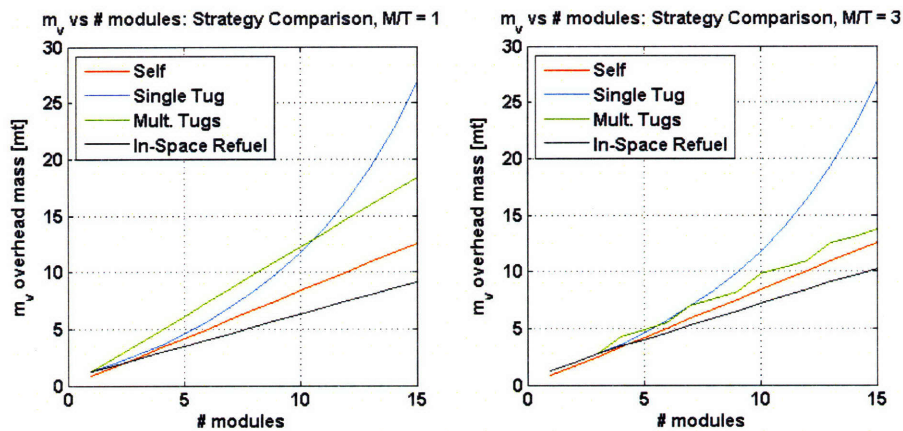


Figure 3.10b: Number of modules comparison, with 420 s Isp.

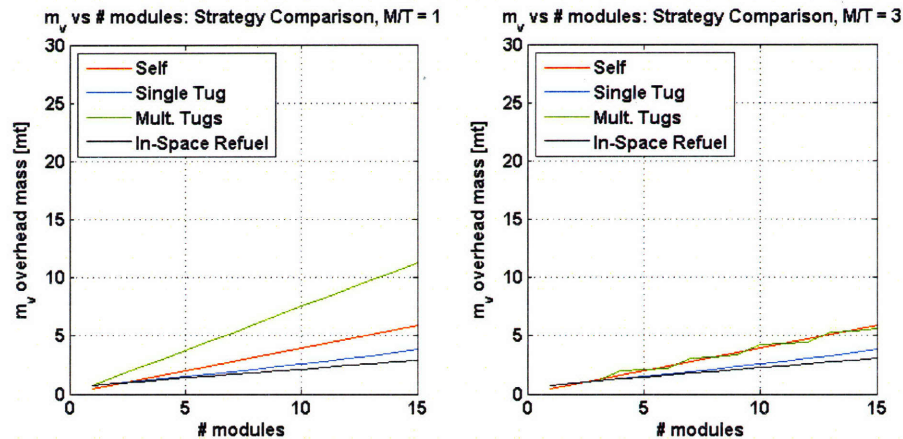


Figure 3.10c: Number of modules comparison, with 1500 s Isp.

were relatively insensitive to changes in this parameter. Reducing the payload mass gives tug-based strategies a slight improvement in overhead mass, but does not change the sorting order of the results; increasing the payload mass slightly increases the overhead mass but, again, does not change the results.

The remaining two parameters – the propellant fraction f_{prp} and the structures fraction f_{str} – were investigated similarly. Reducing f_{prp} from the baseline value 0.12 to 0.05 produced no change in the results; an increase to 0.3 gave only a slight advantage to self-assembly at low numbers of modules. This is to be expected because it increases the impact of the excess propellant that must be carried by the tugs for their return trips. Still, the sensitivity is small. Changes in f_{str} produced virtually no changes in the results. The baseline value of 0.15 was increased to 0.3 and decreased to 0.05 with no effect, probably because this parameter affects both the self-assembly and tug cases nearly equally.

Based on this sensitivity analysis, we can garner increased confidence in this model. The only parameter that shows real sensitivity to changes in assumptions is the fixed engine mass. We expect this parameter to drive the comparison between tug-based strategies and self-assembly strategies. The remaining parameters – tug payload mass, propellant fraction and structures fraction – show relatively little sensitivity to changes in assumptions.

3.5 Assembly Strategy Selection

The results of this tradespace exploration indicate that both tug-based and self-assembly strategies are worthy of further study, because neither was an absolute winner in all assembly scenarios. However, the results clearly indicate that in-space refueling of tugs, as modeled here, is the best assembly strategy (based on our comparison metric) for nearly all assembly tasks. In tasks with very few modules to be assembled, on the other hand, self-assembly often has a lower overhead mass. The single-tug and multiple-tug strategies rarely have lower overhead mass values than either self-assembly or in-space refueling. Based on the launch analysis in Chapter 2 it appears that most lunar architectures with larger launch vehicles (e.g. the 82 mt LV) should proceed with self-assembly as is the current plan. For future Mars missions, however, which might require upwards of 5-6 launches per mission, a refuelable tug-based assembly strategy might be the best option.

3.5.1 Trade Study Conclusions

It is somewhat surprising that both of the non-refueled tug-based strategies performed so poorly in this study. On closer examination, however, this result can be explained. The single tug strategy, as noted earlier, is at an immediate disadvantage at high numbers of modules because it must carry propellant for all its journeys to and from the assembly orbit. The overhead mass therefore increases exponentially, and the strategy is useless for large numbers of modules. The effect can be somewhat lessened by going to a higher Isp propellant (e.g. LH2/LOX in the range of ~400-400 sec) but at this point boil-off issues might start to dominate the problem. Single non-refuelable tugs for on-orbit assembly in LEO might therefore only be viable for proximity operations or once high-thrust, high-Isp electrical propulsion systems become a reality.

The multiple-tug strategy was introduced in an attempt to alleviate this problem. However, by launching multiple tugs, we encounter the same problem as in the self-assembly case: we must launch heavy propulsion, docking, and other hardware quite frequently in order to complete the assembly task. Therefore, in order to make the use of multiple tugs valuable, the right balance must be found between minimizing the number of back-and-forth trips each tug makes, and minimizing the amount of duplicate hardware launched (this balance is controlled by the selection of the M/T parameter). Even with this balance found, the self-assembly case nearly always has a lower overhead mass than the multiple tugs case because the tugs case requires launching *more* excess hardware: not only the propulsion system and propellant tanks, but also the tug payload along with excess propellant for return transfers. The multiple tugs case only appears advantageous in cases where the fixed engine mass is large (rendering the self-assembly 'mass penalty' per module very high). The conclusion therefore is that if the propulsion system hardware is rather light, the use of non-refueled tugs for assembly does not make sense. However, if the propulsion hardware is heavy, non-refueled tugs can indeed be useful.

On the other hand, *refueled* tugs are clearly shown to be the best strategy for on-orbit assembly tasks with more than two or three modules. The strategy performs best when the tug is refueled after assembling only one module. This result is reasonable because rather than launching a new propulsion system on each module (self-assembly), or launching an entirely new tug (propulsion and payload) every few modules, we launch

only the required propellant and tank. The only caveat here is that we do not account for additional propellant required to retrieve each newly launched tank (just as we do not account for propellant for rendezvous with modules, and any excess hardware that may be required to provide attitude control for the tanks⁸). Adding in this relatively small additional propellant requirement might change the results slightly. However, the propellant tanks could also be launched as piggyback payload with the modules; no increased propellant usage would then be incurred.

The self-assembly strategy performs best for tasks with a small number of modules, where other parameters are ‘high-stress’: large modules and/or high assembly orbits. With heavy modules, the addition of a propulsion system is a lower percentage of the total launch mass. With high assembly orbits, the self-assembled modules do not have the tug disadvantage of returning to the parking orbit. However, in most other scenarios, the refueled tug strategy has a lower overhead mass than self-assembly.

A secondary result from this tradespace exploration is the relative lack of sensitivity of the results to changes in three of the most important vehicle design parameters: the tug payload mass, propellant mass fraction, and structures mass fraction. This lack of sensitivity leads to increased confidence in the results of this study (rough estimates and assumptions still probably lead to the correct conclusions).

On the other hand, the results are shown to be sensitive to changes in the engine mass parameter – the *fixed* component of the overhead mass required on each tug or self-assembled module. This result was expected, and indeed provides one of the most important conclusions from this study. When this fixed mass component is increased, the performance of the self-assembly strategy gets worse, and the tug-based assembly strategies become more attractive. As a result, we can conclude that *if the propulsion system mass is high, a tug-based assembly strategy should be used. If the propulsion system mass is low, on the other hand, self-assembly should be considered as a superior alternative.*

⁸ Propellant tanks may be cylindrical in shape with hemispherical end caps and could be attitude stabilized using a simple, low mass tether taking advantage of Earth orbit’s gravity gradient effect.

3.5.2 Future Work

A third result that could be found from a tradespace exploration is an idea of the optimal assembly orbit. However, the level of fidelity of this model is not high enough to capture all the necessary variables. In this study, the assembly orbit simply exacerbates already-present trends. High altitude or high inclination orbits simply increase the ΔV requirements. Future iterations of this model could incorporate relevant orbit perturbations such as drag and solar pressure, which we expect to drive the assembly orbit toward an optimal value. In that particular analysis, the drag-induced altitude losses of modules waiting in an assembly orbit for stack completion will be integrated over time. Thus, a low assembly orbit will incur significant drag losses, while a high assembly orbit is more expensive to reach initially. The optimal assembly orbit is expected to be in between, depending on the total number of modules to be assembled and the expected time interval between successive rendezvous and docking operations.

Additionally, the results of the sensitivity analysis for the Isp parameter showed that very high Isp systems, such as electric tugs, could make a space tug architecture significantly more attractive (lower overhead mass than any options with chemical propulsion). We only touched on this subject briefly, but a more extensive investigation of the potential of electric tugs as assemblers would be enlightening. Two additional elements would be needed in the model: the capability for modeling spiral trajectories and comparing results in terms of time (since electric tugs are generally slow).

Finally, real mission scenarios such as the one described in Chapter 2 feature non-uniform module masses. The model could be extended to handle modules of varying masses that can be described by a vector of module masses or by a distribution function. Also, electrical tugs could be investigated if they appear to offer significant benefits. Allowing for an electrical propulsion tug ($I_{sp} > 1000$ [sec]), will favor the tug, but will cause slower transfers. Non-uniform module masses were not incorporated into this model because they did not add to the objectives of this particular study: to understand the types of tasks for which each strategy is well-suited. Future iterations of the study should focus on the particular strategies and tasks shown to be advantageous and do a more detailed design study; at that point, non-uniform module masses should be incorporated into the model.

3.5.3 Summary of Conclusions

In summary, this assembly trade study accomplished its major objectives of exploring the design space and providing conceptual conclusions about the relative merits of self- and tug-based assembly. The results show that neither the tug case nor the self-assembly case is clearly optimal in all situations, so the trade between the two strategies is worthy of further study. The results also show that the refueled space tug, as modeled here, is a better option than self-assembly for most (but not all) assembly scenarios. The relevant parameters have been identified (vehicle design, orbit design, and assembly requirements) and their impact on the trade has been examined. Sensitivity analysis has been performed to understand the validity of the assumptions inherent in the model. It is clear from this study that a refueled space tug could be a valuable method for on-orbit assembly of various types of modular spacecraft.

4

Conclusions

The preceding two chapters have discussed in-depth studies of two essential components of Earth-to-orbit architectures: the design of architectures that can be easily assembled in orbit, and the assembly methods that make on-orbit assembly efficient and cost-effective. Specific conclusions from each of these studies were provided in each chapter. Here, we synthesize these results in order to draw more general conclusions about the impact of on-orbit assembly on the system-of-systems that makes up a human (or unmanned) space exploration mission.

First, we discuss some of the design strategies identified in this work which have a significant impact on the suitability of an architecture for on-orbit assembly. In that section, we hope to provide useful starting points for designers of next-generation

missions, by giving a number of design techniques that can be used to enhance the affordability of assembled architectures.

Second, we provide a qualitative overview of a few other concerns that impact assemble-able architectures. This section is intended simply to cover a few additional points that were not within the scope of this thesis, but could affect on-orbit assembly.

Finally, we summarize the main sections of this thesis, and provide cohesive conclusions around the impact of on-orbit assembly on next-generation space architectures.

4.1 Design Strategies for Assembly

Throughout this thesis, we have identified a number of design strategies that can significantly impact the reliability and expense of utilizing on-orbit assembly in the context of a space exploration mission. In this context, what we mean by ‘design strategy’ is more along the lines of a ‘design theme’ – concepts and infrastructure that can be built into the design early on, and have a significant impact on the final architecture. Such concepts include reusable in-space infrastructure, on-orbit refueling, high-Isp propulsion, recognizing coupled design spaces, and spacecraft modularity. In this section, we discuss each of these broad design strategies and the manner in which it can impact the design of mission architectures.

One of the common design strategies throughout Chapter 3 is the idea of a *re-usable infrastructure* in space. The key question we addressed was whether it was valuable to invest in such an infrastructure, which might be quite expensive to set up but perhaps less expensive to operate than the non-reusable alternative. This question is relevant for a broad range of space architectures. The conclusion reached also seems applicable to a number of types of missions: in-space infrastructure is only valuable in certain cases. However, in this case (and probably in many others), a reusable infrastructure has the potential to significantly reduce costs, but only if the infrastructure itself is designed for affordable operations and is highly reliable. Here, we found that the space tug infrastructure supporting assembly was only cost-effective (over self-assembly) if in-space refueling was used to support that infrastructure.

In-space refueling thus can be seen as an essential component of re-usable infrastructures, or as a design strategy in its own right. While in-space refueling has not previously been accomplished, it has been shown in this thesis to be a technology worthy of investment, especially for propulsion-intensive operations such as on-orbit assembly, rendezvous, and long-term orbit maintenance. Indeed, without in-space refueling, on-orbit assembly can be best accomplished by installing the same propulsion and guidance hardware on every module, even those with no other use for propulsion; refueling enables a much more efficient and less wasteful solution. Therefore, we conclude that while the technology has not yet been proven, in-space refueling should nevertheless be very seriously considered in any future on-orbit assembly projects because of its potential for significant long-term cost reductions.

Alternatively, or in addition to on-orbit refueling, the use of high Isp electric propulsion systems appears to make the use of space tugs more attractive, as shown in Section 3.4.4. While it was not studied extensively in this thesis, it stands to reason that electric propulsion systems should significantly increase the performance of space tugs, because so little propellant mass must be carried for each maneuver. On-orbit servicing concepts involving electric tugs have been proposed in the past, but none have yet been operated. The only major drawback to such spacecraft is their speed of travel: currently available electric propulsion systems are quite slow, and follow leisurely spiral trajectories rather than fast Hohmann transfers. Because the speed of assembly is generally important (due to boil-off concerns and other issues), electric propulsion may not be a practical choice. If high-thrust systems could be developed, however, they would clearly be the optimal choice. We have shown in this thesis that, like in-space refueling, high Isp propulsion systems are a technology worthy of future investment, because if implemented, such systems could significantly reduce the required overhead mass for on-orbit assembly.

Rather less obvious than the above technology concepts is the idea of *coupled design spaces* and their effect on architecture design. Certainly this is not a new concept in systems engineering, but it has been shown in this thesis that the coupling between launch and assembly has a strong influence on the cost and risk of Earth-to-orbit architectures. To date, much attention has focused on launch costs, while assembly has

been largely ignored. However, we have shown that taking assembly into account during *both* the launch vehicle selection process *and* the design of the vehicles themselves is essential in reducing Earth-to-orbit costs. By so doing, we can design modular spacecraft that fit in launch vehicles and can be assembled efficiently in space. Otherwise, we run the risk of arriving at sub-optimal designs that cannot be easily and cheaply launched *or* assembled. By designing for on-orbit assembly early in the architecture definition stage, the costs of modular architectures can be significantly reduced.

Finally, arguably the most important design strategy is the idea of *modular spacecraft*. Indeed, modularity is one of the key enablers for the research in this thesis; without it, assembly by robots without human assistance would be practically inconceivable. Recall the discussion in Chapter 1 on the extreme complexity of ISS assembly, with its myriad fluid and electrical connections and complex trusses. This type of assembly could not be accomplished by the methods discussed in this thesis; a space tug could not build a truss in space, but it *can* facilitate the docking of two standardized module ports. Modularity plays two key roles in the mission concepts discussed in this thesis: first, we assume that modular spacecraft designs are simple to physically connect so that few complex tasks must be performed in order to assemble them; and second, the idea of modularity allows us to break the functionality of larger monolith spacecraft into a series of smaller modules, so that they fit on smaller launch vehicles. The Space Shuttle, which was not designed to be modular, cannot simply be ‘chunked’ into smaller pieces so that it fits on EELV’s, for example. Modularity thus enables a whole new range of assembly strategies that were previously inconceivable for non-modular construction projects such as trusses or mirrors. These new strategies have the potential to make on-orbit assembly much less costly than it has been in the past, and thereby enable a suite of new exploration mission concepts.

4.2 Additional Architecture Considerations

Aside from the main design strategies identified in this thesis, a number of other considerations impact the Earth-to-orbit architecture of exploration missions. While these ideas are outside the scope of this thesis, they are mentioned here briefly for completeness.

First, a big driver of the launch strategy and the need for on-orbit assembly is the *logistics strategy*. This is especially true for long-term missions such as space stations and planetary bases. If a base is resupplied by infrequent, large missions (e.g. every two years), assembly may be needed in order to launch all the supplies required. However, if more numerous small missions are planned, assembly will be a less important component. In addition, the logistics infrastructure plays a part in assembly requirements. If, for example, consumables such as water, oxygen, or propellant were kept in orbiting depots for pickup by base-bound spacecraft, assembly technologies would be required to ‘fill up’ in orbit. More importantly, the logistics strategy drives the size of each flight ‘stack’, which in turn drives assembly requirements. Thus, the logistics strategy must be kept in mind during the early design process in order to inform planning for on-orbit assembly.

Second, commonality can play a major role in modular spacecraft designs and their ease of assembly. In its simplest conception, commonality might perhaps dictate that all docking ports on all modules be compatible with the same space tug; this is clearly essential for easy on-orbit assembly. However, the concept can be taken much farther, and assemble-able architectures create a fertile environment for common designs. For an excellent example of the use of commonality to minimize hardware development costs, see [Hofstetter 2005]. In addition, common designs across various space exploration missions can make the concept of a reusable assembly infrastructure (as evaluated in this thesis) even more valuable; the same tug could be used to assemble or transport many different types of spacecraft, if they had common docking ports. Thus, commonality in spacecraft design can be a strong enabler for affordable on-orbit assembly.

4.3 Summary & Conclusions

In the first chapter, essential background was provided for the study of on-orbit assembly. It becomes clear on examination of history that assembly projects are technically feasible, but have in the past been complex endeavors requiring significant human involvement and great expense. In order to enable a new generation of more affordable human (and unmanned) space exploration, we hypothesized that the expense and complexity of future assembly projects could be reduced by leveraging past

experience combined with new technologies and design strategies. With modular spacecraft designs and an understanding of the combined launch-and-assembly tradespace, we hoped to develop more affordable on-orbit assembly techniques.

The second chapter developed the combined launch-and-assembly tradespace, in order to understand the coupling between these two components of the Earth-to-orbit architecture. Taking as a case study a lunar/Mars architecture developed for NASA, we examined the options for launching it with an arsenal of rocket options from NASA and industry. Looking at both cost and risk, we concluded that an optimal launch vehicle size does indeed exist for any given transportation architecture. Furthermore, the lowest-risk architecture is not always that with the fewest launches; depending on the nature of the payloads and the sparing strategy, the use of smaller launch vehicles can be equally reliable. These conclusions are based on the idea, proposed in this thesis, that an architecture (set of vehicles) can be *modularized* or ‘chunked’ in order to fit on variously sized launchers (with some modules more ‘replaceable’ than others). Spacecraft can be divided into modules using sophisticated sizing models, or by imposing an estimated mass penalty for the additional hardware required. The key insight from this research was that in order to optimize the Earth-to-orbit architecture, the coupling between launch/assembly and the transportation architecture must be taken into account early in the design process. In other words, the architecture must be designed with the launcher in mind (so that it can be modularized suitably), and the launcher must be chosen based on the characteristics of the spacecraft to be lofted into orbit. The study of this coupling provided an understanding of the combined launch-and-assembly tradespace, which should prove a valuable tool in creating assemble-able architectures for next-generation space exploration.

The third chapter focused on reducing the cost of the on-orbit assembly process itself. An initial assumption was that modular spacecraft designs allowed the complexity of assembly to be reduced such that it can be modeled as a simple rendezvous and docking between two spacecraft, which can be carried out robotically. The basis for this research was the recognition that assembly of modular spacecraft would require propulsion and guidance hardware on every module, seemingly a waste for those that would not need to maneuver under their own power again. Creating a reusable

infrastructure in space to assist in assembly, based on previous on-orbit servicing concepts such as space tugs, could significantly reduce the wasted launch mass for an assemble-able architecture. Our models showed that such an infrastructure did indeed have the potential to significantly reduce the launch mass for on-orbit assembly, if in-space refueling was utilized. However, without in-space refueling or with very large modules, a reusable infrastructure does not provide value, unless new propulsion technologies with much higher propulsive efficiencies than available today are considered. This result allows us to draw two important conclusions: first, in-space refueling is an essential future technology worthy of further research investment; and second, reusable infrastructures for on-orbit assembly have the potential (but are not guaranteed) to significantly increase the affordability of future assembled architectures.

This thesis has thus explored on-orbit assembly from several perspectives, with the unifying theme of increasing the affordability of future human space exploration missions. Armed with the key design strategies, technologies, and assembly methods identified throughout this research, and following the methodologies developed here for analysis of architecture assemble-ability, mission designers should be able to significantly increase the affordability of next-generation space exploration programs.

4.4 Recommendations for Future Work

This thesis intended to explore a broad swath of issues related to on-orbit assembly of space exploration missions, to begin identifying the key design strategies and technologies, and to gain an idea of the design drivers and tradespaces for on-orbit assembly. As a result, more in-depth research can be performed in each of the research areas discussed in this work.

The methodology outlined in Chapter 2 is not entirely complete. Most importantly, the integer optimization problem formulation can be refined to compute results for a larger range of architectures. In addition, other types of problem formulations could be investigated. Finally, the problem could be expanded to include other aspects of the launch packing problem. For example, it would be extremely useful to incorporate the launch volume constraint, which would require the solver to provide the optimal packing solution given both the mass and dimensions of each element.

Potential future work for Chapter 3 was given in Section 3.5.2, and is therefore only summarized here. An optimal assembly orbit could be found by refining the model to incorporate relevant orbit perturbations such as drag and solar pressure. In addition, real mission scenarios could be modeled and evaluated by expanding the model so that it can handle modules of varying mass. A more detailed design study of the assembly strategies and tasks shown here to be advantageous would assist in outlining the tradespace in greater detail. Finally, more detailed investigation of the potential benefits of electric tugs could be carried out.

References

- [AIAA 1992] AIAA, "Guide for the Serviceable Spacecraft Grasping/Berthing/Docking Interfaces," AIAA G-056-1992.
- [Akin 2002] D.L. Akin, M.L. Bowden, "EVA, robotic, and cooperative assembly of large space structures." *2002 IEEE Aerospace Conference Proceedings*, Vol. 7, Big Sky, Montana, Mar 2002.
- [Ayer 2001] S. Ayer, S. England, H. Laird, E. Romo, T. Schuman, "Modular telecommunication satellite network using on-orbit telerobotic assembly." *IEEE Aerospace Conference*, Big Sky, Montana, Mar 2001.
- [Brand 1990] V. Brand, "The Challenge of Assembling a Space Station in Orbit." *AGARD, Space Vehicle Flight Mechanics 8*, 1990.
- [Clark 1987] W. Clark, "On-orbit assembly, integration, and test of a large spacecraft." *EASCON 87: technology for space leadership: 20th annual Electronics and Aerospace Systems Conference*, Washington DC, Oct 1987.

- [Coffman 1997] E.G. Coffman Jr., M.R. Gary, and D.S. Johnson, "Approximation algorithms for bin packing: a survey," *Approximation algorithms for NP-hard problems*. pgs. 46-93. Boston: PWS, 1997.
- [Covault 1997] C. Covault, "Station Faces Difficult Assembly in Orbit." *Aviation Week & Space Technology*. New York: December 8, 1997. Vol. 147, Iss. 23, pg. 47.
- [Crawley 2005] E. Crawley, et al., *Draper/MIT Concept Exploration and Refinement (CE&R) Study, Final Report*. September 2005.
- [Doggett 2002] W. Doggett, "Robotic assembly of truss structures for space systems and future research plans." *IEEE Aerospace Conference Proceedings*, Vol. 7, Big Sky, Montana, Mar 2002.
- [Dornheim 2006] M.A. Dornheim, "Orbital Express to Test Full Autonomy for On-Orbit Service." *Aviation Week & Space Technology*, 4 Jun 2006. Accessed at http://www.aviationnow.com/avnow/news/channel_awst_story.jsp?id=news/aw060506pl.xml.
- [Enright 1998] J. Enright, C. Jilla, and D. Miller, "Modularity and Spacecraft Cost." *Journal of Reducing Space Mission Cost*, 1(2):135-138, 1998.
- [Fehse 2003] W. Fehse, *Automated Rendezvous and Docking of Spacecraft*, Cambridge University Press, Cambridge: 2003.
- [Galabova 2003] K. Galabova, G. Bounova, O. de Weck, and D. Hastings, "Architecting a Family of Space Tugs Based on Orbital Transfer Mission Scenarios." AIAA 2003-6368. *AIAA Space 2003 Conference*, Long Beach, CA, 23-25 September 2003.
- [Goetz 2003] T. Goetz, T. Dark-fox, J. Mayer, "Building the International Space Station: Some Assembly Required." AIAA 2005-2599, *AIAA/ICAS International Air and Space Symposium and Exposition: The Next 100 Years*. Dayton, Ohio, Jul 2003.
- [Gonzalez-Vallejo 1993] J.J. Gonzalez-Vallejo, V.S. Syromiatnikov, "Docking Mechanisms: An Overview of Different Systems." *Fifth European Space Mechanisms & Tribology Symposium: An International Symposium*. Noordwijk, Netherlands, 1993.
- [Gralla 2005] E.L. Gralla, W. Nadir, H. Mamani, and O. de Weck, "Optimal Launch Vehicle Size Determination for Moon-Mars Transportation Architectures." AIAA 2005-6782, *AIAA Space 2005 Conference*, Long Beach, California, Aug-Sep 2005.
- [Hand 2005] D.B. Hand, "Integrated crew exploration and base assembly system for Moon/Mars/Beyond missions." *1st Space Exploration Conference: Continuing the Voyage of Discovery*, Orlando, Florida, Jan-Feb 2005.
- [Hofstetter 2005] W. Hofstetter, P. Wooster, W. Nadir, and E. Crawley. "Affordable Human Moon and Mars Exploration Through Hardware Commonality." AIAA-2005-6757, *AIAA Space 2005*, Long Beach, California, Aug-Sep 2005.
- [Iannotta 2005] B. Iannotta, "DART aims at space rendezvous." *Aerospace America*, March 2005.

- [Isakowitz 2004] S.J. Isakowitz, J.B. Hopkins, J.P. Hopkins Jr., *International Reference Guide to Space Launch Systems, Fourth Edition*, Reston: AIAA, 2004.
- [Kawano 1999] I. Kawano, M. Mokuno, T. Kasai, T. Suzuki, "Result and Evaluation of Autonomous Rendezvous Docking Experiments of ETS-VII." AIAA 99-4073, *AIAA Guidance, Navigation, and Control Conference*, Portland, Oregon, Aug 1999.
- [Langley 1972] R.D. Langley, "Apollo Experience Report - The Docking system," NASA TN D-6854, 1972.
- [Lamassoure 2002] E. Lamassoure, J.H. Saleh, and D.E. Hastings, "Space Systems Flexibility Provided by On-Orbit Servicing: Part 2." *Journal of Spacecraft and Rockets*, Vol. 39, No. 4, Jul-Aug 2002.
- [Lewis 1999] J. Lewis and M.B. Carrol, "Prototype low impact docking system." *AIAA International Space Station Service Vehicle Conference*, 1999.
- [McManus 2003] H. McManus and T. Schuman, "Understanding the Orbital Transfer Vehicle Trade Space." AIAA 2003-6370, *AIAA Space 2003 Conference and Exhibition*, Long Beach, California, Sep 2003.
- [Moe 2005] R. Moe, "An Early In-Space Platform for Technology Development and Demonstration of Robotic Assembly and Servicing." *1st Space Exploration Conference: Continuing the Voyage of Discovery*; Orlando, FL, Jan-Feb 2005.
- [Morgenthaler 1990] G.W. Morgenthaler and M. D'Amara, "Orbital assembly and constructability considerations of candidate manned Mars spacecraft." *The Case for Mars IV: The international exploration of Mars – Consideration for sending humans; Proceedings of the 4th Case for Mars Conference*, Univ. of Colorado, Boulder, Jun 1990.
- [Morgenthaler 1991] G.W. Morgenthaler, "A cost trade-off model for on-orbit assembly logistics." AIAA 91-4148, *AIAA/SOLE 4th Space Logistics Symposium*, Cocoa Beach, Florida, Nov 1991.
- [Moses 2005] R.W. Moses, et al. "Analysis of In-Space Assembly of Modular Systems." *1st Space Exploration Conference: Continuing the Voyage of Discovery*. Orlando, Florida, Jan-Feb 2005.
- [Muller 2002] R.M. Muller, "Assembly and servicing of a large telescope at the International Space Station." *2002 IEEE Aerospace Conference Proceedings*, Vol. 7, Big Sky, Montana, Mar 2002.
- [Nadir 2005] W.D. Nadir, "Multidisciplinary Structural Design and Optimization for Performance, Cost, and Flexibility." M.S. Thesis, Dept. of Aeronautics and Astronautics, Massachusetts Institute of Technology, Feb 2005.
- [NASA 1999] "International Space Station Assembly: A Construction Site in Orbit." *NASA Facts*, IS-1999-06-ISS013JSC, June 1999.
- [NASA 2003] *Columbia Accident Investigation Board Report Volume I*, August 2003.

- [NASA 2004] *The Vision for Space Exploration*, Feb 2004. Accessed at http://www.nasa.gov/mission_pages/exploration/main/index.html.
- [NASA 2005] NASA Docking Systems Technical Interchange Meeting (TIM), Houston, TX, 21 April 2005.
- [NASA 2005a] "NASA's Exploration Systems Architecture Study," NASA-TM-2005-214062, Nov 2005.
- [NASA 2006] "Overview of the DART Mishap Investigation Results," from http://www.nasa.gov/mission_pages/dart/main/index.html, accessed 8 Jun 2006.
- [NASA 2006a] "Ares: NASA's New Rockets Get Names." 6 Jun 2006, accessed at http://www.nasa.gov/mission_pages/exploration/spacecraft/ares_naming.html.
- [NASA/MIT 2005] Various NASA studies and internal MIT models, unpublished.
- [Purves 2002] L.R. Purves, "A method for estimating costs and benefits of space assembly and servicing by astronauts and robots." *2002 IEEE Aerospace Conference Proceedings*, Vol. 7, Big Sky, Montana, Mar 2002.
- [Rumford 2003] T. Rumford, "Demonstration of Autonomous Rendezvous Technology (DART) Project Summary." *Space Systems Technology and Operations*, P. Tchoryk, Jr., and J. Shoemaker, Eds., *Proceedings of SPIE Vol. 5088*, 2003.
- [Saleh 2002] J.H. Saleh, E. Lamassoure, and D.E. Hastings, "Space Systems Flexibility Provided by On-Orbit Servicing: Part 1." *Journal of Spacecraft and Rockets*, Vol. 39, No. 4, Jul-Aug 2002.
- [Senda 2002] K. Senda, Y. Murotsu, A. Mitsuya, H. Adachi, S. Ito, J. Shitakubo, T. Matsumoto, "Hardware Experiments of a Truss Assembly by an Autonomous Space Learning Robot." *Journal of Spacecraft and Rockets*, Vol. 39, No. 2, Mar-Apr 2002.
- [Tobias 1989] A. Tobias, F. Venditti, and N. Cable, "Docking Berthing Systems: Functions and Simulation." *Proceedings of the 2nd European In-Orbit Operations Symposium*, pp. 259-270, 1989.
- [Turner 2001] A.E. Turner, "Cost-Effective Spacecraft Dependent Upon Frequent Non-Intrusive Servicing." *AIAA Space 2001 Conference*, Albuquerque, New Mexico, Aug 2001.
- [Weaver 1987] L.B. Weaver, "The Techniques of Manned On-Orbit Assembly." *Commercial Opportunities in Space*, F. Shahrokhi, C.C. Chao, K.E. Harwell, Eds., Washington DC: 1987/1988.
- [Wertz 1999] J. Wertz and W. Larson, *Space Mission Analysis and Design*. Third Edition. Microcosm, 1999.
- [Wertz 2003] J. Wertz and R. Bell, "Autonomous Rendezvous and Docking Technologies – Status and Prospects." *Proceedings of SPIE Vol. 5088*, 2003.
- [Whelan 2000] D.A. Whelan, E.A. Adler, S.B. Wilson, G. Roesler, "The DARPA Orbital Express Program: Effecting a Revolution in Space-Based Systems." *Small*

Payloads in Space, B.J. Horais, R.J. Twiggs, Eds., *Proceedings of SPIE Vol. 4136*, 2000.

[Zimpfer 2005] D. Zimpfer, P. Kachmar and Seamus Tuohy, "Autonomous Rendezvous, Capture and In-Space Assembly: Past, Present and Future." AIAA 2005-2523. *1st AIAA Space Exploration Conference: Continuing the Voyage of Discovery*. Orlando, Florida, Jan/Feb 2005.

A

Acronyms & Abbreviations

ADBS	Advanced Docking and Berthing System (formerly LIDS)
CE&R	Concept Exploration & Refinement (NASA study)
CEV	Crew Exploration Vehicle
CH ₄ /LO _x	Methane/Liquid Oxygen (Propellant)
CTS	Crew Transfer System
DARPA	Defense Advanced Research Projects Administration
DART	Demonstration of Autonomous Rendezvous Technology
EDS	Earth Departure Stage
EELV	Evolved Expendable Launch Vehicle
ETS-VII	Engineering Test Satellite VII
EVA	Extra-Vehicular Activity

GEO	Geosynchronous Orbit
H2/LOx	Hydrogen/Liquid Oxygen (Propellant)
HLLV	Heavy-Lift Launch Vehicle
HTV	H-II Transfer Vehicle
ISS	International Space Station
IVA	Intra-Vehicular Activity
kg	kilograms
LEO	Low Earth Orbit
LIDS	Low Impact Docking System
LOI	Lunar Orbit Insertion
LV	Launch Vehicle
MIT	Massachusetts Institute of Technology
mt	metric tons
M/T	Modules per Tug/Tank
NASA	National Aeronautics and Space Administration
NASDA	National Aerospace Development Agency of Japan (now JAXA)
TMI	Trans-Moon/Mars Injection

B

Code for Launch Analysis

The following section presents the Matlab model which generated the results discussed in Chapter 2. The code is well-commented (the ‘%’ sign indicates a comment) and should be relatively easy to follow. Please note that the inputs to this model were derived from the study described in [Crawley 2005]. Please refer to Chapter 2 for further explanations.

The ‘main’ function is presented first, and the sub-functions are listed thereafter in alphabetical order.

main.m

```
function OUT = main( IN )
```

```

% -----%
% LAUNCH PACKING TOOL
% Erica L. Gralla
% February 2005
% Version 2: Vector inputs, only "pack" (no rules, no chunking)
%
% DESCRIPTION
% This version of the launch packing tool finds an optimal set of launch
% manifests given an input vector of 'chunk' masses (see below) and a
% launch vehicle size. A full factorial search finds all possible
% combinations of modules on launch vehicles; results are evaluated to find
% the configuration with the lowest surplus launch mass (un-used space on
% the launch vehicle). This version eliminates the rules and 'chunking'
% capabilities of the previous version, in order to accomodate the
% requirements of the transportation team / real options analysis. Results
% are output to a file 'output/results.txt'.
% -----%

% ---- INPUTS & OUTPUTS -----%
%
% Input format:
%
% - A(i)          structure with i entries for i
%                  architectures (or vehicles)
% - A(i).massvec = contains a cell array of the vehicle masses
%                  with their corresponding names
%
%      { 'TMI' 87;
%        'TMI' 87;
%        'Hab' 45;
%        .... .. };
%
% Output format:
%
% - A(i)          structure with i entries for i arch's
% - A(i).results_summary = matrix summarizing the results
%
%      [ lv_capacity  #_launches  total_mass  mass_surplus;
%        ...          ...          ...          ...   ;]
%
% - A text table of the results summary is output to a file:
%   'output/results.txt'
%
% -----%

% FIXED PARAMETERS

global lv_cap;

%%%%%%%%%%
%%%%%%%%%%
% NOTE: For real options study, the launch vehicle capacity is set in a %
% vector on line 93, not here!                                     %
% ----- %                                                    %
max_lv_cap = 82000;                                             %

```

```

min_lv_cap = 82000;
step_lv_cap = 5000;
% ----- %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

print_to_screen = 0;

% ---- INPUT PROCESSING -----%
%
% The processing function produces...
%
% Output format:
%
% - A(i)          contains all the architectures
% - A(i).V(i)     contains all the vehicles present in
%                 each architecture
% - A(i).V(i).name contains the vehicle name
% - A(i).V(i).mass contains the vehicle mass
%
% -----%

% process input
A = process_vector_input( IN );

% loop through all architectures
num_archs = length(A);
for i = 1:num_archs

    max_mass = 0; % initialize
    % find largest module mass
    for v = 1:length( A(i).V )
        if A(i).V(v).mass > max_mass
            max_mass = A(i).V(v).mass;
        end
    end
    % set minimum launch vehicle size
    min_lv_size = step_lv_cap*ceil(max_mass / step_lv_cap);
    if min_lv_size > min_lv_cap
        min_lv_cap = min_lv_size;
    end
    lv_cap_range = min_lv_cap:step_lv_cap:max_lv_cap;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ---- SET LAUNCH VEHICLE CAPACITY HERE ----%

% modify for discrete set of lv_caps
lv_cap_range = [ 82000 ];
%lv_cap_range = [ 115000 105000 ];% 95000 82000 73000 63000 51000 ];%42000 ];

% -----%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

% loop through all lv_cap values
for lv_cap_index = 1:length(lv_cap_range)
    lv_cap = lv_cap_range(lv_cap_index);

    l = 1;

    % just need to figure out vehicle indices, match them up with names
    % at the end.
    manifest = 0;          % initialize
    for v = 1:length(A(i).V)
        manifest(v) = v;    % populate with all vehicle indices
    end

    packing_matrix = do_packing( manifest, A(i) );
    % clean up packing_matrix; eliminate zeros, save to A(i).L(l) one
    % row at a time
    num_rows = size(packing_matrix);
    for r=1:num_rows
        mat_row = packing_matrix(r,:);
        mat_row = sort(mat_row, 'descend');
        ctr = 1;
        e = 1;
        while ( (e>0) & (ctr<=numel(mat_row)) )
            e = mat_row(ctr);
            ctr = ctr+1;
        end
        if e == 0
            man_final = mat_row(1:(ctr-2));
        else
            man_final = mat_row(1:(ctr-1));
        end
        A(i).L(l).manifest = man_final;
        l = l+1;
    end
    if print_to_screen
        fprintf(':: Final manifests are\n')
        disp_manifests( A(i).L )
        % display masses
        fprintf(':: Final manifest masses are\n')
        for ctr = 1:length( A(i).L )
            disp( get_manifest_mass( A(i).L(ctr).manifest, A(i) ) )
        end
        % display mass surpluses
        fprintf(':: Final manifest mass surpluses are\n')
        for ctr = 1:length( A(i).L )
            disp( lv_cap - get_manifest_mass( A(i).L(ctr).manifest, A(i) ) )
        end
    end
    % save results to a matrix
    if lv_cap_index == 1
        A(i).results_summary = zeros(1,4);
    end
    num_launches = length( A(i).L );

```

```

man_mass = 0;
for ctr = 1:num_launches
    man_mass = man_mass + get_manifest_mass( A(i).L(ctr).manifest, A(i) );
end
mass_surp = 0;
for ctr = 1:length( A(i).L )
    mass_surp = mass_surp + lv_cap - get_manifest_mass( A(i).L(ctr).manifest, A(i) );
end
results_row = [ lv_cap num_launches man_mass mass_surp ];
A(i).results_summary(lv_cap_index,:) = results_row;

A(i).L = [];

end % lv_cap_index

% display results matrix
A(i).results_summary;

end % i=arch index

% write results to a file
fid = fopen('output/results.txt','w');
fprintf(fid,'LV_Size_kg | # Ls | Mass | Mass Surp |\n');
for i = 1:length(A)
    fprintf(fid,'\nARCH %3.0f\n',i);
    fprintf(fid,' %6.0f %3.0f %8.0f %8.0f\n',transpose(A(i).results_summary));
end
fclose(fid);

% assign output arg
OUT = A;

```

disp_manifests.m

```

function disp_manifests( IN )

for i = 1:length( IN )
    disp( IN(i).manifest )
end

```

do_modularization.m

```

function [ man_in A_i ] = do_modularization( man_in, A_i )

% -----%
% LAUNCH "CHUNKING" STAGE
% Divide large modules into chunks that fit on a launch vehicle

global lv_cap;
%lv_cap = get_constant('lv_cap');

% need some logic here to enforce same-size chunks for certain
% cases (e.g. TMI manifests). Add later.

```

```

num_elements = length( man_in );
for j = 1:num_elements
    v = man_in(j);           % get the vehicle number of the 'current' chunk

    num_chunks = ceil( A_i.V(v).mass / lv_cap ); % initialize
    A_i.V(v).chunks = num_chunks;           % save
    man_in = [man_in v.*ones(1,num_chunks-1)]; % add v indices to manifest for each chunk
    chunk_mass = get_manifest_mass( [v], A_i ); % initialize
    while chunk_mass > lv_cap
        num_chunks = num_chunks+1;
        A_i.V(v).chunks = num_chunks;
        chunk_mass = get_manifest_mass( [v], A_i );
        man_in = [man_in v];           % add another 'v' to the manifest for each chunk
    end
end

end

```

do_packing.m

```

function out = do_packing( man_in, A_i )

% The input to the function is a particular architecture A(i) and a single
% manifest which needs to be packed into several launches.

print_to_screen = 0;           % controls whether results are printed to screen
global lv_cap;
%lv_cap = get_constant('lv_cap');

% My General Strategy -->
% use julien's script to generate all comb's, e.g.
% partition(length(manifest)) then use those as indices to the manifest
% itself.

all_partitions = partition_take1( length(man_in) ); % call func. below

if print_to_screen
    fprintf(':: Current manifest to be packed is\n')
    disp(man_in);
end

% loop through each 'partition' matrix
num_partitions = length(all_partitions);
for p = 1:num_partitions

    all_manifests{p} = all_partitions{p}; % need to do this element-by-element
    for e = 1:numel(all_partitions{p})
        index = all_partitions{p}(e);
        if index ~= 0
            all_manifests{p}(e) = man_in(index);
        end
    end
end
end
% p=num_partitions in all_partitions

```

```

% if print_to_screen
%   fprintf(':: The all_manifests array is\n');
%   celldisp(all_manifests, ' all_manifests');
% end

% this generates a cell array all_manifests that distributes the
% elements of man_in according to the indices in all_partitions.
% thus, each element of all_manifests contains a matrix; the rows
% of the matrix represent the launch manifests. for example,
%
% all_manifests{1} = [ 1 2 2      --> launch 1
%                   2 5 0 ]    --> launch 2
%

% next, we need to evaluate each set of manifests for the most efficient to
% launch. metric: minimize # launches, then surplus mass?

lm = 1; % initialize launchable manifest counter
for p = 1:length(all_manifests)

    % are all rows launchable?
    manifest_set = all_manifests{p};
    num_rows = size(manifest_set,1);
    launchable = 1; % assume launchable, initially
    for r = 1:num_rows
        man_row = manifest_set(r,:);
        man_row_mass = get_manifest_mass( man_row, A_i );
        if man_row_mass > lv_cap
            % throw this one out --> how?
            launchable = 0;
        end
    end
end
% if print_to_screen
%   fprintf(':: Testing set %4.0f\n', p)
%   fprintf(':: Launchable is %1.0f\n', launchable)
%   disp(manifest_set)
% end
if launchable
    % save somewhere new
    % check for duplicates?? --> not yet
    launchable_manifests{lm} = manifest_set;
    lm = lm+1;
end
end

% if print_to_screen
%   celldisp(launchable_manifests, ' launchable_manifests')
% end

% now we have a cell array of launchable manifests (launchable_manifests);
% loop through that to evaluate their number of launches and then mass
% surplus.

flm = 1; % initialize fewest launches counter
fewest_launches_manifests = cell(1); % initialize to empty
min_num_launches = length( man_in ); % set to max possible # launches

```

```

for lm = 1:length(launchable_manifests)
    num_launches = size( launchable_manifests{lm}, 1 );    % # rows = # launches
    if num_launches < min_num_launches
        min_num_launches = num_launches;
        fewest_launches_manifests = cell(1);            % re-initialize [empty!]
        fewest_launches_manifests{1} = launchable_manifests{lm};
        flm = 1+1;
    elseif num_launches == min_num_launches
        fewest_launches_manifests{flm} = launchable_manifests{lm};
        flm = flm+1;
    end
end

if print_to_screen
    celldisp( fewest_launches_manifests, ' fewLs_manifests')
end

% now we have a cell array of the launchable manifests with the fewest
% number of launches (fewest_launches_manifests). loop through that to find
% one that works; need some criteria to distinguish between them!

% for now, ARBITRARILY choose one (change!!!!!!)
final_manifest = fewest_launches_manifests{1};

out = final_manifest;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% LEO launch architecture option space generator
% From Julien Lamamy
% 16.981, NASA CER Project, Fall 2004

function curpart = partition_take1(card)

% function curpart=partition_take1(card)
%
% Written Monday July 23rd 2002 by Julien-Alexandre Lamamy
% Program that lists all the partitions of a sample space
% the program is recursive
% INPUT: cardinal of the sample space: card
% OUPUT: all partitions of the sample space where an elt is a number

% Definition of the sample space
omega=[1:card];
% ex:omega=[1 2 3 4 ];

% the form used to define a partition is the following:
% let A be a partition of omega=[1 2 3 4]:
% A=[1 0;
% 2 4;
% 3 0]

```



```

% all elements on a row belong to the same subset
% each row is different subset. A is {1}U(2,4)U{3}

% partitions variables are cells
% current partition
curpart={omega(1,1)};

% recursive loop
for i=2:size(omega,2)
    % addnumb is the number of the sample space that is being added to the partitions
    addnumb=omega(i);
    %partition being updated
    uppart={};
    % marker for the updated cell
    k=1;
    for j=1:size(curpart,2)
        % Inside addition
        % add the current number inside existing one subset of the partition considered
        % we are considering the partition curpart{1,j} for this partition we are to add
        % addnum to each subset in order to have new partitions for the increased sample
        % space. num_subsets is the number of subsets in the current partition.
        numb_subsets=size(curpart{1,j},1);
        for cur_subset=1:numb_subsets
            uppart{1,k}=[curpart{1,j} zeros(size(curpart{1,j},1),1)];
            %celldisp(uppart)
            uppart{1,k}(cur_subset,size(uppart{1,k},2))=addnumb;
            %celldisp(uppart)
            % update k
            k=k+1;
        end
        % Outside addition
        % add addnumb to the current partition as a single elt subset
        % it means here creating a new row to the matrix that represents the partition
        % and putting addnum as the first elt of this new row
        uppart{1,k}=[curpart{1,j}; zeros(1,size(curpart{1,j},2))];
        uppart{1,k}(size(uppart{1,k},1),1)=addnumb;
        k=k+1;
    end
    curpart=uppart;
    %celldisp(curpart)
end
% celldisp(curpart);
end

```

get_constant.m

```

function [ output_value ] = get_constant( requested_constant )

% This function returns the requested constant. It is an easier way to
% store constants for use by several matlab functions.

if ( strcmp( requested_constant, 'lv_cap' ) )
    output_value = 40000;
    disp('CALLED LV_CAP CONSTANT FUNCTION -- ERROR')
end

```

```

    % launch vehicle capability, kg
elseif ( strcmp( requested_constant, 'mass_penalty' ) )
    output_value = 1000;
    % modularization mass penalty, kg
elseif ( strcmp( requested_constant, 'PL_fit' ) )
    output_value = 4;
    % flight number for the payload
end

```

get_manifest_mass.m

```
function mass_sum = get_manifest_mass( manifest, A_i )
```

```

% This function calculates the mass of a launch manifest. The vector
% 'manifest' contains a list of indices to the vehicles in the structure
% A_i (representing a single OPN architecture). For each vehicle in the
% manifest, the mass of the vehicle or chunk is found from the structure
% A_i.

```

```

mass_sum = 0; % initialize
if manifest == 0
    mass_sum = 0;
    return
end
for i = 1:length( manifest )

    mass_penalty = get_constant('mass_penalty');

    v = manifest(i);

    if v == 0
        chunk_mass = 0;
    else

        % add mass penalty only if it is 'chunked'
        if A_i.V(v).chunks > 1
            chunk_mass = (A_i.V(v).mass / A_i.V(v).chunks) + mass_penalty;
        else
            chunk_mass = A_i.V(v).mass;
        end

    end

    mass_sum = mass_sum + chunk_mass;
end

```

get_opn_constants.m

```
function output = get_opn_constants()
```

```

% The purpose of this function is to capture the hard-coded aspects of
% using the OPN output in this code. For example, we capture the indexes
% that relate entries in the 'OPN output matrix' to their names (e.g. the

```

```
% first column is the 'CEV', and the second is the 'HAB1'). Output is a
% structure containing all necessary information.
```

```
% re-write this when Bill's matrix changes
```

```
% index for the launching state (the only one i care about)
```

```
output.E_Launching_State = 1;
```

```
% info for 'payload' vehicle
```

```
output.payload_fitnum = get_constant('PL_fit');
```

```
output.payload_name = 'PL';
```

```
% indices for the vehicles (needs to be sequential and increasing)
```

```
output.vehicles.CEVa = 1;
```

```
output.vehicles.CEVb = 2;
```

```
output.vehicles.HAB1 = 3;
```

```
output.vehicles.HAB2 = 4;
```

```
output.vehicles.HAB3 = 5;
```

```
output.vehicles.HAB4 = 6;
```

```
output.vehicles.HAB4b = 7;
```

```
output.vehicles.TMI1 = 8;
```

```
output.vehicles.TMI2 = 9;
```

```
output.vehicles.TMI3 = 10;
```

```
output.vehicles.TMI4 = 11;
```

```
output.vehicles.DSc = 12;
```

```
output.vehicles.DS4 = 13;
```

```
output.vehicles.AS = 14;
```

```
output.vehicles.TEI = 15;
```

```
% number of columns in the OPN matrix
```

```
output.num_matrix_cols = 15;
```

```
end
```

get_vehicle_name.m

```
function output = get_vehicle_name( v )
```

```
% Gets the vehicle name corresponding to a numeric index referring to a
```

```
% column in the opn output matrix. Assumes that the list of indices is
```

```
% sequentially increasing.
```

```
opn_constants = get_opn_constants;
```

```
possible_vehicles = fieldnames(opn_constants.vehicles);
```

```
num_possible_vehicles = length(possible_vehicles);
```

```
for i = 1:num_possible_vehicles
```

```
    vehicle_index = getfield(opn_constants.vehicles, char(possible_vehicles(i)));
```

```
    if vehicle_index == v
```

```
        output = possible_vehicles(i);
```

```
        return
```

```
    end
```

```
end
```

```
output = 0;
```

```
fprintf('ERROR [get_vehicle_name]: vehicle name not present for vehicle index %f, v);
```

process_opninteg_output.m

```
function A = process_opninteg_output( IN )

% ---- INPUT PROCESSING -----%
%
% The following code takes the integration tool outputs as input, and
% produces the following structure:
%
% - A(i)           contains all the OPN architectures
% - A(i).V(i)      contains all the vehicles present in
%                  each architecture
% - A(i).V(i).name contains the vehicle name
% - A(i).V(i).mass contains the vehicle mass
% - A(i).V(i).flt  contains the assigned flt num at launch
% - A(i).V(i).chunks holds the number of 'chunks' a module has
%
% -----%

OPN_CONSTANTS = get_opn_constants;    % get OPN hardcoded stuff
num_archs = length(IN);              % get number of OPN archs
i = 1;                                % architecture index
% loop through all the architectures in the IN struct
for i = 1:num_archs
    v = 1;                             % vehicle index
    % loop through all columns in the OPN matrix
    for j = 1:OPN_CONSTANTS.num_matrix_cols
        % the initial flt num is in 'Earth Launching State'
        flt_num = IN(i).matrix(OPN_CONSTANTS.E_Launching_State,j);
        if flt_num > 0
            A(i).V(v).flt = flt_num;
            A(i).V(v).name = char(get_vehicle_name(j));
            A(i).V(v).mass = getfield(IN(i).masses, char(A(i).V(v).name));
            A(i).V(v).chunks = 1;
            v = v+1;
        end
    end
    % add in the payload mass (not a 'vehicle' in the OPN matrix)
    opn_constants = get_opn_constants;
    pl_name = opn_constants.payload_name;
    pl_mass = getfield(IN(i).masses, pl_name);
    if pl_mass > 0
        A(i).V(v).flt = opn_constants.payload_fitnum;
        A(i).V(v).name = opn_constants.payload_name;
        A(i).V(v).mass = getfield(IN(i).masses, char(A(i).V(v).name));
        A(i).V(v).chunks = 1;
    end
end
end
```

process_vector_input.m

```
function A = process_vector_input( IN )

% ---- INPUT PROCESSING -----%
```

```

%
% Input format:
%
% - A(i).massvec      contains a cell array of the vehicle masses
%                    with their corresponding names
%
%      { 'TMI' 87;
%        'TMI' 87;
%        'Hab' 45;
%        .... ... };
%
% Output format:
%
% - A(i)              contains all the architectures
% - A(i).V(i)         contains all the vehicles present in
%                    each architecture
% - A(i).V(i).name    contains the vehicle name
% - A(i).V(i).mass    contains the vehicle mass
%
% -----%

num_archs = length(IN);          % get num of archs
i = 1;
% loop through all the architectures in the IN struct
for i = 1:num_archs

    num_rows = size(IN(i).massvec,1);
    for j = 1:num_rows
        A(i).V(j).name = IN(i).massvec{j,1};
        A(i).V(j).mass = IN(i).massvec{j,2};
        A(i).V(j).chunks = 1;
    end
end
end

```


C

Code for Assembly Model

This section provides the Matlab code for the assembly model discussed in Chapter 3. It is commented throughout and should be relatively easy to follow; the ‘%’ symbol indicates a comment line. The main routine to run for trade studies is called ‘trades_new.m’, and this file can be used as a starting point for learning how to run the rest of the model. The remainder of the required files are given in alphabetical order.

trades_new.m

```
% compares the five strategies: four plots, each at different M/Tug val
modules_per_tug = [1 3];
y_lim = 30;
```

```

% SELECT PLOT TYPE
% -----

% plot ma vs number of modules
NUMBER_OF_MODULES_PLOT = 1;
num_mod = [1:15];

% plot ma vs module mass
MODULE_MASS_PLOT = 0;
mod_mass = [5000:5000:30000];
number_of_modules_fixed = 5;

ORBIT_ALT_PLOT = 0;
orb_alt = [200:100:1000];
number_of_modules_fixed = 5;

% plot ma vs # mod vs mod mass on a surface
PLOT_STRATEGY_COMPARISON_SURFACE = 0;
surf_num_modules = [1:15];
surf_m_modules = [5000:5000:70000];
surf_m_per_t = 1;
surf_recalc = 1;

% error check
if (NUMBER_OF_MODULES_PLOT + MODULE_MASS_PLOT + ORBIT_ALT_PLOT +
PLOT_STRATEGY_COMPARISON_SURFACE)~=1
    disp('Please select one type of plot');
    return
end

% SINGLE-PARAMETER GRAPHS (TWO-DIMENSIONAL)
% -----

if (~PLOT_STRATEGY_COMPARISON_SURFACE)
    clear params
    clear ma_self ma_tug1 ma_tug2 ma_tug3 ma_tug4 varied_parameter
    % clear output text file
    outfile = fopen('tug_assembly_output.txt','w');
    fclose(outfile);

    load params_chem
    count = 1;

    %%%%%%%%%%%
    % params change for sensitivity
    % params.isp = 200;
    %%%%%%%%%%%

    if NUMBER_OF_MODULES_PLOT
        varied_parameter = num_mod;
        x_factor = 1;
        plot_string = '# modules';
    elseif MODULE_MASS_PLOT
        varied_parameter = mod_mass;
        nm = number_of_modules_fixed;
    end
end

```



```

    x_factor = 10^-3;
    plot_string = 'module mass [mt]';
elseif ORBIT_ALT_PLOT
    varied_parameter = orb_alt;
    nm = number_of_modules_fixed;
    x_factor = 1;
    plot_string = 'orbit altitude [km]';
end

winrows = ceil( max(size(modules_per_tug))/2 );
for plotcount = 1:(max(size(modules_per_tug)))
    m_per_t = modules_per_tug(plotcount);
    params.m_per_t = m_per_t;
    for i = 1:max(size(varied_parameter))
        if NUMBER_OF_MODULES_PLOT
            nm = varied_parameter(i);
        elseif MODULE_MASS_PLOT
            params.m_mod = varied_parameter(i);
        elseif ORBIT_ALT_PLOT
            params.asy_r = varied_parameter(i)+6378;
        end
        ma_self(i) = do_selfassembly(nm,params);
        ma_tug1(i) = do_tugassembly(nm,1,params);
        ma_tug2(i) = do_tugassembly(nm,2,params);
        ma_tug3(i) = do_tugassembly(nm,3,params);
        %ma_tug4(i) = do_tugassembly(nm,4,params);
    end
    subplot(winrows,2,plotcount);
    plot( ...
        varied_parameter*x_factor,ma_self*10^-3,'r-', ...
        varied_parameter*x_factor,ma_tug1*10^-3,'b-', ...
        varied_parameter*x_factor,ma_tug2*10^-3,'g-', ...
        varied_parameter*x_factor,ma_tug3*10^-3,'k-'); %, ...
        %num_mod,ma_tug4*10^-3,'m-');
    title(strcat(['m_v vs ' plot_string ': Strategy Comparison, M/T = ' num2str(m_per_t)
]),'FontWeight','bold','FontSize',12);
    ylabel('m_v overhead mass [mt]','FontWeight','bold','FontSize',12);
    xlabel(plot_string,'FontWeight','bold','FontSize',12);
    legend('Self','Single Tug','Mult. Tugs','In-Space Refuel','Prop. w
Modules','Location','NorthWest');
    axis([0 max(varied_parameter*x_factor) 0 y_lim]);
    grid on
    set(gca,'FontSize',12,'FontWeight','bold');
    count = count + 1;
end
set(gcf,'Color','w');
end

% ----- %
% STRATEGY COMPARISON: SURFACE PLOT
% ----- %

if PLOT_STRATEGY_COMPARISON_SURFACE
    % compares the five strategies: on a surface *****add more*****

    figure(3)

```

```

hold off
clear params
load params_chem

if surf_recalc
    clear mas_self mas_tug1 mas_tug2 mas_tug3 mas_tug4;
    for n = 1:length(surf_num_modules)
        num_mod = surf_num_modules(n);
        num_mod

        for m = 1:length(surf_m_modules)
            m_mod = surf_m_modules(m);
            params.m_mod = m_mod;
            params.m_per_t = surf_m_per_t;
            mas_self(n,m) = do_selfassembly(n,params);
            mas_tug1(n,m) = do_tugassembly(n,1,params);
            mas_tug2(n,m) = do_tugassembly(n,2,params);
            mas_tug3(n,m) = do_tugassembly(n,3,params);
            %mas_tug4(n,m) = do_tugassembly(n,4,params);
        end
    end
end

%my_colormap = [1 0 0; 0 0 1; 0 1 0; 0 0 0; 1 0 1];
my_colormap = [1 0 0; 0 0 1; 0 1 0; 0 0 0];
colormap(my_colormap);

C = ones(size(mas_self)); % self = 1

ma_min1 = min(mas_tug1,mas_self);
C( (ma_min1-mas_self)<0 ) = 2; % tug1 = 2
ma_min2 = min(mas_tug2,ma_min1);
C( (ma_min2-ma_min1)<0 ) = 3; % tug2 = 3
ma_min3 = min(mas_tug3,ma_min2);
C( (ma_min3-ma_min2)<0 ) = 4; % tug3 = 4

ma_plot = ma_min3;

%ma_min4 = min(mas_tug4,ma_min3);
%C( (ma_min4-ma_min3)<0 ) = 5; % tug4 = 5
%ma_plot = ma_min4;

% ma_max_tug0 = max(ma_self,ma_tug0);
% ma_tug0_indices = ceil((ma_max_tug0 - ma_self)./1e6);
% C = ma_tug0_indices+1;
% ma_max_tug1 = max(ma_max_tug0,ma_tug1);
% ma_tug1_indices = ceil((ma_max_tug1 - ma_max_tug0)./1e6);
% C(ma_tug1_indices>0) = 3;
% ma_max_tug2 = max(ma_max_tug1,ma_tug2);
% ma_tug2_indices = ceil((ma_max_tug1 - ma_max_tug1)./1e6);

surf(surf_m_modules.*10^-3,surf_num_modules,ma_plot.*10^-3,C);
xlabel('Module Mass [mt'],'FontWeight','bold');
ylabel('No. Modules','FontWeight','bold');
zlabel('Additional Mass m_a','FontWeight','bold');

```

```

    title(strcat('Strategy Comparison, M/T =
,num2str(surf_m_per_t),'FontWeight','bold','FontSize',12);
    set(gca,'FontSize',12,'FontWeight','bold');
    set(gcf,'Color','w');

end

```

do_hohmann.m

```

function [dv_h, dt_h] = do_hohmann( r1, r2 )
% DO_HOHMANN Calculates the delta-v for a Hohmann transfer between circular
% orbits with radii r1, r2.
%
% Usage: [dv_h,dt_h] = do_hohmann( r1, r2 )
%
% Inputs: r1, r2    orbital radii (km)
% Output: dv_h     delta-v (km/s)
%         dt_h     delta-t (s)

% issue warning if r1, r2 look like altitudes rather than radii
if ( r1<6378 | r2<6378 )
    disp('WARNING: r1 and r2 smaller than Earth radius!')
end

mu = get_constant('mu_E'); % km3/s2

% DELTA-V
dv1 = abs((mu*(2/r1 - 2/(r1+r2)))^(1/2) - (mu*(1/r1))^(1/2)); % km/s
dv2 = abs((mu*(2/r2 - 2/(r1+r2)))^(1/2) - (mu*(1/r2))^(1/2)); % km/s

dv_h = dv1+dv2;

% DELTA-T
ah = (1/2)*(r1+r2);
dt_h = pi*(ah^3/mu)^(1/2);

```

do_incchange.m

```

function dv_i = do_incchange( r, theta )
% DO_INCCHANGE Calculates the delta-v required to perform a simple
% inclination change (from circular orbit to circular orbit) through angle
% theta.
%
% Usage: dv_i = do_incchange( r, theta )
%
% Inputs: theta    angle [deg]
%         r        orbit radius [km]
% Outputs: dv_i    delta-v [km/s]

% issue warning if r looks like altitude rather than radius
if ( r<6378 )
    disp('WARNING: r1 and r2 smaller than Earth radius!')
end

```

```

end

mu = get_constant('mu_E'); % km3/s2

v = (mu/r)^(1/2);

theta_rad = theta*pi/180; % convert to radians
dv_i = 2*v*sin(theta_rad/2);

```

do_selfassembly.m

```

function [ma] = do_selfassembly(num_mod,params)
% DO_SELFASSEMBLY Replaces run_selfassembly. More accurate modeling (no
% longer uses mass penalty).

outfile = fopen('tug_assembly_output.txt','a');

fprintf(outfile, ...
'SELF ASSEMBLY OUTPUT\n\n');
fprintf(outfile, ...
'Number of Modules: %6.0f\n', ...
num_mod);
fprintf(outfile, ...
'Module Mass: %6.0f kg\nMass Penalty: %6.0f kg\nTug Payload Mass: %6.0f kg\n',...
params.m_mod,params.m_pen,params.m_pld);

% user settings
ITER_LIM = 150; % iteration limit for mp_tug calculation
MP_TOL = 1;

% ----- %
% DELTA-V CALCULATIONS
% ----- %

% dv for inclination change
if params.pkg_i ~= params.asy_i
theta = abs(params.pkg_i - params.asy_i); % deg
dv_i = do_incchange( params.pkg_r, theta );
else
dv_i = 0;
end

% dv for hohmann transfer
[dv_h,dt_h] = do_hohmann( params.pkg_r, params.asy_r );

% dv for rendezvous and capture [Saleh/Lamassoure]: could add 120 m/s -- or
% 60 m/s one way

% total delta-v
dv = dv_i + dv_h;

fprintf(outfile, ...

```

```

'Delta-V (one-way): %6.0f m/s\nDelta-V (total): %6.0f m/s\n\n', ...
dv*10^3,dv*num_mod*10^3);

% ----- %
% PROPELLANT MASS CALCULATIONS
% ----- %

% initialize
mp_guess = params.mp_tug;
mp_calc = 0;
iter = 0;

while abs(mp_guess - mp_calc)>MP_TOL
    if iter>0
        mp_guess = mp_calc;
    end
    % initialize
    mp_mod = mp_guess;
    mp_toasy = 0;
    % update dry mass
    m_eng = params.m_pen;           % engine mass (=mass penalty)
    m_pld = params.m_mod;          % module mass (=payload in self-assy)
    m_tank = params.f_prp*mp_mod;   % tank mass
    m_str = params.f_str*(mp_mod+m_tank+m_eng); % structure mass
    m_dry = m_pld+m_tank+m_str+m_eng; % dry mass

    % prop for transfer to assembly orbit
    m0 = mp_mod+m_dry;              % wet mass
    mp_toasy = do_transfer_m0( dv, m0, params );

    mp_calc = mp_toasy;
    iter = iter+1;
    if iter > ITER_LIM
        disp('Iteration limit reached. Exiting...')
        ma = -1;
        return
    end
end

% store mass of propellant calculated
mp_mod = mp_calc;

% find mass overhead ma
ma = (num_mod)*(m_str+m_tank+mp_mod+m_eng);

% print results to text file
fprintf(outfile, ...
'Self Prop Mass: %6.0f kg\nSelf Dry Mass: %6.0f kg\n', ...
mp_mod,m_dry);
fprintf(outfile, ...
' Payload: %6.0f kg\n Structure: %6.0f kg\n Tank: %6.0f kg\n Engine:
%6.0f kg\n', ...
m_pld,m_str,m_tank,m_eng);
fprintf(outfile, ...
'Mass Overhead: %6.0f kg\n\n\n', ...
ma);

```

```
fclose(outfile);
```

do_transfer_m0.m

```
function mp = do_transfer_m0( dv, m0, params )
% DO_TRANSFER_M0 Calculates the mass of propellant (mp) used by a
% spacecraft with an initial mass m0 to perform a burn imparting a specific
% amount of delta-v (dv). (See also do_transfer_mf.)
%
% Usage:    mp = do_transfer_m0( dv, m0 )
%
% Inputs:   dv      amount of delta-v (km/s)
%           m0      initial mass of spacecraft (kg)
% Outputs:  mp      mass of propellant required (kg)
%
% Parameters: isp    spacecraft engine isp (s)

g = get_constant('grav');
%load params_chem
isp = params.isp;

mp = m0*(1-exp(-dv/(isp*g)));

end
```

do_transfer_mf.m

```
function mp = do_transfer_mf( dv, mf )
% DO_TRANSFER_M0 Calculates the mass of propellant (mp) used by a
% spacecraft with a final mass mf to perform a burn imparting a specific
% amount of delta-v (dv). (See also do_transfer_m0.)
%
% Usage:    mp = do_transfer_mf( dv, mf )
%
% Inputs:   dv      amount of delta-v (km/s)
%           mf      final mass of spacecraft (kg)
% Outputs:  mp      mass of propellant required (kg)
%
% Parameters: isp    spacecraft engine isp (s)

g = get_constant('grav');
%load params
isp = params.isp;

mp = mf*(exp(dv/(isp*g))-1);

end
```

do_tugasassembly.m

```
function [ma] = do_tugasassembly(num_mod,strat,params)
```

```

% DO_TUGASSEMBLY replaces run_tugasassembly to clean up code. Fill in the
% rest of description later.

% ----- %
% PARAMETERS, SETTINGS ETC.
% ----- %

% user settings
ITER_LIM = 150;                                % iteration limit for mp_tug calculation
MP_TOL = 1;                                    % tolerance for mp calculation (mp_guess
- mp_calc < MP_TOL)

% housekeeping
mu = get_constant('mu_E');
outfile = fopen('tug_assembly_output.txt','a');

fprintf(outfile, ...
'TUG ASSEMBLY OUTPUT\n\n');
fprintf(outfile, ...
'Number of Modules: %6.0fn', ...
num_mod);
fprintf(outfile, ...
'Module Mass: %6.0f kg\nMass Penalty: %6.0f kg\nTug Payload Mass: %6.0f
kg\nAssembly Strategy: %6.0fn\n',...
params.m_mod,params.m_pen,params.m_pld,strat);

% ----- %
% DELTA-V CALCULATIONS
% ----- %

% dv for inclination change
if params.pkg_i ~= params.asy_i
theta = abs(params.pkg_i - params.asy_i); % deg
dv_i = do_incchange( params.pkg_r, theta );
else
dv_i = 0;
end

% dv for hohmann transfer
[dv_h,dt_h] = do_hohmann( params.pkg_r, params.asy_r );

% dv for rendezvous and capture [Saleh/Lamassoure]: could add 120 m/s -- or
% 60 m/s one way

% total delta-v
dv = dv_i + dv_h;

fprintf(outfile, ...
'Delta-V (one-way): %6.0f m/s\nDelta-V (total): %6.0f m/s\n\n', ...
dv*10^3,dv*num_mod*2*10^3);

% ----- %
% TUG PROPELLANT MASS CALCULATIONS
% ----- %

% tug strategy error checking

```

```

asy_strategy = strat;
if (asy_strategy < 0 || asy_strategy > 4)
    ma = 0;
    disp('ERROR: Cannot read tug strategy from params. ');
    return;
end

% TODO: if strategy is 1, then... else...
if (asy_strategy == 1)
    trip_types = [ num_mod ];
    mp_per_tug = 0;
elseif (asy_strategy == 2 || asy_strategy == 3)
    % multiple tugs
    n_tug = ceil(num_mod/params.m_per_t);
    finaltug_trips = mod(num_mod,params.m_per_t);
trips
    mp_per_tug = [0 0];
    if n_tug > 1
        trip_types = [ params.m_per_t finaltug_trips ];
    elseif (n_tug == 1 && finaltug_trips == 0)
        trip_types = [params.m_per_t finaltug_trips ];
    else
        trip_types = [ finaltug_trips 0 ];
    end
end

index = 1;
clear m_per_tug;

% for each type of trip...
for ntrips = trip_types

    % initialize
    mp_guess = params.mp_tug;
    mp_calc = 0;
    m_mod = params.m_mod;
    iter = 0;

    % iterate to calculate the initial tug propellant required
    while abs(mp_guess - mp_calc)>MP_TOL
        if iter>0
            mp_guess = mp_calc;
        end
        % initialize
        mp_tug = mp_guess;
        mp_toasy = 0;
        mp_topkg = 0;

        % update mass
        m_eng = params.m_pen;
        m_pld = params.m_pld;
        m_tank = params.f_prp*mp_tug;
        m_str = params.f_str*(mp_tug+m_tank+m_pld+m_eng);
        m_dry = m_pld+m_tank+m_str+m_eng;
        m_tug = m_dry;
    end
end

```

% number of tugs required
% last tug might make fewer


```

for i = 1:ntrips

    % prop for transfer to assembly orbit
    m0 = m_mod+mp_tug+m_tug;
    mp_toasy(i) = do_transfer_m0( dv, m0, params );
    mp_tug = mp_tug - mp_toasy(i);

    % prop for transfer to parking orbit
    m0 = m_tug+mp_tug;
    mp_topkg(i) = do_transfer_m0( dv, m0, params );
    mp_tug = mp_tug - mp_topkg(i);

end
mp_calc = sum(mp_toasy) + sum(mp_topkg);
iter = iter+1;
if iter > ITER_LIM
    disp('Iteration limit reached. Exiting...')
    ma = -1;
    return
end
end

% store output for each trip type
mp_per_tug(index) = mp_calc;
m_per_tug(index) = m_tug;
index = index+1;
end

% calculate mass overhead
if (asy_strategy == 1)
    ma = mp_per_tug(1) + m_per_tug;
elseif (asy_strategy == 2)
    if mp_per_tug(2) == 0
        ma = n_tug*(mp_per_tug(1)+max(m_per_tug));
    else
        ma = (n_tug-1)*(mp_per_tug(1)+max(m_per_tug)) + (mp_per_tug(2)+max(m_per_tug));
    end
elseif (asy_strategy == 3)
    m_tank = params.f_prp*mp_per_tug(1);
    if mp_per_tug(2) == 0
        ma = n_tug*(mp_per_tug(1)+m_tank) + max(m_per_tug)-m_tank;
    else
        ma = (n_tug-1)*(mp_per_tug(1)+m_tank) + (mp_per_tug(2)+m_tank) + max(m_per_tug)-
m_tank;
    end
end

% print results to text file
fprintf(outfile, ...
'Tug Prop Mass:   %6.0f kg\nTug Dry Mass:   %6.0f kg\n', ...
mp_per_tug(1),max(m_per_tug));
fprintf(outfile, ...
' Payload:      %6.0f kg\n Structure:      %6.0f kg\n Tank:          %6.0f kg\n Engine:
%6.0f kg\n', ...
m_pld,(max(m_per_tug)-(params.f_prp*mp_per_tug(1))-m_pld-
m_eng),(params.f_prp*mp_per_tug(1)),m_eng);

```

```
fprintf(outfile, ...
'Mass Overhead:  %6.0f kg\n\n\n', ...
ma);

fclose(outfile);
```

get_constant.m

```
function out = get_constant( requested_constant )
% GET_CONSTANT Retrieves the value of the constant specified in the input.
%
% Usage:  out = get_constant( requested_constant )
%
% Inputs: requested_constant  label (string!) e.g. 'mu'
% Outputs: constant          value of constant e.g. 3.986e5

if strcmp( requested_constant, 'mu_E' )
    out = 3.986012e5;          % km3/s2, BMW
end
if strcmp( requested_constant, 'grav' )
    out = .00981;            % km/s2
end
```

D

Assembly Trade Study Data

The following tables give in tabular form the data presented in Figures 3.5, 3.6, and 3.7 of Chapter 3. These are the main trade study plots, which show the variation in overhead mass as a function of the number of modules, module mass, and orbit altitude, respectively.

Table D.1: Overhead mass variation with number of modules.

Number of Modules	Overhead Mass mv [kg], M/T=1				Overhead Mass mv [kg], M/T=3			
	Self-Assembly	Single Tug	Multiple Tugs	In-Space Refueling	Self-Assembly	Single Tug	Multiple Tugs	In-Space Refueling
1	1060	1468	1468	1468	1060	1468	1468	1468
2	2121	2490	2936	2245	2121	2490	2490	2490
3	3181	3667	4405	3022	3181	3667	3667	3667
4	4242	5030	5873	3798	4242	5030	5626	4648
5	5302	6624	7341	4575	5302	6624	6420	5442
6	6362	8505	8809	5352	6362	8505	7333	6355
7	7423	10749	10278	6128	7423	10749	9293	7336
8	8483	13462	11746	6905	8483	13462	10086	8130
9	9543	16799	13214	7682	9543	16799	11000	9043
10	10604	20985	14682	8458	10604	20985	12960	10024
11	11664	26361	16151	9235	11664	26361	13753	10818
12	12725	33516	17619	10012	12725	33516	14666	11731
13	13785	43449	19087	10788	13785	43449	16626	12713
14	14845	58109	20555	11565	14845	58109	17420	13506
15	15906	81818	22024	12342	15906	81818	18333	14419

Table D.2: Overhead mass variation with module mass.

Module Mass [kg]	Overhead Mass mv [kg], M/T=1				Overhead Mass mv [kg], M/T=3			
	Self-Assembly	Single Tug	Multiple Tugs	In-Space Refueling	Self-Assembly	Single Tug	Multiple Tugs	In-Space Refueling
5000	2576	2884	4580	2102	2576	2884	3162	2433
10000	3939	4755	5961	3338	3939	4755	4791	3937
15000	5302	6624	7341	4575	5302	6624	6420	5442
20000	6665	8494	8722	5811	6665	8494	8049	6946
25000	8028	10363	10102	7048	8028	10363	9678	8450
30000	9391	12233	11483	8284	9391	12233	11306	9954

Table D.3: Overhead mass variation with orbit altitude.

Orbit Altitude [km]	Overhead Mass mv [kg], M/T=1				Overhead Mass mv [kg], M/T=3			
	Self-Assembly	Single Tug	Multiple Tugs	In-Space Refueling	Self-Assembly	Single Tug	Multiple Tugs	In-Space Refueling
2 Modules								
200	575	698	1272	689	575	698	698	698
300	1345	1555	2095	1459	1345	1555	1555	1555
400	2121	2490	2936	2245	2121	2490	2490	2490
500	2903	3513	3796	3049	2903	3513	3513	3513
600	3690	4636	4676	3871	3690	4636	4636	4636
700	4484	5874	5576	4712	4484	5874	5874	5874
800	5283	7248	6498	5574	5283	7248	7248	7248
900	6087	8778	7443	6458	6087	8778	8778	8778
1000	6898	10495	8412	7363	6898	10495	10495	10495
5 Modules								
200	1437.813	886.114	3180.065	848.2241	1437.813	886.114	1471.357	872.1968
300	3362.781	3325.819	5238.601	2691.784	3362.781	3325.819	3779	3004.825
400	5301.941	6624.202	7341.173	4574.952	5301.941	6624.202	6419.919	5441.551
500	7257.146	11270.43	9490.273	6499.838	7257.146	11270.43	9470.812	8251.776
600	9225.378	18212.78	11689.92	8469.498	9225.378	18212.78	13035.3	11529.29
700	11209.31	29549.8	13939.51	10484.57	11209.31	29549.8	17259.08	15405.24
800	13208.03	51132.18	16243.92	12548.63	13208.03	51132.18	22347.56	20064.97
900	15218.56	107322.1	18608.02	14665.64	15218.56	107322.1	28608.93	25785.55
1000	17244.88		21030.55	16835.59	17244.88		36525.19	33000.06