Novel Transcription Techniques for Multiple-Spacecraft, Multiple-Target Global Trajectory Optimization

by

Sean W. Napier

B.S., Lehigh University, 2014

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Aerospace Engineering Sciences

2021

Committee Members:

Jay W. McMahon, Chair

Jacob A. Englander

Daniel J. Scheeres

Hanspeter Schaub

Shalom D. Ruben

Napier, Sean W. (Ph.D., Aerospace Engineering Sciences)

Novel Transcription Techniques for Multiple-Spacecraft, Multiple-Target Global Trajectory Optimization

Thesis directed by Prof. Jay W. McMahon

With ever increasingly capable tools available to interplanetary mission designers, newer challenging classes of missions become accessible. Among these new classes of missions are Distributed Spacecraft Missions: designs where multiple spacecraft cooperate to achieve coordinated science objectives. Several applications being explored include, but are not limited to: coordinated launch, coordinated rendezvous, mega-constellation design, precision formation flying (PFF), very long baseline interferometry (VLBI), and distributed aperture space telescopes. These mission architectures promise to widen our gaze on the scientific phenomena in our solar system and beyond. However, current operational state-of-the art global trajectory optimization platforms lack the core capabilities to pose these complex new classes missions. These capabilities include: the ability to model multiple individual trajectory optimization problems as one single coupled trajectory optimization problem, the imposition of coordination constraints and cost functions, and the ability to traverse massive search spaces. In this dissertation, we present a fully automated multi-agent multi-objective technique for solving multi-spacecraft multi-target trajectory optimization problems using a hybrid optimal control approach. We apply this technique to two benchmark problems: an Ice Giants Dual Manifest mission design, and a variable fleet size Very Long Baseline Interferometry mission design. The techniques developed in this work create a general transcription for Multiple Traveling Salesmen Problems applied to global spacecraft trajectory optimization.

Dedication

To my father, Dr. Terrence J. Napier, who taught me to heed the alluring call of the stars.

Acknowledgements

The road to this moment has presented countless unforeseen challenges, including a struggle to find a source of independent funding. My journey at CU Boulder began with one advisor and will end with another. To Jay McMahon, thank you for taking on the monumental task of advising and mentoring me remotely while I have worked to find my way both academically and professionally at NASA Goddard. Your education, patience, and guidance has led me to the point where I can venture forth as a confident burgeoning researcher in astrodynamics. To Jacob Englander– your peerless professional mentorship and devotion to the growth of students like me has been life-changing and has imparted me with experience and knowledge of interplanetary trajectory optimization which I could scarcely find elsewhere. Thank you to all my coworkers and managers past and present at Goddard who proved ever understanding of the challenge of balancing full time work with doctoral research, encouraging me to win IRAD funds to align work with school when possible. I am also deeply grateful to my committee and in particular to Dr. Scheeres whose patience in the face of the obstacles I encountered over the years during completion of courses allowed me to find a stronger student within myself. Thank you to the numerous students at CU Boulder and in the ORCCA lab who supported me in ways large and small, whether it was Mark or Dan letting me crash at their apartments when I came out to visit Boulder to complete milestones such as comps, or Andrew French whose friendship is the reason I survived year one of the PhD program. None of you will be forgotten. And finally, to my loving partner Becca– your support during these past 5 years and daily challenging me to be the best version of myself have brought me to this pivotal moment. I will always treasure you as we continue our lives together.

Contents

Chapter

1 Introduction and Motivation **1**

1.1 Distributed Spacecraft Missions 1

1.1.1 Coordinated Launch: A Dual Manifest Mission to the Ice Giants 1

1.1.2 Distributed Aperture Space Telescopes 2

1.1.3 Very Long Baseline Interferometry 2

1.2 State of the Art of Global Trajectory Optimization 3

1.2.1 Direct versus Indirect Methods 3

1.2.2 Gradient Free Methods 4

1.2.3 Hybrid Optimal Control 6

1.2.4 Multi-Agent Optimization 7

1.3 Approach 8

1.3.1 Thesis Statement 11

1.4 Dissertation Overview 11

1.4.1 Organization 11

1.4.2 Contributions 11

2 Algorithms **13**

2.1 Background 13

2.2 Spacecraft Trajectory Optimization 13

2.3 Inner-Loop Solver . . . 15
2.4 Discrete Parameter Optimization . . . 19
2.5 Multi-Objective Outer-Loop Solver . . . 21
2.6 Multiple Spacecraft, Multiple Objective Global Trajectory Optimization . . . 24

3 Ice Giants Multi-Mission Design **27**
3.1 Background . . . 27
3.2 Problem Formulation . . . 27
3.2.1 MVM Outer-Loop Transcription . . . 29
3.3 Outer-Loop Coordination Objectives (Minimax Approach) . . . 32
3.4 Outer-Loop Coordination Constraints . . . 32
3.4.1 Shared Launch Vehicle Constraint . . . 33
3.4.2 Minimum Number of Identical Flyby Genes Constraint . . . 33
3.4.3 Minimum Number of Shared Trajectory Phases Constraint . . . 35
3.5 Results . . . 35
3.6 Summary . . . 50

4 Very Long Baseline Interferometry Mission Design **52**
4.1 Background . . . 52
4.2 Problem Formulation . . . 53
4.3 Outer-Loop Transcription . . . 57
4.4 Results . . . 68
4.5 Summary . . . 82

5 Conclusions and Future Work **84**
5.1 Conclusions . . . 84
5.2 Future Work . . . 88

Bibliography **89**

Appendix

A Seed Sharing Technique **95**

A.1 Background 95

A.2 Motivation 97

A.3 Definition of partially alike chromosomes 100

A.4 Description of partial seeding technique 101

A.5 Example of a Simple Partial Seeding Problem 105

A.6 Example of a Complicated Problem 105

A.7 Experimental Design 109

A.8 Objectives and Anticipated Findings of the Experiment 110

Tables

Table

3.1 Table of options parameters for the MOMA HOCP setup for both the low and high C_3 regime studies. 39

3.2 Mission itinerary for the minimum ΔV mission of the low C_3 SHFB constraint study (solution in Fig. 3.3). 39

3.3 Mission itinerary for High C_3 SHLV constraint study (solution captured in Fig. 3.5 (a)). 41

3.4 Mission itinerary for High C_3 SHFB constraint study (solution shown in Fig. 3.5 (b)). 41

3.5 Minimum ΔV solution itinerary for the mission in Fig. 3.6. This mission has the same outer-loop decision vector as the mission in Fig. 3.5 (b) (itinerary in table 3.4), but with a Jupiter flyby for the Uranus spacecraft. The total ΔV is significantly higher. The inner-loop runtime was 30 minutes; longer runtimes produced no improvement. 42

3.6 Mission itineraries for two separate single-vehicle missions (solutions in Fig. 3.7), one to Uranus and the other to Neptune, with minimized ΔV for each. Each case ran for 30 minutes. 43

3.7 Mission itineraries for two separate single-vehicle missions (solutions in Fig. 3.8), one to Uranus and the other to Neptune, with minimized ΔV for each. Each case ran for 30 minutes. 43

3.8 Mission itinerary for the "sequentially optimized" version of the optimal SHFB MVM in Fig. 3.5 (b). The separate trajectories are depicted in Fig. 3.9. Comparing against the nominal MVM in Fig. 3.5 (b), the sequential optimization method produces a far more costly mission (2 km/s more) than the coupled MVM method developed in this work. The sequential result is significantly more suboptimal, as hypothesized. Regardless of the particular inner-loop transcription, given **some** inner-loop optimizer, these results are clearly indicative of the need for coupled optimization. . 43

4.1 Average archive lookup durations as a function of entries in the archive. 65

4.2 Options parameters for the inner-loop of the MOMA HOCP scheme. 68

4.3 Options parameters for the outer-loop of the MOMA HOCP scheme. 69

4.4 Total targets observed for zero ΔV as a function of fleet size based on data from generation 40 (Fig. 4.5). Δ_{obs} is the number of observations gained by increasing the fleet size by one. 74

4.5 Total targets observed for zero ΔV as a function of fleet size based on data from generation 100 (Fig. 4.6). 74

4.6 Itinerary for Fig. 4.10. 6 spacecraft observe 14 unique targets over 12.9 years using zero ΔV with 2 simultaneous observations at epoch 8. 74

4.7 Mission Itinerary for Fig. 4.11. 7 spacecraft observe 16 unique targets over 17.5 years using zero ΔV. 75

4.8 Mission Itinerary for the mission in Fig. 4.12. 75

A.1 P_{trio} as a function of N_S. 98

Figures

Figure

2.1 Illustration of the MBH process for an arbitrary cost function f_1 of one decision variable x_1, taken from [35]. 17

2.2 Concept illustration of the Pareto front for a mission with two objectives, f_1 and f_2 [61]. 24

2.3 Flow Chart of algorithm execution. Each inner-loop problem represents a unique candidate mission [61]. 26

3.1 Illustration of the MGA1DSM transcription. This transcription is well suited for use with evolutionary algorithms and is readily applied to trajectory optimization problems with small bodies, planets, or both. 29

3.2 Low C_3 study results: (a) and (b) show, respectively, the Pareto fronts of ΔV versus TOF for the minimum shared flyby genes constraint (SHFB) study, and the minimum number of shared trajectory phases constraint (SHTR) study. In (a), the size of each marker codifies the number of flyby genes shared, while in (b) it codifies the number of shared trajectory phases. The smaller size represents zero shared flyby genes/trajectory phases, and the larger size represents two. 44

3.3 Minimum ΔV solution for the low C_3 SHFB study. Red stars indicate locations of deep space maneuvers. 45

3.4 High C_3 studies: All fronts shown after 24 hours of outer-loop runtime. Pareto fronts are depicted for the shared launch asymptote constraint study (SHLV) (a), SHTR study (b), and SHFB study (c). Note the far more sparse front of the SHTR study. . 46

3.5 (a) The high C_3 SHLV study's minimum ΔV solution and (b), the high C_3 SHFB study's minimum ΔV solution. 47

3.6 This is the high C_3 SHFB case from Fig. 3.5 (b), but rerun with the same launch epoch as Fig. 3.5 (b) to investigate the possibility of a synergistic Jupiter flyby. The mission does not appear to benefit from a Jupiter flyby at the given launch epoch. . 47

3.7 The high C_3 SHFB case from Fig. 3.6 , rerun with the same launch window and C_3 bounds, but as two separate single-vehicle problems. A dual-manifest mission would result the launch asymptote to be the one that least detriments each individual trajectory. With the launch C_3 and asymptotes not constrained to be identical, the trajectories are quite different as is the sum of their ΔV, roughly 7 km/s lower than the coupled MVM, as shown in Table 3.6. 48

3.8 The high C_3 SHFB case from Fig. 3.5 (b), rerun with the same launch window and C_3 bounds as the Fig. 3.5 (b) mission, but as two separate single-vehicle problems. The total ΔV for these missions is 8.110 km/s, cheaper than the 9.438 km/s ΔV of the multi-mission in Fig. 3.5 b. 48

3.9 A co-launched MVM based off the solution Fig. 3.5 (b), but with each spacecraft's trajectory optimized sequentially rather than in parallel as a coupled problem. The resulting trajectories are significantly more costly in terms of ΔV, as discussed in Table 3.8. 49

4.1 Example of the VLBI trajectory optimization problem with a fleet of three spacecraft making two consecutive observations. 56

4.2 Detailed flow chart of inner-loop and outer-loop algorithm interaction. 62

4.3 Left: Number of duplicate chromosomes created at each generation. Right: total duplicate chromosomes discovered as a function of generations. 66

4.4 Left: Cross-section of non-dominated front at generation 10 with 29 solution fleets showing TOF versus observed targets trade. Right: the same front but showing the fleet size versus observed targets cross-section. 76

4.5 Above: cross-sections of the Pareto front after 40 generations. In the highlighted data points, x = TOF in years, y = number of spacecraft, and z = number of targets. 76

4.6 Above: cross-sections of the Pareto front after 100 generations. 77

4.7 A fleet of 3 spacecraft making 1 observation of radio source 124. 77

4.8 This figure shows the frequency distribution of different targets in the Pareto front at generation 10 (left) and 40 (right). 78

4.9 This figure shows the distribution of radio sources observed by the fleets in Fig. 4.10, 4.11, and 4.12. 78

4.10 6-spacecraft 14-target mission from generation 40's Pareto front, leveraging observation multiplicity, with itinerary outlined in Table 4.6. 79

4.11 The most complex member of generation 40's Pareto front: 7 spacecraft observing 16 targets, with itinerary in Table 4.7. 80

4.12 The most complex member of generation 100's Pareto front: 6 spacecraft observing 16 targets, with itinerary outlined in Table 4.8. 81

A.1 Detailed flow chart of inner-loop and outer-loop algorithm interaction with added partial seeding step. 104

A.2 Illustration of the partial seed ranking algorithm. 104

A.3 The inner-loop seed sharing decision tree algorithm. 108

A.4 The seeding of the candidate mission in Eq. A.15 using the archived mission in Eq. A.18. 109

Chapter 1

Introduction and Motivation

1.1 Distributed Spacecraft Missions

With ever increasingly capable tools available to interplanetary mission designers, newer challenging classes of missions become accessible. Among these new classes of missions are Distributed Spacecraft Missions: designs where multiple spacecraft cooperate to achieve coordinated science objectives. Several applications being explored include, but are not limited to: coordinated launch, coordinated rendezvous, mega-constellation design, very long baseline interferometry (VLBI), and distributed aperture space telescopes. These mission architectures promise to widen our gaze on the scientific phenomena in our solar system and beyond. Several applications are discussed in greater context as follows.

1.1.1 Coordinated Launch: A Dual Manifest Mission to the Ice Giants

At the Flagship class level, the Ice Giants, Uranus and Neptune, are of paramount interest in the Planetary Science 2011 Decadal Survey [4]. These targets cannot both be visited by a single spacecraft any time within the next 50 years [46]. Thus, a dual manifest mission becomes a nigh-necessary option for performing science on both of these targets, where two spacecraft co-launch to deep space. This would be more expensive than a mission to just one target, but potentially cheaper than two separate missions [10, 44]. A pre-decadal survey study led by members of NASA/CalTech's Jet Propulsion Laboratory explored a grid of designs for separate ice giant missions, and additionally put forth two point designs for a dual-manifest mission [44].

1.1.2 Distributed Aperture Space Telescopes

Distributed Aperture Space Telescopes make use of a capability known as aperture synthesis, whereby smaller telescopes arrayed far apart use interferometry to mix their received signals to achieve an angular resolution of a single, massive telescope [51]. Perhaps the most profound ground system demonstration of this is the Atacama Large Millimeter Array (ALMA), comprised of 66 mobile radio dishes in northern Chile [1]. Its data has already been used in over 1,000 peer-reviewed articles, for a wide array of radio astronomy research including exoplanet discoveries [1]. The potential to deploy a similar technology in space presents a significant opportunity beyond the scope of this already powerful ground system.

With space telescopes growing ever larger, the eventual result is a telescope too large for one or even several launch vehicles. Thus if multiple smaller units could be deployed in a large formation and slew properly to targets, the result would be the largest distributed array ever conceived. In space, the dominant challenge for the mission design and operations would be precise pointing at the target of interest. The formation would have to be capable of reorienting itself to allocate some spacecraft to point at one target, while some would point at another. Or, the entire formation would potentially reorient to image a new target. Optimizing the trajectories for this formation rapidly becomes an intractable task the more spacecraft there are in the fleet.

1.1.3 Very Long Baseline Interferometry

The Laser Interferometer Gravitational-Wave Observatory (LIGO) is the world's first gravitational wave observatory, consisting of two installations widely separate across the United States [70]. LIGO has made more than six confirmed detections of "black hole merger" events using the ingenuity of machine learning to filter faint signals from saturating noise- signals generated from 1E-10 meter changes in the length of its vacuum tubes, measured precisely by lasers[70, 21]. LISA will perform LIGO's tasks, but in space - leveraging the empty space to create a much larger interferometer antenna [62]. While LIGO's arms are 4 km long, LISA's will be approximately 2.5

million km. The mission will nominally launch in 2034 and use this extremely long interferometry array to detect faint gravitational waves from as early as 1 second after the Big Bang [62]. The mission is a composed of three identical spacecraft which launch and deploy into an equilateral triangular formation which trails roughly 50 million km behind the Earth as it orbits the sun.

The author is directly experienced with preliminary trajectory optimization for this mission. It requires the consideration of extremely tight path constraints. In general, the dominating problem is precise pointing. But whereas distributed aperture telescopes must point precisely at a distant target with simultaneity, LISA's interferometers must point precisely towards each other. Their relative separation distance is tightly constrained, as are the interior angles made by their triangular formation, along with their ecliptic inclination, and distance from the Earth. No global optimization of this mission design has been performed, but this task is likely the focus of continuing design work.

1.2 State of the Art of Global Trajectory Optimization

1.2.1 Direct versus Indirect Methods

Generally speaking in trajectory optimization, there are two classes of optimization methods: direct and indirect. Indirect methods require an analytical formulation of necessary and sufficient conditions for optimality derived in terms of system states and Lagrange multipliers [15]. These conditions are then discretized to form a constrained parameter optimization problem. Direct methods, by contrast, parametrize the problem using physically meaningful decision variables such as ΔV, time of flight, etc. Interplanetary trajectory optimization problems are too large in terms of number of decision variables and constraints to be solved analytically, and therefore require the use of numerical methods. A Nonlinear Programming (NLP) solver is an effective means of performing local gradient-based optimization for both direct and indirect problems. Numerous black box solvers exist for this purpose, including tools such as SNOPT, IPOPT [39, 78] and MATLAB's `fmincon`. However, NLP solvers require a "good" initial guess to converge, and small changes in the initial guess of an indirect method's Lagrange multipliers may result in a large change in the trajectory,

affecting the ability of the optimizer to converge. Additionally, should one decide to change the number of parameters in the optimization problem, the necessary and sufficient conditions must be re-derived in terms of the new system. Finally, direct methods can also eliminate the need to track the sensitive switching function, which presents a major challenge for the indirect methods. Thus indirect methods are highly sensitive approaches for these problems and we defer to direct methods for their versatility [52, 31].

Numerous classes of direct method transcriptions for effectively posing a trajectory optimization problem as a parameter optimization problem have been studied. These include, but are not limited to: simultaneous methods, shooting methods, and the approximation of a trajectory as a series of coast arcs with discrete impulsive maneuvers along each phase [31]. An excellent earlier study surveying and classifying numerical techniques to which such transcriptions might be applied may be found here [13]. Shooting methods solve a boundary value problem by reducing it to an initial value problem, propagating an initial guess over a time arc, and using an NLP solver to adjust the initial guess based on errors in the boundary conditions [52]. The sensitivity of shooting methods, and indeed other direct methods, may be reduced via analytical derivatives of the cost function and boundary conditions with respect to state parameters and/or the use of parallel or multiple shooting techniques [52]. Examples of "multiple coast arcs with N impulsive maneuvers" transcriptions may be found here [48]. Simultaneous methods include, notably, direct collocation methods [52]. Collocation methods approximate a trajectory as some high order polynomial and choose 'collocation points', separated equally in time, at which the system dynamics must be satisfied [43]. Collocation methods have proven effective for the optimization of low thrust trajectories [31, 43].

1.2.2 Gradient Free Methods

In some cases, what are called "evolutionary" or "gradient-free" methods use populations of trial decision vectors to search for an optimal solution. A subclass of these, known as evolutionary algorithms, apply principles derived from natural selection, and can be effective in place of the com-

bination of a direct transcription and NLP solver. These methods generally use a mechanism known as **stochastic gradient descent** to search for a minimum to a cost function. An initially random population of decision vectors undergoes ranking, genetic crossover, and mutation to produce the most "fit" offspring. Several examples of gradient-free methods include Particle Swarm Optimization (PSO), Differential Evolution (DE), binary genetic algorithms, and ant colony optimization (ACO). Each of these methods use metaheuristics (generalized local search metrics) in place of actual gradient information to direct the search for an optimal solution. In PSO, the metaheuristics to drive the search are each particle's inertial, cognitive, and social weights [53]. For ACO, it is a "pheromone" quantity exchanged by the "ants"; this algorithm has enjoyed widespread use for applications ranging from protein folding to rule classification to financial predictions [60, 45, 80]. For DE, the metaheuristic is a difference vector calculated by randomly differencing two or four decision vectors, which then perturbs the current global best solution [67].

The two major variations between methods are their metaheuristics, and the ranking mechanism of one genome against another. These methods lend themselves to problems where an NLP solver encounters great difficulty due to a lack of gradient information in the problem formulation. For example, the authors previously applied DE to the Cassini interplanetary cruise problem, using the MGA1DSM transcription to parametrize candidate trajectories. MGA1DSM, while a facile transcription for rapid prototyping, does not have analytical derivatives, and it was found that DE performed well in place of a gradient-free local optimizer. In general however, when optimizing continuous decision variables, and optimizing spacecraft trajectories specifically, a direct transcription coupled with an NLP solver is less sensitive to the choice of initial guess and better suited to implementing constraints than using evolutionary methods.

The true power of evolutionary methods is twofold. 1) They excel at traversing **integer** decision spaces - where there can be no gradient information - over alternative brute-force grid search methods. 2) While not always desirable for precisely locating a **local** optimum, they can reliably determine the rough location of the global optimum. For these reasons, much research has been done into framing global trajectory optimization problems as Hybrid Optimal Control

Problems (HOCP), with an evolutionary integer algorithm "outer-loop" whose decision vectors are integer genomes encoding continuous trajectory optimization problems for to be solved by an "inner-loop".

1.2.3 Hybrid Optimal Control

Glocker et al. 2001 and Ross et al. 2005 provided solution methods to the HOCP formulation of spacecraft trajectories, where an "outer-loop" handles the discrete (integer) variables, and an "inner-loop" handles the continuous (floating point) variables [65, 77]. Conway et al. 2007 introduced a trade of two hybrid HOCP formulations - a genetic algorithm outer loop wrapped around robust NLP solver inner-loop, followed by a "Branch and Bound" outer-loop wrapped around a genetic algorithm inner-loop [28]. Evolutionary methods have been shown effective on numerous multiple-flyby chemical thrust design problems, as in later work by Gad and Abdelkhalik et al.[6, 7]. Vavrina et al. introduced a HOCP multi-objective framework for low thrust trajectory optimization, which wrapped an evolutionary algorithm around Purdue's GALLOP capability [75, 59]. Separately, Vasile et al. 2011 introduced a multi-objective spacecraft trajectory optimization framework applied to chemical problems such as the Cassini mission [72]. Englander, Vavrina, Ghosh and others have since expanded upon these earlier analyses to include multiple-objectives and low thrust mission design support [42, 29, 35, 30, 28, 23].

The conventional trajectory optimization problem that is solved by what we call the inner-loop has been explored extensively. Englander et al. experimented with various stochastic global search tuning parameters to increase the global problem scope and efficiency of the inner-loop [34]. As of now, a handful of interplanetary trajectory optimization platforms exist, including the Parallel Global Multiobjective Optimizer (PaGMO) and the Evolutionary Mission Trajectory Generator (EMTG) [30, 3]. However, neither of these tools is capable of *multi-spacecraft* interplanetary global trajectory optimization. With many new potential classes of missions motivating the need for such an optimization capability, this proposed work seeks to develop a technique to enable them.

1.2.4 Multi-Agent Optimization

To optimize a coordinated multi-vehicle problem leveraging current cutting edge tools, separate designs would have to be optimized independently and manually/iteratively gridded over to find independent trajectory designs that share the same constraints. For example, consider a mission in which two spacecraft depart from the same launch vehicle along an identical asymptote, then perform independent maneuvers at different points in time to begin journeys to distinct destinations. Using state-of-the-art tools, one would need to create a grid whose points would each be defined by an epoch and a launch asymptote. For each grid point, two separate SVMs would need to be run, both required to share the same launch epoch and asymptote parameters. However, the best design found in this grid can in no way be guaranteed to represent the optimal solution. At best, refining this grid search approach for efficiency would lead to a method that approaches, but remains inferior to, the type of stochastic gradient descent already used by evolutionary methods. The co-launched aspect of this mission, in fact, is a coordination constraint that couples the behavior of the two spacecraft. This coupling is more properly handled with a treatment of two separate SVM decision vectors as one MVM devision vector, with a local optimizer free to rapidly vary and traverse candidate launch asymptotes within a launch window. Furthermore, if one were to add N more spacecraft to the fleet, the problem becomes geometrically intractable to solve as separate subproblems.

Multi-agent optimization is a thriving field in computer science which is applied, in large part, to solve communications network traffic problems. In these problems, the server/client nodes are treated as agents and the communication between any two may be modeled as a time-varying digraph sequence [69]. For more information on computational graph optimization, we refer the reader to the following source which discusses the motorized Traveling Salesman Problem (TSP) [77]. In multi-agent optimization, a computational agent can be either a tool used in the optimization process, the object of the optimization process, or both [69]. For example, PSO is a form of multi-agent optimization where particles are the agents that perform the optimization. Several

multi-agent multi-objective optimization algorithms have been developed and applied to problems ranging network communications to robotics - such as with mechanical rovers coordinating site visits.

Beni et al. 1989 explored decentralized swarm intelligence architectures for robots in terms of classifying predictable versus unpredictable behavior [12]. Sun et al. developed the first non-convex multi-agent optimization algorithm for communications networks [69]. In recent years, multi-objective multi-agent genetic algorithms (MOMA GA) have been proposed, including Aggregation by Variable Objective Weighting, Niched Pareto Genetic Algorithm, Strength Pareto Algorithm (versions I and II), and more [41, 50, 82, 14]. Most of the focus in these algorithms however remains on single objective multi-agent optimization [9]. Additionally, none have been applied to spacecraft trajectory optimization. Yliniemi et al. 2015 used rover site exploration to motivate the development of a continuous-valued Multi-Agent Non-Dominated Sorting Genetic Algorithm (NSGA-II) along with a brand new MOMA GA called The Pareto Concavity Elimination Transformation (PaCCET) [79]. Ghosh and Coverstone et al. explored a multi-agent formulation of the decision vector for low thrust trajectories for propellant-optimal cooperative cubesat maneuvers [37]. They leveraged A HOCP framework as well, but used the outer-loop to optimize the number of discretization points in the low thrust mesh and did not extend this approach to multi-objective, multi-target global optimization problems. Ergo, a host of multi-agent methods and applications exist with different approaches to handling and incentivizing the coordination of the agents involved. This work draws on multi-agent principles to implement a prototype interplanetary global trajectory optimization technique for MVMs.

1.3 Approach

In examining the state of the art, the global trajectory optimization of many different distributed spacecraft mission architectures proves intractable using current trajectory optimization capabilities. When tasked with the global optimization of interplanetary Multi-Vehicle Mission (MVM) trajectories specifically, state-of-the-art techniques are hindered by their need to treat the

MVM as multiple decoupled trajectory optimization subproblems. This shortfall blunts their ability to utilize inter-spacecraft coordination constraints and may lead to suboptimal solutions to the coupled MVM problem. Only a handful of platforms capable of fully automated multi-objective interplanetary global trajectory optimization exist for **single**-vehicle missions (SVM), but none can perform this task for interplanetary MVM.

Framing interplanetary spacecraft trajectory optimization as a hybrid optimal control problem (HOCP) has proven an effective approach [29, 75]. In a HOCP framework, trajectory optimization is a Mixed-Integer Programming (MIP) problem. Some decision variables are discrete (integers) while others are continuous (floating point), necessitating distinct optimization routines for each type. Furthermore, the resulting mission designs are points within a solution space spanned by multiple objectives (i.e., minimum fuel versus minimum time of flight). Thus, in order to effectively characterize the solution space for a given mission design problem, a *multi-objective* HOCP framework is essential. However, while tools exist to solve interplanetary multi-objective HOCPs for a *single* spacecraft, no tool exists to do so for *multi-spacecraft* problems. Addressing the shortcomings of current approaches to MVM optimization, including the methods for handling inter-spacecraft coordination objectives and constraints, are key to enabling the optimization of future interplanetary MVMs.

Operational implementations of state-of-the-art global trajectory optimization techniques are not structured to support multiple-spacecraft global trajectory optimization. In order to solve a problem with a fleet containing a *fixed* number of spacecraft, these implementations would require the analyst to run a large grid of studies, with each grid point representing a particular permutation of constraint values that are applied identically to each of S independent single-vehicle spacecraft trajectory optimizations. Each of the S SVM runs would need to be compared *a posteriori* for adherence to constraints. While it would be possible to set certain boundary constraints, this inherent separation of the MVM already makes it untenable to impose most inter-spacecraft coordination constraints. Now if the analyst is solving a problem where the requisite fleet size to achieve the coordinated science objectives is not known *a priori*, the analyst much choose a range of fleet

sizes and run still more studies, one set of studies for each fleet size. These shortcomings –lack of coordination constraint handling, lack of coordinated objective handling, and artificially inflated computation time due to separate SVM optimization– motivate this thesis work. A transcription which enables global trajectory optimization for multiple-spacecraft multiple-target exploration problems is essential in tackling the next generation of complex space missions.

In this dissertation, we introduce such a technique. We present a Multi-Agent Multi-Objective Hybrid Optimal Control Problem (MOMA HOCP) transcription which resolves many of the shortcomings with state-of-the-art techniques and demonstrate its efficacy on several benchmark global trajectory optimization problems. The technique is fully automated (requiring no analyst-in-the-loop tinkering to explore a given search space), and generic (generally agnostic to a particular trajectory optimization problem). It supports an arbitrary number of spacecraft in a fleet and allows both the fleet size and number of science targets explored by the fleet to vary during the optimization. Several new inter-spacecraft coordination constraints and coordinated science objectives are introduced, and their impact on the solution space of two different benchmark problems is assessed. Finally, a HashMap archive utility is created to prevent repeat evaluations of previous solutions and amplify the reach of the MOMA HOCP technique to explore the solution space using partial seeding. Two benchmark problems are conceived to demonstrate the efficacy of the MOMA HOCP technique: an Ice Giants Multi-Mission (where two spacecraft share a launch asymptote en route to different outer planets), and a science-phase VLBI problem where a variable number of spacecraft coordinate to image multiple distant radio sources.

We then propose, conceive, and apply techniques for more efficient search space traversal as the dimensionality of optimization problems such as these is enormous. We introduce a fast HashMap archive technique to ensure each point in a problem's search space is explored only once. We then introduce an algorithm augmenting this archive into a seed sharing capability in order to potentially expedite the discovery of new solutions and propose an experiment to test its efficacy.

1.3.1 Thesis Statement

A fully automated, multi-objective, multi-agent hybrid optimal control problem framework allows analysts to combine individual spacecraft trajectory optimization problems into a coupled optimization problem, impose inter-spacecraft coordination constraints, and pose coordinated science objectives, enabling the global optimization of new classes of multi-spacecraft, multi-target exploration missions.

1.4 Dissertation Overview

1.4.1 Organization

Chapter 2 details each algorithm applied and/or created for MOMA HOCP transcription. Chapter 3 discusses the problem formulation and results for the Ice Giants Multi-Mission benchmark problem. Chapter 4 discusses the formulation and results for the very long baseline interferometry problem. Finally, chapter 5 summarizes the significant contributions and conclusions of this thesis and discusses our future work. The Appendix discusses the augmentation of the archive utility created in 4 into a novel partial seeding technique to expedite the MOMA HOCP transcription's ability to traverse its search space, to be applied to further research on the VLBI problem.

1.4.2 Contributions

The chief contributions of this dissertation are summarized as follows.

Ice Giants Dual Manifest Study

- introduces a first-of-its-kind MOMA HOCP framework for interplanetary global trajectory optimization
- creates an outer-loop transcription which encodes a coordinated launch asymptote problem
- creates and demonstrates the impact of several inter-spacecraft coordination constraints, including: a coordinated launch asymptote constraint, a shared flyby genes constraint, and

a shared trajectory phases constraint

Very Long Baseline Interferometry Study

- expands the generality of the outer-loop of the MOMA HOCP framework into a transcription which can pose almost any Multiple Traveling Salesmen Problem
- applies a Null Gene technique to allow the number of spacecraft in a fleet to vary
- enables multiple observations at a single epoch when the size of a fleet exceeds 6 spacecraft and incentivizes this behavior with an observation multiplicity cost function
- introduces a fast HashMap method for ensuring no repeated chromosomes are sent to the inner-loop

Chapter 2

Algorithms

2.1 Background

In § 1.2, we discussed the state of the art of global trajectory optimization, identifying key techniques which numerous operational platforms leverage in order to produce compelling trade studies. No technique has been developed which combines: multi-agent techniques, multi-objective optimization, and hybrid optimal control into one framework and applies their combination to trajectory optimization. In this dissertation, we introduce a transcription which does. This transcription uses a multi-agent multi-objective integer genetic algorithm outer-loop (in the form of a multi-agent NSGA-II algorithm) wrapped around a single-objective inner-loop global search method (Monotonic Basin Hopping, MBH). MBH stochastically explores the continuous decision parameter space searching for the global optimum, generating candidate trajectories which are parametrized into decision vectors using a direct transcription. For each benchmark problem, we apply a different direct shooting transcription uniquely suited to that problem. These transcriptions are discussed in detail in the problem formulations of Chapters 3 and 4. The NLP solver, MATLAB's `fmincon`, is used to solve for local optima given the decision vectors created by MBH.

2.2 Spacecraft Trajectory Optimization

Each of the components selected to build the MOMA HOCP technique in this work is complex and nuanced. It is instructive to first consider a basic fundamental trajectory optimization problem and develop a discussion through increasing levels of complexity, introducing each component at

its appropriate level. Let us begin with the discussion of a two-point boundary value problem: the base unit of a trajectory modeled in this dissertation.

In this example, we assume a spacecraft obeys two-body Newtonian gravity. The two bodies include a large primary point mass, and a negligible secondary point mass (the spacecraft). The equations of motion for the spacecraft are given by equation Eq. 2.1.

$$\ddot{\vec{r}}(t) = -\mu \frac{\vec{r}(t)}{|\vec{r}(t)|^3} \tag{2.1}$$

where $\vec{r}(t)$ is the time-dependent cartesian position vector of the spacecraft with respect to the origin of some inertial frame of reference. We can then represent the spacecraft's position-velocity state (Eq. 2.2) and its time derivative (Eq. 2.3).

$$\vec{X}(t)^T = [\vec{r}(t)^T,\ \dot{\vec{r}}(t)^T] \tag{2.2}$$

$$\dot{\vec{X}}(t)^T = [\dot{\vec{r}}(t)^T,\ -\mu \frac{\vec{r}(t)^T}{|\vec{r}(t)|^3}] \tag{2.3}$$

The spacecraft's real physical trajectory is represented by a continuous time series of states. However, in optimization, it is more convenient discretize this trajectory using a transcription which can describe it completely using a limited set of decision variables. A decision variable x_i is a parameter which quantifies an element of a decision-making process, which a decision maker can control. In optimization, the decision maker is the optimization algorithm. With direct trajectory optimization, decision variables may include real parameters of the trajectory, for example: ΔV vectors, position vectors, time of flight (TOF) durations, and spacecraft mass. The decision variables chosen in a particular transcription combine to form the decision vector $\vec{x}$ which encodes a spacecraft trajectory.

The goal of direct trajectory optimization is to find a trajectory whose decision vector minimizes some cost function (Eq. 2.4).

$$J = g(\vec{x}) \tag{2.4}$$

Now suppose we introduce a general constraint on the decision vector using Eq. 2.5.

$$\phi = f(\vec{x}) \tag{2.5}$$

The general trajectory optimization problem in our example is to find a solution decision vector $\vec{x}$ which encodes a trajectory that follows Eq. 2.1 which minimizes the cost $J(\vec{x})$ subject to $\phi(\vec{x})$. The constraint $\phi(\vec{x})$ may be a vector of constraints $\vec{\phi}(\vec{x})$, some linear, others nonlinear, and some be equality or inequality constraints. If any of the constraints or the cost function is a nonlinear function of the decision vector, the trajectory optimization is a nonlinear programming problem. The trajectories in this dissertation are complex nonlinear programming problems- not merely single-arc direct shooting problems, but coupled multi-point shooting problems with control, to be solved by a nonlinear programming (NLP) solver. These problems also contain nonconvex, multimodal, cost and constraint functions, and the existence of a solution can not be known *a priori*. As such, it is difficult to come up with a good initial guess for their solutions, and given the nonintuitive nature of their search spaces, an initial guess generator must be robust absent any analyst intervention.

2.3 Inner-Loop Solver

Many techniques for locating initial guesses from within a search space have been explored. These include, but are not limited to, graphical representations such as Tisserand plots, search space pruning, and seed sharing [68, 48, 22, 66, 46, 54, 57, 36]. There are tradeoffs in each of these methods. Stochastic search methods are at least as efficient as brute force, finding the global optimum in less time and require no analyst intervention, but are inefficient as they require a long runtime to reliably search the space. Graphical representations of the search space can impart intuition about the complex search space to a mission analyst, allowing for quick identification of

candidate solutions once created. However, they: require analyst to intervene in the optimization process, and may not represent an accurate picture of viable initial guesses which satisfy problem dynamics. Search space pruning techniques share similar tradeoffs to graphical methods as they require an analyst to make decisions about which pockets of the search space are worth searching - based on intuition and assumptions which may lead to no solutions being found or cause the analyst to miss the best solutions. However, pruning allows the analyst to completely control the size of the search space, and to some degree all global optimization problems are in practice subjected to pre-pruning of their search space based on mission and time requirements.

In fully automated global trajectory optimization, it is essential to select both an initial guess generator which will facilitate exploration of the global search space, a robust nonlinear programming (NLP) solver to optimize initial guesses, and a trajectory transcription which will enable the NLP solver to find good solutions quickly. In this dissertation, we refer the task of conventional trajectory optimization to an inner-loop of a hybrid optimal control framework. The inner-loop solves a single-objective trajectory optimization problem with no analyst intervention. The inner-loop's routine must reliably find the global optimum, and require no initial guess. It is necessary to wrap a global search method around the local optimizer to enable the inner-loop to routinely arrive with the statistical neighborhood of the global optimum. The stochastic global search method employed in this work is a Monotonic Basin Hopping (MBH) algorithm [17, 35]. This algorithm derives its name from the supposition that the generally nonconvex cost function being optimized contains local optima within **basins**, which can be traversed by a local optimizer.

Once the NLP solver drives a sub-optimal initial guess down to a local minimum (bottom of a basin), MBH perturbs the current decision vector by some random hop distance out of the basin and begins exploring the global space, keeping track of the global best found. This process is repeated until either a maximum number of hops have been performed or an elapsed time has passed (both of which are user-selectable parameters). This process is illustrated in Fig. 2.1. MBH thus hops through the global search space stochastically. Its performance has been greatly improved by using a Nonlinear Programming (NLP) solver to quickly optimize local minima [35].

The following pseudocode captured in **Algorithm 1** describes the version of MBH implemented for this work. Versions of MBH such as that of [34] use Pareto random hops to explore the global space, an approach we adopt in this dissertation as well.

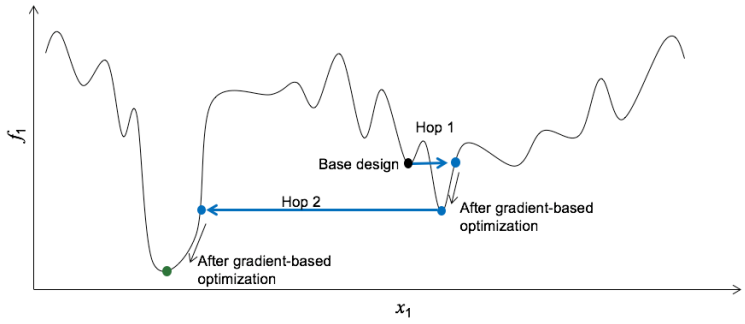


Figure 2.1: Illustration of the MBH process for an arbitrary cost function f_1 of one decision variable x_1, taken from [35].

The fundamental technique of MBH may be tuned in numerous ways. These include but are not limited to: the criteria for hopping out of a basin, the random number distribution from which new decision vectors are pulled (uniform, Gaussian, Pareto, etc.), and the shape parameters of that random number distribution. In this dissertation, we explore two benchmark problems. We employ a different version of MBH in each benchmark problem as we discovered more efficient methodologies during this research. Algorithm 1.1 outlines the pseudocode for the variant used in the Ice Giants Dual Manifest problem, while Algorithm 1.2 outlines the variant used in the VLBI problem. Algorithm 1.1 contains two loops: a global hop loop, and a local hop loop. The purpose of this structure is to enable global search space exploration while also improving the quality of local optima found by the NLP solver. When a local optimum is found, the local hop loop uses a smaller hop size than the global loop, forcing MBH to search for improved solutions nearby. Algorithm 1.2 dispenses with the "local hop" parameters which artificially constrain the hop distance for a certain number of iterations after a local optimum is found. This change was made when it was discovered to be more efficient to simply have one hop size, chosen from a Pareto random distribution, and prioritize exploring the global space rather than the local neighborhood with each hop.

Algorithm 1.1: Monotonic Basin Hopping

Initialize f_{best}

while current iteration < max iterations OR maximum elapsed time not reached

Generate random point x.

Run local optimizer to find point x^* using initial guess x.

$x_{current} = x^*$

if x^* is a feasible point **then**

if $f(x^*) < f_{best}$ **then**

$f_{best} = f(x^*)$

save x^* to archive

end if

while $N_{not\,improve} < N_{max}$

generate x' by randomly perturbing $x_{current}$

Run local optimizer on x' to find x^*

if (x^* is a feasible point) & ($f(x^*) < f(x_{current})$) **then**

$N_{not\,improve} = 0$

$x_{current} = x^*$

if $f(x^*) < f_{best}$ **then**

$f_{best} = f(x^*)$

save x^* to archive

end if

else

$N_{not\,improve} = N_{not\,improve} + 1$

end if

end while

end if

end while

return best x in archive

Algorithm 1.2: Monotonic Basin Hopping (VLBI Variant)

Initialize $\vec{x}_{\text{best}}$, f_{best}

while current iteration $<$ max iterations OR maximum elapsed time not reached

  Generate random vector $\vec{x}$

  Run NLP solver to find point $\vec{x}^*$ using initial guess $\vec{x}$

  if $\vec{x}^*$ is a feasible point **then**

    if $f(\vec{x}^*) < f_{\text{best}}$ **then**

      $f_{\text{best}} = f(\vec{x}^*)$

      save $\vec{x}_{\text{best}} = \vec{x}^*$ to archive

    end if

  end if

  current iteration = current iteration+1

  Update total loop runtime

end while

return $\vec{x}_{\text{best}}$ from archive

2.4 Discrete Parameter Optimization

The inner-loop algorithm optimizes a continuous valued decision vector, but there are numerous trajectory optimization problems which involve discrete variables. Discrete variables are, strictly speaking, integers. These variables may be used to represent the towns in a traveling salesmen problem (TSP), the planets or asteroids in a flyby sequence, the number of thrust arcs on a low thrust trajectory, the number of throttle modes on a thruster, etc. One example of a discrete optimization problem would be: given a final intercept destination at Jupiter, find the flyby sequence of 3 inner planets which a spacecraft may use to reach Jupiter in minimum time. In this example, there are 4 inner planets to choose from: Mercury, Venus, Earth, Mars. We will allow the

same planet to be used multiple times in the flyby sequence. Then given a 3-planet intermediate flyby sequence, there are $4^3 = 64$ possible flyby sequences to choose from. Assuming there are no other variables in the decision vector other than those of the flyby sequence, this problem is small enough to solve with a grid search. However, if we increase the required length of the flyby sequence and/or the number of flyby bodies to choose from, the search space grows geometrically with each additional flyby variable. Thus locating the global optimum quickly becomes intractable using total enumeration, even if we can access substantial parallel computing resources.

The goal of a discrete optimization problem effectively becomes to find the optimal sequence of integers which minimizes some cost function. However, unlike with continuous valued search spaces, there is no cost or constraint Jacobian for the decision vector, thus gradient-free optimization methods are necessary to traverse the search space. Evolutionary methods, a particular class of gradient-free approaches, leverage principles of genetic crossover in biology, treating decision vectors as chromosomes whose genes (decision variables) may be swapped or mutated at random. Evolutionary methods have been demonstrated as effective search space traversal methods, and use the following general process:

1) Start with a randomly generated population of chromosomes.

2) Evaluate the fitness of each chromosome in the population using some cost function.

3) Choose 'parent' chromosomes, according to their fitness, to seed the next generation of chromosomes. In evolutionary algorithm research, giving preference to fitter parents to create children is known as elitism.

4) Select two parents to pass their genes along to a 'child' chromosome. The child chromosome is created by flipping a coin to choose a gene from each parent.

5) Apply a random mutation operator to the child's genes, then return to step 2. After crossover is completed, each gene in the child chromosome has a small chance to be randomly mutated. Using elitism and crossover alone, the solution chromosomes tend to cluster in a subset of the search space. This practice subverts this shortcoming, encouraging continued global search.

This process of searching the decision space for fit solutions and using the fittest members

to seed new generations facilitates the discovery of better solutions to the optimization problem over many generations, but there is no explicit convergence criterion in these approaches. However, evolutionary methods find good solutions far more quickly than total enumeration grid searches, especially for larger search spaces. The trajectory optimization problems we explore in this dissertation, however, are neither discrete nor continuous, but a hybrid of the two. In this dissertation, the parameters being optimized form a mixed-integer programming (MIP) problem. Distinct optimization routines are required for each class of parameter. An effective approach for optimizing a MIP is to create hybrid optimal control problem (HOCP) formulation which combines a gradient-based inner-loop with a gradient-free outer-loop. The outer-loop chooses integer chromosomes which encode distinct continuous-valued trajectory optimization problems to be optimized by the inner-loop.

To expand on the previous example of the minimum-time triple-planetary flyby sequence optimization problem, the inner-loop may choose continuous parameters including the launch window, the characteristic launch energy C_3 (km/s^2), the maneuver vectors, and more. The inner-loop thus optimizes the trajectory through a given flyby sequence for minimum time of flight, and once the solution is returned, the outer-loop compares every flyby sequence chromosome and chooses the ones with the lowest TOF cost to form the pool of parent chromosomes which will seed the next generation. In general, however, global trajectory optimization is not a single-objective, but a multi-objective problem. Numerous different quantities such as TOF and ΔV contribute to the fitness of a given solution.

2.5 Multi-Objective Outer-Loop Solver

To solve a multi-objective global trajectory optimization problem, one common approach is to combine the cost functions into one cost function: $J_{\text{aug}}(\vec{x}) = G(J_1(\vec{x}), J_2(\vec{x}), ..., J_M(\vec{x}))$. This approach is intuitive because an NLP solver can only optimize one cost function at a time. Global trajectory optimization is already time-consuming for single-objective problems. At first glance, in order to optimize a design for multiple objectives, a separate NLP solver run would need to be

carried out for each objective. However, the goal of multi-objective optimization is to generate the Pareto-optimal front of the solution space [26]. This M-dimensional hypersurface, where M is the number of objectives, depicts the fundamental trade-offs between the objective functions considered in a study, beyond which no improvement in the solutions can be made. No point solution on the front completely dominates any other, that is, no solution is superior to another across all its objectives. Thus, traversing from one point on the front to any other in any direction requires degrading the performance of at least one objective. An illustration of this Pareto front is shown in Fig. 2.2. Feasible designs exist in the blue region above the front, optimal solutions upon it, and no designs may exist in the region below it. Note that to traverse this curve to the left or right, at least one objective's fitness must be degraded.

Instead of providing the inner-loop one combined cost function to optimize, we pose the multi-objective problem as a vector of cost functions: $\vec{J\vec{x}} = [J_1(\vec{x}), J_2(\vec{x}), ..., J_M(\vec{x})]$. The inner-loop only optimizes a decision vector for $J_1(\vec{x})$, but the solution it finds still contains data for all $J_2(\vec{x}), ..., J_M(\vec{x})$. Once the inner-loop completes its optimization, the solution is returned to the outer-loop, which then records any of its other cost function values, using non-dominated sort to determine the solution's Pareto rank. A Pareto rank of '1' puts the solution on the first non-dominated front (the Pareto front). If a solution has a rank greater than 1, it does not fall on the Pareto front and represents a suboptimal solution. In this dissertation, these fronts are discrete scatter plots of points rather than smooth curves, as designs do not necessarily exist at all possible values of the cost functions.

The Non-Dominated Sorting Genetic Algorithm-II (NSGA-II) is an ideal method for solving multi-objective hybrid optimal control problems, employing a fast non-dominated sort routine to evaluate a problem spanned by M objectives via the Pareto criterion [26, 30]. It has proven effective on a wide range of flight dynamics applications [29, 35, 30]. The outer-loop, which operates solely on a population of integer decision vectors, selects from integer-encoded menus of caps for different objective functions and thus constrains the inner-loop problem, binning the secondary objective functions to a particular range for each candidate design. Inner-loop designs optimize Δv

exclusively, but the resulting decision vectors carry information regarding any secondary objectives such as TOF. With a single run of the inner loop, all of the multi-objective performance information is returned to the outer-loop for the ranking of the population. Thus, rather than performing an entire optimization run on a given outer-loop vector multiple times (once for each objective) a user may evaluate M objectives in the same timespan it takes to evaluate just one. Further, if M inner-loop runs were performed on a design, one for each objective, there is no guarantee that the resulting vector of cost functions would be Pareto optimal. The NSGA-II algorithm starts with an initial population of size P, then evaluates the fitness of that population by sorting its members onto non-domination fronts according to the domination criterion, given by Eq. 2.6.

$$\begin{aligned} & a \prec b \text{ if} \\ & 1)\ \forall(f_i(a) <= f_i(b))\ \ i \in [1,k] \\ & 2)\ \exists(f_j(a) < f_j(b))\ \ j \in [1,k] \end{aligned} \tag{2.6}$$

That is, solution a dominates solution b if all k cost functions of solution a are less than or equal to the cost functions of solution b, and at least one of a's costs is less than b's. The population of solution chromosomes then undergo crossover and mutation with each other to create the next generation of candidate solutions, and the process repeats, preserving elite members as it proceeds, until a maximum number of generations set by the analyst have elapsed.

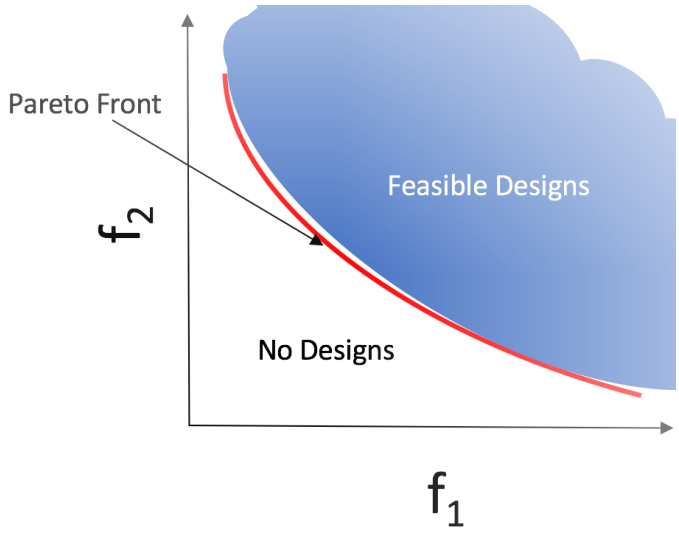


Figure 2.2: Concept illustration of the Pareto front for a mission with two objectives, f_1 and f_2 [61].

2.6 Multiple Spacecraft, Multiple Objective Global Trajectory Optimization

Extensive research has been performed in each of the respective fields which are incorporated in this dissertation. However, no technique has been developed which combines the state-of-the-art from each of them into one capability that can perform multi-spacecraft multi-objective global trajectory optimization for multi-target exploration missions. In this dissertation, we seek to explore new and interesting global trajectory optimization problems involving multiple coordinating spacecraft. To do this, we combine state-of-the-art global trajectory optimization capabilities into a multi-objective, multi-agent (MOMA) HOCP framework. In multi-agent optimization, agents are computational entities with optimizable states and behavior or policies. We have discussed the difficulties encountered in single-spacecraft multi-objective global trajectory optimization, particularly regarding the costs of executing the continuous trajectory optimization and the traversing the massive discrete parameter space. As discussed previously, a single spacecraft multi-objective multi-target exploration problem, such as an interplanetary mission to Jupiter using multiple intermediate flybys, grows geometrically in complexity as the length of the flyby sequence and/or the menu of possible flyby targets increases. This complexity is further increased by the addition

of multiple spacecraft.

Multi-spacecraft multi-target exploration missions require spacecraft to cooperate to achieve coordinated science objectives, for example: visit a maximum number of asteroid targets without any spacecraft intercepting an asteroid already visited by another. Similarly, the trajectories in a fleet of spacecraft must satisfy numerous inter-spacecraft coordination constraints: time between encounters, separation distance, formation geometry, precise pointing, coordinated launch, intercept or rendezvous. No operational capability exists which can perform multiple spacecraft, multiple objective, global trajectory optimization for multi-target exploration missions.

To leverage known operational capabilities to perform this task, the necessary approach would be to decouple the multi-spacecraft problem into separate single-spacecraft optimization problems, each run with single permutation of identical constraints, within a massive grid. This approach is extremely costly in accounting for coordination constraints and objectives in comparison to creating a multi-spacecraft parameterization which allows the multi-spacecraft problem to be solved at once. In this dissertation, we address this gap by creating a fully automated MOMA HOCP framework, comprised of a multi-agent formulation of an NSGA-II outer-loop wrapped around a multi-agent inner-loop which uses MBH and MATLAB's fmincon NLP solver.

Combining these state-of-the-art algorithms into one capability is nontrivial. Leveraging this overall capability to find good solutions to new multi-spacecraft global optimization problems required the creation of new transcriptions for both the inner and outer loops, and new coordination constraints and objectives. Further, to study problems with variable fleet sizes, it was necessary to apply a technique which efficiently varies the outer-loop chromosome size. The Null Gene transcription developed by Englander et al. [28] to vary the size of the planetary or asteroid flyby sequence is leveraged here in a new and general way - to vary the number of spacecraft in a fleet. The high level relationships between the components of this MOMA HOCP technique are shown in Fig. 2.3.

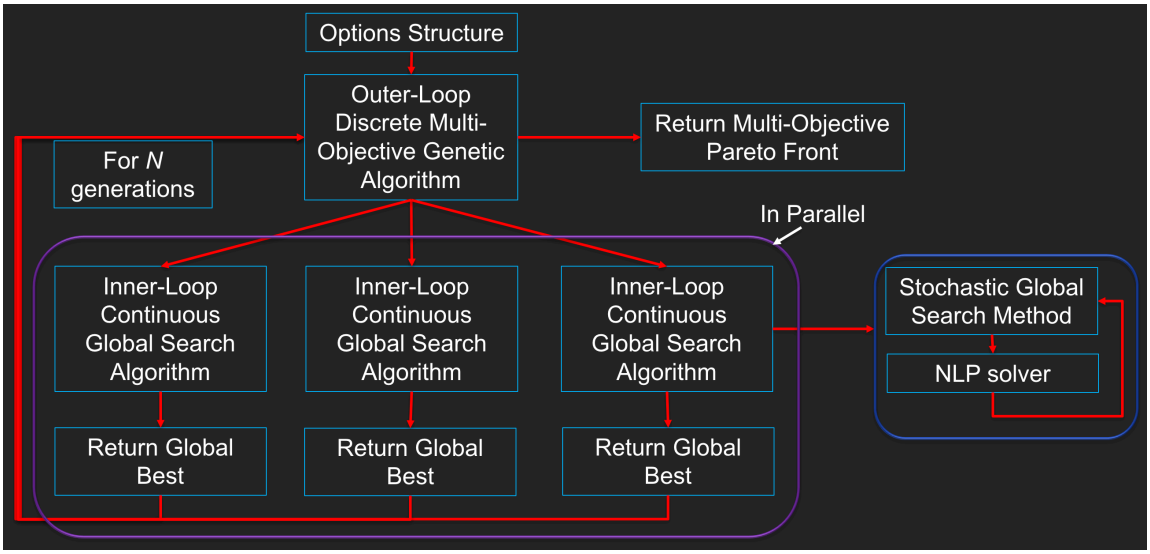


Figure 2.3: Flow Chart of algorithm execution. Each inner-loop problem represents a unique candidate mission [61].

Chapter 3

Ice Giants Multi-Mission Design

3.1 Background

The framework developed in this dissertation is first introduced with three novel coordination constraints to explore different coupled decision spaces. The technique is applied to explore the preliminary design of a dual-manifest mission to the Ice Giants: Uranus, and Neptune, which has been shown to be infeasible using only a single spacecraft anytime between 2020 and 2070. The ice giants had extremely favorable alignment during the launches of the Voyager I and II missions. Indeed, the relative geometry of Uranus and Neptune was similar enough that this would be the most favorable for a hypothetical dual manifest mission for more than 150 years hence.

3.2 Problem Formulation

The primary objective of the analysis in this benchmark problem is to find missions, if they exist, where two spacecraft may share a launch vehicle and perform gravity assist sequences of the inner and outer planets to ultimately intercept the ice giants Uranus and Neptune. Planetary orbit states were acquired from the JPL Horizons database at a single epoch, and were propagated inside the optimizer, along with spacecraft trajectories, using a universal variable Kepler propagator. This was done twofold for ease of prototype development and to avoid the slow runtime of SPICE ephemeris lookups. A single impulsive maneuver is allowed between each pair of flyby bodies. With the inner-loop's functionality validated, the full MOMA HOCP technique was put to work on an Ice Giant Multi-Mission preliminary design. The mission design consists of two high-thrust chemical

propulsion spacecraft, one of which must intercept Uranus and the other Neptune. The MOMA HOCP optimizer is tasked with optimizing the flyby sequences for each spacecraft for total ΔV, while also trading against time of flight (TOF).

Trajectories in this benchmark problem are transcribed using the Vinkó-Izzo Multiple Gravity Assists with one Deep Space Maneuver (MGA1DSM) transcription, a direct method that is easy to formulate and ideally suited to optimization with evolutionary algorithms. This is due to its efficient transcription of a trajectory into few decision variables, and its use of implicit, rather than explicit, constraints to satisfy trajectory continuity [48].

The towns which the "salesmen" (spacecraft) "visit" are planets. All trajectories begin at the Earth, and propagate according to two-body patched conics assumptions. Each mission phase is assigned a TOF and is composed of a forward propagated Kepler arc for some fraction η of that TOF, hereafter known as the Deep Space Maneuver (DSM) Index. This Kepler arc is followed by an impulsive DSM calculated by computing a minimum ΔV Lambert transfer arc to intercept the next flyby target over a duration equal the remaining fractional TOF. For simplicity, the authors chose to only allow for single-revolution Lambert arcs to be calculated. Allowing for multiple-revolution arcs to be calculated would require additional, non-trivial changes to the problem. Specifically, the outer-loop transcription would need to accommodate a boolean decision variable for each phase of a trajectory determining whether the inner-loop would target a multi-revolution or single-revolution transfer. The outer-loop transcription parsing algorithm would need to be altered to accommodate this decision variable. Secondly, for a candidate MVM with M vehicles and N phases per vehicle, this change would result in an increase of $M \times N$ outer-loop decision variables, significantly inflating the decision space of the entire problem. The authors recognize the potential for interesting and meaningful impact on the optimality of inner-loop solutions from implementing such a feature. However, for the purposes of demonstrating the capabilities of the MVM outer-loop in this work, this capability was deemed noncritical. In addition to choices of DSM index and total phase TOF, the first phase begins with the choice of launch date in Modified Julian Date (MJD = JD$-$2,400,000.5), C_3, Right Ascension of Launch Asymptote (RLA), and Declination of Launch Asymptote (DLA),

which all together define the initial position and outgoing velocity of the spacecraft. All subsequent phases are transcribed with four parameters: h_{flyby} (flyby altitude), η, total phase TOF, and the angle β. This angle, β, is defined at a flyby body around the base of a $V_{\infty\, in}$ cone's circumference and determines the position of the outgoing hyperbolic velocity on this cone [48]. The sum of the the TOFs for each phase arc in a spacecraft trajectory is subjected to the global TOF constraint value imposed by the outer loop. Shown in Fig. 3.1, this transcription ensures unpowered flybys.

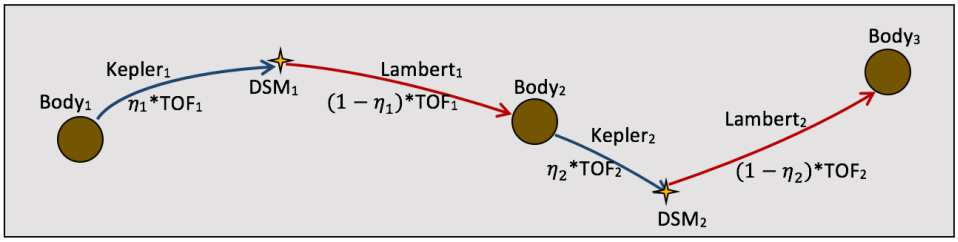


Figure 3.1: Illustration of the MGA1DSM transcription. This transcription is well suited for use with evolutionary algorithms and is readily applied to trajectory optimization problems with small bodies, planets, or both.

3.2.1 MVM Outer-Loop Transcription

The outer-loop problem is defined by a integer decision vector, which in turn is operated upon by the NSGA-II algorithm. A multi-vehicle trajectory may be conceptualized as a 2-dimensional decision vector. Each row would be a vector corresponding to the trajectory of a single spacecraft, while each column contains a gene whose value is the index encoding a particular physically meaningful decision variable from a menu. In the concept of this work, in each row, the first four genes comprise a header encoding bounding parameters: 1) launch window bounds, 2) global TOF bounds, 3) C_3 bounds, and 4) a gene encoding a required minimum number of identical flyby variables in the flyby sequence. This fourth parameter is a coordination constraint that will be discussed in a later section. Beyond this header, all remaining genes encode flyby targets.

The destination body, that is, the final gene in each row, for each spacecraft may be allowed to vary, and is thus included in the decision vector. However for the results presented in this work,

a constraint is imposed to make it fixed; it does not evolve during successive generations of the NSGA-II algorithm. This is done to study problems where the desired final destination is known *a priori*. While the header parameters need not necessarily be identical for both spacecraft, they are constrained to be identical for all spacecraft in this work. For example, consider the dual-spacecraft mission transcribed below. In Eq. 3.1, spacecraft 1's trajectory is represented in the first row, while spacecraft 2's trajectory is represented in the second. In each row, the decision variables are integers representing, in order: launch window, time-of-flight cap, C_3, number of flyby genes required to be shared by all spacecraft in the fleet N_{shfb}, and flyby bodies. Let us assign some example values to this chromosome via Eq. 3.2.

$$X = \begin{bmatrix} LW & T_{cap} & C_3 & N_{shfb} & \text{Planet}_{1,1} & \text{Planet}_{1,2} & \text{Planet}_{1,3} & \text{Destination}_1 \\ LW & T_{cap} & C_3 & N_{shfb} & \text{Planet}_{2,1} & \text{Planet}_{2,2} & \text{Planet}_{2,3} & \text{Destination}_2 \end{bmatrix} \tag{3.1}$$

$$= \begin{bmatrix} 1 & 5 & 2 & 1 & 1 & 10 & 4 & 6 \\ 1 & 5 & 2 & 1 & 2 & 3 & 5 & 7 \end{bmatrix} \tag{3.2}$$

$$= \begin{bmatrix} \{1/1/2030, 5/1/2030\} & 10\,\text{yr} & [50, 60]\,\text{km}^2\text{s}^{-2} & (0\,\text{shfb}) & \text{Venus} & \text{NULL} & \text{Jupiter} & \text{Uranus} \\ \{1/1/2030, 5/1/2030\} & 10\,\text{yr} & [50, 60]\,\text{km}^2\text{s}^{-2} & (0\,\text{shfb}) & \text{Earth} & \text{Mars} & \text{Saturn} & \text{Neptune} \end{bmatrix} \tag{3.3}$$

It is helpful and intuitive to conceptualize an MVM in this way, but ultimately the two dimensional concept is transformed into a 1-dimensional chromosome of independent genes. It is worth noting that this requires a bookkeeping structure to track which variables map to which spacecraft. The transformation is given in Eq. 3.4.

$$\begin{bmatrix} \{\text{shared vars}\} & \{\text{S/C 1's unique vars}\} \\ \{\text{shared vars}\} & \{\text{S/C 2's unique vars}\} \end{bmatrix} \Longrightarrow \begin{bmatrix} X_{\text{shared}} & X_{\text{SC1}} & X_{\text{SC2}} \end{bmatrix} \tag{3.4}$$

This transcription parametrizes a multi-spacecraft problem as a a single trajectory until the occurrence of a boundary event at which point another spacecraft spawns when its journey

requires it diverge. A boundary event in this problem may be a DSM or a planetary flyby. In the 2-dimensional conceptualization of the evaluated chromosome in Eq. 3.3, the number of rows of this chromosome equals the maximum allowed number of spacecraft in a fleet (in this case 2) while the number of columns is equal to the length of the header plus the maximum number of allowed intercept bodies for a spacecraft (which in this case is 4). Note that one gene in equation (2) evaluates to NULL. In the decision variable menu for allowed intercept bodies, which in this work are planets, there are N possibilities to choose from when each intercept body gene in the sequence is created. However, the gene creation process allows for an integer gene to be chosen that is between 1 and $2N$. Thus, there is a 50% chance that an intercept body gene is an integer that does not correspond to an index on the decision variable menu. If and when this happens, the inner-loop ignores that gene since there is no physical intercept body to be reached. The inner-loop thus optimizes a pruned decision vector that does not contain any null genes; only physically meaningful ones.

This **Null Gene** transcription was developed by Englander et al. 2013 to solve the problem of allowing the length of a flyby sequence to vary while maintaining a fixed array size programmatically. In short, the Null Gene transcription efficiently enables the encoding of the *lack* of an intercept body [31]. It applies the insertion and deletion behaviors found in the genetic crossover of natural selection to the optimization of a variable length flyby sequence. Final destination intercept bodies are included in the outer-loop decision vector in order to be passed to the inner-loop. For the problem explored in this work, the destinations are constrained to not crossover or mutate since they are known. Overall, the multi-agent transcription summarized here presents an efficient setup to pose MVM design problems, enabling outer-loop and inner-loop coordination constraints using a header of of bounding parameters and a transformation of the overall problem into a single chromosome.

3.3 Outer-Loop Coordination Objectives (Minimax Approach)

The approach taken for multi-agent objectives is a **weakest link** formulation. For a given objective function, the cost assigned to the fleet is equal to that of the highest cost spacecraft. In the outer-loop, minimal-cost individuals are dominant. Over successive generations, evaluating a fleet's cost this way incentivizes minimization of the cost of all spacecraft in the fleet. This is known in game theory as a **minimax** problem - where the maximum cost (or "loss") across individuals is minimized. This approach has never been applied to a interplanetary MVM design. In the results presented in this work, TOF is a handled as an outer-loop minimax objective, with the cost assigned to a trial fleet equal to the TOF of slowest spacecraft to reach its destination. This coordinated objective approach can be applied to endless other quantities of interest to mission designers such as payload mass, fuel mass, ecliptic inclination change, number of flybys, etc. The choice of using minimax as a coordinated objective approach is driven by its applicability to integer optimization problems, and by its ability to minimize the individual and collective loss by all players in a game (or in this case, all spacecraft in a fleet). Secondly, consider a fleet of identical spacecraft. TOF drives, among other factors, the design of the spacecraft bus. The longer the TOF, the more redundant components needed, so minimizing the longest TOF also minimizes the need for redundant hardware.

3.4 Outer-Loop Coordination Constraints

The dominant challenge of this work is how to efficiently and effectively handle coordination constraints to facilitate robust global optimization of an interplanetary MVM. This challenge is comprised of multiple hurdles including how the outer-loop discretizes continuous-valued constraints, how those constraints are manifested in the inner-loop, classifying these coordination constraints, and understanding their impact on the solution space of an MVM design problem. In this work, three coordination constraints are introduced.

3.4.1 Shared Launch Vehicle Constraint

In this work, all spacecraft in a trial fleet are subject to the initial point coordination constraint of sharing a launch vehicle. This active coordination constraint approach guides the optimizer through the true multi-mission solution space, coupling the performance of one spacecraft to the other. In the outer-loop, this constraint is handled in the header block of the decision vector via genes for each spacecraft encoding identical launch epoch bins, and C_3 bins, and a boolean value enforces that these genes remain identical for all spacecraft in the fleet during the outer-loop's genetic crossover process. At the inner-loop level, any parameters that the outer-loop stipulates must be constrained to be identical are contained within a "shared parameters header". Then, any unique parameters to spacecraft #1 are concatenated to the header, followed by unique parameters to spacecraft #2, etc. Thus the inner-loop makes a transformation of the 2-dimensional MVM vector into a 1-dimensional vector based on shared parameters, and this allows any shared parameters and coordination constraints to be strictly enforced during optimization. The shared launch vehicle constraint manifests in the inner-loop decision vector by holding the launch epoch, C_3, RLA and DLA parameters into the shared header.

3.4.2 Minimum Number of Identical Flyby Genes Constraint

For various reasons, a mission designer may want multiple spacecraft in a fleet to flyby identical targets, not necessarily at the same time. For example, the Voyager spacecraft performed staggered flybys of Jupiter and Saturn to leverage favorable turning angles. These staggered flybys also enabled interesting secondary science objectives including imaging the planets at different points along their orbits. This is an intermediate point constraint that is applied via an outer-loop decision variable menu with genes specifying anywhere from zero to M desired identical flyby genes. If a trial fleet is allowed N intermediate flybys to its destinations, then a user sets the shared flyby genes menu to allow $0 - M$ shared flyby genes (where $M < N$).

One subtle caveat is that given the outer-loop's null gene transcription, flyby genes are not

equivalent to physical flybys. With fixed non-null destination genes, the fleet size is constrained to be constant at the user-specified maximum number. But with intermediate flybys, the number of flybys can and does evolve over generations thanks to the null gene transcription's insertion and deletion behavior. For example, consider the following 2-dimensional conceptualization of an outer-loop decision vector:

$$X = \begin{bmatrix} LW & T_{\text{cap}} & C_3 & (3\,\text{shared flyby genes}) & \text{Venus} & \text{NULL} & \text{NULL} & \text{Jupiter} & \text{Uranus} \\ LW & T_{\text{cap}} & C_3 & (3\,\text{shared flyby genes}) & \text{Venus} & \text{NULL} & \text{NULL} & \text{Saturn} & \text{Neptune} \end{bmatrix} \tag{3.5}$$

In Eq. 3.5, the outer-loop has generated a vector where both spacecraft share at minimum the first three flyby genes, but 2 of those genes were generated null. So functionally, the inner-loop problem is only optimizing trajectories with one intermediate flyby each rather than 3 (gene deletion occurs). However when two trial fleets undergo crossover during NSGA-II, a null intercept body gene may be swapped with a valid intercept body gene, and insertion occurs. Similarly, a null gene can mutate into a valid gene. With the identical flyby genes constraint turned on, the M-flyby-gene long flyby sequence for each spacecraft in the fleet must change identically throughout the mating process, if at all, due to the transformation described in Eq. 3.4.

For lower C_3 trajectories, a longer maximum length for the flyby sequence may be necessary to deliver spacecraft to their destinations, but this length will also increase the runtime of the optimizer. This occurs because 1) in the GA outer-loop, every gene added to a chromosome geometrically increases the size of the search space and 2) at the inner-loop level, a feasible trajectory with more intermediate flybys is geometrically more difficult to find with each additional flyby added. Due to the slow convergence of `fmincon` coupled with the MGA1DSM transcription's lack of analytical derivatives, it was found that candidate missions involving very few intermediate flyby encounters could be studied reliably in a tractable runtime.

3.4.3 Minimum Number of Shared Trajectory Phases Constraint

This boolean constraint enables the transcription of a trajectory where both payloads fly the exact same trajectory for a portion of the mission. This constraint may only be enabled if the minimum shared flyby genes constraint is also enabled. If so, then for any non-NULL flyby target genes, the two spacecraft are required to share the same trajectory until all specified shared flybys are completed. However, if any of the shared flyby genes are NULL, such as in Eq. 3.5, those outer-loop genes are ignored by the inner-loop before the trajectories to them are constructed.

This constraint is enforced by the inner-loop via a header of shared parameters. Just as with the outer-loop vector in Eq. 3.5, the 2D hybrid decision vector conceptualization is transformed into a 1D vector by the inner loop by parsing it into blocks in the following order: a header of shared decision variables including launch asymptote, and shared trajectory phases, followed by unique decision variables for spacecraft 1, then unique decision variables for spacecraft 2, etc. The indices that encode these blocks are stored in a separate structure parsed during the evaluation of an objective function. By making this transformation, the shared trajectory phases constraint is strictly enforced. Were the vector to be handled as 2-dimensional, with one row per spacecraft, a linear constraint would need to be imposed in the inner loop for every decision parameter that is meant to be identical, and the NLP solver will pass through regions of constraint violation with no guarantee of satisfying the constraint. With the transformation to 1 dimension, no extra linear constraints need be imposed and there is no room to violate the coordination constraints. This constraint produced the least feasible solution space, due to the relative geometry of Uranus and Neptune.

3.5 Results

The 2013 Planetary Science Decadal Survey identifies both Uranus and Neptune as high priority science targets for future missions. However, no one spacecraft can perform flybys of both

targets within the next 50 years [46]. This is, in part, due to the lacking conjunction geometry that existed during the Voyager spacecraft launches and occurs roughly once every 170 years. Therefore, in order to visit both targets, two vehicles are needed. This entails two possibilities: either two separate single-vehicle missions, or a dual vehicle multi-mission where both spacecraft are deployed from the same launch vehicle. The approach of designing the latter via cross-referenced independent grid searches for each spacecraft would be both prohibitively costly in terms of computing resources, and insufficient to provide an understanding of the optimal solution space due an ability to impose inter-spacecraft coordination constraints.

Independent searches, filtered *a posteriori* into a constrained space of shared launch opportunities, are likely to miss optimal opportunities and provide only a handful of point solutions rather than a full Pareto-optimal front. Here we present an early analysis of an Ice Giant Multi-Mission design using the novel MOMA HOCP technique described in this work. During the analysis period, the difference in ecliptic right ascension between Uranus and Neptune is considerable. This leads to the performance of one trajectory being inversely proportional to that of the other. Launched along the same asymptote, as the Uranus probe's trajectory improves, the Neptune probe's trajectory worsens and vice versa. Below, in Table 3.1, are the parameters bounding the problem formulation.

Two modestly sized low C_3 studies were run on the CU Boulder Summit cluster followed by several otherwise equivalent high C_3 studies. Each of these studies was multi-threaded and executed on a single node that contained 24 cores, allowed a maximum runtime of 24 hours. The Summit cluster does not have a proprietary MATLAB distributed computing license, so the authors were unable to leverage multiple connected nodes to complete these studies. Each outer-loop decision vector, optimized in parallel, was allotted an inner-loop optimization time of one hour, which yields the optimization of one outer-loop generation per hour. Each of the studies included the shared launch asymptote constraint, but two studies separately included the shared flyby genes constraint and shared trajectory constraint. The low C_3 studies biased results towards higher numbers of intermediate flybys, while the higher C_3 studies biased more direct transfers. The parameters for the low C_3 studies are shown in Table 3.1. Changes made for the high C_3 studies are **bolded**. The choice

of C_3 bounds for the latter study was made based on C_3 values obtained from JPL's pre-decadal study, specifically the point solution for an Ice Giant Dual Manifest mission using an optimal, purpose-built kick stage [44]. In this paper, only high impulse chemical thrust transfer trajectories were examined. In the JPL study, both spacecraft utilized low thrust propulsion systems. Thus an apples-to-apples comparison could not be accomplished in this study. However, were identical inner-loop frameworks implemented, a multi-vehicle outer-loop capable of representing the coupled problem and any appropriate coordination constraints would outperform a grid search outer-loop that optimizes separate single-vehicle subproblems, similar to that in [44].

Fig. 3.2 depicts the Pareto fronts of ΔV versus minimaxed TOF for the fleet at the 22^{nd} outer-loop generation for the minimum identical/SHared FlyBy (SHFB) genes constraint and SHared TRajectory phases (SHTR) constraint studies. Each circle represents a spacecraft fleet. The size of each marker qualitatively reports the total number of intermediate flyby targets used by the spacecraft in the fleet with the least amount of flybys. In these plots, there are two sizes: zero intermediate flybys (smaller marker), and one intermediate flyby (larger marker). The SHTR study exhibits a front with a noticeably sparser tail.

The low C_3 studies both found the minimum ΔV solution to be one that shares no trajectory phases at all, but where each spacecraft performs a flyby of one identical target (Jupiter, white orbit). The result is shown in Fig. 3.3. Red star markers depict events including launch, deep space maneuvers, and flybys. The Neptune probe performs a **solar system escape** maneuver to achieve the energy change needed to reach Neptune. That is, the spacecraft launches from Earth, performs a Jupiter flyby to target the Sun, and performs a solar flyby effectively achieving an Oberth Effect energy increase to reach Neptune, analogous to the maneuver designed by Arora et al. for a Kuiper Belt Object encounter [10]. This mission is summarized in Table 3.5 and has a total ΔV of 18 km/s with a TOF of 12.3 years for the slowest spacecraft to reach its target.

The higher C_3 studies produced results in greater abundance. The Pareto fronts for each are captured in Fig. 3.4. In Fig. 3.4 (b), depicting the front for the SHTR constraint, the structure of the front is markedly sparser than either of its less tightly constrained counterparts. Due to

the unfavorable phasing of Uranus and Neptune during the study window, numerically feasible solutions for the SHTR constraint study were few and far between. No feasible or impractical-but-feasible solutions were found in either the low or high C_3 SHTR studies. Among other concerns to be addressed in future work, a more thorough search is needed. In the SHFB study shown in Fig. 3.4 (c), the Pareto front exhibits a distinct tail at the 10-year TOF mark, where local families of otherwise equivalent ΔV solutions exist for all TOFs over 10-years. Across all studies, the authors find that launch dates later than 1/1/2032 yield markedly fewer, if any, solutions. We conclude that this is due mainly to the fact that the Ice Giants' relative difference in ecliptic right ascension has increased since the epoch of the Voyager probe launches, back when the planets were in conjunction.

In Fig. 3.5 (a), the high C_3 SHLV study's minimum ΔV solution finds a trajectory whose geometry suggests that the Neptune probe would benefit from an Earth flyby, while the Uranus probe could potentially realize a ballistic trajectory with a slightly higher C_3. Since the bounds of this study did not allow for inner planet flybys, an Earth flyby was not an available option. In 8b, the high C_3 SHFB study, the minimum ΔV solution is the most promising MVM solution among all studies run. Its primary shortcoming is the Uranus probe's 7.5 km/s DSM. Secondarily, the 'DSM' on the Neptune trajectory is performed on the launch date, which implies a need for a higher C_3 to eliminate this maneuver. Mission itineraries for these high C_3 SHLV and SHFB missions are given in Table 3.3 and Table 3.3 respectively. We investigated the possibility of leveraging a Jupiter flyby for the solution in Fig. 3.5 (b) - an option that was thought to have been missed by the outer-loop. However, running this new inner-loop case for the same launch date as the previous example, a lower ΔV solution did not in fact exist. The solution discovered is shown in Fig. 3.6.

Additionally, we present a comparison of the solutions in Fig. 3.5 (b) and Fig. 3.6 against the optimization of each of their MVMs as two separate SVMs with the same epoch and C_3 **BUT**

Outer-Loop Parameter	**Value**
Population Size	72
Number of Workers	72
Generations	24
Max. Intermediate Flybys	5
Mutation Rate	10%
Main Objective	minimize fleet ΔV
Secondary Objective(s)	[minimax TOF]
Launch Window Menu	{[1 Jan, 1 May 2030] : 4 month : [1 Sep, 31 Dec 2040]}
Global TOF Cap Menu (years)	[10, 11, ...,16]
Planetary Flyby Menu	[Venus, Earth, Mars, Jupiter, Saturn]
Min. Shared Flyby Genes Menu	[0, 1, 2, 3, 4]
Shared Phases Menu	[0, 1]
C_3 (km^2s^{-2}) Bounds Menu	{[0.0, 2.5] : 2.5 km^2s^{-2} increment : [22.5, 25.0]}
MBH Parameter	**Value**
Max. Global Search Hops	10,000
Local Hop Magnitude	±5% of current decision parameter value
Improvement Criterion	1.0E-5
N_{max}	25
Max. Runtime (minutes)	60
Outer-Loop Parameter	**Value**
Max. Intermediate Flybys	**4**
Planetary Flyby Menu	**[Mars, Jupiter, Saturn]**
C_3 (km^2s^{-2}) Bounds Menu	**{[25.0, 30.0] : 5.0 km^2s^{-2} : [210.0, 220.0]}**

Table 3.1: Table of options parameters for the MOMA HOCP setup for both the low and high C_3 regime studies.

Low C_3, SHFB	**Date**	$C_3(\frac{km^2}{s^2})$	**RLA°**	**DLA°**	$\Delta V(\frac{km}{s})$	**Alt. (r_{planet})**
Spacecraft 1	—	—	—	—	—	—
Launch	1 May 2032	25	319.4	4.1	—	—
DSM 1 (km/s)	5 Jul 2034	—	—	—	4.981	—
Flyby Jupiter	24 May 2036	—	—	—	—	50.0
DSM 2 (km/s)	24 May 2036	—	—	—	2.622	—
Encounter Uranus	7 May 2044	—	—	—	—	—
Spacecraft 2	—	—	—	—	—	—
Launch	1 May 2032	25	319.4	4.1	—	—
DSM 1 (km/s)	11 Jun 2032	—	—	—	4.528	—
Flyby Jupiter	27 Dec 2036	—	—	—	—	50.1
DSM 2 (km/s)	1 Mar 2039	—	—	—	5.591	—
Encounter Neptune	1 Jan 2045	—	—	—	—	—

Table 3.2: Mission itinerary for the minimum ΔV mission of the low C_3 SHFB constraint study (solution in Fig. 3.3).

not constrained to share a launch asymptote. Rather, the two spacecraft are meant to find the optimal mission to each of their own independent targets. We present this comparison to quantify, within the bounds of the problem in this work, the infeasibility of a dual-manifest Ice Giants mission for impulsive high-thrust spacecraft. The MVM solutions represent the "middle-of-the-road" or "least worst" launch asymptote for a co-launched mission. As discussed further in the following results, the least worst co-launched solution is considerably more costly than the total cost of two separately launched missions with different asymptotes. Within the problem setup of this work, if both spacecraft are launched along the same asymptote, a cheaper trajectory to Neptune necessitates a more expensive trajectory to Uranus, and vice versa.

For the MVM solution in Fig. 3.6, its separately optimized SVMs are described in Fig. 3.9 and the corresponding mission itineraries are referenced in Table 3.8. For the MVM of Fig. 3.5 (b), the optimal distinct SVMs at its launch epoch and C_3 are described in Fig. 3.7 and the corresponding mission itineraries are referenced in Table 3.6. Both cases exhibit a significant difference between the optimal coupled problem and the optimal uncoupled problems in terms of trajectory behavior and total ΔV. Both versions of each case (single and multi-vehicle solutions) were run for 30 minutes with identical outer loop vector parameters for equivalent comparison. In the case of the Fig. 3.6 MVM versus corresponding Fig. 3.7, there is a total ΔV difference of approximately 7.0 km/s. For the case of the Fig. 3.5 (b) MVM versus corresponding Fig. 3.8 SVMs, there is a total ΔV difference of approximately 1.2 km/s.

In order for the separate SVM formulation to find solutions that share a launch asymptote using no prior knowledge of the optimal MVM solution, one approach would be to use a grid search method which would take considerably more runtime than the MVM method of this work. Another way to illustrate the need for the MVM capability in this work is to optimize one transfer for a given launch date and C_3, then optimize the second transfer but using the exact launch asymptote of the first transfer.

High C_3, SHLV	**Date**	**C_3 (km^2/s^2)**	**RLA°**	**DLA °**	**ΔV (km/s)**	**Alt. (r_{planet})**
Spacecraft 1	—	—	—	—	—	—
Launch	8 Jun 2030	121.2	40.4	-5.2	—	—
DSM 1 (km/s)	8 Sep 2030	—	—	—	4.546	—
Encounter Uranus	17 Nov 2039	—	—	—	—	—
Spacecraft 2	—	—	—	—	—	—
Launch	8 Jun 2030	121.2	40.4	-5.2	—	—
DSM 1 (km/s)	13 Dec 2033	—	—	—	10.596	—
Flyby Jupiter	12 Oct 2034	—	—	—	—	93.9
DSM 2 (km/s)	20 Oct 2035	—	—	—	0.047	—
Encounter Neptune	9 Oct 2039	—	—	—	—	—

Table 3.3: Mission itinerary for High C_3 SHLV constraint study (solution captured in Fig. 3.5 (a)).

High C_3, SHFB	**Date**	**C_3 (km^2/s^2)**	**RLA °**	**DLA °**	**ΔV (km/s)**	**Altitude (r_{planet})**
Spacecraft 1	—	—	—	—	—	—
Launch	16 May 2030	121.0	339.3	-37.0	—	—
DSM 1 (km/s)	30 May 2034	—	—	—	7.584	—
Encounter Uranus	16 May 2046	—	—	—	—	—
Spacecraft 2	—	—	—	—	—	—
Launch	16 May 2030	121.0	339.3	-37.0	—	—
DSM 1 (km/s)	16 May 2030	—	—	—	1.854	—
Encounter Neptune	16 May 2046	—	—	—	—	—

Table 3.4: Mission itinerary for High C_3 SHFB constraint study (solution shown in Fig. 3.5 (b)).

High C_3, SHFB	**Date**	**C_3 (km^2/s^2)**	**RLA °**	**DLA °**	**ΔV (km/s)**	**Alt. (r_{planet})**
Spacecraft 1	—	—	—	—	—	—
Launch	15 May 2030	121.5	356.5	-51.2	—	—
DSM 1 (km/s)	22 Jan 2033	—	—	—	3.517	—
Flyby Jupiter	19 May 2035	—	—	—	—	105.2
DSM 2 (km/s)	21 Jul 2035	—	—	—	13.118	—
Encounter Uranus	22 May 2040	—	—	—	—	—
Spacecraft 2	—	—	—	—	—	—
Launch	15 May 2030	121.5	356.5	-51.2	—	—
DSM 1 (km/s)	17 May 2030	—	—	—	5.477	—
Encounter Neptune	24 Apr 2046	—	—	—	—	—

Table 3.5: Minimum ΔV solution itinerary for the mission in Fig. 3.6. This mission has the same outer-loop decision vector as the mission in Fig. 3.5 (b) (itinerary in table 3.4), but with a Jupiter flyby for the Uranus spacecraft. The total ΔV is significantly higher. The inner-loop runtime was 30 minutes; longer runtimes produced no improvement.

Finally, we present a comparison between an MVM point solution generated by the coupled MOMA HOCP technique against that generated by sequential optimization of two SVMs with identical asymptotes. Using the same epoch, and C_3 bounds, as those for the optimal MVM in Fig. 3.5 (b), the inner-loop was made to find the optimal Uranus trajectory. Once this trajectory was found, the Neptune trajectory optimization was run but constrained to use the asymptote from the Uranus spacecraft's trajectory. The resulting "multi-spacecraft" mission from the sequential optimization is outlined in Table 3.8, and its separate trajectories are depicted in Fig. 3.9. Compared to its optimal MOMA solution in Table 3.4, the sequentially optimized multi-mission is 2 km/s more expensive, supporting the need for a coupled MOMA HOCP formulation.

High C_3, SHFB	**Date**	**C_3 (km^2/s^2)**	**RLA °**	**DLA °**	**ΔV (km/s)**	**Alt. (r_{planet})**
Spacecraft 1	—	—	—	—	—	—
Launch	16 May 2030	122.8	24.5	-5.7	—	—
DSM 1 (km/s)	10 Sep 2032	—	—	—	4.651	—
Flyby Jupiter	20 May 2035	—	—	—	—	50.1
DSM 2 (km/s)	13 Jul 2035	—	—	—	9.118	—
Encounter Uranus	23 May 2040	—	—	—	—	—
Spacecraft 2	—	—	—	—	—	—
Launch	16 May 2030	125.0	328.8	-4.7	—	—
DSM 1 (km/s)	2 Aug 2030	—	—	—	1.372	—
Encounter Neptune	16 May 2046	—	—	—	—	—

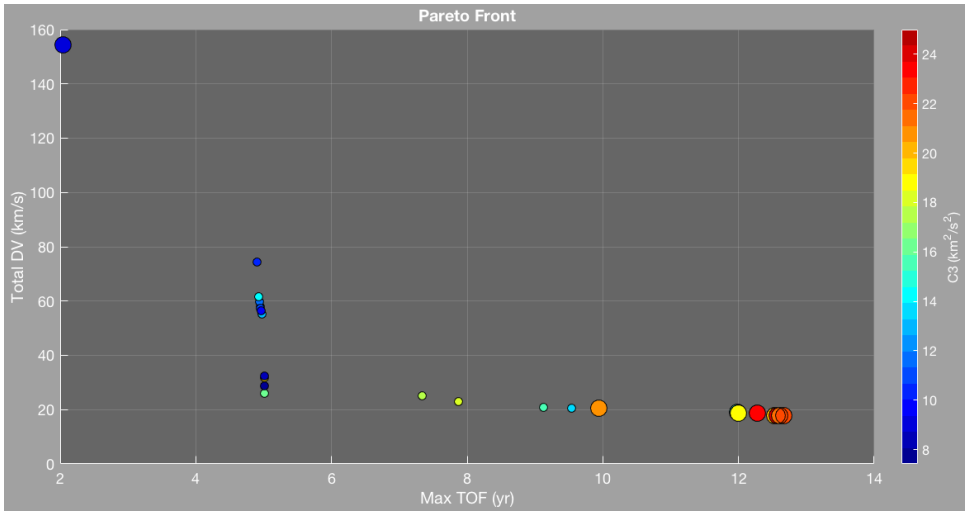
Table 3.6: Mission itineraries for two separate single-vehicle missions (solutions in Fig. 3.7), one to Uranus and the other to Neptune, with minimized ΔV for each. Each case ran for 30 minutes.

High C_3, SHFB	**Date**	**C_3 (km^2/s^2)**	**RLA °**	**DLA °**	**ΔV (km/s)**	**Alt. (r_{planet})**
Spacecraft 1	—	—	—	—	—	—
Launch	16 May 2030	125.0	5.3	-0.8	—	—
DSM 1 (km/s)	22 Oct 2031	—	—	—	6.648	—
Encounter Uranus	16 May 2045	—	—	—	—	—
Spacecraft 2	—	—	—	—	—	—
Launch	16 May 2030	125.0	328.8	-4.7	—	—
DSM 1 (km/s)	2 Aug 2030	—	—	—	1.372	—
Encounter Neptune	15 May 2046	—	—	—	—	—

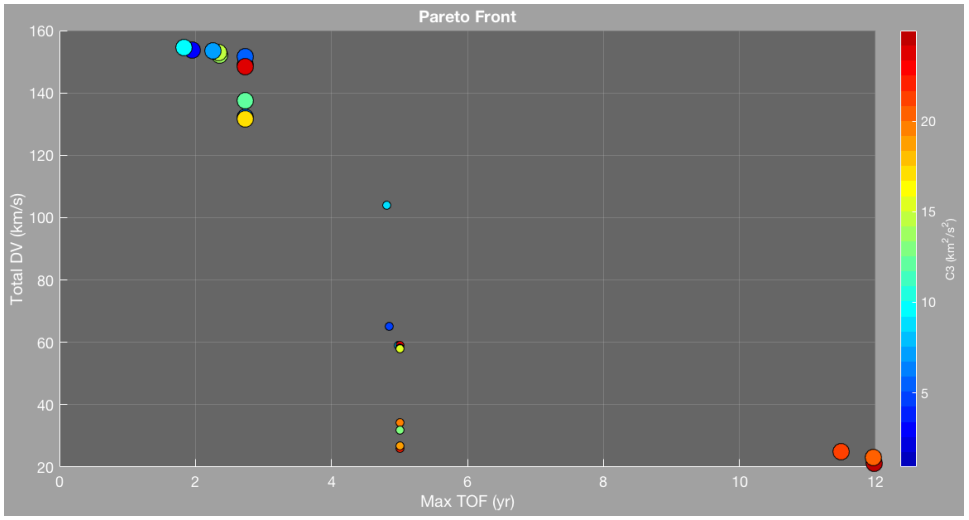
Table 3.7: Mission itineraries for two separate single-vehicle missions (solutions in Fig. 3.8), one to Uranus and the other to Neptune, with minimized ΔV for each. Each case ran for 30 minutes.

High C_3, SHFB	**Date**	**C_3 (km^2/s^2)**	**RLA °**	**DLA °**	**ΔV (km/s)**	**Alt. (r_{planet})**
Spacecraft 1	—	—	—	—	—	—
Launch	16 May 2030	125.0	5.3	-0.8	—	—
DSM 1 (km/s)	22 Oct 2031	—	—	—	6.648	—
Encounter Uranus	16 May 2045	—	—	—	—	—
Spacecraft 2	—	—	—	—	—	—
Launch	16 May 2030	125.0	5.3	-0.8	—	—
DSM 1 (km/s)	28 Aug 2036	—	—	—	4.792	—
Encounter Neptune	16 May 2046	—	—	—	—	—

Table 3.8: Mission itinerary for the "sequentially optimized" version of the optimal SHFB MVM in Fig. 3.5 (b). The separate trajectories are depicted in Fig. 3.9. Comparing against the nominal MVM in Fig. 3.5 (b), the sequential optimization method produces a far more costly mission (2 km/s more) than the coupled MVM method developed in this work. The sequential result is significantly more suboptimal, as hypothesized. Regardless of the particular inner-loop transcription, given **some** inner-loop optimizer, these results are clearly indicative of the need for coupled optimization.



(a)



(b)

Figure 3.2: Low C_3 study results: (a) and (b) show, respectively, the Pareto fronts of ΔV versus TOF for the minimum shared flyby genes constraint (SHFB) study, and the minimum number of shared trajectory phases constraint (SHTR) study. In (a), the size of each marker codifies the number of flyby genes shared, while in (b) it codifies the number of shared trajectory phases. The smaller size represents zero shared flyby genes/trajectory phases, and the larger size represents two.

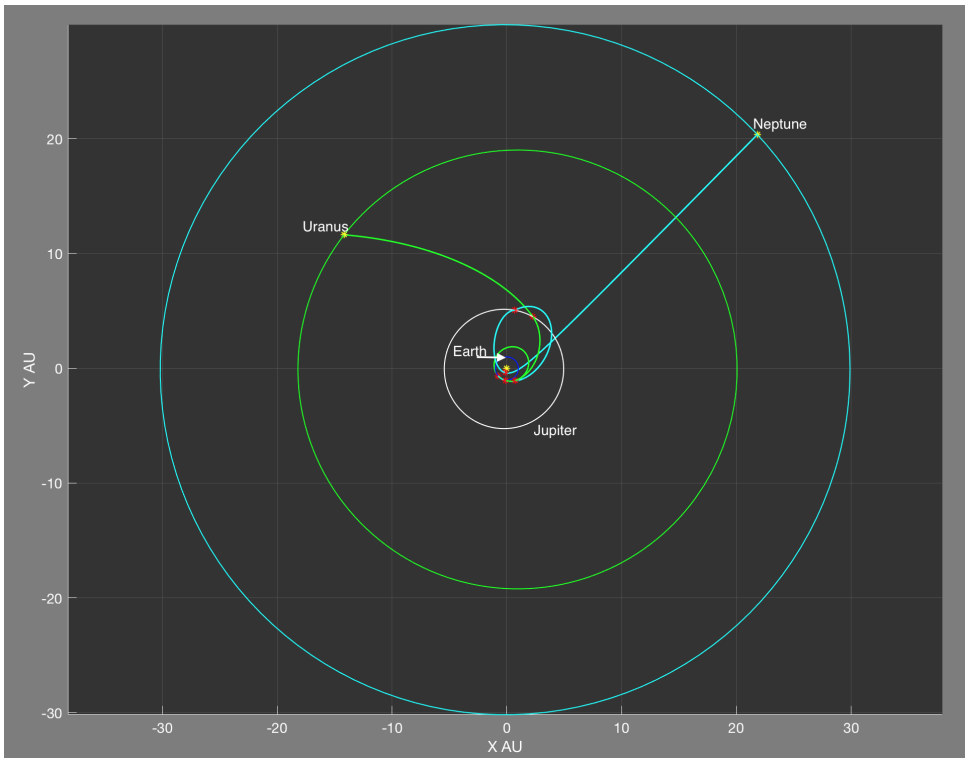
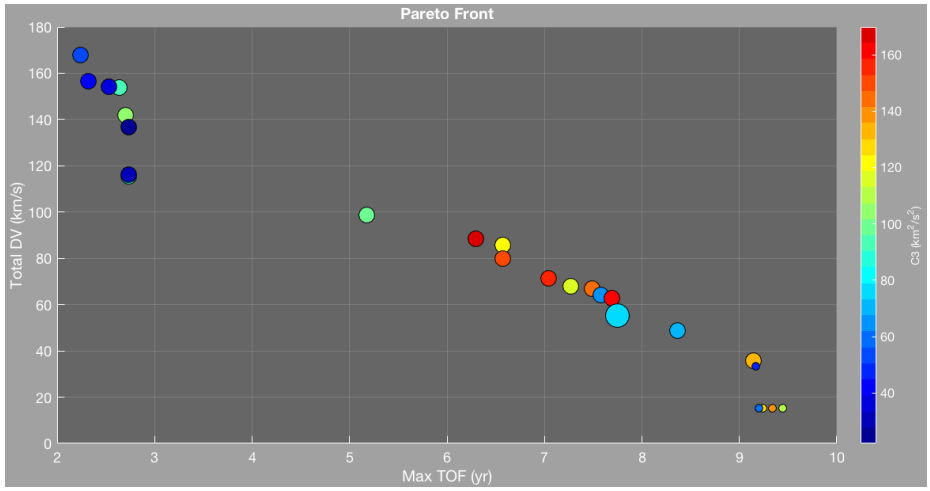
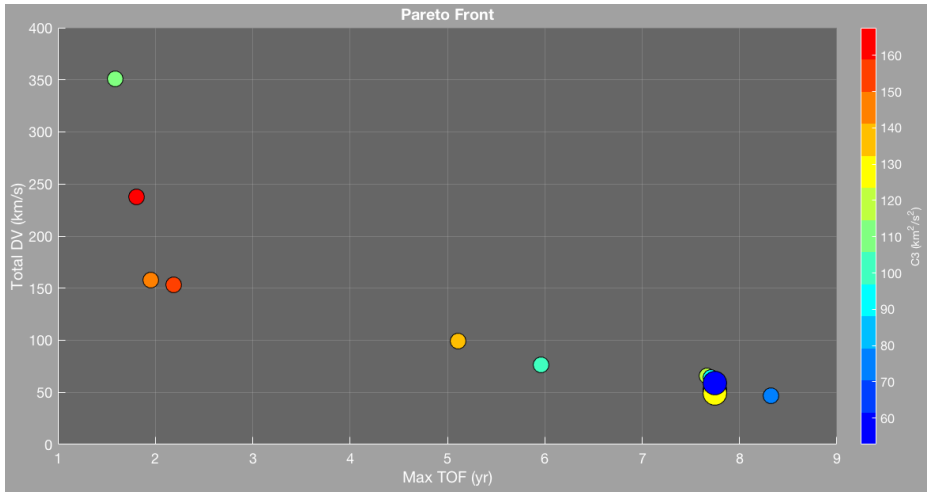


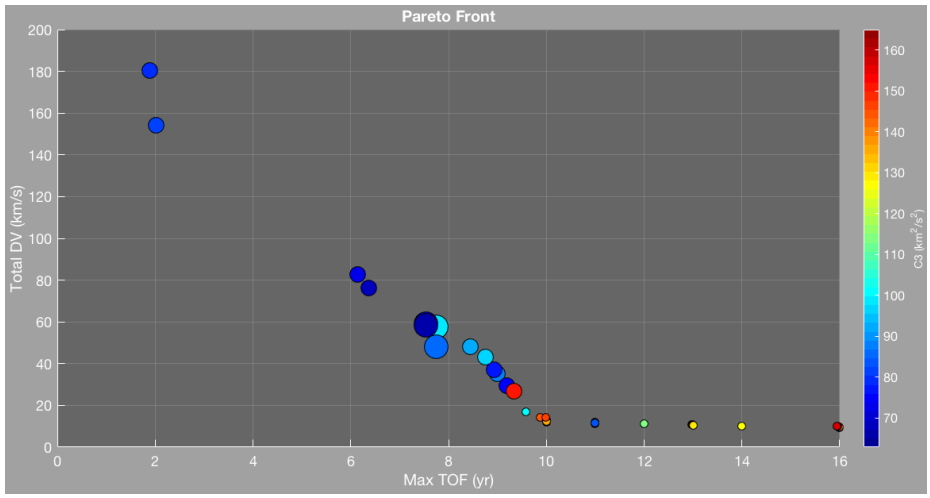
Figure 3.3: Minimum ΔV solution for the low C_3 SHFB study. Red stars indicate locations of deep space maneuvers.



(a)



(b)



(c)

Figure 3.4: High C_3 studies: All fronts shown after 24 hours of outer-loop runtime. Pareto fronts are depicted for the shared launch asymptote constraint study (SHLV) (a), SHTR study (b), and SHFB study (c). Note the far more sparse front of the SHTR study.

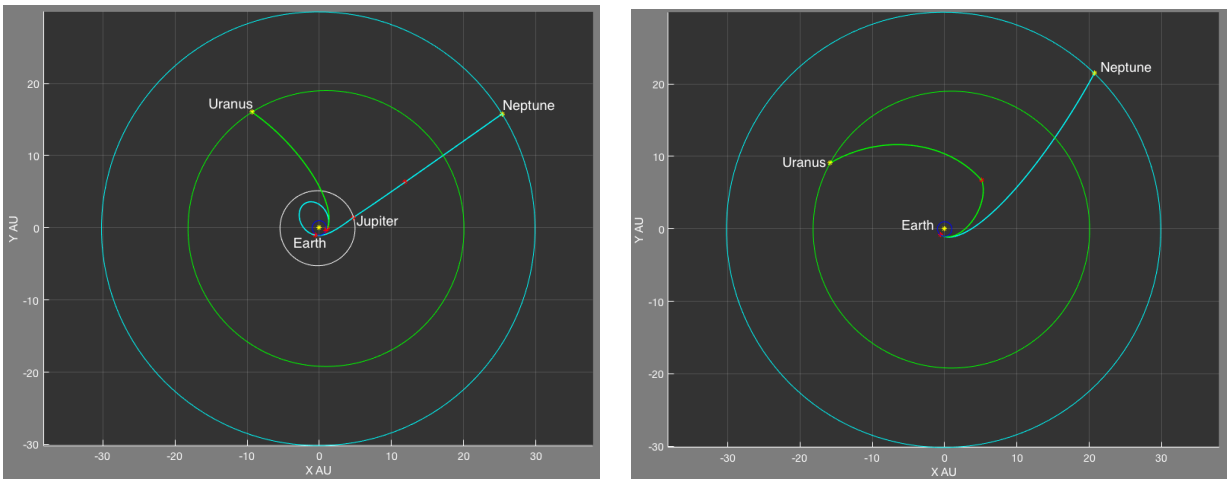


Figure 3.5: (a) The high C_3 SHLV study's minimum ΔV solution and (b), the high C_3 SHFB study's minimum ΔV solution.

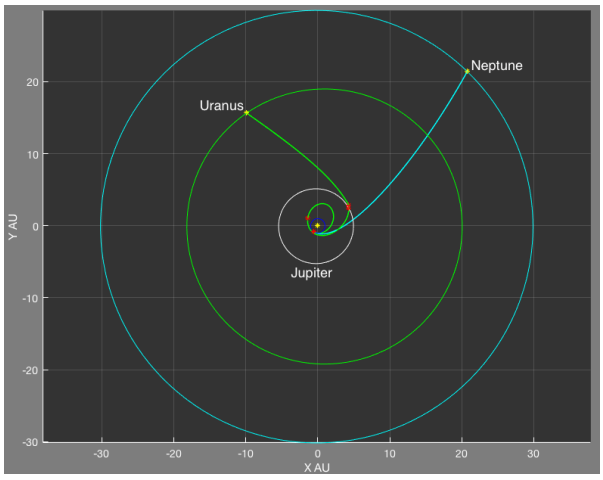


Figure 3.6: This is the high C_3 SHFB case from Fig. 3.5 (b), but rerun with the same launch epoch as Fig. 3.5 (b) to investigate the possibility of a synergistic Jupiter flyby. The mission does not appear to benefit from a Jupiter flyby at the given launch epoch.

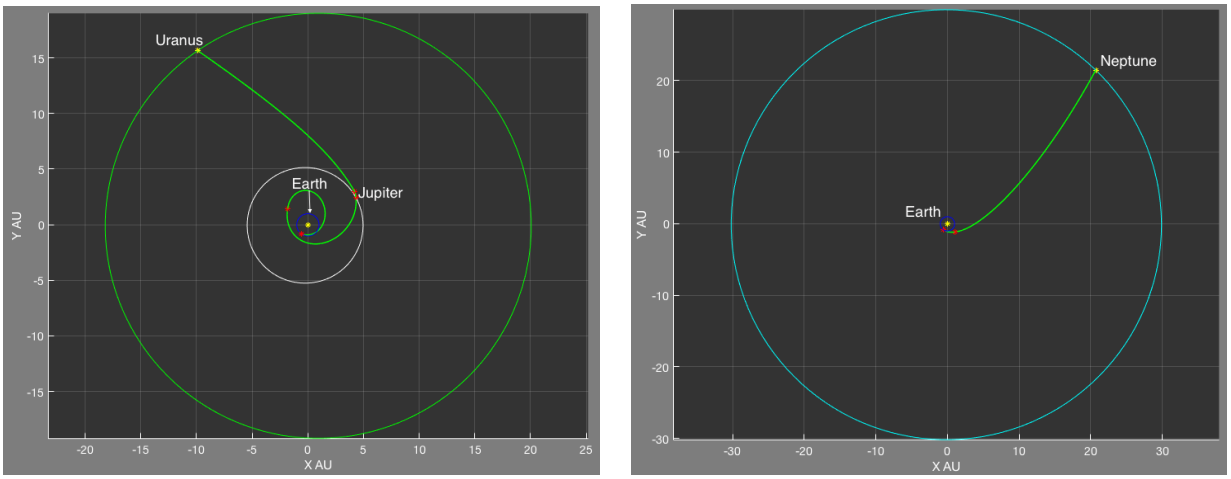


Figure 3.7: The high C_3 SHFB case from Fig. 3.6 , rerun with the same launch window and C_3 bounds, but as two separate single-vehicle problems. A dual-manifest mission would result the launch asymptote to be the one that least detriments each individual trajectory. With the launch C_3 and asymptotes not constrained to be identical, the trajectories are quite different as is the sum of their ΔV, roughly 7 km/s lower than the coupled MVM, as shown in Table 3.6.

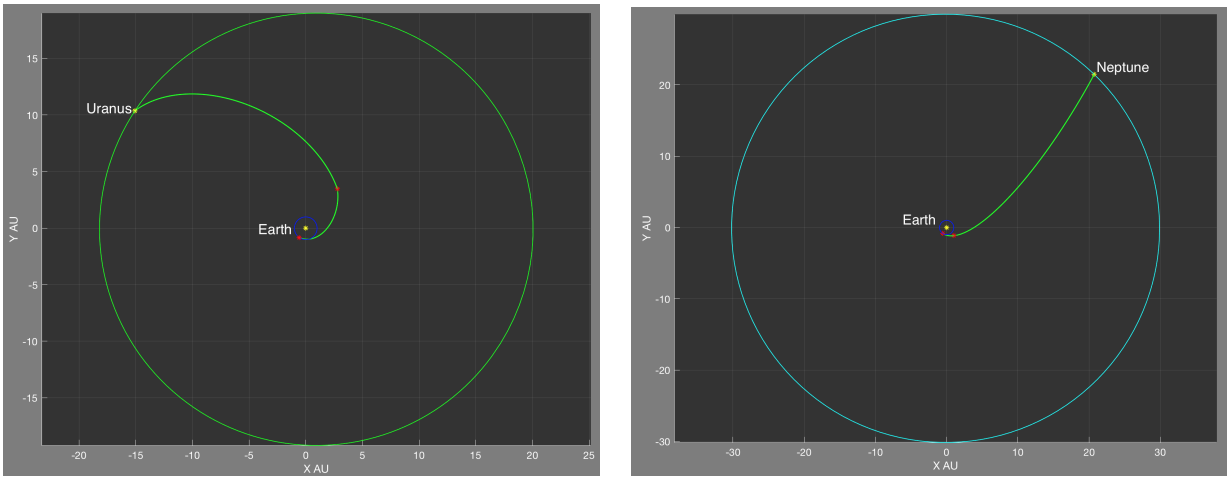


Figure 3.8: The high C_3 SHFB case from Fig. 3.5 (b), rerun with the same launch window and C_3 bounds as the Fig. 3.5 (b) mission, but as two separate single-vehicle problems. The total ΔV for these missions is 8.110 km/s, cheaper than the 9.438 km/s ΔV of the multi-mission in Fig. 3.5 b.

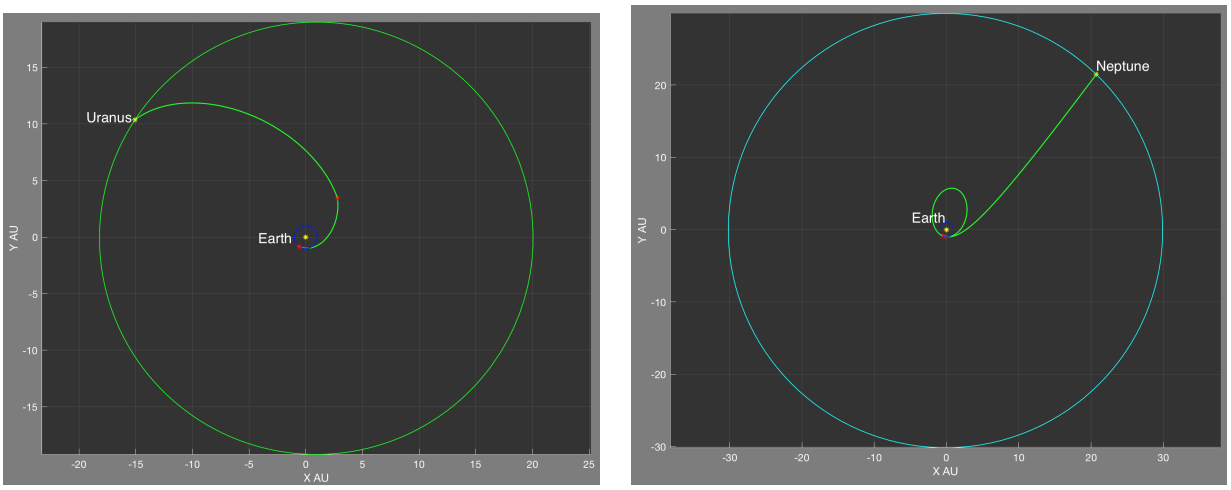


Figure 3.9: A co-launched MVM based off the solution Fig. 3.5 (b), but with each spacecraft's trajectory optimized sequentially rather than in parallel as a coupled problem. The resulting trajectories are significantly more costly in terms of ΔV, as discussed in Table 3.8.

3.6 Summary

In this paper, we have described the design of a novel Multi-Objective, Multi-Agent, Hybrid Optimal Control technique for the global optimization of interplanetary multi-spacecraft trajectories. We have developed an outer-loop transcription that parametrizes an MVM as one coupled problem as opposed to separate subproblems, and presented case studies to demonstrate the need for this transcription when investigating multi-vehicle interplanetary trajectory optimization problems. Trajectories free to optimize separately, launch at the same epoch but not constrained to share a launch asymptote, exhibited completely different behavior compared to the co-launched MVM solution at the same epoch. Further, sequentially optimized single-vehicle trajectories, with one spacecraft constrained **a posteriori** to use the launch asymptote of the previously optimized trajectory, resulted in suboptimal solutions compared to the optimization of the corresponding coupled MVM.

Three novel coordination constraints were investigated and as more were added, they projected different, often costlier, behavior onto the solution space of the problem. The shared launch vehicle constraint introduced the basic coupling of multiple spacecraft trajectories into the same launch asymptote. The SHFB constraint restricted optimization to explore search spaces where one or more intermediate flyby targets would be visited by both spacecraft, and the results behaved differently from those where this constraint was not enforced. The SHTR constraint was found to be ill-suited to the Ice Giants problem during the analysis window due to the difference in ecliptic right ascension of Uranus and Neptune, but may be an important constraint to investigate for different problems. We introduced a minimax method as an inter-spacecraft coordinated objective approach for the TOF cost function to incentivize each spacecraft in a fleet to arrive at their targets in minimum time. The goal of this work was to develop and present a novel approach to posing and optimizing interplanetary MVMs with basic feasibility, and the results of this work demonstrate promise for the current technique.

The transcription presented in this chapter illustrated how a multi-spacecraft problem may

be parametrized into a structure where there is initially a single trajectory with new spacecraft then spawning at boundary events, such as post-launch deep space maneuvers, or planetary flybys. A header of shared parameters by all spacecraft in the fleet precedes gene sequences representing the unique individual traveling salesmen problems for each spacecraft. As developed thusfar, this structure requires shared parameters in the header to pertain to consecutive phases of the fleet's trajectories starting from the first phase. The possibility of having a fleet where shared boundary events are nonconsecutive can not be captured. In this dissertation, we seek to find a more general transcription which can pose any MTSP in the outer-loop. To find such a transcription, we explore a new benchmark problem in chapter 4.

Chapter 4

Very Long Baseline Interferometry Mission Design

4.1 Background

In this chapter, we introduce several new contributions to expand on the proof of concept in Chapter 3. Whereas the previous benchmark problem examined the trajectories of a fleet with two spacecraft whose final destinations were fixed, the final destinations in this problem are free to vary as does the fleet size. The trajectories do not spawn at boundary events along a single initial trajectory, but rather, begin at free points in space. We focus this benchmark's analysis on the science orbit phase of a VLBI mission. The overall MOMA-HOCP transcription employed is now more generalizable for coordinated Multi-Traveling Salesmen Problems (MTSP) beyond this one. We apply s Null Gene transcription to vary the fleet size, and a introduce a cost function for observation multiplicity (multiple VLBI observations taken at the same epoch). We show that through application of these techniques, we can efficiently explore the multi-objective Pareto front across range of fleet sizes and observations taken. Within the Pareto front, many interesting solutions appear, including one mission with a fleet of 7 spacecraft which images 16 radio sources. We also find that in many cases, for a given fleet of spacecraft, allowing multiple VLBI observations to be taken at each epoch yields a mission which images more targets in less time than a mission with only one observation per epoch. In §4.2, we first introduce the GTOC-8 trajectory optimization problem statement as a motivating point for this work, then describe problem we have created which is very different. We then discuss the VLBI problem transcription we have created in §4.2 and provide examples to illustrate how the components behave. We then discuss several interesting

results using this framework in §4.4, discuss how they motivate future work and draw conclusions in §4.5.

4.2 Problem Formulation

This work was inspired by the problem statement of the 8th Global Trajectory Optimization Competition (GTOC-8), which challenged mission designers to observe multiple distant radio sources with VLBI using a fleet of 3 spacecraft, first propelled with some high chemical impulse out of a parking Low Earth Orbit (LEO) onto science trajectories using low thrust, with the opportunity for lunar gravity assists to make dramatic orbit plane changes while saving fuel [49]. At a given epoch, an observation of a radio source may be considered taken if the normal vector to the triangle formed by the 3 spacecraft points within 0.1 degrees of the heliocentric position vector to the distant radio source. The performance index for this problem aimed to maximize the number of observations taken, while also rewarding several factors including: observations of targets with lower declination, triangles with a larger minimum altitude (a metric for the interferometric baseline), and observations of the same target (up to 3 repeat observations are rewarded). The target set contains 420 radio sources, distributed all across the celestial sphere. The explicit GTOC-8 performance index (which is to be maximized) follows:

$$J = \sum_{N_{\text{obs}}} Ph(0.2 + \cos^2\delta) \tag{4.1}$$

where N_{obs} is the total number of observations taken, P is the multiplier that rewards up to 3 repeat observations of a target, h is the minimum altitude of the observation triangle, and δ is the target's declination. The position unit vector to the target is given by

$$\hat{s} = \cos\delta\cos\alpha\,\hat{x} + \cos\delta\sin\alpha\,\hat{y} + \sin\delta\,\hat{z} \tag{4.2}$$

where α and δ are the right ascension and declination of a radio source.

At a given epoch, any 3 spacecraft have the position vectors $\vec{r}_1$, $\vec{r}_2$, $\vec{r}_3$. The unit normal vector to the plane formed by the 3 spacecraft is then:

$$\hat{n} = \frac{(\vec{r}_2 - \vec{r}_1) \times (\vec{r}_3 - \vec{r}_1)}{|(\vec{r}_2 - \vec{r}_1) \times (\vec{r}_3 - \vec{r}_1)|} \tag{4.3}$$

One must also consider the anti-normal vector, which is simply $-\hat{n}$. An observation of a target may be taken when $\hat{s} \cdot \hat{n} = 1$ or $\hat{s} \cdot -\hat{n} = 1$, that is, when the angle θ between $\hat{s}$ and $\mp\hat{n}$ is 0.0°. There is however a maximum 0.1°allowed pointing error.

The dynamics for the spacecraft are two-body Keplerian, but with an added low thrust term with a maximum thrust of 0.1 Newtons.

$$\ddot{\vec{r}} = -\mu \frac{\hat{r}}{r^3} + \frac{\vec{T}}{m} \tag{4.4}$$

where μ is the gravitational parameter of the Earth, $\vec{T}$ is the spacecraft's thrust vector, and m is its mass.

We used this formulation as a motivational starting point, but ultimately created a very different trajectory optimization problem. In our work, we sought to understand trades not captured in the GTOC-8 problem formulation including the trade between fleet size and number of observations, and the relationship between multiple competing cost functions (performance indices). We propagate trajectories according to two-body Keplerian dynamics in the heliocentric regime (where the interferometry baseline is very long due to the size of the spacecrafts' orbits) with semi-major axes on the order of 1AU. We parametrize a trajectory into phases, where a phase is defined by an initial and final state, a propagation time, an observation target, and a single impulsive maneuver performed immediately after the observation of the target. The Keplerian initial state for a spacecraft's first phase is given explicitly in the decision vector while the rest are omitted. The trajectory of a spacecraft is thus:

$$X_{sc} = [X_{0_{\text{kep}}}, \vec{t}, \{\Delta\vec{V}_{ij}\}] \tag{4.5}$$

where $X_{0_{\text{kep}}} = [a, e, i, \Omega, \omega, \nu]$ is the initial state of the spacecraft given by the classical orbit elements, $\vec{t} = [t_1, t_2, ... t_O]$ is the vector of propagation times to each of $\mathbf{O}$ subsequent VLBI observations, and $\{\Delta\vec{V}\} = [\Delta\vec{V}_1, \Delta\vec{V}_2 ..., \Delta\vec{V}_O]$ is the set of $\Delta\vec{V}$ maneuvers performed impulsively

immediately after one observation is taken to target the trajectory towards the next observation. We frame the maneuvers in spherical coordinates with $\Delta\vec{V}_i = [|\Delta\vec{V}_i|, \alpha_i, \delta_i]$ where α and δ are the right ascension and declination of the maneuver and $|\Delta V_i|$ is its magnitude. A single phase of each trajectory begins with an initial Keplerian state, propagates analytically for some duration, arrives at a final Keplerian state at which point an observation is taken, then a maneuver is applied. Note that for a mission with O observations, there are $O-1$ maneuvers. The Keplerian initial state has 6 elements, the $\vec{t}$ component has O elements, and the $\{\Delta\vec{V}\}$ component has $3(O-1)$ elements. Finally, the decision vector describing the trajectories of the entire fleet is a concatenation of the decision vectors of each individual spacecraft within the fleet.

$$\vec{X}_{\text{fleet}} = [\vec{X}_{\text{sc}_1}, \vec{X}_{\text{sc}_2}, ..., \vec{X}_{\text{sc}_S}] \tag{4.6}$$

Thus the decision vector for a fleet with S spacecraft with O epochs at which an observation is taken has a cardinality of:

$$C_{IL} = S(6 + O + 3(O-1)) = S(3 + 4O) \tag{4.7}$$

The goal for a mission with a given number of spacecraft and targets is to minimize the total maneuver cost.

$$J(\vec{X}_{\text{fleet}}) = \sum_{i=1}^{N_S} \sum_{j=1}^{N_{O-1}} |\Delta\vec{V}_{ij}| \tag{4.8}$$

An observation of a target is taken when $\mp\hat{s} \cdot \hat{n} = 1$. This sign dependence in creates a piecewise constraint. So to enable consideration of both the normal and anti-normal to the VLBI triangle for observations while maintaining a smooth constraint function, we instead use the following equation, where p_{err} is the pointing error.

$$p_{\text{err}} = 1 - (\hat{s} \cdot \hat{n})^2 \tag{4.9}$$

A 0.1° pointing error equates to a value of $p_{\text{err}} \approx 3.046\text{e}{-6}$, a value which we will call $p_{\max}$. The angle between $\hat{n}$ and $\hat{s}$ will always vary between 0-180°, and this smooth nonlinear constraint function evaluates to zero in either case, preventing a fork in the optimization. We now have a geometric inter-spacecraft coordination constraint.

This coordination constraint in Eq. 4.9 is applied at each observation taken. We can frame this as a vector of constraints which must always be less than or equal to zero to ensure all observations are taken.

$$F_{O\times 1}(\vec{X}_{\text{fleet}}) = 1 - p_{\max} - (\hat{s} \cdot \hat{n}(\vec{X}_{\text{fleet}}))^2 \tag{4.10}$$

Finally, the two-body dynamics of the problem lack the continuous thrust term of the GTOC-8 problem due to this being an impulsive burn problem, with burns only applied once along a phase after each observation. The spacecraft's mass is also neglected.

$$\ddot{\vec{r}} = -\mu_{\odot}\frac{\vec{r}}{r^3} \tag{4.11}$$

where $\mu_{\odot}$ is the gravitational parameter of the Sun. Fig. 4.1 gives an illustration of the trajectory parameterization in physical space.

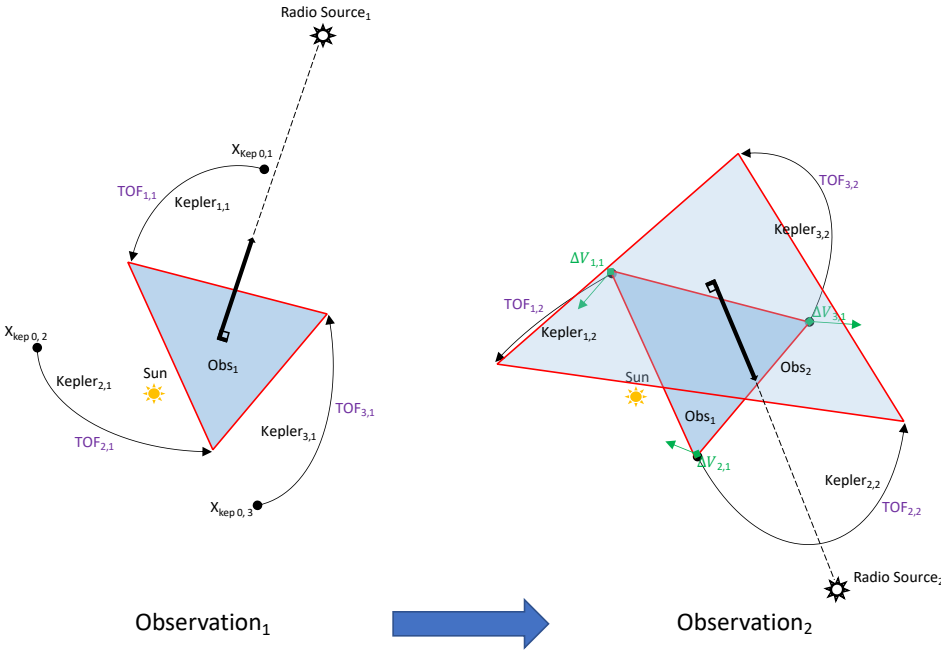


Figure 4.1: Example of the VLBI trajectory optimization problem with a fleet of three spacecraft making two consecutive observations.

In summary, given:

$$\begin{aligned} \ddot{\vec{r}} &= -\mu_{\odot}\frac{\vec{r}}{r^3} \\ \vec{X}_{\text{fleet}} &= [\vec{X}_{\text{sc}_1},\, \vec{X}_{\text{sc}_2},\, ...,\, \vec{X}_{\text{sc}_S}] \\ J(\vec{X}_{\text{fleet}}) &= \sum_{i=1}^{N_S}\sum_{j=1}^{N_{O-1}} |\Delta\vec{V}_{ij}| \\ F_{O\times 1}(\vec{X}_{\text{fleet}}) &= 1 - p_{\max} - (\hat{s}\cdot\hat{n}(\vec{X}_{\text{fleet}}))^2 \end{aligned} \tag{4.12}$$

Solve:

$$\vec{X}^*_{\text{fleet}} \rightarrow \min(J(\vec{X}_{\text{fleet}})) \text{ subject to: } F_{O\times 1}(\vec{X}_{\text{fleet}}) \leq 0_{O\times 1} \tag{4.13}$$

4.3 Outer-Loop Transcription

The outer-loop of this work transcribes a multi-spacecraft multi-target trajectory optimization problem as a decimal integer chromosome. To capture all the information about discretizing the trajectory optimization problem to be passed to the inner-loop, the following details must be encoded within the chromosome structure.

1) Maximum number of spacecraft in the fleet.

2) Number of epochs at which a VLBI observation of a radio source may be taken.

3) The sequence of radio source targets.

4) For each target in the sequence, a trio of spacecraft within the fleet which will, at the epoch of observation, form a triangle whose normal or anti-normal vector must point at the target. This approach is generally extensible for MTSP i.e. "which salesmen visit which town?". A chromosome thus has the structure:

$$X_{OL} = \underbrace{\boxed{N_S}\boxed{N_O}}_{\text{Header}}\underbrace{\boxed{S_{1,1}}\boxed{S_{2,1}}\boxed{S_{3,1}}\boxed{T_1}}_{\text{Observation 1}}\underbrace{\boxed{S_{1,2}}\boxed{S_{2,2}}\boxed{S_{3,2}}\boxed{T_2}}_{\text{Observation 2}}...\underbrace{\boxed{S_{s,o}}\boxed{S_{s,o}}\boxed{S_{s,o}}\boxed{T_o}}_{\text{Observation O}} \tag{4.14}$$

where N_S is the maximum number of spacecraft allowed in the fleet, N_O is the maximum number of observation epochs allowed. $S_{1,j}$, $S_{2,j}$, $S_{3,j}$ and T_j are the integers representing each of the 3 spacecraft in the fleet which take part in the observation of target T_j. The size of the

chromosome is determined entirely by the two header parameters. Assuming that only one observation per epoch may take place, the length of a chromosome is $C_{OL} = 2 + 4N_O$. However, by introducing the opportunity for multiple observations at one epoch, the length instead becomes $C_{OL} = 2 + 4\lfloor \frac{N_S}{3} \rfloor N_O$. The outer-loop's genetic algorithm generates the initial population at random, then subjects its members to crossover and mutation based on their fitness. These processes consistently create chromosomes which do not transcribe a real trajectory optimization problem, so they must be pruned into their reduced "legal" form before they can be executed by the inner-loop. This is why the header contains maxima for the number of observation epochs and spacecraft, rather than their actual values (derived through pruning). The rules governing the legality of the chromosome include the following:

1) An observation is taken by a trio of spacecraft, each represented by a unique integer value; no repetition. For example, the spacecraft trio sequence $\boxed{1}\boxed{2}\boxed{3}$ is valid but $\boxed{2}\boxed{2}\boxed{3}$ is invalid.

2) One spacecraft may only take part in one observation per epoch. For example, if a candidate chromosome contains 6 spacecraft, it has an observation multiplicity value of 2. A given observation epoch may then have 2 observations, encoded as:

$$\boxed{S_{1,1}}\boxed{S_{2,1}}\boxed{S_{3,1}}\boxed{T_1}\boxed{S_{1,2}}\boxed{S_{2,2}}\boxed{S_{3,2}}\boxed{T_2} = \boxed{1}\boxed{3}\boxed{5}\boxed{140}\boxed{2}\boxed{4}\boxed{6}\boxed{16} \tag{4.15}$$

In the example in Eq. 4.15, spacecraft 1, 3, and 5 are tasked with forming the interferometer triangle which observes target 140 while at the same epoch, spacecraft 2, 4 and 6 observe target 16. If instead the sequence read:

$$\boxed{S_{1,1}}\boxed{S_{2,1}}\boxed{S_{3,1}}\boxed{T_1}\boxed{S_{1,2}}\boxed{S_{2,2}}\boxed{S_{3,2}}\boxed{T_2} = \boxed{1}\boxed{3}\boxed{5}\boxed{140}\boxed{1}\boxed{4}\boxed{6}\boxed{16} \tag{4.16}$$

the latter spacecraft trio sequence would be treated as invalid because spacecraft 1 may only take part in one observation at a time, and the observation of target 16 would be pruned from the final candidate chromosome. When the reduction process occurs, every $\lfloor \frac{N_S}{3} \rfloor$ observation blocks of the crude chromosome will be interpreted as simultaneous in the bookkeeping.

3) While the position vector to each distant radio source is mapped to a unique static integer ID in a 'Universe' structure, the integer IDs of spacecraft in a candidate chromosome are dynamic:

their final values change depending on the pruning conducted according to the first two rules. In general, neither the number of observation targets nor the number of spacecraft in the final chromosome match their pre-pruned quantities. For example, if after the first two rules are applied and the chromosome is:

$$\boxed{N_S}\boxed{N_O}\boxed{S_{1,1}}\boxed{S_{2,1}}\boxed{S_{3,1}}\boxed{T_1}\boxed{S_{1,2}}\boxed{S_{2,2}}\boxed{S_{3,2}}\boxed{T_2} = \boxed{7}\boxed{2}\boxed{1}\boxed{3}\boxed{5}\boxed{140}\boxed{2}\boxed{4}\boxed{7}\boxed{16} \tag{4.17}$$

we note that in this posed 7-spacecraft / 2-epoch problem, there are only 6 unique spacecraft IDs. To reduce these integer IDs to their legitimate values, we sort them, purge any duplicate values, and assign them new values based on their index in the unique-valued sorted list.

4) An outer-loop chromosome is of fixed size C_{OL} for a given (outer-loop) problem. The first three reduction rules allow many opportunities to delete genes from the chromosome. The final means of deletion is given by the null gene transcription, which is applied to both the target IDs and spacecraft IDs. Whenever either a target or spacecraft ID is generated it is given a 50/50 chance of being lesser or greater than the maximum number of possible targets or spacecraft in the candidate pool respectively. The purpose of the null gene transcription is to allow the insertion or deletion of genes during the crossover and mutation steps. This technique was applied in [29] to vary the length of flyby targets in an interplanetary cruise trajectory. Here, we leverage the capability to insert or delete spacecraft from the fleet as well as radio source targets. The chromosome is stored in a "Fleet" structure along with the reduced chromosome, and an array of integers which encodes the actual (reduced, non-null) number of observations at each of the N_O epochs in the crude chromosome. The sum of the values in this "observations per epoch" array gives the number of actual observations in the reduced chromosome, acting as a bookkeeping variable determining when the inner-loop must propagate the trajectories to a new epoch. In the reduced chromosome then, the value of N_O reduced to the number of elements in the observation bookkeeping array which are nonzero (i.e. which epochs have more than zero observations). Similarly, the value of N_S in the reduced chromosome is changed to the length of the sorted unique array of spacecraft IDs.

To visualize the overall chromosome reduction process, consider the following crude chromo-

some:

$$X_{OL_{\text{crude}}} = \boxed{7}\boxed{2}\boxed{1}\boxed{3}\boxed{5}\boxed{140}\boxed{2}\boxed{2}\boxed{7}\boxed{19}\boxed{2}\boxed{5}\boxed{7}\boxed{13}\boxed{2}\boxed{3}\boxed{1} \tag{4.18}$$

Here we have a fleet with a maximum size of 7 spacecraft and maximum of 2 observation epochs. Since $\lfloor \frac{7}{3} \rfloor = 2$, we may have at most two observations per epoch. We then parse this chromosome as shown below.

$$X_{OL_{\text{crude}}} = \underbrace{\boxed{7}\boxed{2}}_{\text{[Header]}} \quad \underbrace{\boxed{1}\boxed{3}\boxed{5}\boxed{140}}_{\text{[Observation 1, Epoch 1]}} \quad \underbrace{\boxed{2}\boxed{2}\boxed{7}\boxed{19}}_{\text{[Observation 2, Epoch 1]}} \quad \underbrace{\boxed{2}\boxed{5}\boxed{7}\boxed{13}}_{\text{[Observation 1, Epoch 2]}} \quad \underbrace{\boxed{2}\boxed{3}\boxed{1}\boxed{600}}_{\text{[Observation 2, Epoch 2]}} \tag{4.19}$$

Immediately, observation 2 at epoch 2 is pruned because there are 420 targets to choose from and 600 is therefore a null gene.

$$X_{OL} = \underbrace{\boxed{7}\boxed{2}}_{\text{[Header]}} \quad \underbrace{\boxed{1}\boxed{3}\boxed{5}\boxed{140}}_{\text{[Observation 1, Epoch 1]}} \quad \underbrace{\boxed{2}\boxed{2}\boxed{7}\boxed{19}}_{\text{[Observation 2, Epoch 1]}} \quad \underbrace{\boxed{2}\boxed{5}\boxed{7}\boxed{13}}_{\text{[Observation 1, Epoch 2]}} \tag{4.20}$$

Now, the spacecraft trio for observation 2 at epoch 1 is illegal as spacecraft 2 cannot form a triangle with itself and spacecraft 7.

$$X_{OL} = \underbrace{\boxed{7}\boxed{2}}_{\text{[Header]}} \quad \underbrace{\boxed{1}\boxed{3}\boxed{5}\boxed{140}}_{\text{[Observation 1, Epoch 1]}} \quad \underbrace{\boxed{2}\boxed{5}\boxed{7}\boxed{13}}_{\text{[Observation 1, Epoch 2]}} \tag{4.21}$$

We are now left with 2 observation epochs each with one legal observation. However, to determine how many actual spacecraft the fleet will have, we perform the uniqueness sort of the spacecraft integers.

$$\text{unique sorted spacecraft IDs} = [1,\ 2,\ 3,\ 5,\ 5,\ 7] \rightarrow [1,\ 2,\ 3,\ 5,\ 7] \rightarrow [1,\ 2,\ 3,\ 4,\ 5] \tag{4.22}$$

Thus the actual fleet whose trajectories the inner-loop solver shall optimize contains 5 spacecraft and 2 observation epochs each with one observation. The final reduced chromosome is:

$$X_{OL} = \underbrace{\boxed{5}\boxed{2}}_{\text{[Header]}} \quad \underbrace{\boxed{1}\boxed{3}\boxed{5}\boxed{140}}_{\text{[Observation 1, Epoch 1]}} \quad \underbrace{\boxed{2}\boxed{4}\boxed{5}\boxed{13}}_{\text{[Observation 1, Epoch 2]}} \tag{4.23}$$

Note that multiple different crude chromosomes could map to the same reduced chromosome. This feature is elaborated on in further detail in the outer-loop solver discussion. In a problem

where the size of the chromosome is fixed, the null gene technique allows us to study problems of variable size while preserving the fixed structure, i.e., without the need to come up with clever ways to deal with asymmetric length vectors. In a problem with say 3 maximum targets and 2 maximum observation epochs, but absent the possibility of null genes, every chromosome evaluated would have the same number of spacecraft and targets. It would not be readily possible to vary the number of targets or spacecraft and extract the Pareto front between these parameters. To further illustrate the need for the null gene technique, consider the following genetic crossover.

$$\text{Parent}_{1,\text{ crude}} = \underbrace{\boxed{7}\boxed{2}}_{\text{[Header]}} \quad \underbrace{\boxed{1}\boxed{3}\boxed{5}\boxed{140}}_{\text{[Observation 1, Epoch 1]}} \quad \underbrace{\boxed{2}\boxed{2}\boxed{7}\boxed{19}}_{\text{[Observation 2, Epoch 1]}} \quad \underbrace{\boxed{2}\boxed{5}\boxed{7}\boxed{13}}_{\text{[Observation 1, Epoch 2]}} \quad \underbrace{\boxed{2}\boxed{3}\boxed{1}\boxed{600}}_{\text{[Observation 2, Epoch 2]}} \tag{4.24}$$

$$\text{Parent}_{2,\text{ crude}} = \underbrace{\boxed{7}\boxed{2}}_{\text{[Header]}} \quad \underbrace{\boxed{2}\boxed{5}\boxed{14}\boxed{18}}_{\text{[Observation 1, Epoch 1]}} \quad \underbrace{\boxed{2}\boxed{4}\boxed{7}\boxed{109}}_{\text{[Observation 2, Epoch 1]}} \quad \underbrace{\boxed{5}\boxed{6}\boxed{7}\boxed{22}}_{\text{[Observation 1, Epoch 2]}} \quad \underbrace{\boxed{2}\boxed{4}\boxed{6}\boxed{400}}_{\text{[Observation 2, Epoch 2]}} \tag{4.25}$$

Assume that we end up with the following child chromosome, created by choosing each gene at random with 50% probability from either parent.

$$\text{Child}_{\text{ crude}} = \underbrace{\boxed{7}\boxed{2}}_{\text{[Header]}} \quad \underbrace{\boxed{2}\boxed{3}\boxed{14}\boxed{140}}_{\text{[Observation 1, Epoch 1]}} \quad \underbrace{\boxed{2}\boxed{4}\boxed{7}\boxed{19}}_{\text{[Observation 2, Epoch 1]}} \quad \underbrace{\boxed{2}\boxed{6}\boxed{7}\boxed{13}}_{\text{[Observation 1, Epoch 2]}} \quad \underbrace{\boxed{2}\boxed{4}\boxed{6}\boxed{400}}_{\text{[Observation 2, Epoch 2]}} \tag{4.26}$$

Given that there are 420 possible radio sources to observe and there may be at most 7 spacecraft in the fleet, gene "14" in observation 1, epoch 1 is a null spacecraft gene. During the reduction process, the observation 1, epoch 1 block is "deleted" (ignored) by the inner-loop. Additionally, the two observations at epoch 2 share spacecraft number 6, so the latter is deleted despite it now having a non-null target gene of "400". With these two blocks pruned, the spacecraft IDs are reduced to reveal the actual number of spacecraft in the fleet whose trajectories will be optimized by the inner-loop.

$$\text{Child} = \underbrace{\boxed{4}\boxed{2}}_{\text{[Header]}} \quad \underbrace{\boxed{1}\boxed{2}\boxed{4}\boxed{19}}_{\text{[Observation 1, Epoch 1]}} \quad \underbrace{\boxed{1}\boxed{3}\boxed{4}\boxed{13}}_{\text{[Observation 1, Epoch 2]}} \tag{4.27}$$

The child chromosome now has 3 fewer spacecraft and 2 fewer observations than allowed by the bounds of the outer-loop problem, resulting in a very different trajectory optimization problem for the inner-loop. Note that here we've omitted the mutation operation step for simplicity, but in the algorithm, mutation occurs before reduction. This includes a 10% chance that each gene in the crude child chromosome mutates, but a further check against the archive of past solutions to ensure *unique* exploration of the search space. The NSGA-II algorithm uses elitism to preserve the fittest members of the population, seeding children from predominantly elite parents (those on the Pareto front). However, without mutation, the algorithm would tend to get stuck in a sub-optimal solution space. Mutation preserves the diversity of the population and furthers the exploration of the search space. The rules for chromosome reduction impact the MOMA HOCP structure and this effect is illustrated in Fig. 4.2.

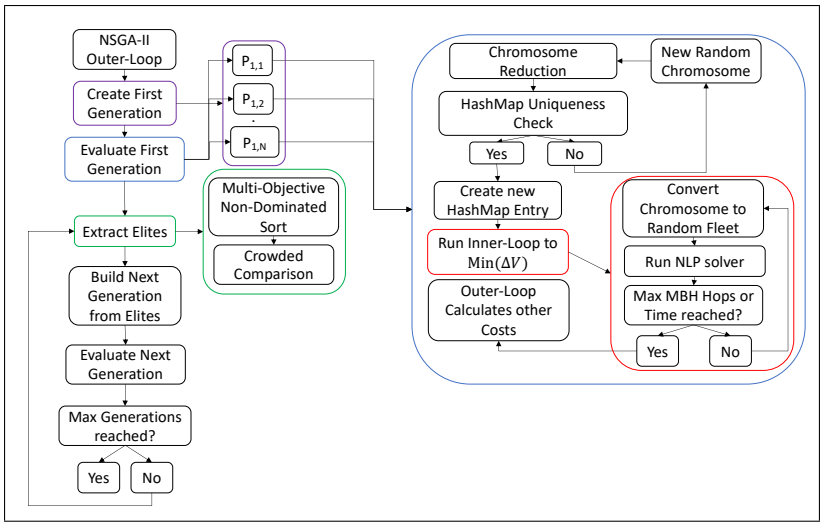


Figure 4.2: Detailed flow chart of inner-loop and outer-loop algorithm interaction.

The search space of this problem is vast, well beyond the computing resources available to the authors during the course of this work. Even for a version of the problem orders of magnitude smaller, a brute force search for solutions would be untenable. While genetic algorithms are proven superior resources to brute force in large combinatorial spaces, their stochastic search process must be kept as efficient as possible. One aspect of this efficiency is preventing the algorithm from re-

evaluating a chromosome it has already evaluated. There are many ways to check for re-evaluation, the most straightforward of which is to log the chromosome's sequence in a text file. Every time a new chromosome is created, the text file may be searched for the new sequence to see if it has been tried before and if so, generate a new trial chromosome. The check must be repeated until an unevaluated chromosome is created or until it is deemed impossible to avoid re-evaluation (though the latter condition is unlikely to be reached in large decision spaces like this one).

However, as the list of previously evaluated chromosomes lengthens, the lookup time for file reading becomes slow compared to other lookup alternatives. If the list contains on the order of 5,000 elements, then performing a lookup from a key-value pair hashmap is multiple times quicker. Hashmap lookup time relative to the linear lookup time of file reading becomes exponentially quicker as the list grows from 5,000 to 10,000 to 100,000 elements and so on. To check for previous evaluation, we create a hashmap with a key-value pair corresponding to the chromosome sequence and a boolean `is_tried` respectively. Specifically the key is created as follows.

1) A crude chromosome is stripped to its reduced form, e.g.:

$$X_{OL} = \boxed{5}\boxed{2}\boxed{1}\boxed{3}\boxed{5}\boxed{140}\boxed{2}\boxed{4}\boxed{5}\boxed{13} \tag{4.28}$$

2) The "observations per epoch" integer array is appended to it, creating an augmented reduced chromosome. This is necessary to distinguish otherwise identical chromosomes with different observation multiplicities. Continuing using the example in step 1, we have:

$$X_{OL_{\text{aug}}} = \boxed{5}\boxed{2}\boxed{1}\boxed{3}\boxed{5}\boxed{140}\boxed{2}\boxed{4}\boxed{5}\boxed{13}\boxed{1}\boxed{1} \tag{4.29}$$

since there are two epochs and two observations, yielding an "observations per epoch" vector of $\boxed{1}\boxed{1}$.

3) 9 bits are required to store an integer as high as 420 (the maximal value of a target's ID). Therefore each decimal integer in the augmented reduced chromosome is converted to a 9-bit binary string (with the rightmost bit representing 2^0), yielding a sequence of bit-strings which may

become quite large as the number of spacecraft and observation epochs increases:

$$X_{OL_{\text{aug}}} = \underbrace{\boxed{5}\boxed{2}}_{\text{[Header]}} \quad \underbrace{\boxed{1}\boxed{3}\boxed{5}\boxed{140}}_{\text{[Observation 1, epoch 1]}} \quad \underbrace{\boxed{2}\boxed{4}\boxed{5}\boxed{13}}_{\text{[Observation 1, epoch 2]}} \quad \underbrace{\boxed{1}\boxed{1}}_{\text{[number of obs. each epoch]}} \tag{4.30}$$

$$\begin{aligned}\text{Key} = [&\underbrace{\text{`000000101'}}_{\boxed{5}}, \underbrace{\text{`000000010'}}_{\boxed{2}},\\ &\underbrace{\text{`000000001'}}_{\boxed{1}}, \underbrace{\text{`000000011'}}_{\boxed{3}}, \underbrace{\text{`000000101'}}_{\boxed{5}}, \underbrace{\text{`010001100'}}_{\boxed{140}},\\ &\underbrace{\text{`000000010'}}_{\boxed{2}}, \underbrace{\text{`000000100'}}_{\boxed{4}}, \underbrace{\text{`000000101'}}_{\boxed{5}}, \underbrace{\text{`000001101'}}_{\boxed{13}},\\ &\underbrace{\text{`000000001'}}_{\boxed{1}}, \underbrace{\text{`000000001'}}_{\boxed{1}}]\end{aligned} \tag{4.31}$$

4) All the 9-bit values presented as character strings are concatenated into one character string. This creates a unique key for a given crude chromosome. Once the outer-loop creates this chromosome, the map (stored in memory to be referenced during the execution of the outer-loop) is updated with the entry:

$$M\langle \text{Key}, \text{Value}\rangle = \langle \text{`000000101000000010...000000001'}, \texttt{true}\rangle \tag{4.32}$$

Thus whenever a new crude chromosome is created, it is reduced, its key is generated, and the outer-loop checks the HashMap stored in memory for the existence of the key.

The HashMap archive utility yields significant savings during the MOMA HOCP execution. On a test archive comparison using an archive of 10,000 keys, the HashMap required an average lookup time of 15 milliseconds (ms) versus an average text file lookup time of 171 ms. This HashMap geometrically outperforms the text file lookup time as the number of entries increases. In this test case, the text file archive lookup is more than 11 times slower than the HashMap. Over the course of 10,000 lookups, this average 156 ms difference yields a total of over 26 minutes of runtime saved. In this chapter, the maximum inner-loop runtime is set to 30 minutes, thus almost an entire inner-loop optimization time is saved be using the HashMap lookup. At worst, if the performance offset

Entries	File Lookup (ms)	HashMap Lookup (ms)
300	6.3	6.4
1,000	19.1	6.9
10,000	171.7	15.1

Table 4.1: Average archive lookup durations as a function of entries in the archive.

remained constant, this would amount to a savings of almost 44 hours if the outer-loop evaluated one million solutions. Table 4.1 shows the performance discrepancy as a function of the number of entries in the archive. For a small number of archive entries, the text file lookup is faster. In an archive with 300 entries, the average lookup time between the two methods is approximately equal. Beyond this value, the the text file approach becomes geometrically slower.

The purpose of performing a uniqueness check using an archive is to ensure that the search space is efficiently explored, with no repeat evaluations of identical chromosomes. Each time a new chromosome is created, there is a non-negligible chance that it is a duplicate of one which has already been solved by the inner-loop. We catalogued the incidence of duplicate chromosome discovery during the outer-loop execution for a VLBI outer-loop problem with a population size of 72 chromosomes that evolve over 100 generations. The evolutionary process was found to generate 15.17 chromosomes per generation which already existed in the HashMap archive, before finding one that was unique, for a total of 1,517 duplicate chromosomes over the entire program execution.

Thus, to explore the same number of unique chromosomes in the search space without invoking the HashMap archive to check for uniqueness, the analyst should expect to explore an excess of 1500 repeat chromosomes. In the VLBI problem presented in this dissertation, the inner-loop execution time is capped at 30 minutes. This amounts to a total runtime savings of 758.5 hours when the HashMap is invoked, assuming every chromosome's trajectory optimization problem is solved in serial. Using the parallel computing capabilities of the CU Boulder Summit cluster available during this dissertation (24 CPUs), the runtime to explore an identical number of unique search space points with the HashMap is 6.25 days versus 7.60 days without; a 20% savings. These data

demonstrate that the HashMap archive is an essential component of the MOMA HOCP framework. Fig. 4.3 illustrates the number of duplicate chromosomes created per generation as well as the total duplicates found over the course of the outer-loop run. For the first 10 generations (720 chromosomes created), the outer-loop has no trouble finding unique chromosomes, but it then quickly encounters duplicates as the HashMap archive begins to populate. The growth in the total number of duplicates encountered is roughly linear. This is due to the massive size of the outer-loop search space. It will not become prohibitively difficult to find unique chromosomes within the scope of this dissertation, so we do not observe an exponential increase in the rate of duplicate chromosome discovery.

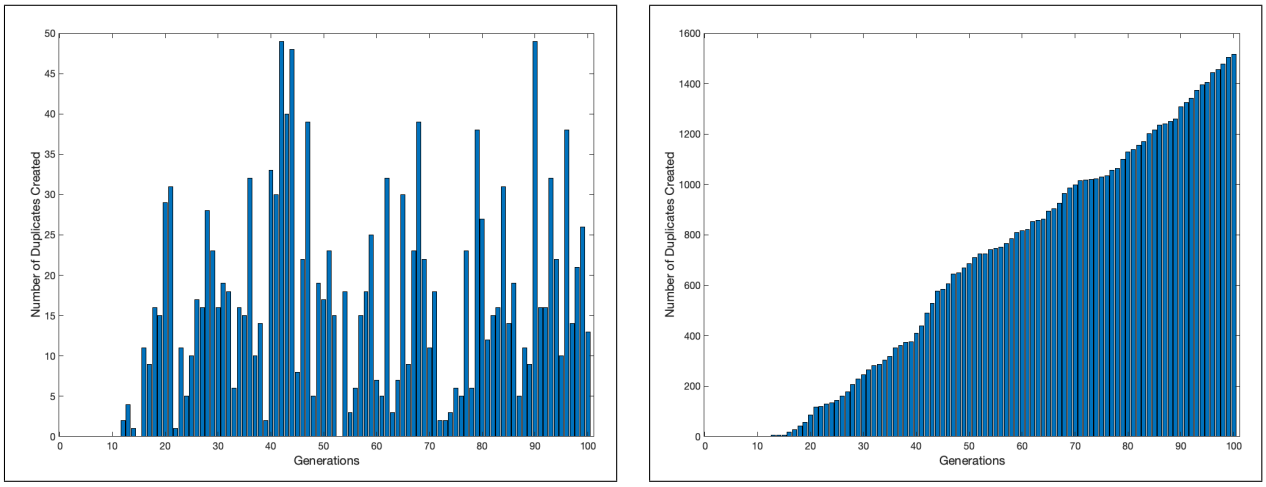


Figure 4.3: Left: Number of duplicate chromosomes created at each generation. Right: total duplicate chromosomes discovered as a function of generations.

Outer-Loop Coordination Objectives

For this benchmark problem, we sought to explore the trade space between 5 objectives. The outer-loop ranks chromosomes with **lower-valued** cost functions as dominant, minimizing the cost functions of each solution mission. Thus cost functions must be scaled accordingly based on whether the analyst desires to maximize or minimize a key quantity. The "primary" objective is the one optimized explicitly by the inner-loop.

$$J_{\Delta V_{\text{tot}}} = \sum_{i=1}^{N_S} \sum_{j=1}^{N_{O-1}} |\Delta \vec{V}_{ij}| \tag{4.33}$$

is the total ΔV for the fleet, computed directly by summing up each inner-loop $|\Delta\vec{V}|$ magnitude decision variable for each spacecraft at each observation epoch (apart from the first one, at which no $|\Delta\vec{V}|$ is applied).

"Secondary" objectives are calculated by the outer-loop after the inner-loop optimization is complete.

$$J_{\text{TOF}} = \sum_{i=1}^{N_O} t_i \tag{4.34}$$

where $t_i \in \vec{t}$ is each TOF decision variable in the inner-loop decision vector.

$$J_{SC} = \vec{X}_{OL}(1) \tag{4.35}$$

is the total number of spacecraft in the fleet.

$$J_{N_{\text{targ}}} = \frac{1}{\text{length}(\vec{X}_{OL} == \text{“target”})} \tag{4.36}$$

is the inverse of the total number of targets observed by the fleet. This causes the number of targets in a mission to be maximized.

$$J_{M_{\text{max}}} = \frac{1}{\max(\text{number of observations per epoch})} \tag{4.37}$$

is the inverse of the maximum number of observations at an epoch, causing the observation multiplicity to be maximized.

Rather than amalgamate these 5 individual cost functions into one, the non-dominated sort step in the NSGA-II algorithm assigns each solution mission a non-domination rank based on its vector of cost functions. As mentioned previously, solution A dominates solution B if all of solution A's costs are less than or equal to those of solution B AND at least one of its cost functions is less than the corresponding cost for B.

Inner-Loop Parameter	Value
Maximum MBH Hops	100
Maximum Run Time (minutes)	30
Maximum NLP Solver Iterations	50,000
Cost Function	$J_{\Delta V_{\text{tot}}}$
Initial Semimajor Axis bounds (AU)	[0.5, 1.5]
Initial Eccentricity bounds	[0.0, 0.8]
Initial Inclination bounds (deg)	[0.0, 45.0]
Initial RAAN bounds (deg)	[0.0, 360.0]
Initial AOP bounds (deg)	[0.0, 360.0]
Initial True Anomaly bounds (deg)	[0.0, 360.0]
Phase TOF bounds (days)	[14, 1000]
ΔV magnitude bounds (km/s)	[0.0, 0.3]

Table 4.2: Options parameters for the inner-loop of the MOMA HOCP scheme.

4.4 Results

Since each outer-loop chromosome encodes an inner-loop trajectory optimization problem, which itself may take a long time to optimize, the inner-loop becomes the bottleneck to traversing a meaningful fraction of the outer-loop search space. During a series of test runs on inner-loop problems of varying complexity, we determined a 30-minute execution time cap for the MBH algorithm would be sufficient to locate the global optimum. A 100-hop (iteration) cap was also set, and the MBH algorithm terminates at whichever condition comes first. There may be cases where a fleet's trajectories may be feasible or optimal if given longer to optimize, but we discovered most solutions of interest could be found within 30 minutes and judged the greater priority to be exploring the outer-loop search space. We set the inner-loop parameters for the study in Table 4.2.

The inner-loop execution is one of two major bottlenecks in the execution of the transcription we present. It must be overcome via substantial distributed computing resources. If the outer-loop problem were as simple sending a 3-spacecraft fleet to conduct N_O unique observations of radio sources in serial, pulling from a pool of N_T targets, the size of the search space is:

$$ {}^{N_T}P_O = \frac{N_T!}{(N_T - N_O)!} \tag{4.38} $$

Outer-Loop Parameter	**Value**
Population Size	72
Number of Workers	24
Generations	100
Maximum Fleet Size	7
Maximum num. Observation Epochs	30
Mutation Rate	10%
Main Objective	$J_{\Delta V_{\text{tot}}}$
Secondary Objective(s)	$[J_{\text{TOF}}, J_{SC}, J_{N_{\text{targ}}}, J_{M_{\text{max}}}]$

Table 4.3: Options parameters for the outer-loop of the MOMA HOCP scheme.

Given the bounds on number of targets and epochs in this problem (420 and 30 respectively), this formula shows that the search space is already enormous. Accounting for all the possible permutations of spacecraft trios and possible observation multiplicity at each epoch, exploring the entire search space by brute force, even with massive computing resources, is absolutely untenable. The best we can do is: apply what computing resources are available, discover interesting results along the way, hypothesize that the patterns we discover hold across different subsets of the global search space, and test this hypothesis in the course of future work.

The University of Colorado at Boulder possesses a distributed computing resource known as the Summit Cluster. We used a 24-core node to execute a 100 outer-loop generation job. For this work, we determined an outer-loop population size of 72 chromosomes and a cap of 100 generations to be the tractable problem size. With 24 cores, 24 inner-loop problems may be run in parallel. At 30 minutes runtime each, with perfect efficiency, this means each generation completes in 90 minutes, yielding a total execution time for the study of 150 hours or 6.25 days. This equates to evaluating 1152 chromosomes every 24 hours. Each outer-loop generation's results are archived as the algorithm proceeds, enabling us to build on results and continue exploring of the search space at any point during the execution of the nominal job. In summary, we set the outer-loop parameters for this study in Table 4.3.

We conceived the study in this work to demonstrate how a novel techniques for multi-objective

optimization, variable fleet size, and observation multiplicity could enhance the ability of mission analysts to pose, solve, and understand multi-spacecraft multi-target exploration problems like this one. The allowable fleet size for this study ranged between 3 and 7 spacecraft. This range captures the minimum proof-of-concept span for these contributions. A candidate VLBI mission is not viable without a minimum of 3 spacecraft due to the observation geometry requirement. Adding a 4^{th} spacecraft increases the opportunity to achieve more observations, by some unknown Δ_{obs}. Adding a 5^{th} to the fleet has a similar effects, though we hypothesize the returns diminish slightly. A fleet of 6 spacecraft should have ample opportunity to image more targets than 5 because not only does it add another observer, but now there are enough spacecraft to allow two observations at a time. This feature may thus reduce the overall duration of the mission, theoretically allowing the fleet to image twice the number of targets as a 3-spacecraft fleet but in the same amount of time. Finally, we allow for the possibility of a 7-spacecraft fleet to estimate the diminishing Δ_{obs} gained by adding just one more spacecraft.

Ideally, we aimed to find missions with static orbits for all spacecraft, requiring no ΔV. Therefore initially, the problem was posed with TOF as the inner-loop cost function and maneuvers were omitted from the transcription. The TOF between encounters was constrained to be 14 days at minimum. This value was selected as an estimate of the turn-around time for the mission operations team to prepare for the next observation. The TOF cost was traded at the outer-loop level against J_{SC} and $J_{N_{\text{targ}}}$. This approach yielded a tenuous Pareto front, where the algorithm struggled to find many solutions at all, let alone solutions with a high ratio of $\frac{J_{N_{\text{targ}}}}{J_{SC}}$. Moreover, the solutions it *did* find contained phase propagation times at the minimum 14 day value. With no other metric for optimality, the inner-loop had minimized cost by directly minimizing the TOF variables in the decision vector. Thus the missions it did find observed very few targets because it is difficult to observe a large quantity of targets in such a short time. It became apparent that in order to find a broader diversity of solutions, maneuvers must be included in the inner-loop transcription. While ideally these maneuvers would optimize to zero, their inclusion in the transcription facilitated the discovery of better solutions, with phase TOF free to relax to its maximum if needed. This change

successfully fleshed out the diversity of the Pareto front, however it lead to the discovery of static orbits (zero magnitude maneuvers) with very long mission durations, often exceeding 10 years. Nevertheless, the forthcoming solutions present a depiction of the optimal solution space for the bounds set on this particular problem.

Fig. 4.4 displays two different cross-sections of the 5-dimensional Pareto front for this study. Blue points represent fleets where one observation was taken at one epoch. Red points indicate fleets where at least one epoch yielded 2 simultaneous observations ($J_{M_{\max}} = \frac{1}{2}$). A 2-dimensional Pareto front will visually present a clear intuitive trade curve as depicted in Fig. 2.2, and similarly each 2-dimensional cross-section of a 3-dimensional Pareto front will also present a smooth curve. However, a Pareto front whose dimensionality D exceeds 3 will not do so as its non-domination pattern exists only in $\mathbb{R}_D$ while we are only able to plot the front in $\mathbb{R}_3$. The results in the right portion of Fig. 4.4 show the best case trade of observed targets versus fleet size, and we quantify this trade in Table 4.4. We display the Pareto front at three different intermediate generations of the algorithm's execution in Fig. 4.4, 4.5 and 4.6 to identify interesting behavioral trends and show how the population evolves over time.

There are 29, 40, and 40 missions on the Pareto fronts for generations 10, 40 and 100 respectively. The number of front members generally increases as the number of generations increases, but this does not always happen. The cap on the front size, per the framework of the NSGA-II algorithm, is 2× the outer-loop population size, which in this case would be 144. Additionally, as more dominant solutions are found over generations, previously dominant solutions are deleted from the front when they are supplanted, as occurred between generations 40 and 100 in this study, leading to a plateau in the number of front members over time.

As the algorithm continues to run, we observe expected trends beginning to take shape. We observe a greater prevalence of solutions with higher observation multiplicity (multiplicity =1 in blue, 2 = red), enabling the imaging of more targets in a given amount of time. The over-arching structure appears to be achieved after 10 generations, with the front generally not expanding beyond its extrema by generation 40 but rather fleshing out its diversity. In the left of Fig. 4.5, several data

points are highlighted to compare how increasing the fleet size to 6 often enables a given number of targets to be imaged in *half* the time as a fleet of size 3-5. After 100 generations we observe that the ratio of targets to spacecraft has been maximized, with a substantial increase in targets observed by the best 3-spacecraft fleet, and the maximum quantity of targets previously observable with no fewer than 7 spacecraft now observed by a fleet of only 6 (Fig. 4.6). Note that all but one 6-spacecraft mission on the front coordinates multiple observations during a single epoch at least once. The 7-spacecraft 16-target solution still exists in the solution set, but it no longer falls upon the non-dominated front as the 6-spacecraft 16-target solution is more dominant because it achieves the same TOF, ΔV, and number of targets using 1 fewer spacecraft. As shown in Fig. 4.12 and outlined Table 4.8, it accomplishes this by imaging 2 targets during its 10^{th} observation epoch, while the 7-spacecraft solution from generation 40 used no observation multiplicity. A dominant 7-spacecraft solution (with a greater number of targets) likely exists in the search space, but was not found during this work. We quantify the Δ_{obs} quantity for the final generation in Table 4.5. After 100 generations, we discover that for fleets ranging in size from 3-6 spacecraft can at best expect to observe an additional two targets per spacecraft added to the fleet. This trend may be very different after 100 additional generations, just as it it varied over the first 100 generations of this study. This trend will continue to be examined in future work.

We also display the frequency of different targets' occurrence along the Pareto front at generation 10 and generation 40 in Fig. 4.8 to show how the final front may appear mostly unchanged over time in terms of its cost functions, but the underlying choices of targets within those optimal solutions changes significantly. It is interesting to observe the frequency of various targets that appear in the sequences of missions along the non-dominated front. Initially, we may expect targets with very similar coordinates to be the easiest to observe in sequence because very little ΔV is required to change the orbit planes of the fleet to observe them all. Similarly, it may be desirable to fly a mission with some targets nearly normal to a given orbit plane while even more targets are nearly anti-normal to the plane. However, with fairly relaxed orbit and TOF parameters, we observe very different behavior, with solution missions observing targets at widely varying

coordinates using zero ΔV.

Finally, we highlight several point solution missions: a simple 3-spacecraft 1-target mission to illustrate the mission concept in physical space (Fig. 4.7), a 6-spacecraft 14-target solution to display a far more complex solution that also leverages multiple observations at one epoch (Fig. 4.10), the most complex mission after 40 generations (7 spacecraft, 16 targets) (Fig. 4.11), and the most complex mission after 100 generations (Fig. 4.12). In Fig. 4.7, the 3 spacecraft trajectories are shown in white, propagating to the states (marked by magenta stars) where the VLBI observation triangle is formed. The translucent triangle formed by their positions contains a vector located at its centroid which is parallel to the position vector of the radio source. The sun is situated at the origin, marked with a yellow star. The mission in Fig. 4.10 uses zero ΔV in exchange for 12.9 years total time of flight and its mission itinerary is given in Table 4.6. All observations are shown in the same figure to illustrate the non-intuitive complexity of missions like this one which the algorithm finds. Observations that occur simultaneously are color-coded as magenta triangles with magenta normal vectors. Fig. 4.11 displays the mission with 7 spacecraft imaging 16 targets with zero ΔV in exchange for 17.5 years total time of flight and its itinerary is outlined in Table 4.7. There are no multi-observation epochs in this solution, but its complexity is otherwise staggering.

Each mission on the Pareto fronts in Fig. 4.4, 4.5, and 4.6 uses zero ΔV, therefore its fleet's trajectories are static orbits. This property ensures that the observations of all targets its sequence will repeat cyclically over a period equal to the duration of the nominal mission, which may add long-term scientific value. The targets' coordinates in each of the 3 highlighted missions are shown in Fig. 4.9, where N_S is the number of spacecraft in the mission and N_O is the number of observations the fleet makes. The targets in these missions' observation sequences are spread out all across the sky. There is considerable overlap between the targets observed by the fleets in Fig. 4.10 and Fig. 4.12, including incidentally the same pair of targets at each mission's multi-observation epoch.

N_S	$N_{\mathbf{Targ}}$	$\frac{J_{N_{\mathbf{targ}}}}{J_{SC}}$	$\Delta_{\mathbf{obs}}$	TOF (years)
3	9	3.0	N/A	14.3
4	10	2.5	1	11.8
5	13	2.6	3	16.8
6	15	2.5	2	20.1
7	16	2.3	1	17.5

Table 4.4: Total targets observed for zero ΔV as a function of fleet size based on data from generation 40 (Fig. 4.5). Δ_{obs} is the number of observations gained by increasing the fleet size by one.

N_S	$N_{\mathbf{Targ}}$	$\frac{J_{N_{\mathbf{targ}}}}{J_{SC}}$	$\Delta_{\mathbf{obs}}$	TOF (years)
3	10	3.3	N/A	11.2
4	12	3.0	2	13.9
5	14	2.8	2	17.4
6	16	2.7	2	17.5
7	10	1.4	—	7.2

Table 4.5: Total targets observed for zero ΔV as a function of fleet size based on data from generation 100 (Fig. 4.6).

Fig. 4.10 Mission Itinerary

Epoch ID	**Spacecraft Trio**	**Radio Source ID**	**Source α°**	**Source δ °**	**TOF (years)**
1	2-4-6	352	347.56319	41.75708	0.3
2	3-4-5	221	321.37855	2.63145	0.07
3	2-3-5	341	203.87887	42.64595	0.4
4	3-4-5	96	84.34929	-28.09646	1.6
5	2-5-6	414	5.34961	76.87223	0.04
6	1-3-4	97	94.89564	-32.6827	0.04
7	2-3-5	350	321.51644	40.91679	1.5
8	1-3-5	58	311.60358	-51.32555	0.96
8	2-4-6	339	178.73763	36.56666	0.96
9	1-3-5	399	151.78999	73.55852	0.7
10	1-2-5	403	268.67363	71.39317	1.47
11	1-5-6	389	260.35891	59.51633	1.97
12	3-4-5	80	245.14647	-42.82787	2.62
13	2-4-6	51	200.69828	-49.92761	1.21

Table 4.6: Itinerary for Fig. 4.10. 6 spacecraft observe 14 unique targets over 12.9 years using zero ΔV with 2 simultaneous observations at epoch 8.

Fig. 4.11 Mission Itinerary

Epoch ID	Spacecraft Trio	Radio Source ID	Source α°	Source δ °	TOF (years)
1	1-2-3	51	200.69828	-49.92761	1.1
2	3-4-6	72	141.98388	-38.5905	1.3
3	3-5-6	22	36.98163	-63.48211	0.7
4	3-4-6	237	125.2422	9.52288	2.0
5	1-2-3	194	48.79947	0.17478	0.5
6	2-4-7	214	248.77494	-1.81304	1.3
7	1-3-4	55	265.90339	-47.6815	0.9
8	2-4-5	206	169.416	-1.98228	1.1
9	1-4-6	200	110.41998	-0.00215	0.4
10	1-5-6	250	259.49626	12.40535	0.8
11	2-4-5	75	179.9135	-36.72425	1.6
12	2-3-4	185	321.02702	-12.50944	1.0
13	2-4-5	43	77.40982	-48.80318	1.2
14	5-6-7	394	358.96459	73.96146	1.6
15	1-2-3	195	59.11799	0.1232	0.9
16	4-5-6	332	88.01375	39.94318	1.1

Table 4.7: Mission Itinerary for Fig. 4.11. 7 spacecraft observe 16 unique targets over 17.5 years using zero ΔV.

Fig. 4.12 Mission Itinerary

Epoch ID	Spacecraft Trio	Radio Source ID	Source α°	Source δ °	TOF (years)
1	2-3-6	276	167.12468	18.58792	0.7
2	1-4-5	400	180.79924	69.38621	0.6
3	1-2-3	271	114.02224	22.72405	1.8
4	3-4-5	341	203.87887	42.64595	1.0
5	2-3-4	143	242.76637	-22.44479	0.3
6	4-5-3	161	73.34137	-9.86491	1.0
7	1-4-5	22	36.98163	-63.48211	2.5
8	1-2-4	148	298.58878	-19.39339	1.4
9	2-3-5	350	321.51644	40.91679	1.2
10	1-3-5	58	311.60358	-51.32555	0.7
10	2-4-6	339	178.73763	36.56666	0.7
11	1-3-5	399	151.78999	73.55852	0.5
12	1-2-5	403	268.67363	71.39317	2.5
13	1-5-6	389	260.35891	59.51633	0.8
14	3-4-5	80	245.14647	-42.82787	1.7
15	2-4-6	51	200.69828	-49.92761	0.6

Table 4.8: Mission Itinerary for the mission in Fig. 4.12.

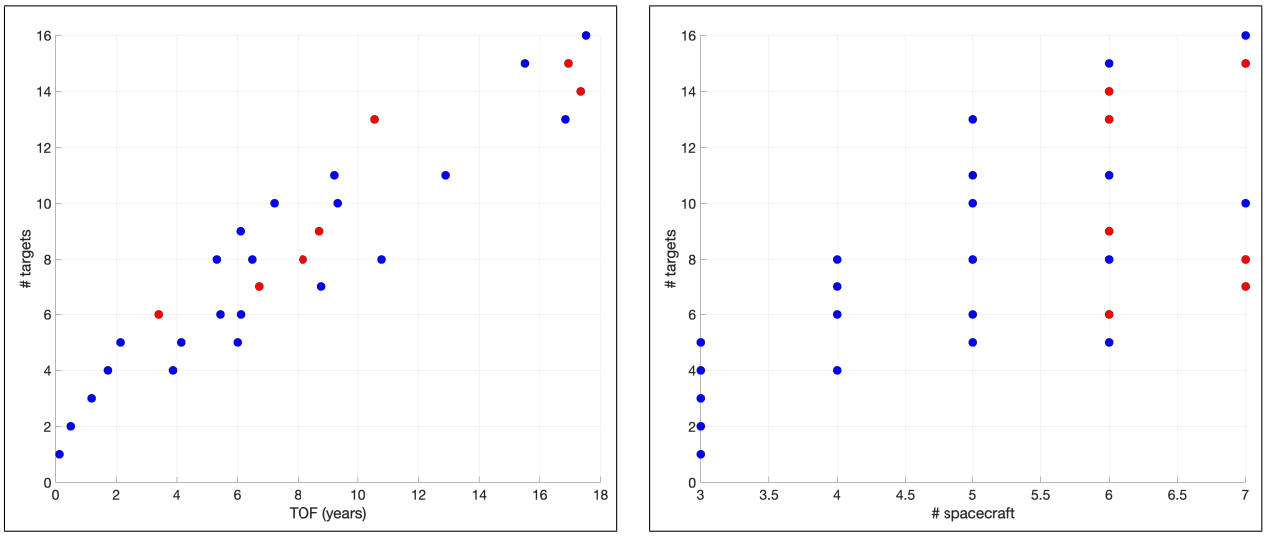


Figure 4.4: Left: Cross-section of non-dominated front at generation 10 with 29 solution fleets showing TOF versus observed targets trade. Right: the same front but showing the fleet size versus observed targets cross-section.

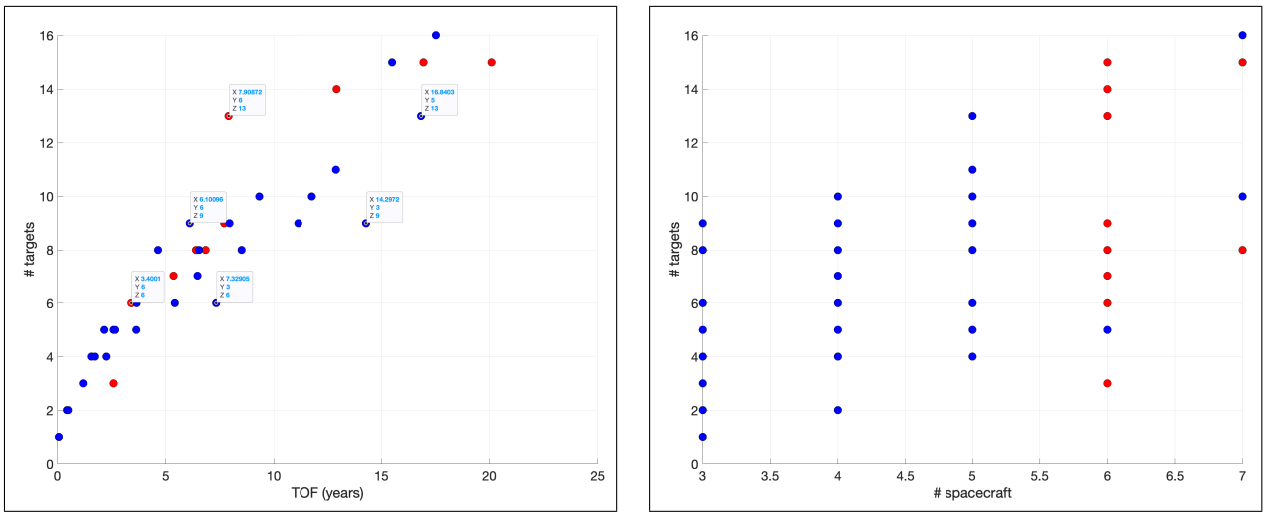


Figure 4.5: Above: cross-sections of the Pareto front after 40 generations. In the highlighted data points, x = TOF in years, y = number of spacecraft, and z = number of targets.

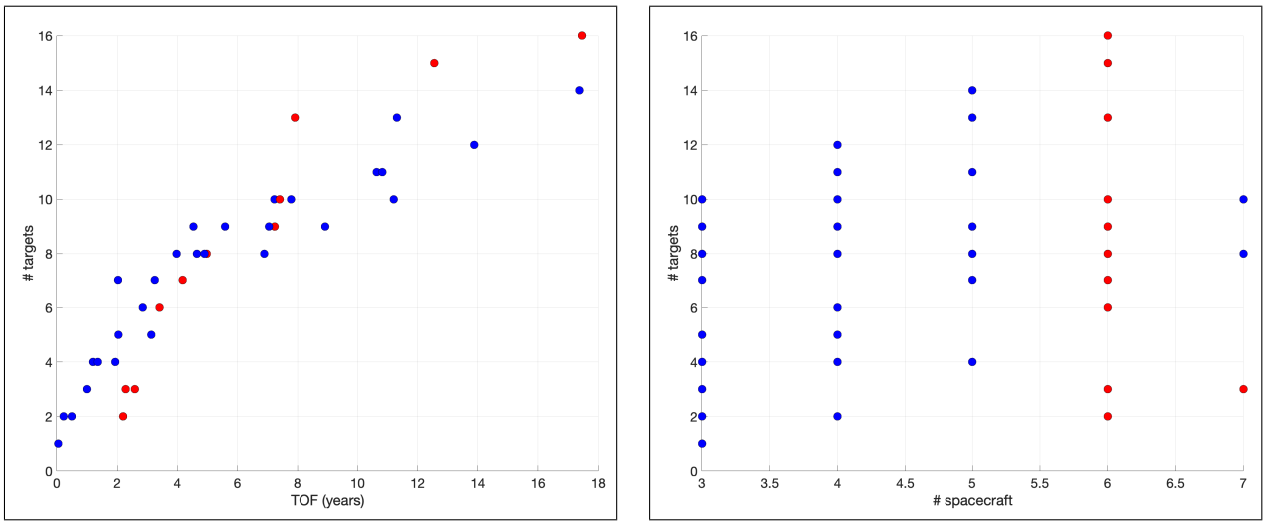


Figure 4.6: Above: cross-sections of the Pareto front after 100 generations.

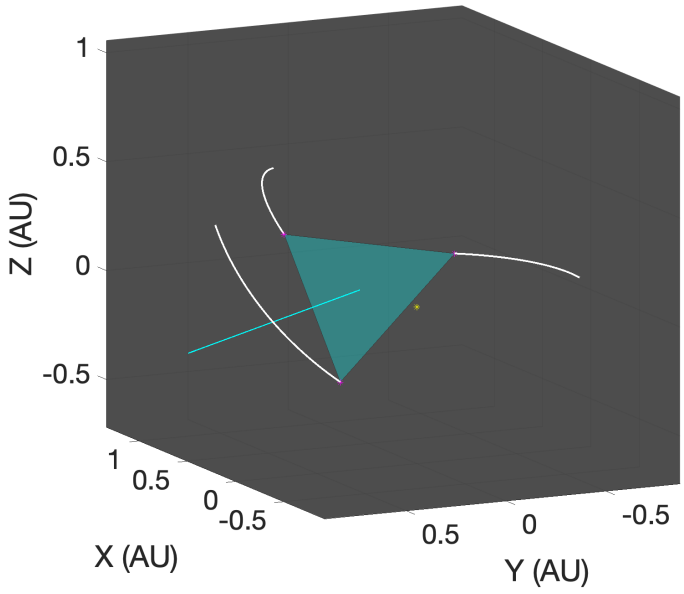


Figure 4.7: A fleet of 3 spacecraft making 1 observation of radio source 124.

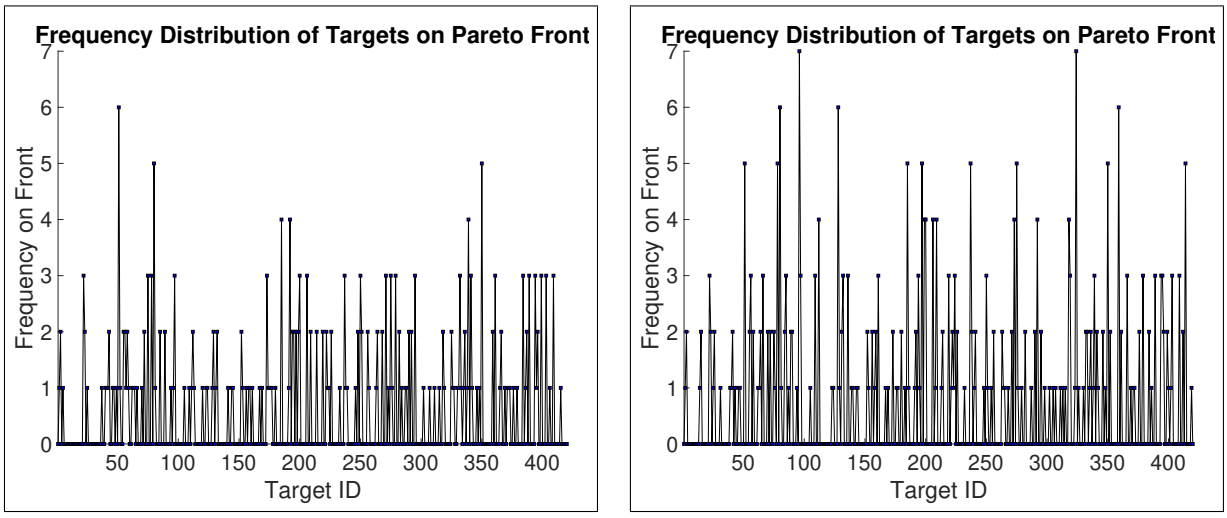


Figure 4.8: This figure shows the frequency distribution of different targets in the Pareto front at generation 10 (left) and 40 (right).

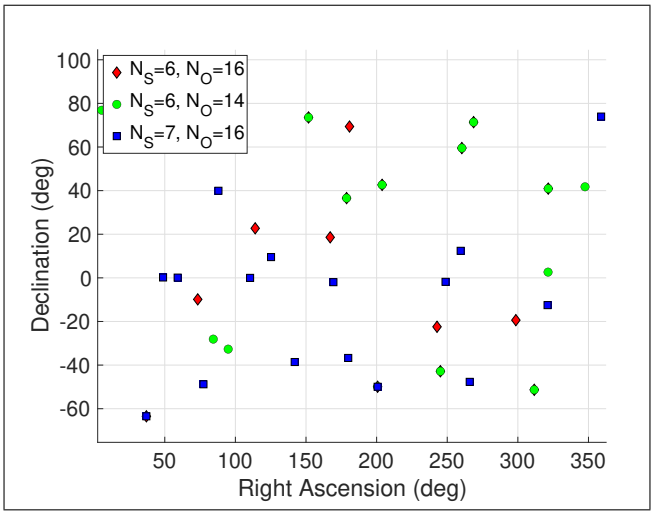


Figure 4.9: This figure shows the distribution of radio sources observed by the fleets in Fig. 4.10, 4.11, and 4.12.

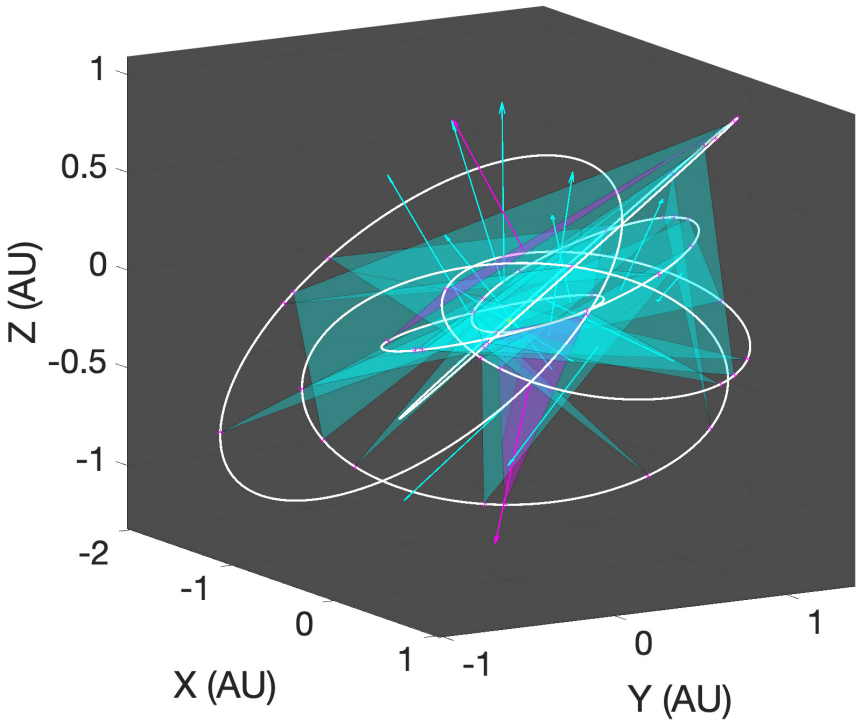


Figure 4.10: 6-spacecraft 14-target mission from generation 40's Pareto front, leveraging observation multiplicity, with itinerary outlined in Table 4.6.

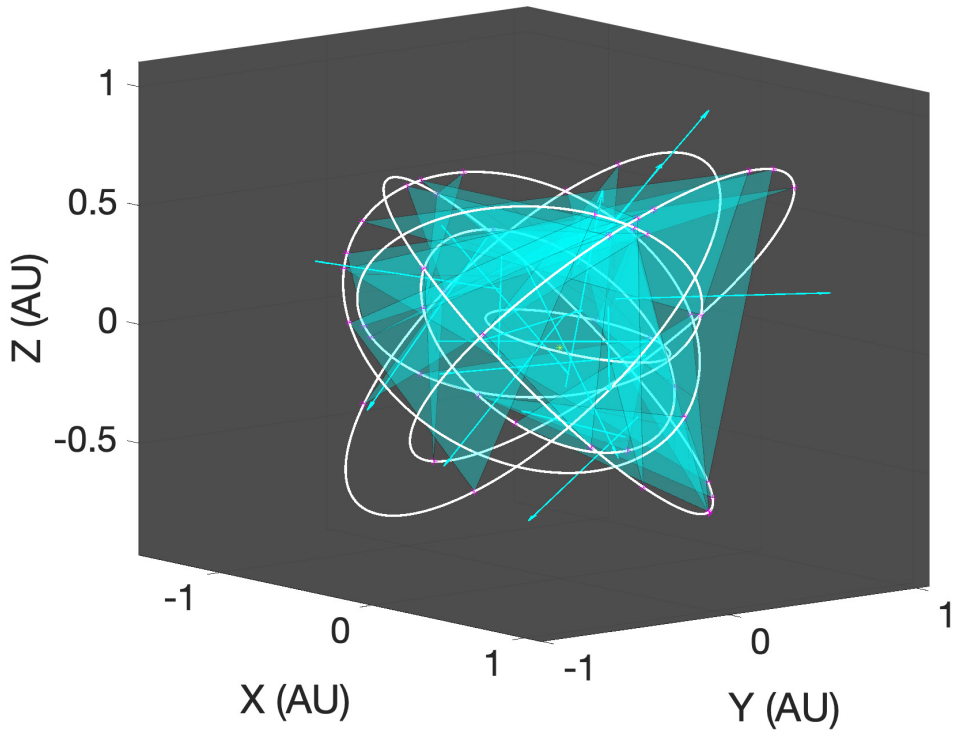


Figure 4.11: The most complex member of generation 40's Pareto front: 7 spacecraft observing 16 targets, with itinerary in Table 4.7.

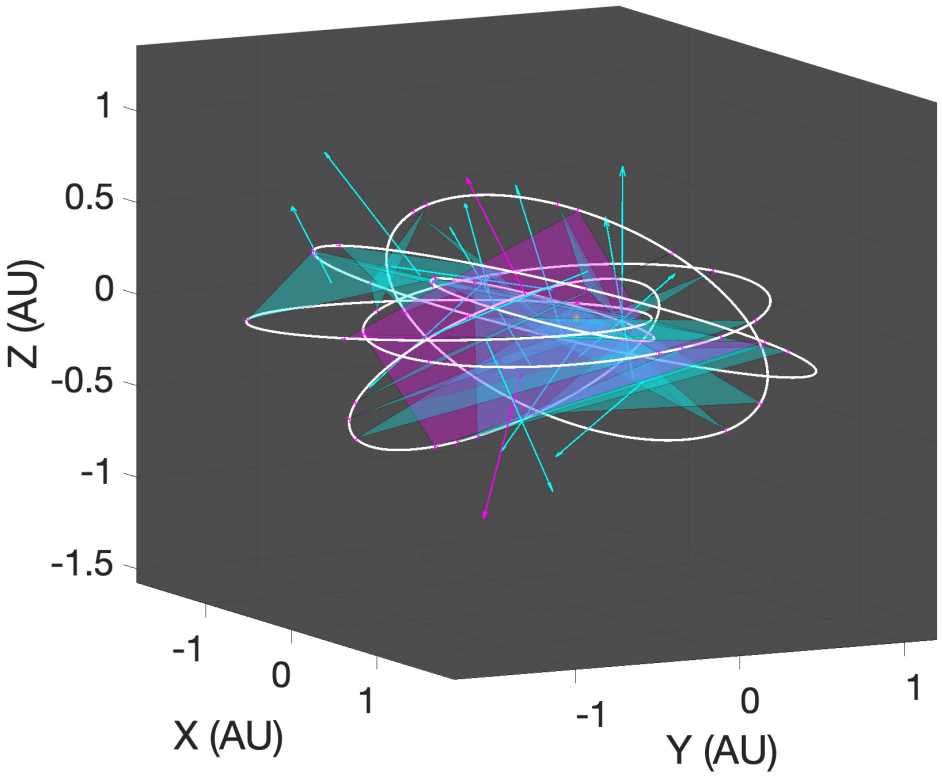


Figure 4.12: The most complex member of generation 100's Pareto front: 6 spacecraft observing 16 targets, with itinerary outlined in Table 4.8.

4.5 Summary

In this work, we have created and demonstrated the utility of a general Multi-Objective, Multi-Agent Hybrid Optimal Control Problem (MOMA HOCP) transcription for coordinated Multiple Traveling Salesmen Problems (MTSP) in context of global trajectory optimization, applied to a Very-Long Baseline Interferometry (VLBI) problem. We have leveraged a Null Gene transcription to vary the number of spacecraft in a fleet, allowing for the characterization of fleet size as a cost function to be traded against others including quantity of observed radio sources, time of flight (TOF), and more. This allowed us to quantify the trade of fleet size versus observed targets –a key trade study for mission analysts lacking *a priori* knowledge of the number of spacecraft required to achieve coordinated science objectives. In allowing for a fleet size greater than or equal to 6 spacecraft, we allowed members of the fleet to coordinate multiple observations at the same epoch, allowing us to capture solutions along the Pareto front with increased quantity of observed targets without increased TOF. We observed certain expected trends, including missions with $2S$ spacecraft observing the same number of targets as missions with S spacecraft in half the time, while other unexpected trends emerged, such as the prevalence of solutions where targets were not closely grouped in the sky. The outer-loop of this transcription is extensible to a general coordinated MTSP class of optimization problems. The transcription is a scaffold for traversing combinatorially vast discrete-valued search spaces, encoding which "salesmen" visit which "town" and enabling efficient characterization of a D-dimensional Pareto front–capabilities which are essential for exploration of multi-spacecraft multi-target exploration problems.

We briefly discussed the trend in popularity of different targets over successive generations of Pareto fronts. We note that an analogous trend not of targets but of target *sequences* is also an interesting point of investigation, and will study this trend in future work. The greatest limitation placed on techniques such as the transcription in this paper is the quantity of available distributed computing resources. While this novel transcription is a scaffold to traverse a vast decision space, a high quantity of computing resources is needed to sufficiently do so, and any efficiency measure that

can be taken must be. We have utilized a HashMap as an archive of previously solved solutions to prevent the outer-loop from re-evaluating the same chromosome multiple times, facilitating quicker traversal compared to a file I/O comparison approach. This technique may be augmented further to allow an archive comparison not of two *entire* chromosomes, but of *partially alike* chromosomes. For example, if the chromosome $\boxed{1}\boxed{2}\boxed{3}$ is created and the chromosome $\boxed{1}\boxed{2}\boxed{4}$ is stored in the archive, the archive utility could return the $\boxed{1}\boxed{2}$ portion of the optimal $\boxed{1}\boxed{2}\boxed{4}$ solution fleet's inner-loop decision vector as a partial-seed for the optimization of $\boxed{1}\boxed{2}\boxed{3}$. This seeding procedure may at best greatly speed up the inner-loop optimization of $\boxed{1}\boxed{2}\boxed{3}$, but is no worse of a seeding method than pulling the entire trajectory's initial guess decision vector from a uniform random distribution (as is currently done). Via partial seeding, the inner-loop would only be generating the initial guess for the fleet's decision variables from $\boxed{2}\boxed{3}$ at random instead of doing so for the entire mission. In short, introducing a partially alike seeding capability only stands to improve search space traversal.

Chapter 5

Conclusions and Future Work

This dissertation has explored the creation and novel application of several techniques for global trajectory optimization which combine to form a general MOMA HOCP transcription for the global trajectory optimization of multiple-spacecraft multiple-target mission design. This chapter summarizes the chief contributions of this dissertation and results produced therein, and discusses areas of future work to further the work presented here.

5.1 Conclusions

In this work, we created a global trajectory optimization transcription which frames multiple-spacecraft multiple-target missions as Multiple-Objective Multiple-Agent Hybrid Optimal Control Problems (MOMA HOCP). We selected the NSGA-II algorithm to optimize the discrete decision variables in an outer-loop and the MBH stochastic global search method wrapped around an NLP solver to optimize the continuous decision variables in an inner-loop. We parameterized the trajectories for each benchmark optimization problem using shooting transcriptions with coast arcs punctuated with a single impulsive burn along each phase, with a phase defined between two "targets".

The MOMA HOCP transcription encapsulates the trajectory optimization for an entire fleet of spacecraft into a single decision vector rather than multiple separate decision vectors for separate optimization problems. This encapsulation enabled the imposition of numerous inter-spacecraft coordination constraints and coordinated science objectives while removing the need for an alternative

brute force grid search approach.

The outer-loop component of the MOMA HOCP transcription uses a "chromosome" structure to discretize an inner-loop multi-spacecraft trajectory optimization problem. The chromosome is composed of a header to constrain the size of both the inner-loop trajectory optimization and the outer-loop search, as well as a Multiple Traveling Salesmen Problem (MTSP) gene sequence which encodes which targets in the search space will be explored in which order by which salesmen (i.e. spacecraft). When this chromosome is interpreted by the inner-loop as a real trajectory optimization problem, a single cost function is minimized, after which the solution fleet structure is returned to the outer-loop where any additional costs a mission analyst wants to trade are computed (without re-running the inner-loop). The NSGA-II algorithm efficiently searches the discrete decision space for favorable trajectory optimization problems to be posed and ranks their multi-objective performance to find the Pareto front.

In the IGMM benchmark problem, coordination constraints explored included a shared launch asymptote constraint for two spacecraft, a shared trajectory phase constraint (where multiple spacecraft must flyby the same planet or a sequence of planets on their overall journeys before diverging), and a shared flyby target constraint (two spacecraft must flyby the same body but not necessarily at the same time). With the shared launch asymptote constraint, only a single trajectory exists until the first maneuver is applied at which point a new trajectory for one spacecraft spawns at this maneuver, as a boundary point. The header structure of the outer-loop chromosome was used to capture the shared parameters of both spacecraft in the mission, reducing the size of the chromosome and the already massive outer-loop search space. This same boundary point spawning is applied in the case of the shared trajectory phase constraint- with only one trajectory existing until the two spacecraft need to flyby different targets. We demonstrated by example in this benchmark problem that sequential optimization of separate single vehicle trajectories yielded suboptimal results compared to those using the MOMA HOCP technique for the multi-mission.

In the VLBI mission design, the MOMA HOCP technique was further developed to more efficiently explore the outer-loop search space. An archive utility was created to prevent the re-

evaluation of previously optimized chromosomes. If the outer-loop creates a chromosome which is identical to that of a mission which has already been optimized, a check is performed and a new candidate chromosome is created. While a standard technique for an archive is to write past solutions to a text file, we use a HashMap structure as a computationally faster strategy over file I/O. The HashMap used in the VLBI benchmark problem mapped a bit string, which encodes a chromosome, to a boolean value indicating whether the chromosome had been tried or not. In both the IGMM and VLBI benchmark problems, complex rules had to be created to bookkeep reduce, and interpret the outer-loop chromosome into a real trajectory optimization problem for the inner-loop.

The VLBI problem also introduced new coordination constraints. The chief constraint was an observation geometry constraint on the trajectories of a given trio of spacecraft at a given epoch to facilitate the interferometry of a distant radio source. A VLBI triangle must be formed at a given observation epoch whose normal (or anti-normal) vector points to within a small tolerance of the the direction to the distant radio source. The GTOC-8 problem statement applied a piecewise constraint to accomplish this while we introduced a smooth constraint to aid the optimizer. Another constraint (which was interpreted by the outer-loop as a cost function) was the requirement for multiple trios of spacecraft to attempt observations at the same time if possible, to reduce the overall mission TOF cost. This feature often enabled 6-spacecraft fleets to often image the same number of targets as a 3-spacecraft fleet in half the time.

Finally, applied in the IGMM problem to vary the length of the planetary flyby sequence, a Null Gene transcription was applied in the VLBI work to vary the size of the fleet as well, demonstrating an elegant technique for mission analysts to capture the trade of fleet size against any other cost functions without using brute force to run separate studies with fixed fleet sizes. This technique is especially useful when a mission analyst does not know *a priori* the requisite fleet size to achieve the mission's objectives.

This research has shown that a MOMA HOCP transcription can be an effective capability for solving multiple-spacecraft multiple-target global trajectory optimization problems. The technique

developed and iteratively improved upon in this work was demonstrated on two benchmark problems in the deep space regime, but is by no means constrained to that regime. Indeed, we conclude that the overall transcription created in this work may be generally applied for the cislunar regime, the LEO regimes, and other MTSP beyond the scope of trajectory optimization altogether. This is due to its encapsulation of all the information required to pose and solve a MTSP: a chromosome which encodes which salesmen visit which towns in what order. The outer-loop chromosome structure is parsed and interpreted by an inner-loop whose transcription is arbitrary and problem-specific, and is problem agnostic to choice of MTSP.

We applied the first proof-of-concept transcription to an Ice Giants Multi-Mission design problem with several novel coordination constraints and discovered how these constraints induce costly changes in the trajectories optimized by the algorithm. The particular trajectory optimization problem statement was shown to prove physically untenable for operational mission design because of practical limitations on fuel due to the unfavorable relative positions of the ice giants; NASA's Voyager missions truly capitalized on a rare opportunity where the heavens aligned. We demonstrated that a MOMA HOCP technique is far more effective for posing the multi-spacecraft multi-target problem as a single rather than multiple separate trajectory optimization problems.

We applied a more matured version of this transcription to a very long baseline interferometry problem and applied a null gene transcription to vary the size of a fleet. We discovered numerous missions which imaged over a dozen distant radio sources. The dominant solutions were static constellations which used zero ΔV to image target sequences in non-intuitive ways. We expected dominant solutions to image sequences of radio sources which were all clustered together with very similar right ascension and declination ordered pairs, but instead found missions which image a sequence of targets all across the sky. Moreover, these solutions being static constellations will cause their observation patterns to repeat cyclically, enhancing the long term scientific value of the mission. We allowed the size of the fleet to range up to 7 spacecraft, which enabled multiple trios of spacecraft within the fleet to image targets at the same time, as expected.

Finally, we discussed how the HashMap archive technique is being augmented into a partial-

seeding algorithm for the inner-loop to decrease its runtime, and we hypothesize we will find a significant speed improvement. If this technique proves effective, it will not only aid in the trajectory optimization but also in quicker traversal of the outer-loop search space.

5.2 Future Work

The results presented in this dissertation merit future research on numerous fronts. One particular area of work is to create techniques that improve the tractability of the outer-loop search space. NP-Hard hybrid optimal problems such as those in this work are generally intractable for two reasons: insufficient distributed computing resources with which to traverse the discrete outer-loop search space, and the runtime cost of solving a global trajectory optimization problem with the inner-loop– a single point within the outer-loop search space. Whether a mission analyst applies techniques ranging from brute force to genetic algorithms to neural networks, the determining factor on whether a statistically representative fraction a search space can be traversed is the number of processors available. Because of this, efficiency measures need to always be taken to accelerate the traversal of a search space.

The HashMap archive contribution in the VLBI work was demonstrated to first order as quick check to avoid re-evaluation in the outer-loop. In the Appendix, we discuss how this structure can be leveraged further as a seeding capability to potentially widen the second bottleneck in global trajectory optimization: the runtime of a specific trajectory optimization problem. Future work is to carry out numerical validation of the hypotheses of the partial seeding algorithm development. These implementations will at worst add no efficiency or reach to the MOMA HOCP transcription while at best they will transform the technique into one with significant reach compared to other state-of-the-art methods. These tasks will serve to create a framework for global trajectory optimization that serves mission analysts exploring massive search spaces, enabling the design of new classes of distributed spacecraft missions.

Bibliography

[1] Atacama large millimeter array. https://www.almaobservatory.org/en/home/.

[2] Emtg (evolutionary mission trajectory generator). https://sourceforge.net/projects/emtg/.

[3] Pagmo (parallel global multiobjective optimizer). http://pagmo.sourceforge.net/pagmo/index.html.

[4] Vision and voyages for planetary science in the decade 2013-2022. The National Academies Press, 2011.

[5] Matlab, October 2021. http://www.mathworks.com/products/matlab/.

[6] Gad A and O Abdelkhalik. Hidden genes genetic algorithm for multi-gravity-assist trajectories optimization. Journal of Spacecraft and Rockets, 48:629–641, 2011.

[7] Gad A and O Abdelkhalik. Dynamic-size multi-population genetic optimization for multi-gravity-assist trajectories. Journal of Guidance, Control, and Dynamics, 35:520–529, 2012.

[8] Bernardetta Addis, Andrea Cassioli, Marco Locatelli, and Fabio Schoen. A global optimization method for the design of space trajectories. Computational Optimization and Applications, 48(3):635–652, jun 2009.

[9] Andranik Akopov. A multi-agent genetic algorithm for multi-objective optimization. In IEEE International Conference on Systems, Man, and Cybernetics. Institute of Electrical and Electronics Engineers, 2013.

[10] Nitin Arora, Nathan Strange, and Alkalai L. Trajectories for a near term mission to the interstellar medium. In AAS/AIAA Astrodynamics Specialist Conference, 08 2015.

[11] M Ayre, D Izzo, and L Pettazzi. Self assembly in space using behaviour based intelligent components. In Proceedings of the TAROS (Towards Autonomous Robotic Systems). Imperial College, London, 09 2005.

[12] G Beni and J Wang. Swarm intelligence in cellular robotic systems. Proceedings of NATO Advanced Workshop on Robots and Biological Systems, 06 1989.

[13] J Betts. Survey of numerical methods for trajectory optimization. Journal of Guidance, Control, and Dynamics, 21:193–207, 1998.

[14] S Bleuer, M Brack, L Thiele, and Zitzler. E. Multiobjective genetic programming: Reducing bloat by using spea 2. Congress on Evolutionary Computation, 2001.

[15] A Bryson and Y Ho. Applied optimal control: Optimization, estimation, and control. IEEE Transactions on Systems Man and Cybernetics, 9(6):366–367, 1979.

[16] A.E. Bryson and Y. Ho. Applied Optimal Control. Taylor and Francis, 1975.

[17] Andrea Cassioli, Dario Izzo, David Di Lorenzo, Marco Locatelli, and Fabio Schoen. Global optimization approaches for optimal trajectory planning. In Springer Optimization and Its Applications, pages 111–140. Springer Science+ Business Media, 2012. http://dx.doi.org/10.1007/978-1-4614-4469-5_5.

[18] M. Ceriotti and M. Vasile. Automated multigravity assist trajectory planning with a modified ant colony algorithm. Journal of Aerospace Computing, Information and Communication, 7(9):261–293, 2010.

[19] Dean J. Chai, Steven Z. Queen, and Samuel J. Placanica. Precision closed-loop orbital maneuvering system design and performance for the magnetospheric multiscale formation. In AAS/AIAA Space Flight Mechanics Meeting, 2015.

[20] C. M. Chilan and B. A. Conway. A space mission automaton using hybrid optimal control. Advances in the Astronautical Sciences, 127:259–276, 2007.

[21] LIGO Scientific Collaboration. Detections. https://www.ligo.org/detections.php.

[22] B.A. Conway. Spacecraft Trajectory Optimization. Cambridge University Press, 2010.

[23] V. Coverstone-Caroll, J.W. Hartmann, and W.J. Mason. Optimal multi-objective low-thrust spacecraft trajectories. Computational Methods in Applied Mechanical Engineering, 186:387–402, 2000.

[24] Victoria Coverstone-Carroll. Near-optimal low-thrust trajectories via micro-genetic algorithms. Journal of Guidance, Control, and Dynamics, 20(1):196–198, jan 1997.

[25] K. Deb. Multi-Objective Optimization Using Evolutionary Algorithms. John Wiley & Sons, Inc., 2001.

[26] Kalyan Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsgaii. IEEE Transactions on Evolutionary Computation, 6, 08 2002.

[27] J. Englander, M. Vavrina, and D. Hinckley. Global optimization of low-thrust interplanetary trajectories subject to operational constraints. In AAS/AIAA Astrodynamics Conference, Napa, Ca, February 2016.

[28] Jacob Englander. Automated trajectory planning for multiple-flyby interplanetary missions. 04 2013.

[29] Jacob Englander, Bruce Conway, and Trevor Williams. Automated mission planning via evolutionary algorithms. Journal of Guidance, Control, and Dynamics, 35, 10 2012.

[30] Jacob Englander, Matthew Vavrina, and A Ghosh. Multi-objective hybrid optimal control for multiple-flyby low-thrust mission design. In AAS/AIAA Space Flight Mechanics Meeting, 01 2015.

[31] Jacob A Englander and Bruce A Conway. An automated solution of the low-thrust interplanetary trajectory problem. Journal of Guidance, Control, and Dynamics, 2016.

[32] Jacob A Englander and Bruce A Conway. An automated solution of the low-thrust interplanetary trajectory problem. Journal of Guidance Control and Dynamics, 40(1):15–27, 2016.

[33] Jacob A. Englander, Donald H. Ellison, and Bruce A. Conway. Global optimization of low-thrust, multiple-flyby trajectories at medium and medium-high fidelity. In AAS/AIAA Space-Flight Mechanics Meeting, Santa Fe, NM. AAS, January 2014.

[34] Jacob A. Englander and Arnold C. Englander. Tuning monotonic basin hopping: Improving the efficiency of stochastic search as applied to low-thrust trajectory optimization. In 24th International Symposium on Space Flight Dynamics, 2014.

[35] Jacob A Englander and Alexander R Ghosh. Coupled low-thrust trajectory and systems optimization via multi-objective hybrid optimal control. In AAS/AIAA Spaceflight Mechanics Meeting, pages 11–15, 2015.

[36] Gordon Fraser and Andrea Arcuri. The seed is strong: Seeding strategies in search-based software testing. In Proceedings of the 2012 IEEE Fifth International Conference on Software Testing, Verification and Validation, pages 121–130. IEEE, 2012.

[37] Alexander Ghosh and Victoria Coverstone. Optimal cooperative cubesat maneuvers obtained through parallel computing. Acta Astronautica, 107:130–149, 2015.

[38] Pradipto Ghosh and Bruce A Conway. A direct method for trajectory optimization using the particle swarm approach. In 21 st. AAS/AIAA Space Flight Mechanics Meeting, New Orleans, Louisiana, February 2011.

[39] Phillip E Gill and Michael A Saunders. Snopt : An sqp algorithm for large-scale constrained optimization. SIAM Journal on Optimizaton, 12(4):979–1006, 2002.

[40] Yanping Guo and Robert W. Farquhar. New horizons mission design. In New Horizons, pages 49–74. Springer Science+ Business Media.

[41] P Hajena and Lin C. Genetic search strategies in multicriterion optimization design. Structural Optimization, 4:99–107, 06 1992.

[42] John W. Hartmann, Victoria L. Coverstone-Caroll, and Steven N. Williams. Optimal interplanetary spacecraft trajectories via a pareto genetic algorithm. The Journal of the Astronautical Sciences, 46:267–282, 1998.

[43] Jonathan Herman. Improved collocation methods to optimize low-thrust, low-energy transfers in the earth-moon system. 2015.

[44] Mark Hofstdater and Amy Simon. Ice giants pre-decadal survey mission study report. 06 2017.

[45] Xiao-Min Hu, Jun Zhang, xiao Jing, and Yun Li. Protein folding in hydrophobic-polar lattice model: A flexible ant-colony optimization approach. Protein and peptide letters, 15:469–77, 02 2008.

[46] Kyle M Hughes, Jeremy M Knittel, and Jacob A Englander. Gravity-assist trajectories to the ice giants: An automated method to catalog mass- or time- optimal solutions. In AAS/AIAA Astrodynamics Specialist Conference, 08 2017.

[47] D. Izzo, D. Hennes, and A. Riccardi. Constraint handling and multi-objective methods for the evolution of interplanetary trajectories. AIAA Journal of Guidance, Control, and Dynamics, online early edition, 2014.

[48] Dario Izzo. Global optimization and space pruning for spacecraft trajectory design. pages 178–201, 2010.

[49] Dario Izzo, Daniel Hennes, Marcus Märtens, Ingmar Getzner, Krzysztof Nowak, Anna Heffernan, Stefano Campagnola, Chit Hong Yam, Naoya Ozaki, and Yoshihide Sugimoto. Gtoc8: Results and methods of esa advanced concepts team and jaxa-isas. In AAS/AIAA Spaceflight Mechanics Meeting, 02 2016.

[50] Horn J and Nafpliotis N. Multiobjective optimization using the niched pareto genetic algorithm. 07 1993.

[51] Kenneth Kellerman. Radio interferometry and aperture synthesis. Encyclopedia Britannica, 2021.

[52] Matthew P Kelly. Transcription methods for trajectory optimization: a beginners tutorial. arXiv:1707.00284 [math], 07 2017.

[53] J. Kennedy and R. Eberhart. Particle swarm optimization. In Proceedings of ICNN'95 - International Conference on Neural Networks, volume 4, pages 1942–1948, 1995.

[54] J Knittel, K Hughes, J Englander, and B Sarli. Automated sensitivity analysis of interplanetary trajectories for optimal mission design. In International Symposium on Space Flight Dynamics, 06 2017.

[55] James Longuski, José Guzman, and John Prussing. Optimal Control with Aerospace Applications. Springer Science+ Business Media, 2014.

[56] James M. Longuski and Steve N. Williams. Automated design of gravity-assist trajectories to mars and the outer planets. Celestial Mechanics and Dynamical Astronomy, 52(3):207–220, 1991. http://dx.doi.org/10.1007/BF00048484.

[57] Roberto Lopez-Herrejon, Javier Ferrer, Francisco Chicano, Alexander Egyed, and Enrique Alba. Comparative analysis of classical multi-objective evolutionary algorithms and seeding strategies for pairwise testing of software product lines. In IEEE Congress on Evolutionary Computation. Institute of Electrical and Electronics Engineers, 2014.

[58] R. Timothy Marler and Jasbir S. Arora. Survey of multi-objective optimization methods for engineering. Structural and multidisciplinary optimization, 26(6):369–395, 2004.

[59] T Troy McConaghy, Theresa J Debban, A.E. Petropoulos, and J.M. Longuski. An approach to design and optimization of low-thrust trajectories with gravity-assists. Advances in the Astronautical Sciences, 2002.

[60] Daniel Merkle and Martin Middendorf. Ant colony optimization, marco dorigo, thomas stützle. mit press (2004), isbn: 0-262-04219-3. European Journal of Operational Research, 168:269–271, 01 2006.

[61] Sean Napier, Jay McMahon, and Jacob Englander. A multi-objective, multi-agent for the global optimization of interplanetary trajectories. The Journal of the Astronautical Sciences, 67, 08 2020.

[62] NASA. Lisa: Laser interferometer space antenna. https://lisa.nasa.gov/.

[63] A. D. Olds, C. A. Kluever, and M. L. Cupples. Interplanetary mission design using differential evolution. Journal of Spacecraft and Rockets, 44(5):1060–1070, 2007.

[64] Jeffrey Parker. Low-energy ballistic lunar transfers. The Journal of the Astronautical Sciences, 58, 07 2011.

[65] I Ross and C D'Souza. A hybrid optimal control framework for mission planning. Journal of Guidance, Control, Dynamics, 28:686–697, 2005.

[66] Abolfazl Shirazi, Josu Ceberio, and Jose Lozano. Spacecraft trajectory optimization: A review of models, objectives, approaches and solutions. Progress in Aerospace Sciences, 102, 08 2018.

[67] Rainer Storn and Kenneth Price. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization, 11:341–359, 01 1997.

[68] NJ Strange and JM Longuski. Graphical method for gravity-assist trajectory design. Journal of Spacecraft and Rockets, 39(1):9–16, January-February 2002.

[69] Y Sun, G Scutari, and D Palomar. Distributed nonconvex multiagent optimization over time-varying networks. 11 2016.

[70] IPAC Communications & Education Team. Ligo: Laser interferometer gravitational-wave observatory. `https://www.ligo.caltech.edu/page/what-is-ligo`.

[71] M. Vasile and P. De Pascale. Preliminary design of multiple gravity-assist trajectories. Journal of Spacecraft and Rockets, 43(4):794–805, 2006.

[72] M Vasile and F Zuiani. Macs: A hybrid multiobjective optimization algorithm applied to space trajectory optimization. Journal of Aerospace Engineering, 225:1211–1227, 2011.

[73] Massimiliano Vasile, Edmondo Minisci, and Marco Locatelli. Analysis of some global optimization algorithms for space trajectory design. Journal of Spacecraft and Rockets, 47(2):334–344, mar 2010.

[74] M Vavrina, J Englander, and D Ellison. Global optimization of n-manuever, high-thrust trajectories using direct multiple shooting. In AAS/AIAA Space Flight Mechanics Meeting. AAS/AIAA, 02 2016.

[75] Matthew Vavrina and Kathleen Howell. Multiobjective optimization of low-thrust trajectories using a genetic algorithm hybrid. Advances in the Astronautical Sciences, 134, 02 2009.

[76] G. M. Viswanathan, M.G.E. da Luz, E.P. Rapaso, and H.E. Stanley. The Physics of Foraging. Cambridge University Press, 2011.

[77] O von Stryk and M Glocker. Numerical mixed-integer optimal control and motorized traveling salesman problems. European Journal of Control, 35:519–533, 2001.

[78] A Wächter and Lorenz Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Research Report RC 23149, IBM T. J. Watson Research Center, Yorktown, USA, 1 2006.

[79] Logan Yliniemi. Multi-objective optimization in multiagent systems. 04 2015.

[80] Yu-Dong Zhang, Shuihua Wang, and Genlin Ji. A rule-based model for bankruptcy prediction based on an improved genetic ant colony algorithm. Mathematical Problems in Engineering, 2013:1–10, 11 2013.

[81] Guoxiang Zhao and Minghui Zhu. Pareto optimal multirobot motion planning. IEEE Transactions on Automatic Control, 6, 09 2021.

[82] Escart Zitzler and Lothar Thiele. Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. IEEE Transactions on Evolutionary computation, 3, 1999.

Appendix A

Seed Sharing Technique

A.1 Background

Global trajectory optimization is inherently difficult due to many factors including the complexity and/or sensitivity of the dynamical models, the often non-analytical cost and constraint function Jacobians, the high quantity of local optima, and the size of the global search space. To obtain a locally optimal trajectory for a given problem, a good initial guess is needed. Myriad initial guess generation methods have been developed for different classes and sizes of trajectory optimization problems. For example, a conventional approach for finding transfer trajectories to a libration point orbit in the restricted 3-body problem (RTBP) is to generate the stable and unstable manifolds of the orbit and seed an NLP solver with a trajectory on the manifold. Chemical high impulse interplanetary trajectories such as those in this dissertation use Keplerian dynamics to approximate trajectories as conjoined conic sections. Indeed, the choices of initial guess methods are highly dependent upon the sensitivity, dynamics, and parameterization of the trajectory optimization problem. The inner-loop of this work uses a stochastic search method, MBH, to draw initial guess trajectories from a uniform random distribution in the first iteration and perturb them using a Pareto random distribution for subsequent iterations. MBH is thus the method used to find good initial guesses and search the global space of a given trajectory optimization problem.

Many other techniques for quickly locating good initial guesses from a search space have been explored. These include, but are not limited to, graphical representations such as Tisserand plots, search space pruning, and seed sharing [68, 48, 22, 66, 46, 54, 57, 36]. There are tradeoffs in each of

these methods. Stochastic search methods are at least as efficient as brute force, finding the global optimum in less time and require no analyst intervention, but are inefficient as they require a long runtime to reliably search the space. Graphical representations of the search space can impart intuition about the complex search space to a mission analyst, allowing for quick identification of candidate solutions once created. However, they: require analyst to intervene in the optimization process, and may not represent an accurate picture of viable initial guesses which satisfy problem dynamics. Search space pruning techniques share similar tradeoffs to graphical methods as they require an analyst to make decisions about which pockets of the search space are worth searching - based on intuition and assumptions which may lead to no solutions being found or cause the analyst to miss the best solutions. However, pruning allows the analyst to completely control the size of the search space, and to some degree all global optimization problems are in practice subjected to pre-pruning of their search space based on mission and time requirements.

Seed sharing describes a general practice of solving a subset of a problem space and using the results of the subset to seed candidate solutions in the subset's neighborhood. This practice is preferable when the number of different possible inputs in a search space is high, but the difference between nearby inputs is small and an analyst seeks to study most if not the entire search space without reducing its size. This practice has been applied extensively for interplanetary global trajectory optimization of single-spacecraft missions in an operational environment [54, 46]. In this chapter, we propose a novel seed sharing technique for multiple-spacecraft global trajectory optimization, applied to the vast search space of the VLBI problem. This technique tackles the problem of seeding a mixed-integer decision vector by identifying *partially alike* chromosomes.

We discuss the scope of the VLBI search space and motivation for this technique in § A.2. We then define criteria for partially alike chromosomes in § A.3 and describe the technique in § A.4. We prime further discussion using a simple example in § A.5 followed by a complicated example in § A.6. We then propose an experiment to test this capability in § sec:expseed and discuss anticipated findings of the experiment in § A.8. The development and future results of the work this chapter will inform the future work of this dissertation, as multi-spacecraft global

trajectory optimization problems pose complex high-dimensionality search spaces which will require new traversal algorithms to locate optimal solutions quickly.

A.2 Motivation

Throughout this dissertation, we have identified many key techniques that contribute to the successful formulation of a transcription for the global trajectory optimization of multi-spacecraft multi-target exploration missions. We have also discovered several key bottlenecks with these techniques. The most significant is the availability of ample distributed computing resources. The benchmark problems posed in this dissertation are combinatorially vast.

In the VLBI problem, as posed in chapter 4, we can estimate the size of the outer-loop search space by considering the two outer-loop bounding parameters: number of spacecraft in the fleet N_S and number of observation epochs N_O. For simplicity, let us consider the chromosomes for a VLBI problem where: N_S and N_O are fixed, only one observation is allowed at each epoch, each spacecraft trio is legal (sorted in ascending order and each ID unique), and each observation target gene is a unique non-null one. If the fleet size is 3, then there is only one possible legal trio for each observation: $1-2-3$ and the number of chromosomes in this problem is given by Eq. 4.38. If the fleet size is greater than 3, the number of possible chromosomes multiplies by an iterated function $P_{\text{trio}}(N_S)$ which quantifies the number of legal trio permutations that may occur at each observation epoch for a given N_S. The hypothetical value of $N_S = 2$ constitutes a fleet of 2 spacecraft which can not form a triangular formation. Let us define $P_{\text{trio}}(2) \equiv 0$, the base value of the P_{trio} iterated function. Then, $P_{\text{trio}}(3) = 1$. Table A.1 shows the manually calculated values of P_{trio} for different values of N_S. We may express the value of P_{trio} using the expression in Eq. A.1, where $\delta(N_S)$ is an iterative function increment which must be added to the first two terms $N_S - 1$ and $P_{\text{trio}}(N_S - 1)$. Its expression is given in Eq. A.2. However, in the case of $N_S = 2$, we define the base value of $\delta(N_S) \equiv -1$.

N_S	P_{trio}
2	0
3	1
4	4
5	10
6	20
7	35

Table A.1: P_{trio} as a function of N_S.

$$\begin{cases} P_{\text{trio}}(N_S) = (N_S - 1) + P_{\text{trio}}(N_S - 1) + \delta(N_S) : & N_S \geq 3 \\ P_{\text{trio}}(N_S) = 0 : & N_S < 3 \end{cases} \tag{A.1}$$

$$\begin{cases} \delta(N_S) = \delta(N_S - 1) + (N_S - 3) : & N_S \geq 3 \\ \delta(N_S) = -1 : & N_S < 3 \end{cases} \tag{A.2}$$

Using the base values for P_{trio} and $\delta(N_S)$, we can solve Eq. A.1 for a fleet size of 3 spacecraft, following the steps in Eq. A.3.

$$\begin{cases} P_{\text{trio}}(N_S) = (N_S - 1) + P_{\text{trio}}(N_S - 1) + \delta(N_S) \\ = (N_S - 1) + P_{\text{trio}}(N_S - 1) + [\delta(N_S - 1) + (N_S - 3)] \\ = 2N_S - 4 + P_{\text{trio}}(N_S - 1) + \delta(N_S - 1) \\ = 2(3) - 4 + P_{\text{trio}}(2) + \delta(2) \\ = 2(3) - 4 + 0 + -1 \\ = 1 \end{cases} \tag{A.3}$$

This relation holds for all fleet sizes, and we can iteratively use it to thus compute that the number of permutations of trios for a fleet of 8 spacecraft at a given observation epoch is 56. For the simplified VLBI problem defined in this section, the size of the outer-loop search space is given by Eq. A.4. For a fixed fleet size of 7, a fixed number of observation epochs of 30, and a search space of 420 targets, this evaluates to $D_{OL} = \frac{35*420!}{390!} \approx 6 \times 10^{79}$ legal chromosomes.

$$D_{OL} = \frac{P_{\text{trio}}(N_S)N_T!}{(N_T - N_O)!} \tag{A.4}$$

This is an absolutely enormous problem. What's more, in this dissertation, N_S and N_O are varied using a Null Gene transcription, multiplying the size of the outer-loop search space even further. The search space includes a chromosome for every possible fleet size versus every possible number of observations. Its size is given by equation A.5.

$$D_{OL_{\text{full}}} = \sum_{N_S=3}^{7} \sum_{N_O=1}^{30} \frac{P_{\text{trio}}(N_S)420!}{(420 - N_O)!} \tag{A.5}$$

Additionally, the time required to traverse the full search space is further increased by the duration required by the inner-loop solver to perform the trajectory optimization for each point in the space– on the order of hours per point solution. These two bottlenecks: the time cost of solving for a point solution, and the size of the outer-loop search space, motivate the creation and application of new techniques to speed up the search space traversal. The HashMap archive utility introduced in the VLBI problem design is a first order step in expediting this traversal, preventing previously solved outer-loop chromosomes from being passed to the inner-loop using the quick lookup of a hash function. We seek to extend this technique further by seeding the cumbersome inner-loop problem using archived solutions from the HashMap when possible. If a new candidate outer-loop chromosome is created that is similar (but not identical) to an archived solution, then seeding parts of the candidate's mission with the identical phases of the archived solution will at worst yield no benefit, but at best will greatly accelerate the convergence time of the inner-loop.

Genetic algorithms such as NSGA-II facilitate a much more efficient traversal of the search space by choosing handfuls of points in the search space at random and evaluating their multi-objective costs, approximating the otherwise non-existent gradient information of the discrete search space. The results of the NSGA-II algorithm reveal which missions' sequences have which cost. If we can further make an association between *fractional* mission sequences and a vector of costs, this may allow for even quicker traversal of the outer-loop search space. The creation of a partial-seeding

technique in this dissertation could serve to reduce the cost at the outer-loop and inner-loop level. If a partially-seeded chromosome is run by the inner-loop and is solved significantly quicker than seeded from random, this also enables the outer-loop to resume searching the discrete space sooner.

A.3 Definition of partially alike chromosomes

We define a partially alike chromosome as one which shares at least one gene with another chromosome. However, the shared genes are not necessarily located at the same indices within their chromosomes. The order of genes within the chromosome matters because the locations of different genes indicate specific context for how the chromosome is evaluated. In the work presented in this dissertation, the chromosomes in the outer-loop use sequences of genes to transcribe a trajectory. A single gene is insufficient to determine whether two chromosomes (missions) could share identical mission segments; a trajectory phase is always defined between at least two boundary events. The number of genes in a sequence that warrants a "partially" alike designation varies per trajectory optimization problem, but for our purposes that number is always at least two. This harkens to the nature of a TSP, where in the Ice Giant Multi-Mission problem, two spacecraft (salesmen) each perform flybys (visits) of sequences of planets (towns), occasionally visiting the same planet at the same time. In the VLBI problem, multiple spacecraft (salesmen) simultaneously take interferometric observations of (visit) multiple distant radio sources (towns). The journey a salesmen takes to visit a town is constrained by the road map and is uniquely defined by the salesmen's ID and the initial and final towns on that journey, and their journeys may be considered similar if they contain an identical partial sequence of towns.

Let us define the minimum comparable segment length L_{min} of an outer-loop chromosome. When comparing 2 chromosomes for similarities, they chromosomes must share a minimum number of genes in sequence to be considered partially alike. In this work, the salesmen are always spacecraft, but their visits and town vary entirely based on the trajectory optimization problem statement. In all cases however, the spacecraft must follow distinct trajectories to achieve their goals. Each spacecraft's trajectory encompasses a series of targets. The base unit of a trajectory in

this work is a phase. A given spacecraft's trajectory phase begins with an outgoing position/velocity state which allows the spacecraft to visit a target and ends with with new position/velocity state that facilitates a visit of its next target.

For the ice giants problem, at the outer-loop level, an entire phase is transcribed by 2 genes: the IDs for the outgoing and incoming flyby planets. Therefore for that problem, $L_{min} = 2$. In the VLBI work, each phase of mission is defined by a unique trio of spacecraft following their trajectories to image a distance radio source. The previously observed radio source, the 3 spacecraft, and next radio source genes give $L_{min} = 5$. To generalize: two outer-loop chromosomes are partially alike if they have at least one identical phase sequence. The number of parameters used to define a phase in the outer-loop is problem-subjective. Further, careful bookkeeping is required to parse a chromosome to ensure a gene sequence being compared actually represents a phase. For instance, while the length of the chromosomes need not be identical nor the gene sequence be located at the same indices in both chromosomes, a *header* gene sequence must not be compared to a *phase* sequence.

A.4 Description of partial seeding technique

The partial likeness check of the newly created candidate chromosome is performed following reduction, then uniqueness check, using a HashMap structure similar to the one created for the work in chapter 4. However whereas the value corresponding to the archive key in prior work was merely a boolean, it is now the key's corresponding solution fleet structure. This change is necessary in order to access the physical trajectories of the fleet, to use for seeding the candidate, in the event that a partially alike archived solution is found. The new map structure is summarized in Eq. A.6

$$M\langle \text{Key}, \text{Value} \rangle = \langle \text{bit string}, \text{Fleet} \rangle \tag{A.6}$$

To illustrate the partial likeness check, consider the following chromosome structure (Eq.A.7).

$$X_{OL} = \underbrace{\boxed{N_S}\boxed{N_O}}_{\text{Header}} \underbrace{\boxed{S_{1,1}}\boxed{S_{2,1}}\boxed{S_{3,1}}\boxed{T_1}}_{\text{Observation 1}} \underbrace{\boxed{S_{1,2}}\boxed{S_{2,2}}\boxed{S_{3,2}}\boxed{T_2}}_{\text{Observation 2}} ... \underbrace{\boxed{S_{s,o}}\boxed{S_{s,o}}\boxed{S_{s,o}}\boxed{T_o}}_{\text{Observation O}} \tag{A.7}$$

As described in detail in chapter 4, this structure denotes a header containing the number of spacecraft and the number of observation epochs in the trajectory optimization problem, followed by a sequence of VLBI observations of different targets taken by different trios of spacecraft. We may look at this chromosome in a slightly different way, in terms of trajectory phases. The trajectories of a given trio of spacecraft are defined with between two boundaries (observation targets) Eq. A.8.

$$X_{OL} = \underbrace{\boxed{N_S}\boxed{N_O}}_{\text{Header}} \underbrace{\boxed{S_{1,1}}\boxed{S_{2,1}}\boxed{S_{3,1}}\boxed{T_1}}_{\text{Phase 1**}} \underbrace{\boxed{T_1}\boxed{S_{1,2}}\boxed{S_{2,2}}\boxed{S_{3,2}}\boxed{T_2}}_{\text{Phase 2}} ... \underbrace{\boxed{T_{o-1}}\boxed{S_{s,o}}\boxed{S_{s,o}}\boxed{S_{s,o}}\boxed{T_o}}_{\text{Phase O}} \tag{A.8}$$

Phase 1 is a special phase because the initial boundary point is undefined. There are infinitely many solutions to the VLBI problem in the case of one observation target; infinitely many states that a trio could take which form a triangle whose normal or anti-normal vector points to the distant radio source T_1. Propagation is not even necessary unless a problem is posed involving a sequence of multiple targets whose arrangement in the sky merits it. Therefore, in the partial likeness test, if the gene sequence for phase 1 of a candidate chromosome matches that of an archived solution, we ignore this similarity unless there is also an identical phase sequence defined explicitly between two targets later in the chromosome.

To determine the degree of similarity between a candidate and an archived chromosome, we apply the following process.

1) Identify the indices in the archive key (bit string) which delimit each phase gene sequence.

2) Identify each phase gene sequence in the candidate key.

3) For each phase gene sequence in the candidate key, check for the existence of that sequence at each initial phase index in the archive key.

4) Each time a phase gene sequence match is found, increment the rank (number of identical phases) that the archive key shares with the candidate key.

The above 4 steps are repeated until every single archive key in the HashMap has been searched. This linear search is necessary in order to find all partially alike keys, if any. If there are multiple partially alike archive keys to the candidate key, the first criterion for selecting the key whose fleet will partially seed the candidate's inner-loop decision vector is the one with the maximum number of identical phase gene sequences. The purpose of this criterion is to minimize the portion of the inner loop decision vector which is seeded at random, maximizing the chance of finding a feasible or optimal solution quickly. However, if multiple archive keys have the same number of identical phases to the candidate, then the one with the *most consecutive identical phases* is chosen as the seed mission. This second criterion is derived from the fact that, all things being equal, a seed solution with a consecutive chain of phases constitutes a more completely solved problem than a seed with multiple nonconsecutive phases –in which case maneuver targeting must be done to create a complete trajectory for unseeded phases. If there are still multiple identically ranked keys according to these two criteria, we employ one last deterministic check for seed selection. If any archive keys have the same fleet size as the candidate, that mission is chosen to seed. If there are still multiple seed contenders after these 3 criteria are applied, one is chosen from the list at random.

The third criterion derives from the fact that a fleet's trajectories are coordinated. Though implicitly so in many cases, this coordination impacts the usefulness of a phase from one fleet to another. For example a solution fleet with 5 spacecraft might have an identical phase to a solution mission with 4 spacecraft. However, if the candidate mission to be partially seeded has a fleet size of 5, then all things being equal, the genes from the 5-spacecraft archive mission are preferable because coordination of the first 4 spacecraft trajectories with a 5th was taken into account. This accounting is necessary for the candidate's 5-spacecraft mission. Fig. A.1 shows how the overall MOMA HOCP transcription is modified with the addition of the partially alike seeding technique. Fig. A.2 illustrates the entire process of this partial seed ranking technique.

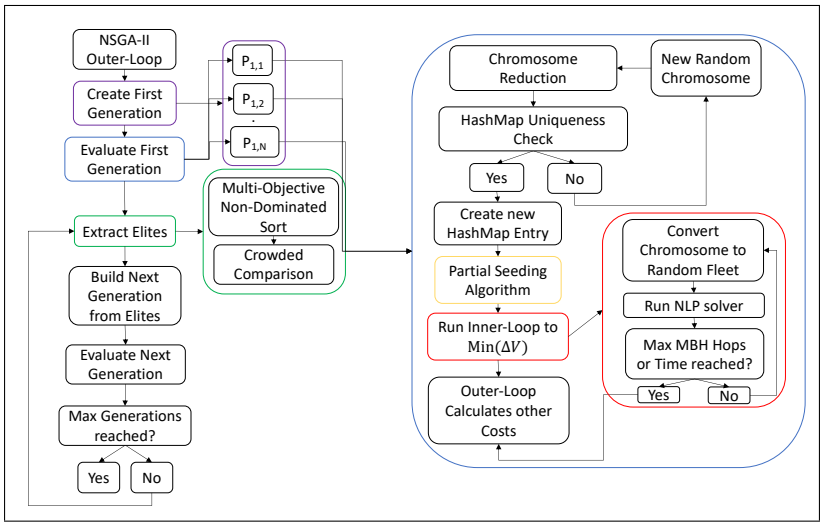


Figure A.1: Detailed flow chart of inner-loop and outer-loop algorithm interaction with added partial seeding step.

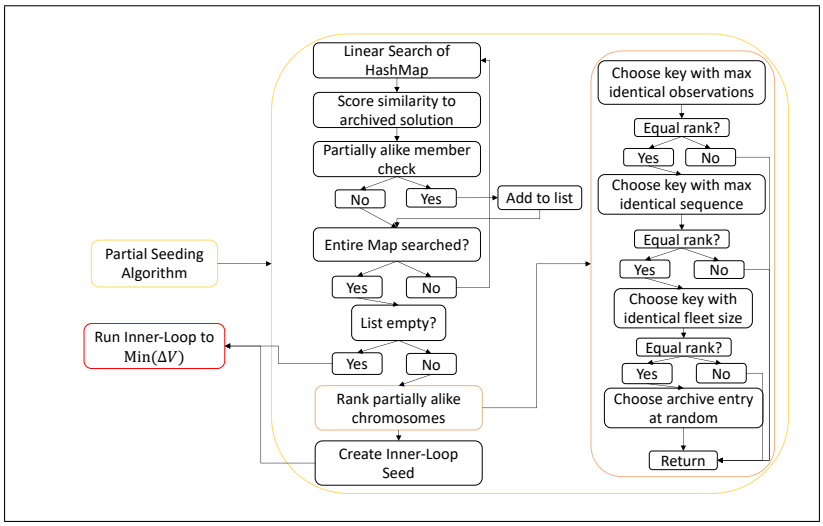


Figure A.2: Illustration of the partial seed ranking algorithm.

A.5 Example of a Simple Partial Seeding Problem

Assuming $L_{\min} = 2$ and that any 2 sequential genes constitute a mission phase, consider the following two integer chromosomes.

$$X_{\text{candidate}} = \boxed{1}\boxed{2}\boxed{3} \tag{A.9}$$

$$X_{\text{archive}} = \boxed{1}\boxed{2}\boxed{4} \tag{A.10}$$

Here we observe that the archived chromosome is partially alike to the candidate chromosome, with a similarity rank of 1. The phase sequence $\boxed{1}\boxed{2}$ is shared between them. Thus, the candidate's inner-loop decision parameters for the phase punctuated by "towns" $\boxed{1}$ and $\boxed{2}$ would be initialized to the values for the archived solution fleet's trajectories corresponding to this same phase. Those for the next phase, towns $\boxed{2}$ and $\boxed{3}$ would be chosen from a bounded uniform random distribution, as is already done by default.

A.6 Example of a Complicated Problem

Returning to the parametrization of the VLBI problem from Chaper 4, consider the following partial seeding problem. We assume there are no multi-observation epochs in this example.

$$X_{\text{candidate}} = \boxed{6}\boxed{3}\boxed{1}\boxed{2}\boxed{3}\boxed{11}\boxed{4}\boxed{5}\boxed{6}\boxed{72}\boxed{1}\boxed{4}\boxed{5}\boxed{13} \tag{A.11}$$

$$X_{1,\text{ archive}} = \boxed{6}\boxed{2}\boxed{1}\boxed{3}\boxed{5}\boxed{11}\boxed{4}\boxed{5}\boxed{6}\boxed{72} \tag{A.12}$$

$$X_{2,\text{ archive}} = \boxed{5}\boxed{2}\boxed{1}\boxed{2}\boxed{3}\boxed{11}\boxed{1}\boxed{4}\boxed{5}\boxed{101} \tag{A.13}$$

$$X_{3,\text{ archive}} = \boxed{5}\boxed{3}\boxed{1}\boxed{2}\boxed{3}\boxed{11}\boxed{1}\boxed{3}\boxed{4}\boxed{72}\boxed{1}\boxed{4}\boxed{5}\boxed{13} \tag{A.14}$$

In this more complicated problem, we have 3 solutions in the archive, two of which are partially like the candidate chromosome. We can use the rules in section A.4 to determine which archived chromosome is the most like the candidate. For clarity we parse the chromosomes into header and phases and compare similarities.

$$X_{\text{candidate}} = \underbrace{\boxed{6}\boxed{3}}_{\text{Header}}\underbrace{\boxed{1}\boxed{2}\boxed{3}\boxed{11}}_{\text{Phase 1}}\underbrace{\boxed{11}\boxed{4}\boxed{5}\boxed{6}\boxed{72}}_{\text{Phase 2}}\underbrace{\boxed{72}\boxed{1}\boxed{4}\boxed{5}\boxed{13}}_{\text{Phase 3}} \tag{A.15}$$

$$X_{1,\text{ archive}} = \underbrace{\boxed{6}\boxed{2}}_{\text{Header}}\underbrace{\boxed{1}\boxed{3}\boxed{5}\boxed{11}}_{\text{Phase 1}}\underbrace{\boxed{11}\boxed{4}\boxed{5}\boxed{6}\boxed{72}}_{\text{Phase 2}} \tag{A.16}$$

$$X_{2,\text{ archive}} = \underbrace{\boxed{5}\boxed{2}}_{\text{Header}}\underbrace{\boxed{1}\boxed{2}\boxed{3}\boxed{11}}_{\text{Phase 1}}\underbrace{\boxed{11}\boxed{1}\boxed{4}\boxed{5}\boxed{101}}_{\text{Phase 2}} \tag{A.17}$$

$$X_{3,\text{ archive}} = \underbrace{\boxed{5}\boxed{3}}_{\text{Header}}\underbrace{\boxed{1}\boxed{2}\boxed{3}\boxed{11}}_{\text{Phase 1}}\underbrace{\boxed{11}\boxed{1}\boxed{3}\boxed{4}\boxed{72}}_{\text{Phase 2}}\underbrace{\boxed{72}\boxed{1}\boxed{4}\boxed{5}\boxed{13}}_{\text{Phase 3}} \tag{A.18}$$

We observe that archived chromosome 1's phase 2 matches the candidate's phase 2, archive 2's phase 1 matches the candidate's phase 1, and that both archive 3's phase 3 and phase 1 match those of the candidate. While archive 2 shares the first phase genes of the candidate, it shares no other phases so its similarity rank is zero. Both archive 1 and 3 share 1 nontrivial phase with the candidate, but since archive 3 also shares phase 1, it outranks archive 1. Therefore, the candidate will have phases 1 and 3 seeded from archive 3, while phase 2 will be seeded from a combination of uniform random parameters and shooting based on the seeded phases adjacent to it.

Once the outer-loop seed is selected, the seed sharing algorithm for the inner-loop is invoked to extract the necessary trajectory data for the phases to be seeded. The algorithm is a decision tree which fills in the unseeded phases of each spacecraft's trajectory within the fleet. First, the algorithm chooses a TOF from a bounded uniform random distribution for each unseeded phase. Then for each spacecraft, it determines which phases include the spacecraft partaking in a VLBI observation. Of those, it determines which phases are being seeded and which are not. Four possible seeding patterns may occur: 1) the spacecraft is only included in unseeded phases, 2) it

is only included in phase one (the phase is seeded), 3) it is only included in a single seeded phase that is not phase one, or 4) it is mentioned in multiple seeded phases between which there are one or more unseeded phases which may or may not include it. In case 1), the spacecraft's entire trajectory is chosen using the current uniform random distribution method. In cases 2) and 3), forward and/or backwards propagated Kepler arcs are generated from the seeded state to fill in the rest of the trajectory. In case 4), a Lambert solver is used to target the maneuvers to transfer the spacecraft between "bookending" states. Then if there is more than one unseeded phase between any two seeded phases, the Lambert state at the final bookend may be used to propagate a Kepler arc backwards in time through each intermediate phase. This process must then be repeated for every instance of a case 4) pattern in the chromosome. The seed sharing technique is depicted in its entirety in Fig. A.3.

The inner-loop seeding process for this example in Eq. A.15 is captured in Fig. A.4. In this example, we have seed data for spacecraft 1, 2 and 3 for phase 1 as well as data for spacecraft 1, 4, and 5 in phase 3. Phase 2 is unseeded, ergo, the initial and final states of all spacecraft during this phase must be solved for. We begin by choosing the time of flight for this phase from a uniform random distribution. Recall from Chapter 4 that the phase TOF is bounded between 14 and 365 days. For spacecraft 1, we know its final state in phase 1 and its initial state in phase 3. To complete its mission trajectory, we use the Lambert solver to target the maneuvers required to transfer between these two states within the chosen phase 2 TOF. We may not use Kepler propagation because the two states are not guaranteed to fall on the same conic section. For spacecraft 2 and 3, we have no information about their states in phases 2 and 3, so we may simply use forward Kepler propagation without maneuvers to generate their trajectories during these phases. In the case of spacecraft 4 and 5, the only information we have is their trajectory data for phase 3. To obtain the trajectories which lead to the desired states in phase 3, we can back-propagate Kepler arcs without maneuvers. Finally, spacecraft 6 has no seed data associated with it, so its trajectory parameters are chosen entirely from the bounded uniform random distribution. Once all the targeting of solve for parameters is completed, the entire fleets trajectories have been enumerated and this resulting

seed is the first initial guess passed to the NLP solver in the inner-loop.

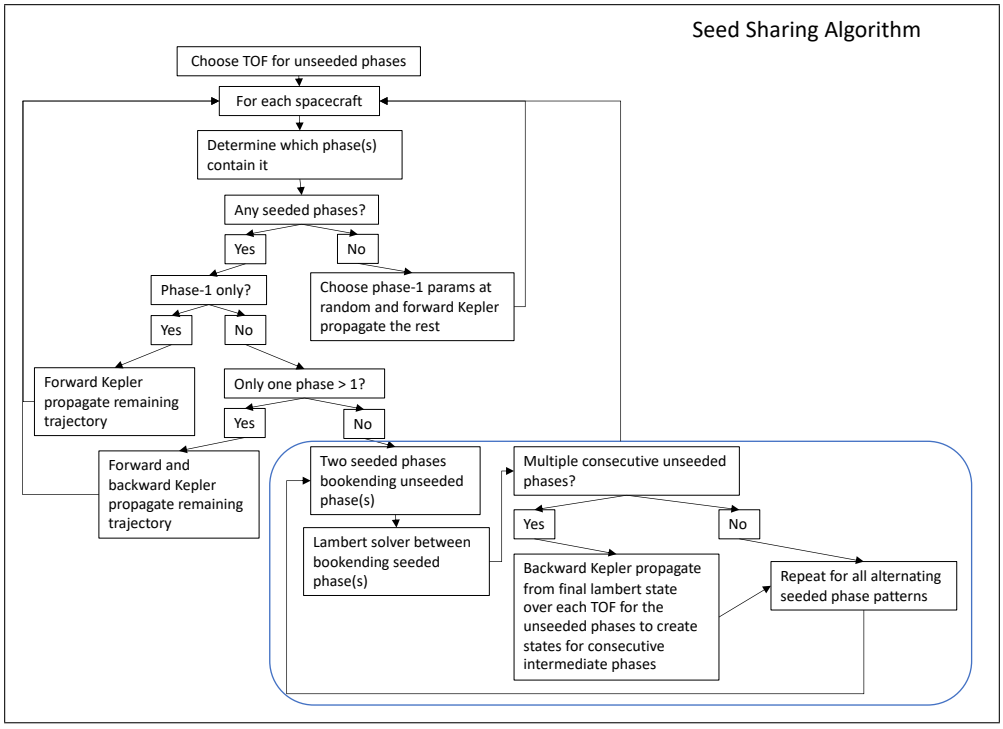


Figure A.3: The inner-loop seed sharing decision tree algorithm.

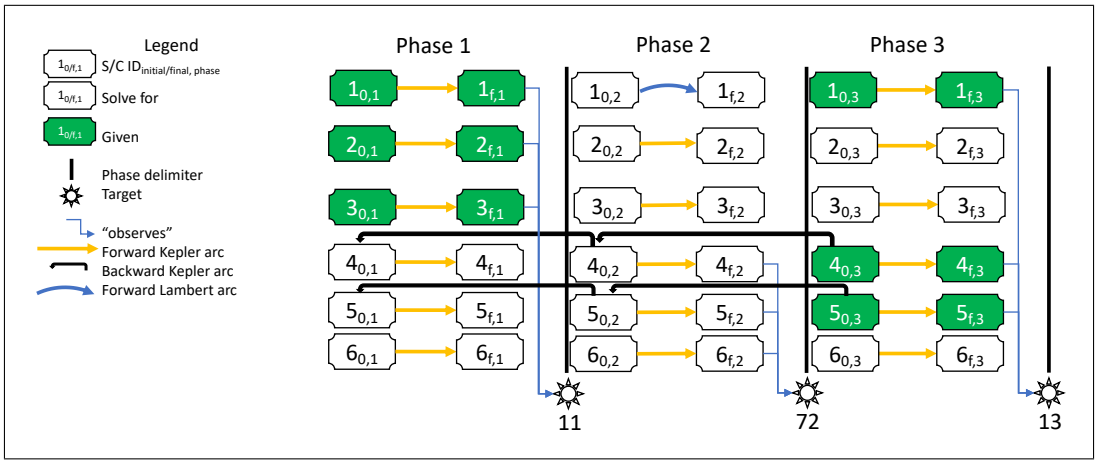


Figure A.4: The seeding of the candidate mission in Eq. A.15 using the archived mission in Eq. A.18.

A.7 Experimental Design

The results of the work in Chapter 4 yielded a pool of over 1000 converged missions. These mission fleet structures will be used as the first test data set using the new HashMap archive. The proposed experiment is outlined thusly.

1) Create a brand new generation of candidate chromosomes as usual in NSGA-II.

2) Duplicate this generation, and used partial seeding on one group, no seeding on the other.

3) Perform evaluation of the experimental and control group in parallel.

4) The MBH inner-loop execution on a particular chromosome stops if and when the first $\Delta V = 0$ solution is found and the elapsed MBH hop time and number of hops are recorded as fields in the solution fleet data structure.

5) Repeat this entire process for an NSGA-II run of 100 generations.

6) Store solution fleets every generation for each group (not just the Pareto fronts) for analysis afterward.

7) Calculate the mean and standard deviation of the MBH hops and runtime parameters for each group. Draw conclusions on efficacy of this seeding technique.

A.8 Objectives and Anticipated Findings of the Experiment

The ultimate goals of creating and applying this novel seeding technique are twofold. 1) Achieve quicker convergence, on average, for a solution mission to the global optimum via the inner-loop. 2) Identify "fitter" sequences to amplify the reach of the outer-loop algorithm. By enhancing the HashMap archive into seed sharing capability, the-outer loop will be able to "learn" which phase sequences are best combined to produce longer, more complex missions.

Currently, the outer-loop generates new chromosomes via genetic crossover of elite parents followed by random mutation. These chromosomes are then initialized into missions by the inner-loop by drawing parameters from a bounded uniform random distribution– a process which is inefficient, relying on MBH to find a data point which leads the NLP solver to the global optimum over many iterations. The seed sharing capability will potentially expedite the inner-loop and outer-loop, but also introduces a potential inflection point in the outer-loop design of future work. This seed sharing technique could be leveraged to create a different method of child chromosome creation: smart seeding.

We can introduce a criterion (i.e. iteration) at which point the outer-loop transitions from creating chromosomes via crossover and mutation to creating them by stitching together multiple fit sequences from the archive. For example, as the outer-loop evaluates more and more generations, two particular target sequences may appear frequently on the non-dominated front: a 4-target sequence and a completely different 3-target sequence. After the outer-loop plateaus and struggles to add new solutions to the Pareto front, it could attempt to search the decision space by chaining together, for instance, the 4-target and 3-target sequences and solve for a 7-target sequence. The outer-loop could also choose to prepend, append or bookend these converged sequences with a randomly generated sequence to solve for, say, a 25-target mission. Smart seeding thus has the potential to facilitate continued exploration of the outer-loop search space and find complex solutions that would take far longer to find purely stochastically.