A Parallel Processing and Diversified-Hidden-Gene-Based Genetic Algorithm Framework for Fuel-Optimal Trajectory Design for Interplanetary Spacecraft Missions

a project presented to
The Faculty of the Department of Aerospace Engineering
San José State University

in partial fulfillment of the requirements for the degree
Master of Science in Aerospace Engineering

by

Dhathri H. Somavarapu

May 2017

approved by

Dr. Kamran Turkoglu
Faculty Advisor

The Designated Thesis Committee Approves the Thesis Titled

A PARALLEL PROCESSING AND DIVERSIFIED-HIDDEN-GENE-BASED GENETIC ALGORITHM FRAMEWORK FOR FUEL-OPTIMAL TRAJECTORY DESIGN FOR INTERPLANETARY SPACECRAFT MISSIONS

by

Dhathri H. Somavarapu

APPROVED FOR THE DEPARTMENT OF AEROSPACE ENGINEERING

SAN JOSÉ STATE UNIVERSITY

May 2017

Dr. Kamran Turkoglu	Department of Aerospace Engineering
Dr. Nikos Mourtos	Department of Aerospace Engineering
Prof. Jeanine Hunter	Department of Aerospace Engineering

ABSTRACT

A PARALLEL PROCESSING AND DIVERSIFIED-HIDDEN-GENE-BASED GENETIC ALGORITHM FRAMEWORK FOR FUEL-OPTIMAL TRAJECTORY DESIGN FOR INTERPLANETARY SPACECRAFT MISSIONS

by Dhathri H. Somavarapu

This thesis proposes a new parallel computing Genetic Algorithm framework for designing fuel-optimal trajectories for interplanetary spacecraft missions. The framework can capture the deep search-space of the problem with the use of a fixed chromosome structure and hidden-genes concept, can explore the diverse set of candidate solutions with the use of the Adaptive and Twin-Space Crowding techniques, can execute on any High-Performance Computing (HPC) platform with the adoption of the portable Message Passing Interface (MPI) standard. The algorithm is implemented in C++ with the use of the MPICH implementation of the MPI standard. The algorithm uses a patched-conic approach with two-body dynamics assumptions. New procedures are developed for determining trajectories in the V_∞-Leveraging legs of the flight from the launch and non-launch planets, and deep-space maneuver legs of the flight from the launch and non-launch planets. The chromosome structure maintains the time of flight as a free parameter within certain boundaries. The fitness or the cost function of the algorithm uses only the mission ΔV, and does not include time of flight. Optimization is conducted with two variations for the mission gravity-assist sequence, the 4-gravity-assist and the 3-gravity-assist, with a maximum of 5 gravity-assists allowed in both the cases. In both the variations, an optimal trajectory is found with a mission cost (total ΔV) comparable to the cost of the bench mark Cassini 2 mission of Gad and Abdelkhalik [1].

DEDICATION

To my loving mother, for her unwavering support.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincere gratitude to Dr. Kamran Turkoglu, my advisor, whose guidance has been a tremendous factor in my educational and research achievements. His drive to meet the highest of standards in the work produced by his students has helped me greatly and is highly respected and appreciated.

Second, I would like to thank Dr. Nikos Mourtos and Prof. Jeanine Hunter from the Aerospace Engineering department for being generous with their time and knowledge to serve on my thesis committee.

Third, I would like to thank Mr. Sean Fritz, a former classmate of mine and a current employee of the Lockheed Martin Space Company, for introducing me to Genetic Algorithms, for countless discussions on topics related to my thesis, and for allowing me to use a gravity-assist feasibility method that he developed along with Dr. Turkoglu.

Last but not the least, I would like to thank my Mother and Sister, for their unwavering support throughout my long pursuit of this degree.

TABLE OF CONTENTS

CHAPTER

LIST OF TABLES

Table

LIST OF FIGURES

Figure

LIST OF PROCEDURES

CHAPTER 1

INTRODUCTION

Humans have long aspired to explore other worlds in search of resources and extraterrestrial life. While all other major planets in the Solar system are currently not hospitable to forms of life as we know it, the planetary moons such as Europa, Titan, and Enceladus are believed to have underneath their outer crusts liquid oceans that could potentially support microbial life forms such as those that exist on Earth [4–6]. Despite widespread agreement based on existing data indicating the existence of a salt-water ocean underneath Europa's icy crust, this remains to be confirmed by future missions [4]. The same can be said about the two moons of Saturn. Given the significance of proving the existence of salt-water oceans and possible microbial life on these moons, there is significant interest within the scientific community in pursuing missions to these moons of the outer planets.

Missions to planetary moons are usually designed in two phases. The first phase is the interplanetary voyage to the sphere-of-influence of the parent planet. The second phase involves designing trajectories to do one or more of three things: (1) multiple fly-bys of the moons, (2) launching a probe to the surface of a moon, or (3) getting into and maintaining an orbit around one or more of the moons. In either of those 3 phases or in the voyage to the parent planet, any number of planetary or moon gravity-assists and deep-space maneuvers (DSMs) are used. This thesis proposes a framework for determining a gravity-assist based fuel-optimal trajectory to a parent planet, such as Saturn or Jupiter.

A space mission strives to maximize the payload mass, while minimizing the

launch energy and total ΔV required to achieve the mission. Hollenbeck [7] introduced the concept of an extra-deep space maneuver for decreasing launch energy and total ΔV, calling it ΔV-Earth-Gravity-Assist. Sims and Longuski [8] used the term V_∞-leveraging in their expanded analysis of the ΔV-Earth-Gravity-Assist maneuver, to formalize the deep-space maneuver recommended by Hollenbeck [7]. Sims, Longuski, and Staugler [9] extended this analysis to a more generalized V_∞-leveraging technique to apply to any Solar planet, especially to missions to inner planets. Brinckerhoff and Russel [10] successfully applied the V_∞-leveraging technique to the problem of a phase-fixed Jovian moon tour, albeit with more flight time than that of a regular Hohmann transfer. Strange, Compagnola, and Russell [11] developed a novel non-tangential V_∞-leveraging technique to achieve effective gravity-assists around low-mass moons in terms of time of flight. This would otherwise be impractical, given the insufficient bending provided by the low mass moons using the traditional V_∞-leveraging technique. Compagnola, Russell, and Strange [12] utilized this non-tangential V_∞-leveraging technique to design an optimal mission to place an orbiter around the moon Enceladus of the planet Saturn with a ΔV requirement of only 445 m/s over that of 4 km/s for the regular Hohmann transfer, at expense, however, of extending the flight time to 2.7 years.

The V_∞-leveraging technique, along with gravity-assists and other deep-space maneuvers, has become a mainstay of interplanetary missions. The challenge of finding the correct sequence of these operations for a given launch and target date, however, is immense. This is because of the depth of the search space involved in finding an optimal solution. To address this immense challenge, this thesis explores a class of non-deterministic evolutionary algorithms, Genetic Algorithms, known to provide near-global optimal solutions, from a large search space of the problem

domain. Genetic Algorithms use selection, crossover, and mutation operators on the candidate solutions to mimic the natural evolutionary processes found in nature. This allows for a near-optimal solution in the best-case scenario. Genetic Algorithms are "non-deterministic" because they may in some cases lead to local-optimal or practically infeasible solutions. Because of the search space depth, it is standard practice to limit the design space of the problem to a prescribed number of gravity-assists, V_∞-leveraging maneuvers and general deep-space maneuvers. Not all missions have the same number of design parameters (genes). Gad and Abdelkhalik [1] presented a novel approach where the number of design parameters (genes) are fixed for all conceivable problems with some of the parameters (genes) designated as "hidden" depending on the nature of the particular problem being solved. These "hidden" genes are not used in the fitness evaluation of a candidate solution. This more generalized Genetic Algorithm, which applies to any kind of interplanetary mission problem, can provide the optimal sequence of maneuvers-as well as the magnitudes of velocities and locations of the maneuvers-for the available launch and target dates. In their analyses of known missions to Mars, Jupiter, and Mercury, their algorithm could generate the actual known optimal solutions, in some cases, with improvements. Gad and Abdelkhalik [13] presented another novel approach to this trajectory optimization problem using the variable size design parameters (variable-size genes in a chromosome). In this approach, Gad and Abdelkhalik [13] restricted the problem design space to one that obeys the solutions to multiple-revolution Lambert's problem, within the realm of the two-body dynamics model.

Gad and Abdelkhalik's [1] hidden-gene Genetic Algorithm works in two phases because of the prohibitive computational cost (time) involved in implementing that algorithm directly in a single phase. The first phase computes

the optimal sequence of gravity-assist planets. The second phase refines the first-phase solution by adding deep-space maneuvers (DSMs). The algorithm proposed in this thesis employs the same concept of hidden genes. However, in this thesis, the algorithm is improved in terms of its computational cost by employing an industry standard parallel computation framework known as the Message Passing Interface [14], thereby avoiding the need to separate the algorithm into two phases.

Achieving population diversity is a very common challenge in Genetic Algorithms. Population diversity enables the Genetic Algorithm to explore vast swathes of the problem search-domain, thus increasing the likelihood that the solution will be globally optimum, thereby preventing the algorithm from getting stuck at a local optimum. Two different techniques, Niching and Crowding, have emerged during the past several years as solutions to this challenge. Beasly, Bull, and Martin [15] originally proposed the Niching technique as a means of achieving population diversity in retrieving solutions to a multi-modal optimization problem. The use of Niching technique requires the knowledge of the Niche-radius apriori. The Niche-radius is *not* known apriori for the problem of this thesis and is highly likely that it is not constant. Due to this, the Niching technique is *not* considered. Crowding is a technique that determines the selection of individuals from a current generation to carry over to the subsequent generation, in such a way that the population diversifies with each generation. Some crowding techniques require the knowledge of search space. However, the Twin-Space crowding technique proposed by Chen, Chou, and Liu [3] does *not* require prior knowledge of the search space to produce offspring. The Twin-Space Crowding technique has shown to diversify the population significantly with little to no knowledge of the search space. The problem of this thesis requires that the Genetic Algorithm explore as much of the search space as possible with as little knowledge of the search space as possible. Due

to this, this thesis utilizes the Twin-Space crowding technique proposed by Chen, Chou, and Liu [3]. Population diversity is also highly dependent on the crossover and mutation probabilities in the Genetic Algorithm. Srinivas and Patnaik [16] proposed the concept of adaptive crossover and mutation probabilities for each chromosome based on the knowledge of the cumulative and individual fitness/cost characteristics of the population. In this approach, the most fit chromosomes are protected from being disrupted, increasing the possibility of carrying them over to next generation. At the same time, chromosomes with less than average fitness of the population are disrupted with higher crossover and mutation probabilities to help infuse the population with potentially new and unexplored solution candidates, in a maximization problem. In this thesis, the adaptive crossover and mutation probabilities technique of Srinivas and Patnaik [16] is employed. The technique is adapted to the minimization problem of this thesis as described in section 4.2.5.

The orbital mechanics procedures developed in this thesis make use of two-body orbital dynamics. In the actual missions, when a spacecraft flies by a planet for a gravity-assist, the effects of the moons of the planet on the resultant trajectory of the spacecraft must be considered. Developing an algorithm to consider n-body effects during a gravity-assist is very complex and may not be necessary during the preliminary analysis of the optimal trajectory candidates. In practice, the preliminary analysis only considers two-body dynamics. The candidate trajectories determined from the preliminary analysis are further refined for determination of feasibility by taking the n-body effects into consideration. For example, the Cassini mission to Saturn was designed in two phases as described by Peralta and Flanagan [17]. The VVEJGA trajectory of the Cassini mission was developed using two optimization programs developed at the Jet Propulsion Laboratory. The first program, MIDAS, uses the two-body orbital dynamics and

the patched conic method to determine the preliminary feasible trajectories. The second program, PLATO, uses multi-conic (n-body) propagation methods to refine the feasible trajectories for safety of the spacecraft and success of the mission. The refinement of preliminary feasible trajectories is not considered in this thesis. The goal of this thesis is to facilitate the preliminary analysis. Hence the use of the two-body dynamics is justified.

This study was prompted by the need for an improved means of interplanetary trajectory design accessible in the academia. Given the interest in future missions to Jupiter's Europa [18], Saturn's Enceladus and Titan moons [19], the need for charting fuel-optimal trajectories to the parent planets Jupiter and Saturn is immense. The trajectories determined using the algorithm developed here can be used for initial trade studies on candidate trajectories.

CHAPTER 2

PROBLEM STATEMENT AND THESIS OUTLINE

2.1 Problem Statement

This thesis focuses on the problem of developing a computationally efficient general algorithm framework for fuel-optimal interplanetary trajectory and mission design within the Solar System. The requirements for this algorithm are as follows:

(1) Because of the vastness of the search space involved in this problem, the algorithm must be capable of generating and evaluating diversified candidates from the problem search space.

(2) The algorithm should be reasonably fast, i.e., finishing in days, as opposed to several weeks, and in hours rather than several days, depending on the size of the search space.

(3) The algorithm should be generic enough to accommodate variable number of problem parameters among competing candidates for an optimal solution.

2.2 Thesis Outline

The algorithm developed in this thesis is presented in the following manner:

(1) The various appropriate orbital mechanics problems utilized are discussed in Chapter 3.

(2) The Genetic Algorithm, along with the chromosome structure and the Twin-space Crowding technique, is presented in Chapter 4.

(3) Implementation and the parallelization mechanism are explained in Chapter 5.

(4) The results obtained by applying the algorithm to the problem of finding a fuel-optimal trajectory to Saturn are presented in Chapter 6.

(5) Conclusions and recommendations are given in Chapter 7.

CHAPTER 3

ORBITAL MECHANICS FUNDAMENTALS

This thesis employs various solutions to two-body problems in astrodynamics. The basic approach used is patched-conic [20]. In the actual missions, the n-body effects on the spacecraft must be considered for a gravity-assist maneuver, due to the presence of moons of the gravity-assist planet within the sphere-of-influence of the gravity-assist planet. In practice, for an interplanetary mission design, preliminary analysis on the possible candidate trajectories is conducted using two-body dynamics and patched-conic method. The candidate solutions obtained from the preliminary analysis are refined for feasibility in the presence of n-body effects in the gravity-assist maneuvers involved in a trajectory. Since the objective of this thesis is the development of an efficient algorithm for the preliminary analysis, the refinement process considering the n-body effects is not considered. Thus, the gravity-assist feasibility procedure developed by Fritz and Turkoglu [21], which does *not* take n-body effects into account is employed in this thesis. This chapter outlines and describes the various maneuvers used in the solution to the fuel-optimal trajectory design problem. This work is restricted to trajectories with either multiple gravity-assists (MGA) only, or gravity-assists with one single deep-space maneuver (DSM) in between each of the possible gravity-assist maneuvers (MGA-1DSM). When the trajectory calls for consecutive gravity-assists from the same planet, the V_∞-leveraging maneuver (VILM) is used.

3.1 Kepler's Problem

In the realm of classical orbital mechanics, the problem of tracking a celestial object's position and velocity as a function of time is known as Kepler's problem. The problem addressed by this thesis requires that the position and velocity vectors of all planets and the spacecraft be known at all times under consideration. In this thesis, ephemerides of the planets are known apriori using the Horizons tool, provided by the Jet Propulsion Laboratory [22]. For tracking the position and velocity of the spacecraft, a universal variable-based solution provided by Curtis [20] in Matlab has been converted into C++.

3.2 Lambert's Problem

The problem of finding required velocities, when two positions and time-of-flight in between are given, is known as Lambert's problem [20]. In this problem, a single revolution of the celestial body around the central body of gravitational influence is assumed. In this thesis, the universal variable-based solution to this problem provided by Curtis [20] in Matlab has been converted into C++.

3.3 Multiple-Revolution Lambert's Problem

This problem is a variation of the regular Lambert's problem, involving multiple revolutions of the celestial body around the central body. In this thesis, a novel method developed by Izzo [23] is employed for solving multiple-revolution Lambert's problem.

3.4 Gravity-Assist Dynamics

The gravity-assist maneuver helps to gain or shed the mechanical energy of the spacecraft, depending on the mission requirement. There are two kinds of gravity-assist maneuvers: non-powered and powered. This thesis employs both the types of the gravity-assist maneuver. When the leg of the flight is a Lambert's leg, the powered gravity-assist maneuver is employed. When the leg of the flight includes a deep-space maneuver, the non-powered gravity-assist maneuver is used. In non-powered gravity-assist maneuvers, the incoming and outgoing V_∞ of the spacecraft with respect to the planet is the same in magnitude. In powered gravity-assist maneuvers, they are not equal, because a Δv maneuver is conducted at the periapse of the hyperbolic trajectory with respect to the planet. The mechanical energy gained or shed is significant, helping to reduce the cost of the mission in terms of fuel required.

For non-powered gravity-assist maneuvers,

$$\left|v_\infty^-\right| = \left|v_\infty^+\right| = v_\infty \tag{3.1}$$

$$sin(\frac{\delta}{2}) = \frac{\mu_p}{\mu_p + r_{per} v_\infty^2} \tag{3.2}$$

and,

$$\left|\Delta v_{nps}\right| = \left|v_\infty^+ - v_\infty^-\right| = 2 v_\infty sin(\frac{\delta}{2}) \tag{3.3}$$

For powered gravity-assist maneuvers,

$$\Delta v_{ps} = (v_{s/c}^+)_{req} - (v_{s/c}^+)_{nps} \tag{3.4}$$

where, $(v^+_{s/c})_{req}$ is the spacecraft's required outgoing heliocentric velocity and,

$$(v^+_{s/c})_{nps} = v_p - v^+_\infty \tag{3.5}$$

Here v_p represents the heliocentric velocity of the gravity-assist planet.

Knowing the radius of the periapse, r_p, of the hyperbolic trajectory of the spacecraft and the incoming v_∞ of the spacecraft, enables us to solve the gravity-assist maneuver.

3.4.1 Gravity-Assist Feasibility

A special case in this study requires determination of feasibility of gravity-assist from a planet, given the required parameters for the gravity-assist. The required parameters are: the inbound and outbound heliocentric velocity vectors of the spacecraft, the heliocentric velocity vector of the gravity-assist planet, the radius of the gravity-assist planet, the gravitational parameter of the gravity-assist planet, and the tolerance for the bending angle of the hyperbolic trajectory of the spacecraft from the gravity-assist. A method developed by Fritz and Turkoglu [21] is used to determine the feasibility of the gravity-assist from the given planet. This method applies the Newton-Raphson iteration scheme, for determining feasibility.

3.5 Deep-Space Maneuver Modeling

A deep-space maneuver aids with conducting a non-powered gravity-assist maneuver. When employing a deep-space maneuver, the standard practice is to conduct a ΔV maneuver at a location in the transfer orbit, in such a way that the spacecraft can get a free (non-powered) gravity-assist from another planet. During a

leg of the flight, the position of velocity of the spacecraft at the starting planet are known. The time of flight from the starting planet to the position in transfer orbit where the deep-space maneuver is to be conducted is also known. Using the solution to Kepler's problem, the exact position and velocity vectors of the spacecraft (in the transfer orbit) are calculated for the deep-space maneuver. An instantaneous tangential ΔV burn is assumed at this location. The position vector obtained from the solution to Kepler's problem is used in the subsequent procedure, to determine the required velocity vector at this location.

To determine the velocity vector of the deep-space maneuver, we first consider the following known parameters: (1) the position and velocity vectors of the ending planet in the current leg of flight and (2) the time of flight from the deep-space maneuver location to the ending planet. Using these data, Lambert's problem is solved, to determine the required velocity vectors at the deep-space maneuver location and that of the ending planet.

3.6 V_∞-Leveraging Maneuver

The V_∞-leveraging maneuver is defined as a relatively small deep-space maneuver to modify V_∞ at a body such as Earth [9]. The maneuver, when timed properly, in conjunction with a gravity-assist from the same body, can significantly reduce the launch energy requirement [7]. It should be noted here that this technique can be applied to any planetary body or moon from which multiple gravity-assists are sought. It should also be noted that the method for determining the maneuver details (such as location, magnitude, and direction) is numeric in nature. Because of this, the problem domain and the design or solution space can be extended to include trajectories that involve multiple revolutions of a planet and

the spacecraft. In this thesis, the time-of-flight parameter for a leg of the flight is chosen arbitrarily, within certain boundaries. It is therefore beneficial to consider trajectories that involve multiple revolutions of the planet or the spacecraft.

3.6.1 A Procedure for V_∞-Leveraging Maneuver

The following procedure is employed in solving for the parameters of the V_∞-Leveraging Maneuver.

(1) First, the position and velocity vectors of the spacecraft are determined at the location of the DSM using the solution to Kepler's problem.

(2) Second, Kepler's problem is used again to verify that the maneuver is possible without a DSM. If such a trajectory is feasible, the procedure concludes there.

(3) Third, if such a trajectory is not feasible, the solution to Lambert's problem(s) is employed to verify if the trajectory is feasible with a DSM.

There are two different conditions under which V_∞-leveraging maneuver is employed in the current study.

3.6.2 The V_∞-Leveraging Maneuver from the Launch Planet

In this special case, a gravity-assist is sought from Earth after launching from Earth. In this case, the required hyperbolic excess velocity V_∞ at launch is not known, since the goal is to determine a DSM that would minimize this quantity. For this reason, a procedure is employed that iterates over a range of values for V_∞ to determine the value that results in minimum total ΔV with a DSM. The procedure

from section 3.6.1 is used repeatedly with different inputs bases on the V_∞ value of the current iteration.

3.6.3 The V_∞-Leveraging Maneuver from a Non-Launch Planet

In this special case, gravity-assists are sought from a non-launch planet sequentially, e.g., seeking gravity-assist from Mars after already flying by Mars immediately prior to the desired gravity-assist. In this scenario, the outbound heliocentric velocity vector of the spacecraft after the first gravity-assist from the planet is not known, because of the presence of the VILM DSM between the two gravity-assists. This velocity vector is required to know the position vector of the spacecraft at the VILM DSM location (given as a fraction of the time-of-flight of the entire leg of the flight between two planetary gravity-assists) and its corresponding velocity vector. To address this problem, the gravity-assist periapse radius and orientation (angle ζ) of the plane of gravity-assist are used as the problem parameters. In other words, the non-launch VILM procedure requires these two parameters for inputs. Using these two parameters, the heliocentric outbound velocity vector of the spacecraft from the first gravity-assist from the planet is determined using the equations listed as follows:

Figure 3.1 depicts the orientation of the outbound V_∞ of the spacecraft with respect to the planet. Here the vectors $\bar{b}_1$, $\bar{b}_2$, and $\bar{b}_3$ are defined as follows by Molenaar [2]:

$$\bar{b}_1 = \frac{\vec{V}_{\infty_{in}}}{\left\| \vec{V}_{\infty_{in}} \right\|_2}$$

$$\bar{b}_2 = \frac{\bar{b}_1 \times \vec{r_{pl}}}{\left\| \bar{b}_1 \times \vec{r_{pl}} \right\|_2}$$

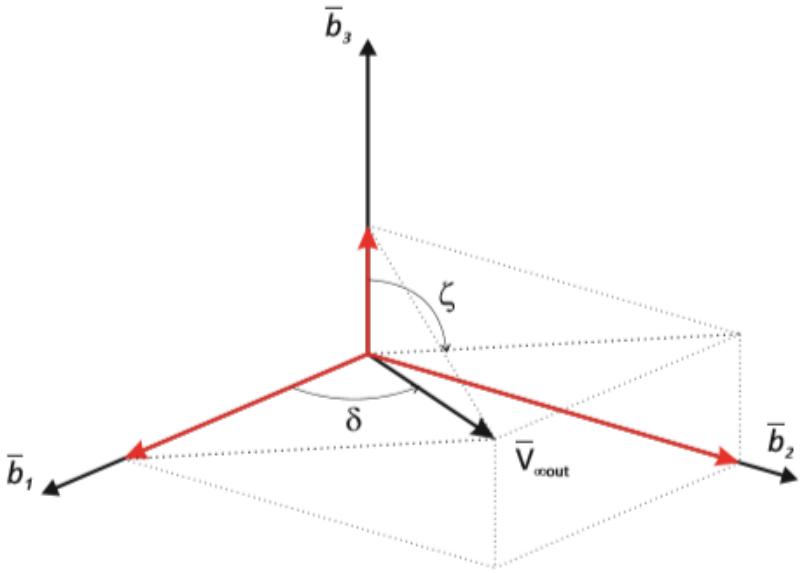

Figure 3.1: Gravity-Assist Orientation [2].

$$\bar{b}_3 = \bar{b}_1 \times \bar{b}_2$$

The angle δ represents the gravity-assist rotation/bending angle, while ζ represents the gravity-assist plane orientation angle, with δ is obtained as follows:

$$\delta = 2 \arcsin(\frac{1}{e}) \tag{3.6}$$

where the eccentricity e is calculated as

$$e = 1 + \frac{r_p \left\| \vec{V}_{\infty_{in}} \right\|_2^2}{\mu_{pl}} \tag{3.7}$$

The vector rotation with angles δ and ζ yields the following expression for $\vec{V}_{\infty_{out}}$:

$$\vec{V}_{\infty_{out}} = \vec{V}_{\infty_{in}}[cos(\delta)\bar{b}_1 + sin(\delta)sin(\zeta)\bar{b}_2 + sin(\delta)cos(\zeta)\bar{b}_3] \tag{3.8}$$

The outbound heliocentric velocity vector is obtained as follows:

$$\vec{V}_{out} = \vec{V}_{pl} + \vec{V}_{\infty_{out}} \tag{3.9}$$

Once $\vec{V}_{out}$ is computed, using the position vector $\vec{r}$ of the planet, the method described in sub-section 3.6.1 is used to compute the optimal DSM, to re-encounter the planet for a second gravity-assist.

CHAPTER 4

THE PROPOSED GENETIC ALGORITHM

4.1 The Basic Genetic Algorithms

Genetic Algorithms are a class of Evolutionary Algorithms that take an initial population or pool of candidate solutions from the problem search space, usually generated randomly within the bounds of the problem parameters, using an iterative process and evolutionary biological operators such as selection, crossover and mutation to repeatedly and progressively improve the initial population or pool toward the optimal solution(s) based on the fitness criteria for candidate solutions. The general structure of a Genetic Algorithm is shown in Figure 4.1.

Two important tasks required in the Genetic Algorithm are problem-specific. The first is the definition of the structure of the candidate solution, known as the chromosome in the parlance of Genetic Algorithms. The second task is the definition of the "fitness or cost" of the candidate solution, usually defined by the objective function for the optimization problem. It is possible for a candidate solution or chromosome to be either fixed in length or variable, depending on the problem at hand. Most or all of the individual parameters of the chromosome, also known as genes, are used in the computation of fitness.

4.1.1 Selection

The Genetic Algorithms employ various schemes to select the candidate solutions for the next generation from the current generation of candidate solutions. All the selection schemes use the fitness or the cost of the candidate solutions in the

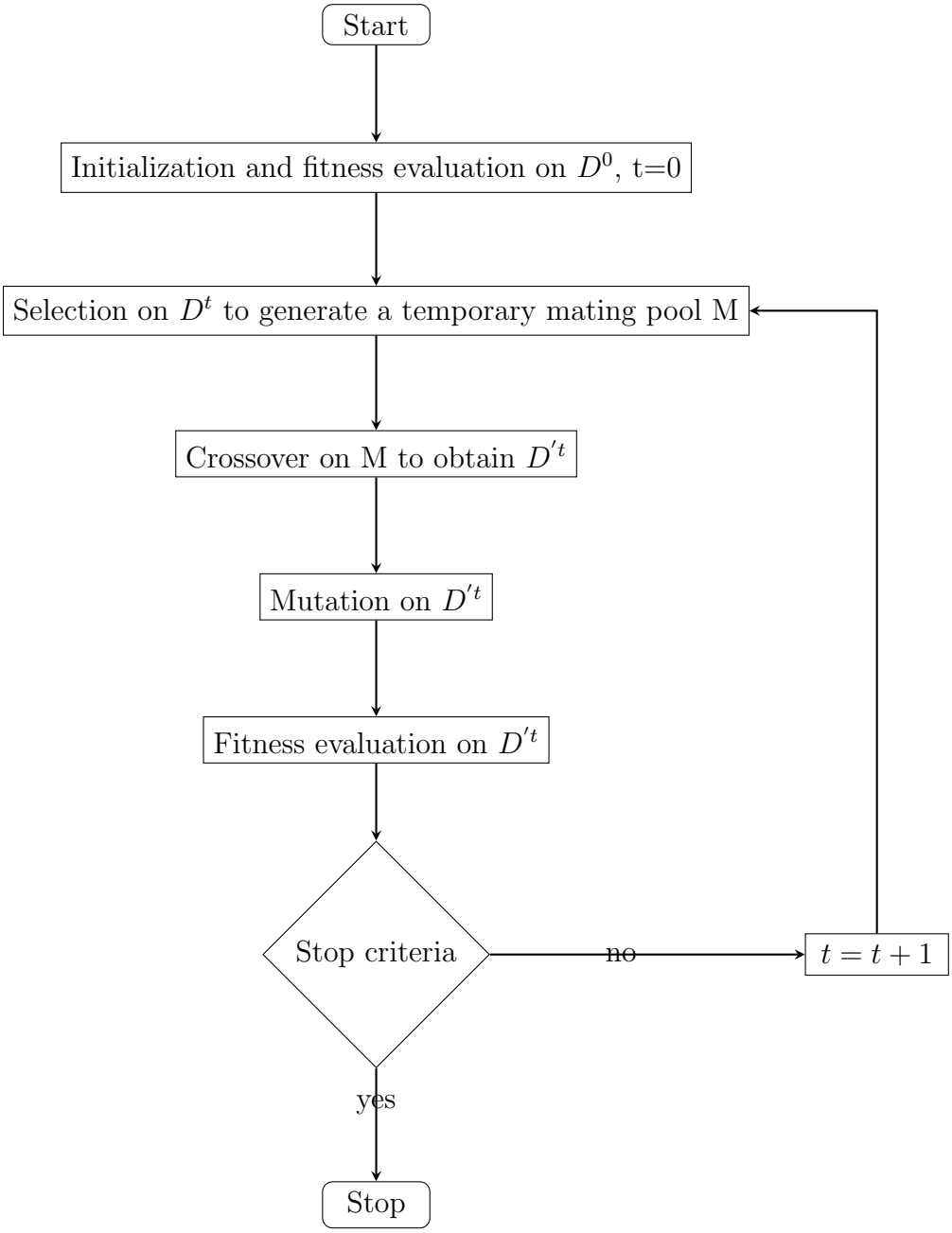

Figure 4.1: Genetic Algorithm Flow Chart

population as the criteria for selection. The purpose of selection is to carry forth the most fit candidate solutions to the next generation so that algorithm gets closer to the optimal solution. In all selection schemes, two candidate solutions often called parents, are selected to be passed along to the subsequent operators Crossover and Mutation. The most predominant selection schemes are the Roulette Wheel or the Fitness Proportionate selection and the Tournament selection.

The Roulette Wheel or The Fitness Proportionate Selection Scheme

The Roulette Wheel or the Fitness Proportionate selection works as follows:

(1) Compute the sum of the fitness of all candidate solutions in the population.

(2) Normalize the fitness of each of the candidate solution with the sum of fitness so that the fitness values fall between 0 and 1 for each candidate solution.

(3) Sort the candidate solutions based on the fitness value in descending order.

(4) Draw a random number between 0 and 1.

(5) The first candidate solution with the fitness value above the random number drawn is selected for next generation.

(6) Repeat random number draw and candidate solution selection N number of times, where N is the population size.

Tournament Selection Scheme The tournament selection, where tournament size is k, works as follows:

(1) Select k number of candidate solutions from the current population at random.

(2) Sort the k candidate solutions based on their fitness value in descending order.

(3) Pick the first candidate solution in the list, i.e. the candidate with the best fitness is selected.

4.1.2 Crossover

The crossover operator is equivalent to mating and reproduction of children in nature. The purpose of crossover is to diversify the next generation of population to get closer to the optimal solution for the problem. There are two most predominantly used crossover techniques, known as the Single-point crossover and the Two-point crossover. In both the techniques a threshold called crossover threshold is used to swap the genes of the parents to produce the children. In Single-point crossover, a single cutoff point is chosen randomly. Genes from parent 1 before the cutoff point and from parent 2 after cutoff point are used to generate child 1. Genes from parent 1 after the cutoff point and from parent 2 before the cutoff point are used to generate child 2. In Two-point crossover, two cutoff points are selected randomly. The two parts of the chromosome from parent 1 before the first cutoff point and after the second cutoff point and one part from parent 2 between the two cutoff points is merged to produce child 1. Similarly, the two parts of the chromosome from parent 2 before the first cutoff point and after the second cutoff point and one part from parent 1 between the two cutoff points are chosen to produce child 2.

4.1.3 Mutation

The mutation operator mimics the biological mutation process and works on individual genes of the chromosome. In practice, most predominantly used mutation scheme is the Gaussian mutation scheme. In this scheme, each gene has a predefined mutation threshold. During mutation, a random number is generated for each gene in the chromosome. If the random number is below the mutation threshold for the gene under consideration, another random number is generated from the Gaussian distribution. This Gaussian random number is multiplied with the standard deviation for the gene and the result is added to the current value for the gene. If the final value of the gene falls outside the boundaries for the gene, the value of the gene is adjusted to fit either minimum or maximum boundary for the gene as appropriate. The purpose of mutation is to diversify the next generation of the population to increase the possibility of finding the optimal solution.

4.2 Elements of the Proposed Genetic Algorithm

4.2.1 The Chromosome Structure

In the problem solved for this thesis, not all candidate solutions are the same in number of parameters or genes. It is possible for various candidate solutions to have different number of gravity-assists on the way to the target planet. It is also possible for various candidate solutions to have or not have a deep-space maneuver between two gravity-assist planets. In other words, the chromosome can have variable number of parameters or genes. In literature, so far there have been two main approaches to the problem of capturing variable number of genes in a chromosome. One is the hidden-gene concept proposed by Gad and Abdelkhalik [1]. Gad and Abdelkhalik [13] also propose the variable-size chromosome. In the

variable-size chromosome approach, the population of candidate solutions varies in size and in each iteration of the Genetic Algorithm the size of each candidate solution in the population keeps varying based on the fitness. In the hidden genes approach, a fixed size chromosome is proposed that can capture *all* possible candidate solutions. However, *not* all genes may be active or effective for the candidate solution. Some candidate solutions can have less than the maximum number of allowed gravity-assists, in which case the planets for gravity-assists is limited to only those up to the active number of planets for gravity-assists. This concept is illustrated with the proposed hidden-gene chromosomes in Figure 4.2.

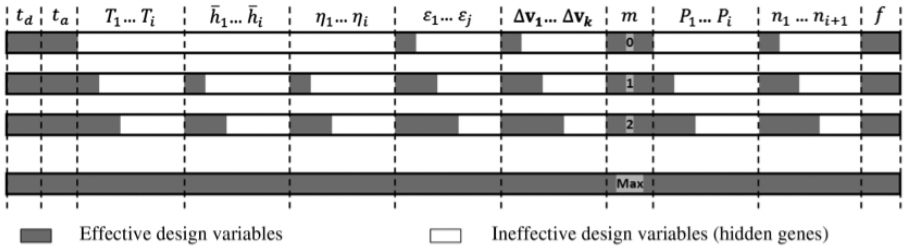

Figure 4.2: Hidden Gene Chromosomes by Gad and Abdelkhalik [1].

Figure 4.2 is showing the fixed size of the chromosome limited by a maximum of possible gravity-assists en-route to the target planet. The variable m denotes the actual number gravity-assists applicable in each candidate solution and is always less than or equal to the maximum number of gravity-assists allowed in the problem. t_d and t_a denote time of departure from Earth and time of arrival at the target planet respectively. i can vary from *1* to m. T_i denotes the time of flight in each leg flight to each of the gravity-assist planets. P_i denotes the identifier for the planet of the gravity-assist. f denotes either prograde or retrograde motion. h_i and η_i denote height of gravity-assist maneuver and a rotation angle in gravity-assist mechanism used by Gad and Abdelkhalik [1] respectively. ϵ_i, representing a fraction

of the T_i (a value between 0.1 and 0.9), is an epoch at which a deep-space maneuver is conducted in i^th leg of the flight. n_i indicates the number of deep-space maneuvers in a given leg of flight. Finally, ΔV_i represents the magnitude of a deep-space maneuver.

Gad and Abdelkhalik [1] use a two-phase approach. The first phase determines a solution containing an optimum number of gravity-assists. The second phase refines the optimal solution from first phase by introducing deep-space maneuvers in various legs of the optimal solution. According to them, this reduces the total time complexity of their algorithm significantly, despite not specifying the performance metrics of their algorithm in their published work on hidden-gene Genetic Algorithm. One possible drawback of their algorithm, that contributes to the increased time complexity if it were to be executed in one single phase, is the inclusion of the magnitudes ΔV_i of deep-space maneuvers in the chromosome. Given the vast range of values for this parameter, it is not possible to capture feasible values of this parameter with a population of 100 of less for example. This increases the problem search space immensely and thus increases the time complexity of the algorithm.

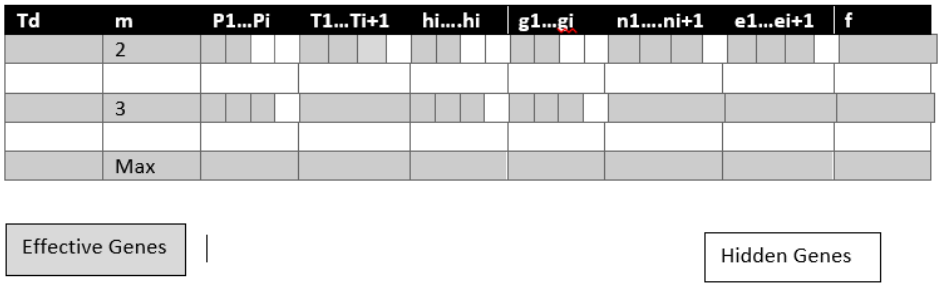

Figure 4.3: Proposed Hidden Gene Chromosomes

In this thesis, the general concept of hidden-gene chromosome is adapted, but with different number of parameters for the fixed chromosome, yet capturing the

Table 4.1: Genes of the Proposed Hidden Gene-based Chromosomes

Gene Name	**Description**
T_d	Time of departure captured as a Julian date value
m	Number of gravity-assists en-route to target planet
P_i	Integer identifier for the gravity-assist planet for leg i
T_i	Time of flight in seconds in a given leg of flight i
h_i	Height of periapse of gravity-assist trajectory around a given planet in leg i
g_i	Gravity-assist plane orientation angle in the gravity-assist planet centered frame of reference in leg i
n_i	0 or 1, indicating whether there is a deep-space maneuver in leg i, this must be set to 1 if a VILM is involved
e_i	A fraction between 0.1 and 0.9 of the time of flight for leg i, indicating the epoch at which the deep-space maneuver is conducted
f	Direction of flight, 0 for prograde, 1 for retrograde

problem search space at hand completely. The new chromosome structure proposed is shown in figure 4.3. Table 4.1 describes the genes of the proposed chromosome from this figure. As can be seen by comparing the figures 4.2 and 4.3, there are a number differences in the approach taken in the proposed chromosome structure. First is that time of arrival is not fixed, and is left out to be determined as the sum of the randomly chosen times-of-flight in each chromosome. This is done so to find the most optimal solution in terms of fuel consumption, albeit at the expense of mission time. Second, the gravity-assist mechanism used in this thesis is different from that used by Gad and Abdelkhalik [1]. Because of this difference the gravity-assist plane altitude and orientation angle are considered as genes in the proposed chromosome structure. Third, ΔV for the deep-space maneuver is *not* considered to be a part of the chromosome structure. The magnitude and direction of the deep-space maneuver is computed from other genes of the chromosome. This is done so to reduce the computation time of the algorithm. When deep-space maneuver ΔV is included in the chromosome structure, the number of chromosomes in the population to be evaluated to determine an optimal solution increases by multiple fold and results in increased time complexity for the algorithm. Because of the proposed new structure, the algorithm can be executed in one single phase as opposed to the two phases in which Gad and Abdelkhalik [1] execute their algorithm. Fourth, although a chromosome can have a deep-space maneuver in any leg of flight, the proposed algorithm does not always include the deep-space maneuvers in a leg of flight, it does so only when a better solution cannot be found using the regular Lambert's trajectory.

4.2.2 The Fitness of the Chromosome

The fitness function of the proposed chromosome structure from figure 4.3 needs to account for only the effective genes of the given chromosome. Procedure 4.1 captures the computation of fitness for the proposed chromosome structure.

Procedure 4.1 Procedure for Computation of the Fitness of a Chromosome

1: **procedure** FITNESS(*chromosome*,*flyby-limit*,*source*,*target*,*mu*,r_p,*ephemerides*,$config$)

Input:

chromosome $\rightarrow$ Chromosome for which the fitness is to be computed

flyby-limit $\rightarrow$ Maximum number of gravity-assists allowed in the GA

source $\rightarrow$ Identifier for the source planet for the trajectory

target $\rightarrow$ Identifier for the target planet for the trajectory

mu $\rightarrow$ A list of gravitational parameters of the Solar planets

r_p $\rightarrow$ A list of radii of the Solar Planets

ephemerides $\rightarrow$ Ephemerides of the Solar Planets in the time span chosen for the GA

$config$ $\rightarrow$ A list of configuration parameters for the GA

Output:

Fitness (Total ΔV of the trajectory) of the chromosome

2: *soilimits* $\leftarrow$ a composite list of minimum and maximum distances for the Sphere-of-Influence of Solar planets

3: $dV_{total} \leftarrow \infty$

4: *leo-height* $\leftarrow$ $config$.*leo-height*

5: $T_d \leftarrow chromosome.T_d$

6: $m \leftarrow chromosome.m$

7: $P_i \leftarrow chromosome.P_i$

Procedure 4.1 Procedure for Computation of the Fitness of a Chromosome (continued)

$T_i \leftarrow chromosome.T_i$
$h_i \leftarrow chromosome.h_i$
$g_i \leftarrow chromosome.g_i$
$n_i \leftarrow chromosome.n_i$
$e_i \leftarrow chromosome.e_i$
$directionn \leftarrow chromosome.f$
$dir \leftarrow (direction == 0)?true : false$
for $i \leftarrow$ source to target **do** ▷ Initialize the position and velocity vectors of the planets in P_i at T_i
    $rv \leftarrow$ VEC-INTERP$(T_d + \frac{\Sigma T_i(1:i)}{24*60*60}), ephemerides.slice(P_i(i)))$
    $r(i) \leftarrow rv.\vec{r}$
    $v(i) \leftarrow rv.\vec{v}$
end for
if $source == P_i(0)$ **then** ▷ Compute the feasibility and ΔV for the first leg flight
    $result \leftarrow$ VILT-LAUNCH($\vec{r}$(source), $\vec{v}$(source), $vecr(P_i(0))$, $T_i(0)$, $e_i(0)$, mu(Sun), soilimits(source))
    $V_\infty \leftarrow result.V_\infty$
else if $n_i(0) == 1$ **then**
    $result \leftarrow$ LAUNCH-DSM$(r(source), v(source), T_i(0), e_i(0), r(P_i(0)), direction, mu(Sun))$
    $V_\infty \leftarrow result.V_\infty$
else
    $lOutput \leftarrow$ LAMBERT$(r(source), r(P_i(0)), T_i(0), mu(Sun), dir)$
    if The Single-revolution Lambert is Not Feasible **then**
        $lOutput \leftarrow$ MULTI-REV-LAMBERT$(r(source), r(P_i(0)), T_i(0), mu(Sun), dir)$
    end if
    if Either of Lambert solutions converged **then**
        $v_{sc_{leo}} \leftarrow \sqrt{\frac{mu(source)}{r_p(source)+leo\text{-}height}}$
        $v_{sc} \leftarrow \sqrt{\left\| lOutput.\vec{V}1 - \vec{v}(source) \right\|^2 + \frac{2mu(source)}{r_p(source)+leo\text{-}height}}$
        $dV(1) \leftarrow \left| v_{sc} - v_{sc_{leo}} \right|$
        $result \leftarrow Result(0, dV(1), lOutput.\vec{V}2, 0)$
    else
        $result \leftarrow Result(\infty, \infty, \vec{empty}, \infty)$
    end if
end if
if $result.V_\infty == \infty$ or $result.\Delta V == \infty$ **then**
    return ∞
end if
$safety \leftarrow true$
$angle\text{-}tol \leftarrow 1e-3$

Procedure 4.1 Procedure for Computation of the Fitness of a Chromosome (continued)

45: **for** $i \leftarrow 2$ to m-1 **do** ▷ Determine the feasibility and compute the ΔV for each of the intermediate legs of flight
46: $p_1 \leftarrow P_i(i)$
47: $p_2 \leftarrow P_i(i+1)$
48: **if** $p_1 == p_2$ **then** ▷ This is a V_∞-Leveraging leg
49: $result \leftarrow$ FLYBY-VILT-NON-LAUNCH$(\vec{r}(i), \vec{v}(i), result.\vec{V}, h_i(i) \star r_p(p_1) + r_p(p_1), g_i(i), \vec{r}(i+1), T_i(i+1), e_i(i+1), mu(p_1), safety, soilimits(p_1))$
50: **else if** $n_i(i+1) > 0$ **then**
51: $result \leftarrow$ FLYBY-WITH-DSM$(\vec{r}(i), \vec{v}(i), result.\vec{V}, h_i(i) \star r_p(p_1) + r_p(p_1), g_i(i), \vec{r}(i+1), T_i(i+1), e_i(i+1), direction, mu(p_1), mu(Sun), safety)$
52: **else**
53: $lOutput \leftarrow$ LAMBERT$(\vec{r}(i), \vec{r}(i+1), T_i(i+1), mu(Sun), dir)$
54: **if** The Single-revolution Lambert is Not Feasible **then**
55: $lOutput \leftarrow$ MULTI-REV-LAMBERT$(\vec{r}(i), \vec{r}(i+1), T_i(i+1), mu(Sun), dir)$
56: **end if**
57: **if** Neither of Lambert solutions converged **then**
58: **return** ∞
59: **end if**
60: $dVt \leftarrow$ FLYBY$(result.\vec{V}, lOutput.\vec{V}1, \vec{v}(i), r_p(p_1), mu(p_1), angle\text{-}tol)$
61: **if** $dVt == \infty$ **then**
62: **return** ∞
63: **end if**
64: $result \leftarrow Result(0, dVt, lOutput.\vec{V}2, 0)$
65: **end if**
66: **if** $result.\Delta V == \infty$ **then**
67: **return** ∞
68: **end if**
69: $dV(i+1) \leftarrow result.dV$
70: **end for**

Procedure 4.1 Procedure for Computation of the Fitness of a Chromosome (continued)

if $target == P_i(m)$ **then**
  $result \leftarrow$ FLYBY-VILT-NON-LAUNCH$(\vec{r}(m), \vec{v}(m), result.\vec{V}, h_i(m) \star r_p(P_i(m)) + r_p(P_i(m)), g_i(m), \vec{r}(target), T_i(m), e_i(m), mu(P_i(m)), safety, soilimits(P_i(m)))$
else if
  then $result \leftarrow$ FLYBY-WITH-DSM$(\vec{r}(m), \vec{v}(m), result.\vec{V}, h_i(m) \star r_p(P_i(m)) + r_p(P_i(m)), g_i(m), \vec{r}(target), T_i(m), e_i(m), direction, mu(P_i(m)), mu(Sun), safety)$
else
  $lOutput \leftarrow$ LAMBERT$(\vec{r}(m), \vec{r}(target), T_i(m), mu(Sun), dir)$
  if The Single-revolution Lambert is Not Feasible **then**
    $lOutput \leftarrow$ MULTI-REV-LAMBERT$(\vec{r}(m), \vec{r}(target), T_i(m), mu(Sun), dir)$
  end if
  if Neither of Lambert solutions converged **then**
    return ∞
  end if
  $dVt \leftarrow$ FLYBY$(result.\vec{V}, lOutput.\vec{V}1, \vec{v}(m), r_p(P_i(m)), mu(P_i(m)), angle\text{-}tol)$
  if $dVt == \infty$ **then**
    return ∞
  end if
  $result \leftarrow Result(0, dVt, lOutput.\vec{V}2, 0)$
end if
if $result.dV == \infty$ **then**
  return ∞
end if
$dV(last) \leftarrow result.dV$
$dV_{total} \leftarrow V_\infty + \Sigma|dV|$
return dV_{total}
end procedure

4.2.3 The Genetic Operators

Selection In the proposed algorithm, a variation of the Roulette Wheel or Fitness Proportionate selection scheme is used. In a deviation from the standard form of this selection scheme, the cost or fitness of the chromosome is *not* normalized. In the current algorithm, there is a possibility of the cost or fitness being ∞, and this does not lend itself well to normalization.

Crossover The proposed algorithm uses the Single-point crossover scheme. In a deviation from the standard Genetic Algorithms, the crossover threshold or probability changes for each pair of parent chromosomes, based on the fitness value of the best parent, and the average and minimum fitness values of the population.

Mutation The proposed algorithm uses the Gaussian mutation scheme. In a deviation from the standard practice of each of the individual genes of the chromosome having a specific mutation threshold or probability, the proposed algorithm uses a single mutation threshold for all genes. However, the algorithm uses adaptive mutation probabilities that are defined in each generation based on the fitness value of the chromosome, the average and minimum fitness values of the population. Due to this approach, the mutation probability is fixed for all the genes for the chromosome.

4.2.4 The Diversification of Population using a Crowding Technique

Basic Genetic Algorithms have a tendency of exploring a small search space of the problem domain and repeated consideration of the same sub-optimal chromosomes generation after generation. Generally, the more diverse the

population becomes in each generation of computation, the more of the problem search space explored. Diverse population is key to finding global optimum. Otherwise the GA might get stuck at the local minima. In the current study, a special crowding technique called "Twin-Space Crowding" [3] is used to maintain population diversity, which aids on the optimal convergence characteristics. Figure 4.4, reproduced from Chen, Chou, and Liu [3] shows the application of this special technique to the basic GA. Here two additional steps are added to basic GA to introduce the capability to diversify the population generation over generation. After creating the offspring from parent population, the offspring fitness is computed and is used in application of Twin-Space crowding technique to determine a diverse population for next generation of computation.

4.2.5 The Diversification of Population using an Adaptive GA Technique

The crowding technique does a great job of carrying over most fit solutions to the next generation, while also diversifying the population with solutions from currently unexplored search space. However, the speed at which the population diversifies is a function of the crossover and mutation probabilities (p_c and p_m). If these probabilities are constant for the entire execution of the algorithm, the solution convergence is not fast and may get stuck at local optima. If the offspring population constructed is homogeneous, the diversification process and hence the algorithm slows down. It is more efficient to use variable crossover and mutation probabilities determined from the fitness characteristics of the population, to prevent premature convergence and explore more of the search space. Srinivas and Patnaik [16] introduced the relationship between the average and best fitness values ($\bar{f}$ and f_{max}) of the population as the decisive factors in tuning the crossover and

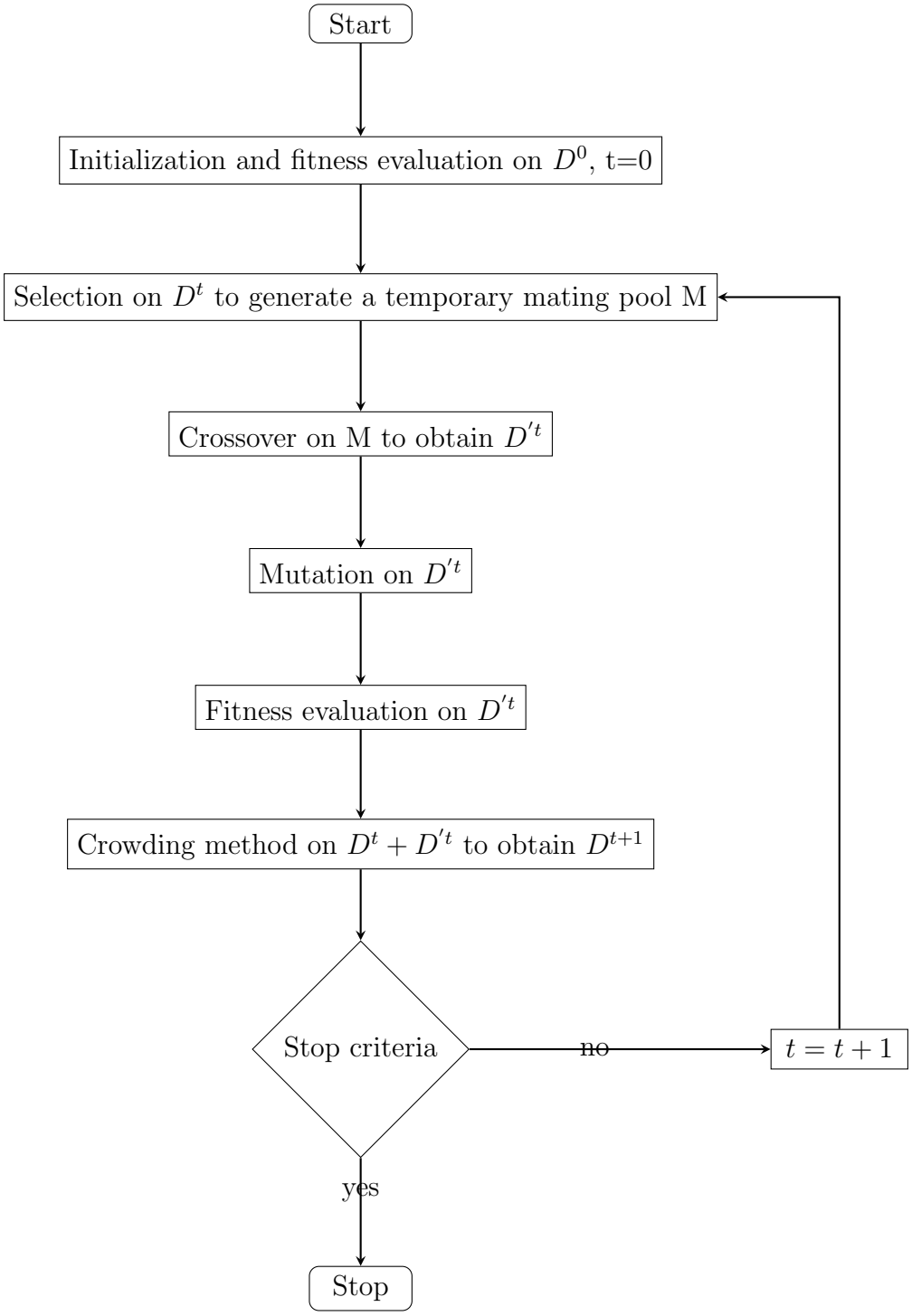

Figure 4.4: TCGA Flow Chart [3]

mutation probabilities over the execution span of the algorithm for a maximization problem. If the difference between average and best fitness values of the population is small, the population is deemed homogeneous, and hence higher values of crossover and mutation probabilities must be used to diversify the offspring constructed from the population. Similarly, if the difference is higher, the population is diverse and lower values of crossover and mutation probabilities must be used to preserve diversity. In other words, p_c and p_m must vary inversely with $f_{max} - \bar{f}$. Srinivas and Patnaik [16] also reasoned that p_c and p_m must vary per chromosome depending on the difference between the best fitness value of the population and fitness of chromosome, $f_{max} - f\prime$. If p_c and p_m solely depended on $f_{max} - \bar{f}$, both the near-optimal and sub-optimal chromosomes will be equally disrupted, potentially causing divergence in the algorithm. Hence, p_c and p_m must also vary per chromosome directly with $f_{max} - f\prime$. Srinivas and Patnaik [16] use tuning constants k_1, k_2, k_3, and k_4 to maintain the probabilities to a range of [0,1]. Srinivas and Patnaik [16] use the following equations to formalize the inverse and direct relationships with various fitness values for determining p_c and p_m for each chromosome:

$$p_c = k_1 \frac{f_{max} - f\prime}{f_{max} - \bar{f}}, f\prime \geq \bar{f} \tag{4.1}$$

$$p_c = k_3, f\prime < \bar{f} \tag{4.2}$$

and

$$p_m = k_2 \frac{f_{max} - f\prime}{f_{max} - \bar{f}}, f\prime \geq \bar{f} \tag{4.3}$$

$$p_m = k_4, f\prime < \bar{f} \tag{4.4}$$

The problem of this thesis is a minimization problem. So, the equations 4.1 - 4.4 are adjusted for a minimization problem as follows:

$$p_c = k_1 \frac{f\prime - f_{min}}{\bar{f} - f_{min}}, f\prime \leq \bar{f} \tag{4.5}$$

$$p_c = k_3, f\prime > \bar{f} \tag{4.6}$$

and

$$p_m = k_2 \frac{f\prime - f_{min}}{\bar{f} - f_{min}}, f\prime \leq \bar{f} \tag{4.7}$$

$$p_m = k_4, f\prime > \bar{f} \tag{4.8}$$

4.2.6 The Termination Criteria

In the proposed algorithm, a minimum number of generations are evaluated. After that, algorithm is terminated if it cannot improve the fitness of the winning chromosome for more than another minimum number of generations.

CHAPTER 5

THE IMPLEMENTATION AND PARALLELIZATION WITH MPI

The proposed algorithm is implemented in C++ using a parallel computing framework, the Message Passing Interface (MPI). Thus, the algorithm can be executed on any High-Performance Computing (HPC) environment.

5.1 The Implementation

The chromosome pool is represented in C++ using matrices. The Armadillo C++ linear algebra library developed by Sanderson and Curtin [24] is used to do so. The min, max and sort functions from this library are used extensively. This library also has a reliable uniform and Gaussian random number generation functionality necessary in the Genetic Algorithms.

5.1.1 Interpolation of Ephemerides

The ephemerides collected from the Horizons tool [22] are in a day granularity. However, the time-of-flight gene of the chromosome is expressed in seconds. Due to this discrepancy, an interpolation scheme developed by Fritz and Turkoglu [21] is used to derive the ephemerides of the planets for the exact times-of-flight specified in the gene.

5.1.2 Orbital Mechanics Procedures

A leg of the flight is a flight sequence between any two planets in a trajectory. There are several flavors of a leg of the flight depending on the combination of

various genes for that leg of the flight in the chromosome, listed as follows:

(1) Single-revolution Lambert's leg

(2) Multiple-revolution Lambert's leg

(3) V_∞-Leveraging launch leg

(4) Launch leg including a deep-space maneuver

(5) V_∞-Leveraging leg from a non-launch planet

(6) Non-launch leg including a deep-space maneuver

The single-revolution Lambert's leg is solved using the universal variable based solution from Curtis [20]. The multiple-revolution Lambert's leg is solved using the fast solution developed by Izzo [23]. Rest of the 4 types of leg of flight are solved as follows.

A Procedure for V_∞-Leveraging Launch Leg This procedure is required when a leg of the flight has the launch planet as the source and target planets. The purpose of this procedure is to leverage a deep-space maneuver to reduce the launch energy of the spacecraft. The procedure limits the launch v_∞ to a range of $[0.1, 5.0]$ km/s. The procedure attempts to solve the V_∞-leveraging maneuver repeatedly using the launch v_∞ values in the given range with increments of 0.1 km/s. The solution parameters that solve the V_∞-leveraging maneuver with minimum v_∞ are picked as the solution for the leg of flight. If the range of v_∞ values does not yield a solution to the V_∞-leveraging maneuver, ∞ is returned as the solution to that leg of flight, resulting in the termination of fitness computation

for the chromosome in the current leg of flight. Procedure 5.2 documents the pseudo-code for this procedure.

Procedure 5.2 Procedure for V_∞-Leveraging Launch Leg

1: **procedure** VILT-LAUNCH($\vec{r}_{pl}$,$\vec{v}_{pl}$,$\vec{r}_{pl_t}$,t,ϵ,μ,$soilimits$)

Input:

$\vec{r}_{pl}$ $\rightarrow$ Position vector of the launch planet at launch

$\vec{v}_{pl}$ $\rightarrow$ Velocity vector of the launch planet at launch

$\vec{r}_{pl_t}$ $\rightarrow$ Position vector of the launch planet at the end of the current leg

t $\rightarrow$ Time of flight for the current leg

ϵ $\rightarrow$ Fraction of time of flight where deep-space maneuver is to be conducted

μ $\rightarrow$ The gravitational parameter of the Sun

$soilimits$ $\rightarrow$ The limits of the Sphere-of-Influence (SOI) for the launch planet

Output:

Result $\rightarrow$ A composite object containing solution parameters

2: $min\text{-}dVp \leftarrow \infty$

3: $min\text{-}dVap \leftarrow \infty$

4: $min\text{-}err \leftarrow \infty$

5: $T_d \leftarrow t \star \epsilon$

6: $vnorm \leftarrow \left\|\vec{v}_{pl}\right\|_2$

7: $\vec{dir} \leftarrow \frac{\vec{v}_{pl}}{vnorm}$

8: $lambert\text{-}dir \leftarrow 0$

9: **for** $dVp \leftarrow 0.1$ to 5.0 **do**

10: $\vec{V} \leftarrow \vec{dir} \star (vnorm + dVp)$

11: $result \leftarrow$ VILT-KEPLER-LAMBERT($\vec{r}_{pl}, \vec{V}, \vec{r}_{pl_t}, t, \epsilon, soilimits, \mu, lambert\text{-}dir$)

Procedure 5.2 Procedure for V_∞-Leveraging Launch Leg (continued)

```
12:         if result.error < min-err and dVp + result.dV < min-dVp+min-dVap
  then
13:             min-err ← result.error
14:             min-dVap ← |result.dv|
15:             min-dVp ← dVp
16:             V_r ← result.V
17:             R_d ← result.R_d
18:             V_d ← result.V_d
19:         end if
20:     end for
21:     return Result(min-dVp, min-dVap, R_d, V_d, T_d, V_r, min-err)
22: end procedure
```

A Procedure for Launch Leg with Deep-Space Maneuver This procedure is required when the chromosome has a deep-space maneuver specified between the launch planet and the target planet in the current leg of flight, and the target planet is *different* from the launch planet. The purpose of this procedure is to minimize the launch energy of the spacecraft. Although the chromosome has a gene value indicating the use of a deep-space maneuver in this type of leg of flight, the use of a deep-space maneuver is optional. The deep-space maneuver is only used if a direct Lambert's transfer between the planets is not more economical in terms of launch energy. The procedure first attempts to compute the ΔV for a Lambert's transfer, if such a trajectory is feasible at all. The procedure then attempts to determine the trajectory with deep-space maneuver to target planet by repeatedly using the launch v_∞ values in the range of $[0.1, 5.0]$ km/s with increments of 0.1 km/s. The solution parameters that determine the trajectory with minimum v_∞ are picked. This minimum v_∞ is compared to the ΔV that would be required for a Lambert's transfer, if Lambert's transfer were feasible. The solution parameters corresponding to the minimum of these two values are returned. Procedure 5.3

documents the pseudo-code for this procedure.

Procedure 5.3 Procedure for Launch Leg including a Deep-Space Maneuver

1: **procedure** LAUNCH-DSM($\vec{r}_{pl}$,$\vec{v}_{pl}$,t,ϵ,$\vec{r}_{pl_t}$,dir,μ)

Input:

$\vec{r}_{pl}$ $\rightarrow$ Position vector of the launch planet and spacecraft at launch

$\vec{v}_{pl}$ $\rightarrow$ Velocity vector of the launch planet and spacecraft at launch

t $\rightarrow$ Time of flight for the current leg

ϵ $\rightarrow$ Fraction of time of flight where deep-space maneuver is to be conducted

$\vec{r}_{pl_t}$ $\rightarrow$ Position vector of the launch planet at the end of the current leg

dir $\rightarrow$ 0 or 1 indicating prograde or retrograde motion respectively

μ $\rightarrow$ The gravitational parameter of the Sun

Output:

Result $\rightarrow$ A composite object containing solution parameters

2: $min\text{-}vinf \leftarrow \infty$

3: $min\text{-}dV \leftarrow \infty$

4: $min\text{-}dV\text{-}tot \leftarrow \infty$

5: $T_d \leftarrow t \star \epsilon$

6: $vnorm \leftarrow \left\|\vec{v}_{pl}\right\|_2$

7: $v\vec{dir} \leftarrow \frac{\vec{v}_{pl}}{vnorm}$

8: $lOutput1 \leftarrow$ LAMBERT($\vec{r}_{pl}, \vec{r}_{pl_t}, t, \mu, dir$)

9: **if** Single-revolution Lambert's solution does not exist **then**

10: $lOutput1 \leftarrow$ MULTI-REV-LAMBERT($\vec{r}_{pl}, \vec{r}_{pl_t}, t, \mu, dir$)

11: **end if**

12: **for** $vinf \leftarrow 0.1$ to 5.0 **do**

13: $\vec{v}_m \leftarrow v\vec{dir} \star (vnorm + dVp)$

14: $rv \leftarrow$ KEPLER($\vec{r}_{pl}, \vec{v}_m, t \star \epsilon, \mu$)

A Procedure for V_∞-Leveraging Leg from a Non-Launch Planet

This procedure is required for determining the trajectory in a leg of flight when the source and target planets of the leg are the *same* non-launch planet. A deep-space maneuver may be required in this case. A non-powered gravity-assist maneuver is conducted at the beginning of this leg. The outbound heliocentric velocity vector from the gravity-assist maneuver is used in to determine the resultant trajectory to the target planet in the current leg. A deep-space maneuver is only used if necessary. Procedure 5.4 lists the pseudo-code in detail.

Procedure 5.3 Procedure for Launch Leg including a Deep-Space Maneuver (continued)

if A solution to Kepler's problem is found **then**
  $lOutput \leftarrow \text{LAMBERT}(rv.\vec{r}, \vec{r}_{pl_t}, t \star (1-\epsilon), \mu, dir)$
  if Single-revolution Lambert's solution did not coverge **then**
    $lOutput \leftarrow \text{MULTI-REV-LAMBERT}(rv.\vec{r}, \vec{r}_{pl_t}, t \star (1-\epsilon), \mu, dir)$
  end if
  if Either of Lambert's solutions coverged **then**
    $dV \leftarrow \left\| lOutput.\vec{V}1 - rv.\vec{v} \right\|_2$
    if $min\text{-}dV\text{-}tot > vinf + dV$ **then**
      $min\text{-}dV\text{-}tot \leftarrow vinf + dV$
      $min\text{-}vinf \leftarrow vinf$
      $min\text{-}dV \leftarrow dV$
      $\vec{V} \leftarrow lOutput.\vec{V}2$
      $\vec{R}_d \leftarrow rv.\vec{r}$
      $\vec{V}_d \leftarrow lOutput.\vec{V}1$
    end if
  end if
end if
end for
if Either of direct Lambert's solutions converged **then**
  $dVl \leftarrow \left\| lOutput1.\vec{V}1 - \vec{v}_{pl} \right\|_2$
  if $dVl < min\text{-}dV\text{-}tot$ **then**
    return $Result(dVl, 0, \vec{R}_d, \vec{V}_d, t, lOutput1.\vec{V}2, 0)$
  end if
end if
return $Result(min\text{-}vinf, min\text{-}dV, \vec{R}_d, \vec{V}_d, T_d, \vec{V}, 0)$
end procedure

Procedure 5.4 Procedure for V_∞-Leveraging Leg from a Non-Launch Planet

1: **procedure** FLYBY-VILT-NON-LAUNCH($\vec{r}_{pl}$,$\vec{v}_{pl}$,$\vec{v}_{sc_{in}}$,rp,ζ,$\vec{r}_{pl_t}$,t,ϵ,μ_{pl},μ_S,$safety$,$soilimits$)

Input:

$\vec{r}_{pl}$ $\rightarrow$ Position vector of the planet at the start of the leg

$\vec{v}_{pl}$ $\rightarrow$ Velocity vector of the planet at the start of the leg

$\vec{v}_{sc_{in}}$ $\rightarrow$ Inbound heliocentric velocity vector of the spacecraft at the start of the leg

rp $\rightarrow$ Periapse radius of the hyperbolic trajectory of the spacecraft around the planet at the start of the leg

ζ $\rightarrow$ Orientation of the hyperbolic trajectory of the spacecraft around the planet at the start of the leg

$\vec{r}_{pl_t}$ $\rightarrow$ Position vector of the planet at the end of the current leg

t $\rightarrow$ Time of flight for the current leg

ϵ $\rightarrow$ Fraction of time of flight where deep-space maneuver is to be conducted

μ_{pl} $\rightarrow$ The gravitational parameter of the planet

μ_S $\rightarrow$ The gravitational parameter of the Sun

$safety$ $\rightarrow$ 1 or 0 indicating whether or not to consider safety of the spacecraft, to make sure it does not crash into or get dangerously close to the Sun respectively

$soilimits$ $\rightarrow$ The Sphere-of-Influence (SOI) limits for the planet

Output:

Result $\rightarrow$ A composite object containing solution parameters

2: $\vec{V}_{\infty_{in}} \leftarrow \vec{v}_{sc_{in}} - \vec{v}_{pl}$

3: $vinf\text{-}in \leftarrow \left\| \vec{V}_{\infty_{in}} \right\|_2$

4: $\vec{i} \leftarrow \frac{\vec{V}_{\infty_{in}}}{vinf\text{-}in}$

A Procedure for Non-Launch Leg including a Deep-Space Maneuver

This procedure is necessary to determine trajectory for a leg of flight containing *different* source and target planets. The deep-space maneuver is used only when a Lambert's transfer trajectory is not feasible or is not more economical than the trajectory with the deep-space maneuver in terms of fuel. First a gravity-assist maneuver is conducted about the source planet. The resultant outbound heliocentric velocity vector of the spacecraft is used in subsequent steps to determine the trajectory for the current leg of flight. Procedure 5.5 lists the pseudo-code in detail.

Procedure 5.4 Procedure for V_∞-Leveraging Leg from a Non-Launch Planet (continued)

$\vec{j} \leftarrow \vec{i} \times \vec{v}_{pl}$
$\vec{j} \leftarrow \frac{\vec{j}}{\|j\|_2}$
$\vec{k} \leftarrow \vec{i} \times \vec{j}$
$e \leftarrow 1 + \frac{rp \star vinf - in^2}{\mu_{pl}}$
$\delta \leftarrow 2 \sin \frac{1}{e}$
$\vec{V}_{\infty_{out}} \leftarrow \vec{V}_{\infty_{in}}[cos(\delta)\vec{i} + sin(\delta)sin(\zeta)\vec{j} + sin(\delta)cos(\zeta)\vec{k}]$
$dir \leftarrow 0$
$result \leftarrow$ VILT-KEPLER-LAMBERT$(\vec{r}_{pl}, \vec{V}_{\infty_{out}}, \vec{r}_{pl_t}, t, \epsilon, soilimits, \mu_S, dir)$
if safety is set to 1 and $result.dV < \infty$ **then**
    $[a, e] \leftarrow$ AE-FROM-RV$(result.\vec{R}_d, result.\vec{V}_d, \mu_S)$
    $rp_{sc} \leftarrow a(1 - e^2)$
    if $rp_{sc} < 10\%$ of an AU **then**
        return $Result(0, \infty, \vec{empty}, \vec{empty}, t \star \epsilon, \vec{empty}, \infty)$
    end if
end if
return $result$
end procedure

Procedure 5.5 Procedure for Non-Launch Leg including a Deep-Space Maneuver

1: **procedure** FLYBY-WITH-DSM($\vec{r}_{pl}$,$\vec{v}_{pl}$,$\vec{v}_{sc_{in}}$,rp,ζ,$\vec{r}_{pl_t}$,t,ϵ,dir,μ_{pl},μ_S,$safety$)

Input:

$\vec{r}_{pl}$ $\rightarrow$ Position vector of the planet at the start of the leg

$\vec{v}_{pl}$ $\rightarrow$ Velocity vector of the planet at the start of the leg

$\vec{v}_{sc_{in}}$ $\rightarrow$ Inbound heliocentric velocity vector of the spacecraft at the start of the leg

rp $\rightarrow$ Periapse radius of the hyperbolic trajectory of the spacecraft around the planet at the start of the leg

ζ $\rightarrow$ Orientation of the hyperbolic trajectory of the spacecraft around the planet at the start of the leg

$\vec{r}_{pl_t}$ $\rightarrow$ Position vector of the planet at the end of the current leg

t $\rightarrow$ Time of flight for the current leg

ϵ $\rightarrow$ Fraction of time of flight where deep-space maneuver is to be conducted

dir $\rightarrow$ 0 or 1 indicating prograde or retrograde motion respectively

μ_{pl} $\rightarrow$ The gravitational parameter of the planet

μ_S $\rightarrow$ The gravitational parameter of the Sun

$safety$ $\rightarrow$ 1 or 0 indicating whether or not to consider safety of the spacecraft, to make sure it does not crash into or get dangerously close to the Sun respectively

Output:

Result $\rightarrow$ A composite object containing solution parameters

2: $\vec{V}_{\infty_{in}} \leftarrow \vec{v}_{sc_{in}} - \vec{v}_{pl}$

3: $vinf\text{-}in \leftarrow \left\| \vec{V}_{\infty_{in}} \right\|_2$

4: $\vec{i} \leftarrow \frac{\vec{V}_{\infty_{in}}}{vinf-in}$

A Procedure for determining V_∞-Leveraging Trajectory This procedure is required to determine a V_∞-Leveraging trajectory in a leg of flight. In this case both the source and target planets of the leg are the *same*. Because of this a deep-space maneuver may be required in the current leg of flight. The procedure computes the position and velocity vectors of the spacecraft at the expected deep-space maneuver location. At the deep-space maneuver location, an instantaneous tangential ΔV maneuver is assumed. It is the goal of this procedure to determine the minimum such ΔV burn to determine a fuel-optimal trajectory to target location of the planet. Procedure 5.6 lists the pseudo-code in detail for this procedure.

Procedure 5.5 Procedure for Non-Launch Leg including a Deep-Space Maneuver (continued)

$\vec{j} \leftarrow \vec{i} \times \vec{v}_{pl}$
$\vec{j} \leftarrow \frac{\vec{j}}{\|j\|_2}$
$\vec{k} \leftarrow \vec{i} \times \vec{j}$
$e \leftarrow 1 + \frac{rp \star vinf - in^2}{\mu_{pl}}$
$\delta \leftarrow 2 \sin \frac{1}{e}$
$\vec{V}_{\infty_{out}} \leftarrow \vec{V}_{\infty_{in}}[cos(\delta)\vec{i} + sin(\delta)sin(\zeta)\vec{j} + sin(\delta)cos(\zeta)\vec{k}]$
$isLambertSafe \leftarrow true$
$lOutput1 \leftarrow$ LAMBERT$(\vec{r}_{pl}, \vec{r}_{pl_t}, t, \mu_S, dir)$
if Single-revolution Lambert trajectory does not exist **then**
    $lOutput1 \leftarrow$ MULTI-REV-LAMBERT$(\vec{r}_{pl}, \vec{r}_{pl_t}, t, \mu_S, dir)$
end if
if Either of Lambert's trajectories exists **then**
    $dVl \leftarrow \left\| lOutput1.\vec{V}1 - \vec{V}_{\infty_{out}} \right\|_2$
    $\vec{Vl} \leftarrow lOutupt1.\vec{V}2$
    $[a, e] \leftarrow$ AE-FROM-RV$(\vec{r}_{pl}, lOutput1.\vec{V}1)$
    $rp_{sc} \leftarrow a(1 - e^2)$
    if rp_{sc} <10% of an AU **then**
        $isLambertSafe \leftarrow false$
    end if
end if
$rv \leftarrow$ KEPLER$(\vec{r}_{pl}, \vec{V}_{\infty_{out}}, t \star \epsilon, \mu_S)$
$lOutput \leftarrow$ LAMBERT$(rv.\vec{r}, \vec{r}_{pl_t}, t \star (1 - \epsilon), \mu_S, dir)$
if Single-revolution Lambert trajectory does not exist **then**
    $lOutput \leftarrow$ MULTI-REV-LAMBERT$(rv.\vec{r}, \vec{r}_{pl_t}, t \star (1 - \epsilon), \mu_S, dir)$
end if
if Either of Lambert's trajectories exists **then**
    $dV \leftarrow \left\| lOutput.\vec{V}1 - rv.\vec{v} \right\|_2$
    $lOutput.\vec{V}2$
end if

Procedure 5.5 Procedure for Non-Launch Leg including a Deep-Space Maneuver (continued)

```
    if safety is set to 1 then
        [a, e] ← AE-FROM-RV(rv.r⃗, lOutput.V⃗1)
        rp_sc ← a(1 − e²)
        if rp_sc <10% of an AU then
            if dVl == ∞ then
                return Result(0, ∞, empty⃗, empty⃗, t ⋆ ϵ, empty⃗, ∞)
            else if isLambertSafe is true then
                return Result(0.0, dVl, r⃗_pl, lOutput1.V⃗1, t, V⃗l, 0)
            end if
        end if
    end if
    if dV < dVl then
        return Result(0, dV, rv.r⃗, lOutput.V⃗1, tof ⋆ ϵ, V⃗, 0)
    end if
    if isLambertSafe is true then
        return Result(0.0, dVl, r⃗_pl, lOutput1.V⃗1, t, V⃗l, 0)
    end if
    return Result(0.0, ∞, empty⃗, empty⃗, t, empty⃗, 0)
end procedure
```

Procedure 5.6 Procedure for V_∞-Leveraging Maneuver

1: **procedure** VILT-KEPLER-LAMBERT($\vec{r}$,$\vec{v}$,$\vec{r}_t$,t,ϵ,$soilimits$,μ,dir)

Input:

$\vec{r}$ $\rightarrow$ Position vector of the launch planet at launch

$\vec{v}$ $\rightarrow$ Velocity vector of the launch planet at launch

$\vec{r}_t$ $\rightarrow$ Position vector of the launch planet at the end of the current leg

t $\rightarrow$ Time of flight for the current leg

ϵ $\rightarrow$ Fraction of time of flight where deep-space maneuver is to be conducted

$soilimits$ $\rightarrow$ The limits of the Sphere-of-Influence (SOI) for the launch planet

μ $\rightarrow$ The gravitational parameter of the Sun

dir $\rightarrow$ 0 or 1 indicating the prograde or retrograde motion respectively

Output:

Result $\rightarrow$ A composite object containing solution parameters

2: $rv \leftarrow$ KEPLER($\vec{r}, \vec{v}, t \star \epsilon, \mu$)

3: **if** The Kepler's solution did not converge **then**

4: **return** $Result(0, \infty, \vec{empty}, \vec{empty}, t, \vec{empty}, \infty)$

5: **end if**

5.2 The MPI Standard

The Message Passing Interface (MPI) is a platform-independent standard for message communication and coordination of program execution in parallel computing environments. The first version (1.0) of MPI was released in June of 1994. The latest version of MPI (3.1) was published in June of 2015. The main advantage of the MPI standard is its portability. There are several open-source

Procedure 5.6 Procedure for V_∞-Leveraging Maneuver (continued)

$rvt \leftarrow \text{KEPLER}(rv.\vec{r}, rv.\vec{v}, t \star (1-\epsilon), \mu)$
if The Kepler's solution converged **then**
  $err \leftarrow \|rvt.\vec{r} - \vec{r}_t\|_2$
  if $err > soilimits.min$ and $err < soilimits.max$ **then**
    return $Result(0, 0, rv.\vec{r}, rv.\vec{v}, t \star \epsilon, rvt.\vec{v}, 0)$
  end if
end if
$lOutput1 \leftarrow \text{LAMBERT}(rv.\vec{r}, \vec{r}_t, t \star (1-\epsilon), \mu, dir)$
if Single-revolution Lambert trajectory does not exist **then**
  $lOutput1 \leftarrow \text{MULTI-REV-LAMBERT}(rv.\vec{r}, \vec{r}_t, t \star (1-\epsilon), \mu, dir)$
end if
if Either of Lambert's trajectories exists **then**
  $dVl \leftarrow \left\| lOutput1.\vec{V}1 - rv.\vec{v} \right\|_2$
  return $Result(0, dVl, rv.\vec{r}, rv.\vec{v}, t \star \epsilon, lOutput1.\vec{V}2, 0)$
end if
return $Result(\infty, \infty, \vec{empty}, \vec{empty}, t \star \epsilon, \vec{empty}, \infty)$
end procedure

implementations of MPI available today. MPICH [25] and OPENMPI [26] are the most prevalent open-source implementations of the MPI standard. In this thesis, MPICH implementation of MPI is used.

5.3 Parallelization with MPI

The MPI_Send and MPI_Recv functions are used extensively in coordinating the communication between the master and worker cores. The fitness computation is distributed to all the cores used in the program. Master core divides and distributes the population for parent or offspring generation equally to all the available cores using the MPI_Scatter function. All the cores compute the fitness of the sub-pool distributed to them from either the parent pool or offspring pool and send the results back to master core using the MPI_Gather function. Master core is responsible for executing all the other Genetic Operators (selection, crossover, and mutation) as well as the Operators of the TCGA (crowding method). Master core is also responsible for testing the termination criteria and communicating termination of algorithm to all the worker cores.

5.4 HPC Platform of Choice

There is a myriad of HPC platforms out there. However, the San Jose State University is yet to establish an HPC environment accessible to students as of this writing. Due to this, the HPC platform of choice is a small cluster of micro-controllers, the ODROID XU4's, manufactured by HardKernel [27], established in the Control Science and Dynamical Systems (CSDy) laboratory of the Aerospace Engineering department at San Jose State University. These micro-controllers are very user friendly and yet powerful. The ODROID XU4 has

two types of ARM $Cortex^{TM}$ processors, the 2 GHz A-15 and 1.2 GHz A-7 processor. There are 4 cores in each of these two processors, giving a total of 8 cores for the ODROID XU4. The ODROID XU4 has 2GB of LPDDR3 RAM along with support for Gigabit Ethernet for inter-node communication. A cluster of 7 ODROID XU4's is established, with a total core capacity of 56.

CHAPTER 6

FUEL-OPTIMAL TRAJECTORIES TO SATURN

6.1 An Optimal Earth-Saturn Trajectory with 4 Gravity-Assist Maneuvers

For finding an optimal trajectory to Saturn with 4 gravity-assist maneuvers, the proposed algorithm is tuned with the following configuration. Table 6.1 lists the various configuration parameters for the GA. Table 6.2 lists the lower and upper bounds, and the standard deviation for all the genes of the chromosome. The resultant optimal trajectory is shown in Figure 6.1. Figure 6.2 shows the minimum and average ΔV over the Genetic Algorithm generations. It also shows the total number of feasible solutions found in each generation. Figure 6.3 shows the minimum ΔV over the generations of the Genetic Algorithm. The total ΔV for the mission is 10.018 km/s with a mission time of 19.06 years. Table 6.3 lists all the parameters of this trajectory.

6.2 An Optimal Earth-Saturn Trajectory with 3 Gravity-Assist Maneuvers

For finding an optimal trajectory to Saturn with 3 gravity-assist maneuvers, the proposed algorithm is tuned with the following configuration. Table 6.4 lists the various configuration parameters for the GA. Table 6.5 lists the lower and upper bounds, and the standard deviation for all the genes of the chromosome. The resultant optimal trajectory is shown in Figure 6.4. Figure 6.5 shows the minimum and average ΔV over the Genetic Algorithm generations. It also shows the total

Table 6.1: Configuration of the Algorithm for 4 Gravity-Assists

Parameter	**Description**	**Value**
LEO Height	Height of the LEO parking orbit of the spacecraft for a Lambert's launch	500 km
Population or Pool Size	Size of the population for the GA	280
Target Planet Id	Integer identifier for the target planet, Saturn	6
Ephemerides Start Date	Start date of the Ephemerides downloaded from JPL	01-01-2020
Ephemerides End Date	End date of the Ephemerides downloaded from JPL	12-31-2055
Ephemerides Granularity	Granularity of the Ephemerides downloaded from JPL	1 day
Termination Tolerance	Tolerance for mission ΔV, over a given number of generations	0.01 $\frac{km}{s}$
Convergence Generations	Minimum number of generations for which ΔV is within termination tolerance	50
Minimum Generations	The minimum number of generations to execute for the GA	50
Maximum Generations	The maximum number of generations to execute before termination of the GA	1000

Table 6.2: The GA Gene Configuration for 4 Gravity-Assists

Gene	**Lower Bound**	**Upper Bound**	**Standard Deviation**
T_d	01-01-2020	12-31-2020	30 days
m	4	5	1
P_i	2, Venus	5, Jupiter	1
T_i	3 months	72 months	2 months
h_i	$0.1 * r_p$	$10 * r_p$	$0.1 * r_p$
g_i	0 radians	2π radians	0.1 radians
n_i	0	1	1
e_i	$0.1 * T_i$	$0.9 * T_i$	$0.05 * T_i$
f	0	1	0

Table 6.3: 4 Gravity-Assist Fuel-Optimal Earth-Saturn Trajectory Parameters

Trajectory Parameter	**Value**
Launch Date	07-06-2020 8:52 PM
Launch V_∞	0.1 km/s
DSM 1 Date	03-21-2021 7:59 PM
DSM 1 ΔV	0.4267 km/s
Earth Gravity-Assist Date	02-21-2022 2:45 AM
DSM 2 Date	04-23-2024 6:53 PM
DSM 2 ΔV	0.2264 km/s
Earth Gravity-Assist Date	02-20-2025 2:45 AM
Mars Gravity-Assist Date	06-25-2028 3:28 AM
Mars Gravity-Assist ΔV	7.561 km/s
Jupiter Gravity-Assist Date	09-30-2033 0:16 AM
Jupiter Gravity-Assist ΔV	1.2627 km/s
Saturn Rendezvous Date	04-29-2-39 1:15 AM
Total Mission ΔV	10.018 km/s
Total Mission Time	19.06 years

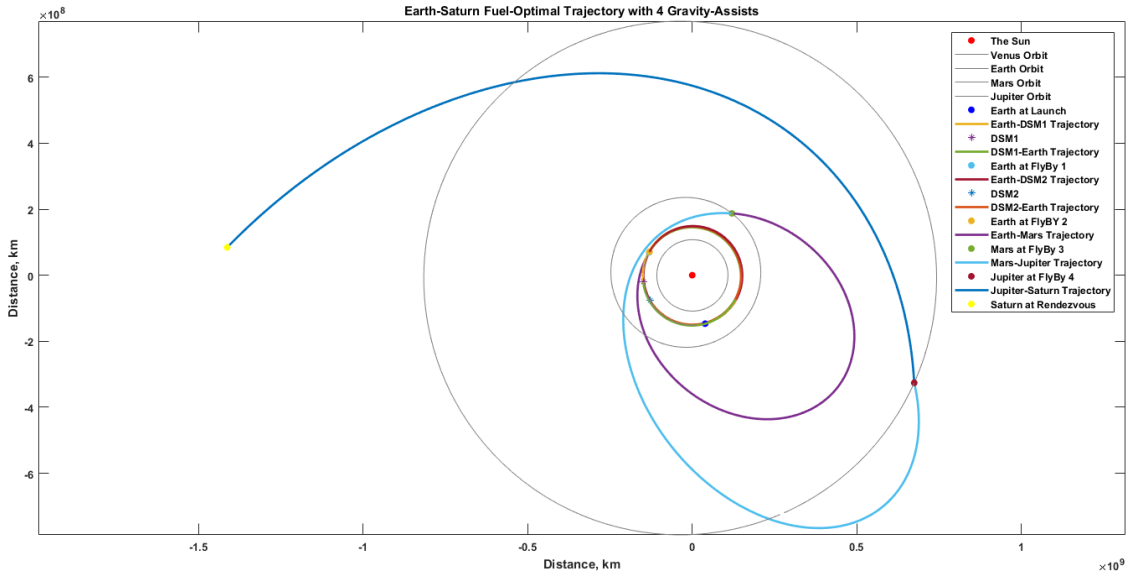

Figure 6.1: An Earth-Saturn Optimal Trajectory with 4 Gravity-Assist Maneuvers

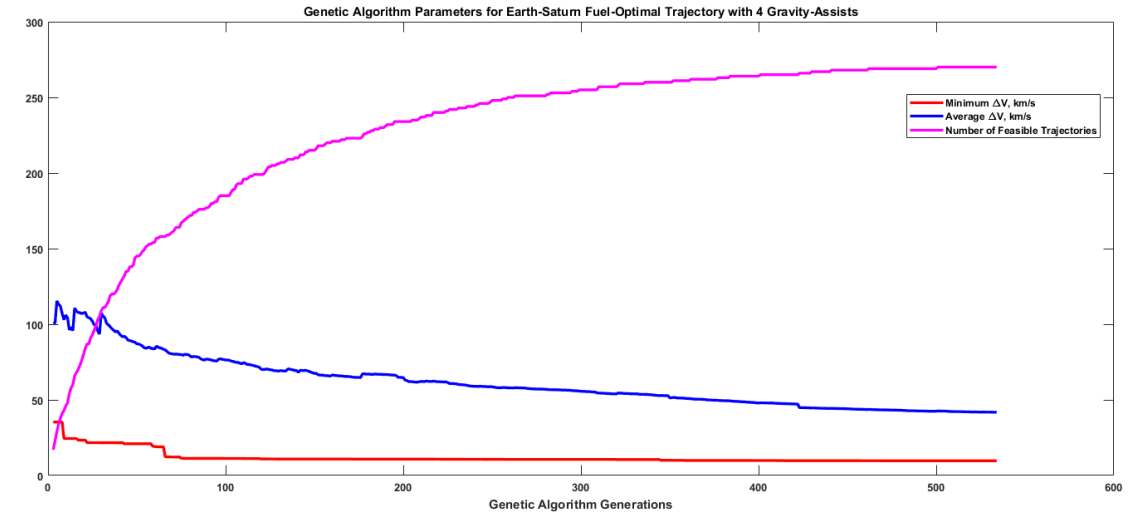

Figure 6.2: The Parameters of the Genetic Algorithm for 4 Gravity-Assists

number of feasible solutions found in each generation. Figure 6.6 shows the minimum ΔV over the generations of the Genetic Algorithm. The total ΔV for the mission is 11.2426 km/s with a mission time of 14.365 years. Table 6.6 lists all the parameters of this trajectory.

Table 6.4: Configuration of the Algorithm for 3 Gravity-Assists

Parameter	**Description**	**Value**
LEO Height	Height of the LEO parking orbit of the spacecraft for a Lambert's launch	500 km
Population or Pool Size	Size of the population for the GA	280
Target Planet Id	Integer identifier for the target planet, Saturn	6
Ephemerides Start Date	Start date of the Ephemerides downloaded from JPL	01-01-2020
Ephemerides End Date	End date of the Ephemerides downloaded from JPL	12-31-2055
Ephemerides Granularity	Granularity of the Ephemerides downloaded from JPL	1 day
Termination Tolerance	Tolerance for mission ΔV, over a given number of generations	0.01 $\frac{km}{s}$
Convergence Generations	Minimum number of generations for which ΔV is within termination tolerance	50
Minimum Generations	The minimum number of generations to execute for the GA	50
Maximum Generations	The maximum number of generations to execute before termination of the GA	1000

Table 6.5: The GA Gene Configuration for 3 Gravity-Assists

Gene	**Lower Bound**	**Upper Bound**	**Standard Deviation**
T_d	01-01-2020	12-31-2020	30 days
m	3	5	1
P_i	2, Venus	5, Jupiter	1
T_i	3 months	72 months	2 months
h_i	$0.1 * r_p$	$10 * r_p$	$0.1 * r_p$
g_i	0 radians	2π radians	0.1 radians
n_i	0	1	1
e_i	$0.1 * T_i$	$0.9 * T_i$	$0.05 * T_i$
f	0	1	0

Table 6.6: 3 Gravity-Assist Fuel-Optimal Earth-Saturn Trajectory Parameters

Trajectory Parameter	**Value**
Launch Date	11-15-2020 10:56 AM
Launch V_∞	0.1 km/s
DSM 1 Date	01-12-2022 12:26 PM
DSM 1 ΔV	0.4267 km/s
Earth Gravity-Assist Date	09-01-2022 11:42 AM
DSM 2 Date	10-12-2024 2:49 AM
DSM 2 ΔV	0.2264 km/s
Earth Gravity-Assist Date	08-13-2025 0:35 AM
Mars Gravity-Assist Date	07-09-2029 7:01 PM
Mars Gravity-Assist ΔV	7.561 km/s
Saturn Rendezvous Date	01-12-2035 9:33 PM
Total Mission ΔV	11.2426 km/s
Total Mission Time	14.365 years

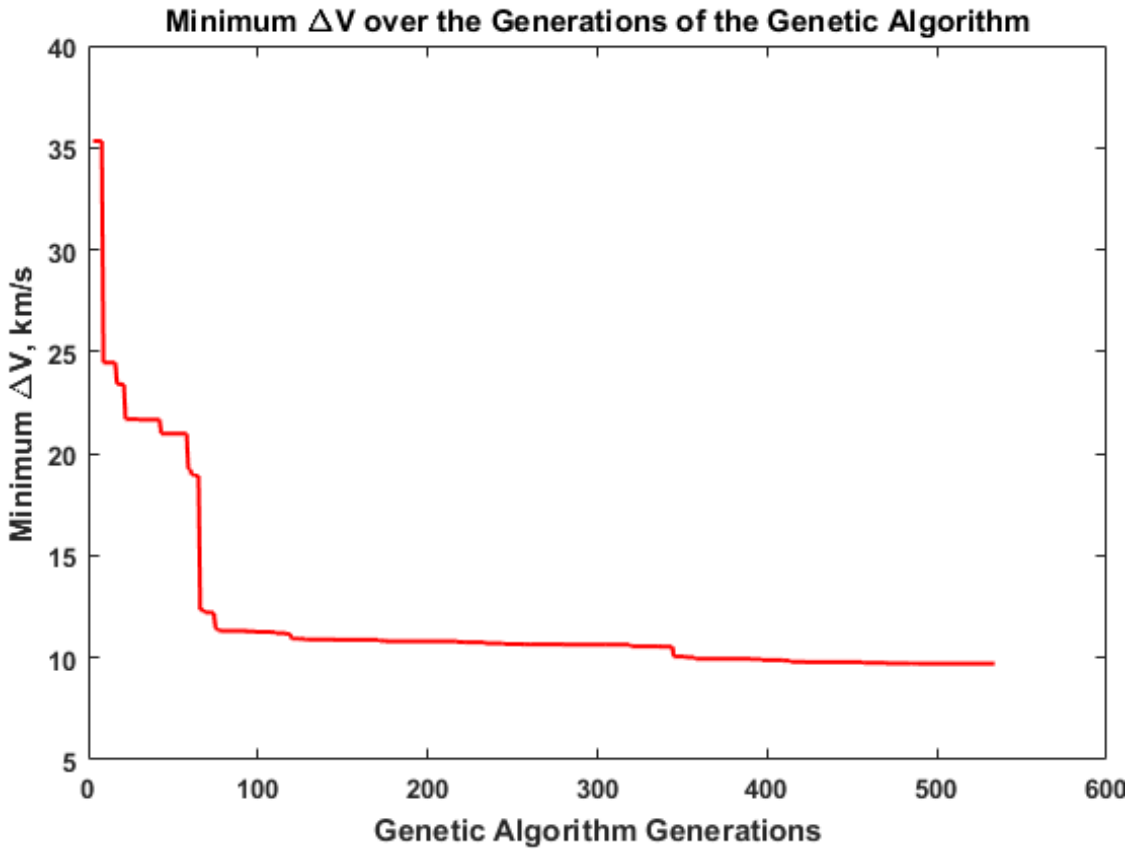

Figure 6.3: The Minimum ΔV over the Generations of the Genetic Algorithm for 4 Gravity-Assists

6.3 Comparison of the Optimal Trajectories

The main difference between the 4-gravity-assist trajectory and the 3-gravity-assist trajectory is the extended mission time in the 4-gravity-assist trajectory, albeit with an improvement in the mission cost (total ΔV) by 1.2242 km/s. In both the trajectories, the ΔV for Mars Gravity-Assist is very high at 7.561 km/s.

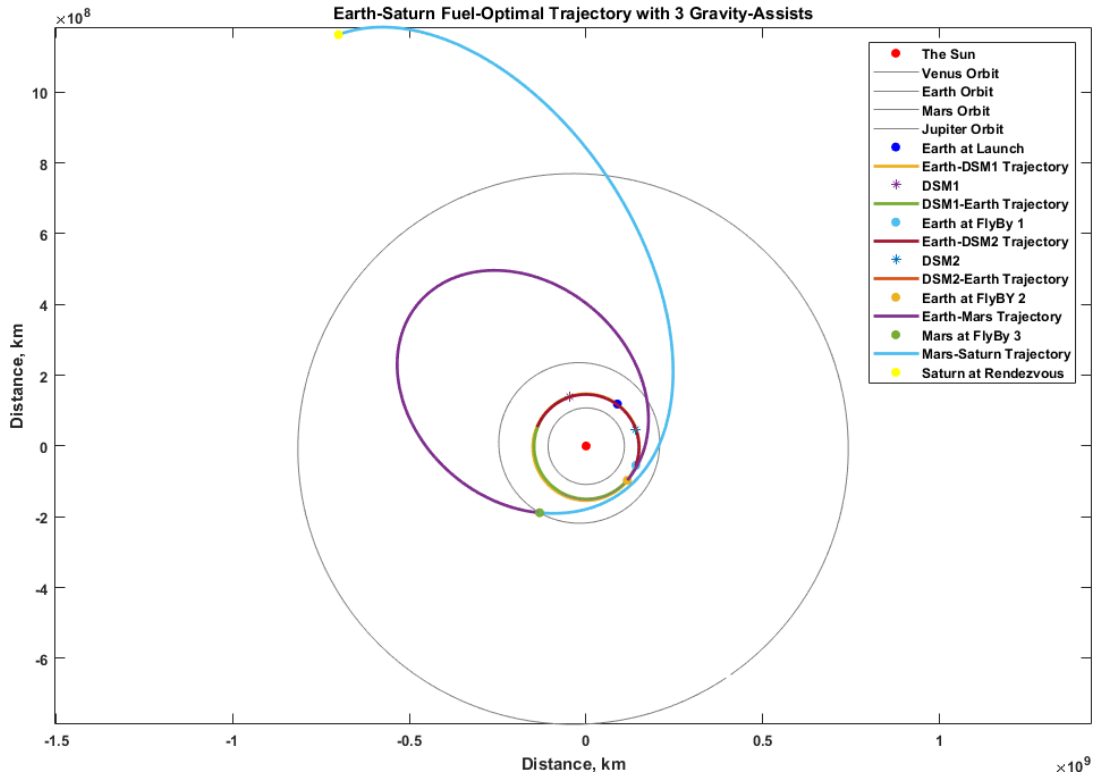

Figure 6.4: An Earth-Saturn Optimal Trajectory with 3 Gravity-Assist Maneuvers

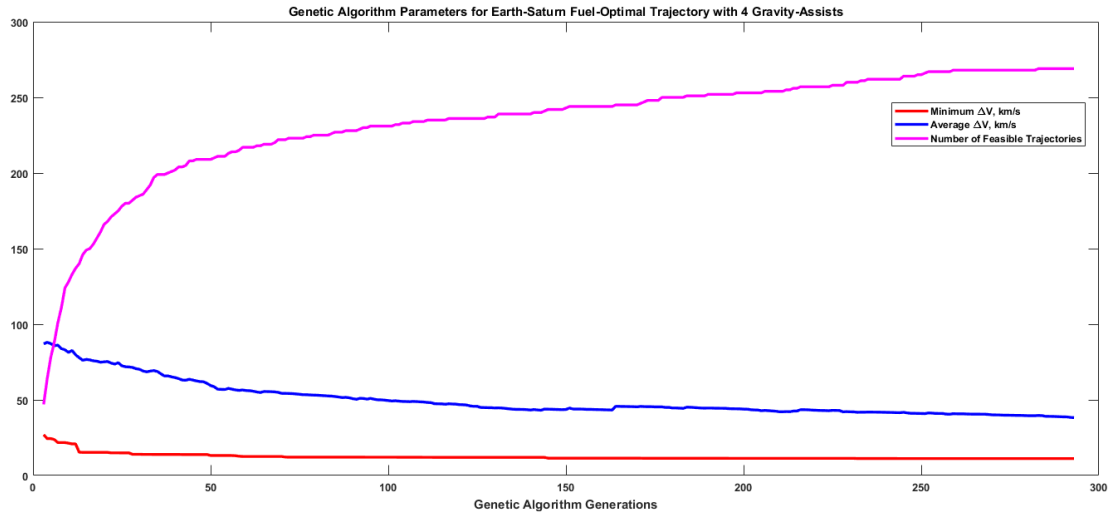

Figure 6.5: The Parameters of the Genetic Algorithm for 3 Gravity-Assists

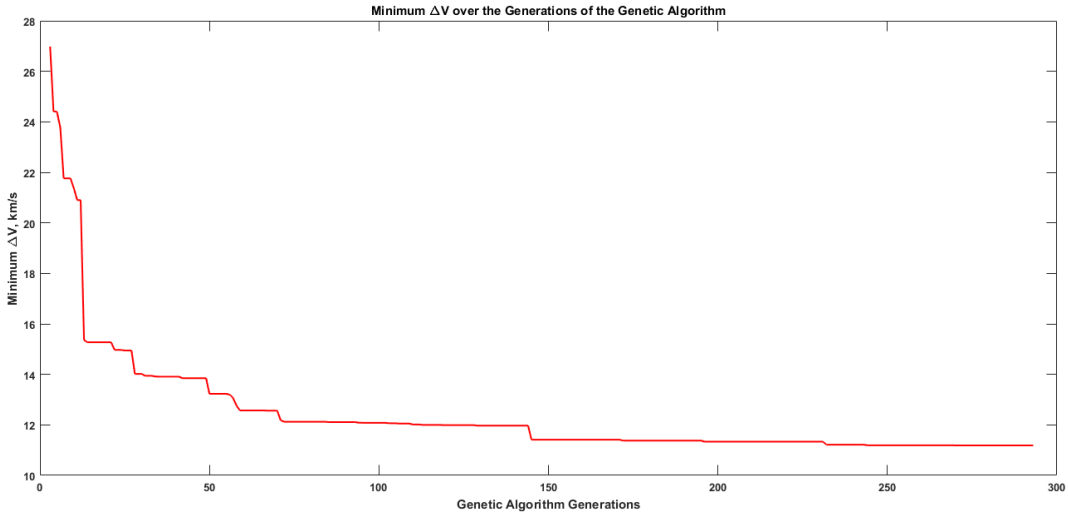

Figure 6.6: The Minimum ΔV over the Generations of the Genetic Algorithm for 3 Gravity-Assists

6.4 Performance of the Adaptive Twin-Space Crowding Genetic Algorithm

For the 4-gravity-assist trajectory, the algorithm converged at the 534th generation. Figure 6.2 shows the average and minimum ΔV values of the population as well as the total number of feasible solutions in the population over all of the 534 generations. As can be seen in Figure 6.2, the number of feasible solutions in the population raises steeply till 125th generation to reach a value of 200 out of a total of 280 solution candidates in the population. Correspondingly, the minimum and average ΔV values also steeply decrease during this span. After the 125th generation, the number of feasible raises steadily but slowly. Accordingly, the minimum and average ΔV values also decrease slowly but steadily until convergence. Similar trend is also observed in the 3-gravity-assist trajectory case in Figure 6.5. In this case, the algorithm converged at the 293rd generation. Number of feasible solutions in the population quickly gets to 200 out of possible 280

candidate solutions by the 40th generation. After this the number raises slowly but steadily. A similar trend is observed in minimum and average ΔV values. This is due to the combination of adaptive and Twin-Space crowding techniques employed. An attempt is made to use adaptive diversification technique alone for the Genetic Algorithm. However, it is observed that the convergence is not as effective as when the two diversification techniques are combined. The algorithm executed for 173 minutes for the 4-gravity-assist trajectory and for 110 minutes for the 3-gravity-assist trajectory. The scale of calculations involved in interplanetary travel is astronomical. Given this, the execution times of the algorithm in these two cases are deemed efficient.

CHAPTER 7

CONCLUSIONS AND RECOMMENDATIONS

The algorithm proposed in this thesis is producing optimal solutions with economical costs. The two solutions found using this algorithm are yielding costs that are comparable to the Cassini 2 mission cost (8.385 km/s) found by Gad and Abdelkhalik's [1] hidden-gene based algorithm. However, the algorithm is not doing well in terms of the mission time. This is because the cost/fitness function does not include the mission time. The time-of-flight gene of the chromosome is also crucial to the mission time. The proposed algorithm uses common minimum, maximum and standard deviation values for *all* the Solar planets. This is not an ideal choice. These GA parameters should be tuned according to the pair of planets involved in each leg of the flight. For example, the time-of-flight in a Venus-Earth leg will be way smaller to that of an Earth-Jupiter leg. Since the goal of this thesis is to determine fuel-optimal trajectories, this work is left as a recommendation for future work. To complete the mission design for a real mission, it is necessary to consider the n-body effects in space. The process for refining the preliminary optimal trajectories found using the proposed algorithm, to make sure that the spacecraft does not crash into any of the known celestial bodies, is quite complex and cannot be generalized. However, it can be attempted on a case by case basis and left as a recommendation for future work. It is possible to further optimize the solutions with the use of more than one deep-space maneuver in a leg of the flight. This is also left as a recommendation for future work. Another interesting problem that is worth solving is the trajectory design for the moon tour or orbiter missions of the

parent planet. This problem requires solutions to gravity-assist maneuvers from low-mass moons. This is also left as a recommendation for future work.

BIBLIOGRAPHY

[1] A. Gad and O. Abdelkhalik, "Hidden genes genetic algorithm for multi-gravity-assist trajectories optimization," *Journal of Spacecraft and Rockets*, vol. 48, no. 4, 2011.

[2] S. Molenaar, "Optimization of interplanetary trajectories with deep space maneuvers - model development and application to a uranus orbiter mission," 2009.

[3] C.-H. Chen, J.-H. Chou, and T.-K. Liu, "A novel crowding genetic algorithm and its applications to manufacturing robots," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 3, 2014.

[4] N. Europa, "Europa - in depth," 2016.

[5] N. Titan, "Titan: Saturn's largest moon," 2016.

[6] L. Spilker, "Cassini: Mission to saturn: Enceladus," 2016.

[7] G. Hollenbeck, *New Flight Techniques for Outer Planet Missions*. 1975.

[8] J. A. Sims and J. M. Longuski, *Analysis of V(infinity) leveraging for interplanetary missions*. 2004.

[9] J. A. Sims, J. M. Longuski, and A. J. Staugler, "V8 leveraging for interplanetary missions: Multiple-revolution orbit techniques," *Journal of Guidance, Control, and Dynamics*, vol. 20, no. 3, pp. 409–415, 1997.

[10] A. T. Brinckerhoff and R. P. Russell, *PATHFINDING AND V-INFININTY LEVERAGING FOR PLANETARY MOON TOUR MISSIONS*. 2009.

[11] N. J. Strange, S. Campagnola, and R. P. Russell, "Leveraging flybys of low mass moons to enable an enceladus orbiter," *Advances in the Astronautical Sciences*, vol. 135, no. 3, 2009.

[12] S. Campagnola, R. P. Russell, and N. Strange, "A fast tour design method using non-tangent v-infinity leveraging transfer," *Celestial Mechanics and Dynamical Astronomy*, vol. 108, no. 2, 2010.

[13] A. Gad and O. Abdelkhalik, "Dynamic-size multiple populations genetic algorithm for multigravity-assist trajectories optimization," *Journal of Guidance, Navigation and Control*, vol. 35, no. 2, 2012.

[14] MPI-Standards, "Mpi documents," 2017.

[15] D. Beasly, D. R. Bull, and R. R. Martin, "A sequential niche technique for multimodal function optimization," 2017.

[16] M. Srinivas and L. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 24, no. 4, pp. 656–667, 1994.

[17] F. Peralta and S. Flanagan, "Cassini interplanetary trajectory design," *Control Engineering Practice*, vol. 3, no. 11, pp. 1603–1610, 1995.

[18] B. Buffington, "Trajectory design concept for the proposed europa clipper mission (invited)," *AIAA/AAS Astrodynamics Specialist Conference*, 2014.

[19] P. Tsou, D. E. Brownlee, C. P. McKay, A. D. Anbar, H. Yano, K. Altwegg, L. W. Beegle, R. Dissly, N. J. Strange, and I. Kanik, "Life: Life investigation for enceladus a sample return mission concept in search for evidence of life," *Astrobiology*, vol. 12, no. 8, pp. 730–742, 2012.

[20] H. D. Curtis, *Orbital mechanics for engineering students*. Butterworth-Heinemann, 3 ed., 2013.

[21] S. Fritz and K. Turkoglu, *Optimal Trajectory Determination and Mission Design for Asteroid/Deep Space Exploration via Multi-Body Gravity Assist Maneuvers*. IEEE, 2016.

[22] A. Chamberlin, "Horizons web-interface," 2016.

[23] D. Izzo, "Revisiting lambert's problem," *Celestial Mechanics and Dynamical Astronomy*, vol. 121, no. 1, pp. 1–15, 2014.

[24] C. Sanderson and R. Curtin, "Armadillo: a template-based c++ library for linear algebra," *The Journal of Open Source Software*, vol. 1, no. 2, 2016.

[25] MPICH, "Mpich overview — mpich," 2017.

[26] OPENMPI, "Open mpi: Open source high performance computing," 2017.

[27] HardKernel, "Odroid — hardkernel," 2017.