c

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_1 -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

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 con rmed by future missions [4]. The same can be said about the two moons of Saturn. Given the signi cance of proving the existence of salt-water oceans and

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_1 -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, 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 re ne the feasible trajectories for safety of the spacecraft and success of the mission. The re nement 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 justi ed.

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 e cient 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 diversi ed candidates from the problem search space.

(2) The algorithm should be reasonably fast, i.e., nishing 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 nding a fuel-optimal trajectory to Saturn are presented in Chapter 6.

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

CHAPTER 3

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 nding required velocities, when two positions and time-of- ight 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 in uence 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

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

$$(v^{+}_{s=c})_{nps} = v_p \quad v^{+}_{1} \qquad (3.5)$$

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_1 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 V 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_1 -Leveraging Maneuver

The V_1 -leveraging maneuver is defined as a relatively small deep-space maneuver to modify V_1 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- ight parameter for a leg of the ight is chosen arbitrarily, within certain boundaries. It is therefore bene cial to consider trajectories that involve multiple revolutions of the planet or the spacecraft.

3.6.1 A Procedure for V_1 -Leveraging Maneuver

The following procedure is employed in solving for the parameters of the V_1 -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 di erent conditions under which V_1 -leveraging maneuver is employed in the current study.

3.6.2 The V_1 -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_1 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_1 to determine the value that results in minimum total V with a DSM. The procedure from section 3.6.1 is used repeatedly with different inputs bases on the V_1 value of the current iteration.

3.6.3 The V_1-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

Figure 3.1: Gravity-Assist Orientation [2].

$$\tilde{V}_{out} = \tilde{V}_{pl} + \tilde{V}_{1\ out} \tag{3.9}$$

Once $\tilde{V}_{out}$ is computed, using the position vector $\tilde{r}$ 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

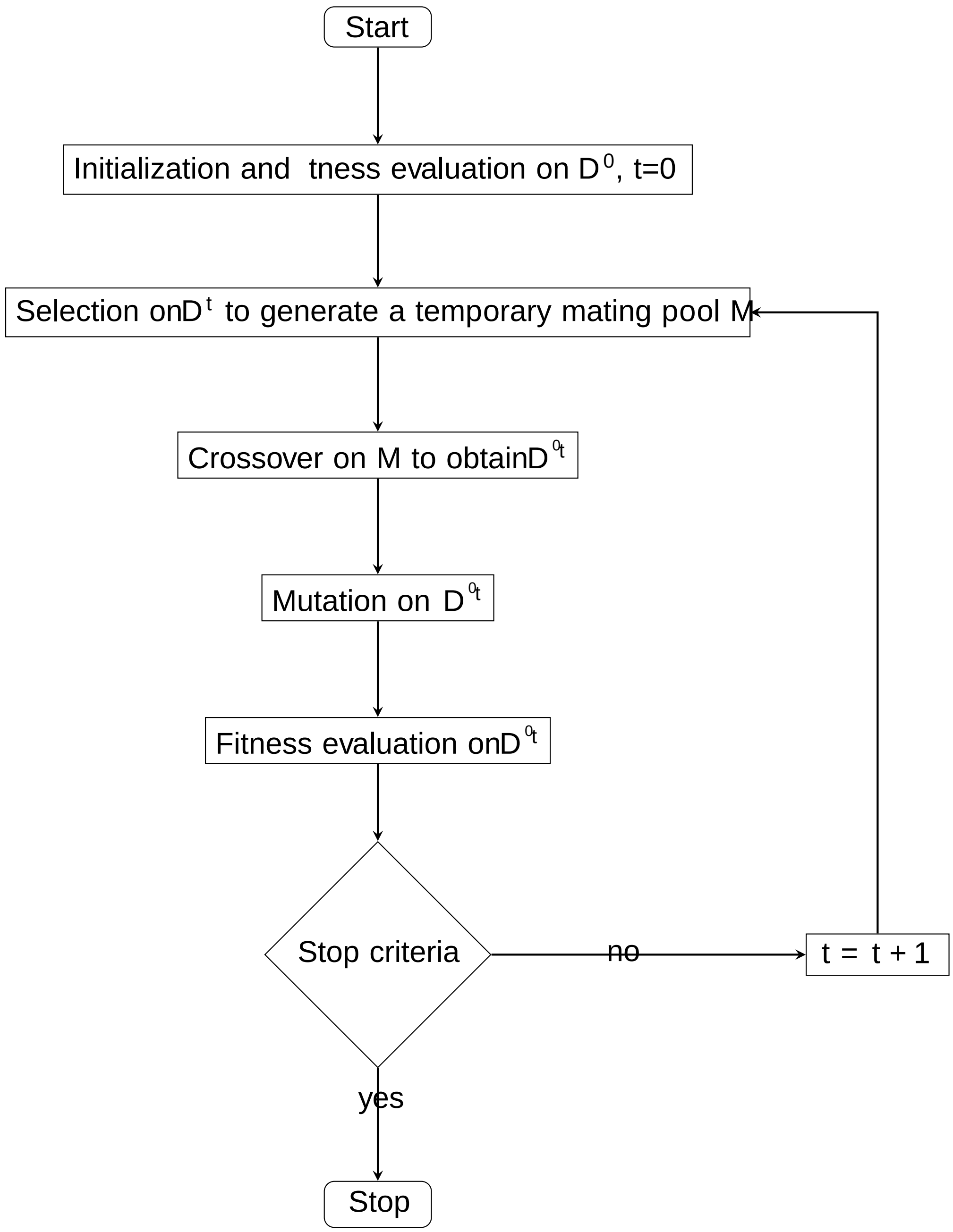


Figure 4.1: Genetic Algorithm Flow Chart

population as the criteria for selection. The purpose of selection is to carry forth the most t 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 tness of all candidate solutions in the population.

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

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

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

(5) The rst candidate solution with the tness 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 tness value in descending order.

(3) Pick the rst candidate solution in the list, i.e. the candidate with the best tness 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 cuto point is chosen randomly. Genes from parent 1 before the cuto point and from parent 2 after cuto point are used to generate child 1. Genes from parent 1 after the cuto point and from parent 2 before the cuto point are used to generate child 2. In Two-point crossover, two cuto points are selected randomly. The two parts of the chromosome from parent 1 before the rst cuto point and after the second cuto point and one part from parent 2 between the two cuto points is merged to produce child 1. Similarly, the two parts of the chromosome from parent 2 before the rst cuto point and after the second cuto point and one part from parent 1 between the two cuto 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 prede ned 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 nal value of the gene falls outside the boundaries for the gene, the value of the gene is adjusted to t 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 nding 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 di erent 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

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.

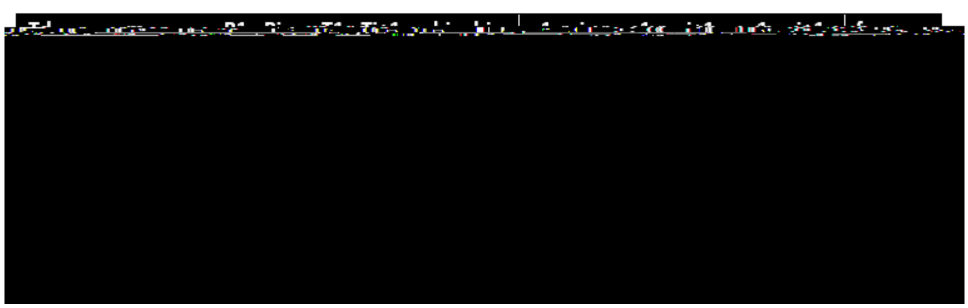


Figure 4.3: Proposed Hidden Gene Chromosomes

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 identi er for the gravity-assist planet for leg i
T_i	Time of ight in seconds in a given leg of ight 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 ight for leg i, indicating the epoch at whir

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 tness function of the proposed chromosome structure from gure 4.3 needs to account for only the e ective genes of the given chromosome. Procedure 4.1 captures the computation of tness for the proposed chromosome structure.

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

T_i chromosome:T_i
h_i chromosome:h_i
g_i chromosome:g_i
n_i chromosome:n_i
e_i chromosome:e_i
directionn chromosome:f
dir (direction == 0)? true : false
for i source to target do

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

```
for i  2 to m-1 do    . Determine the feasibility and compute the V for
  each of the intermediate legs of ight
    p1   Pi(i)
    p2   Pi(i + 1)
    if p1 == p2 then                              . This is a V1 -Leveraging leg
        result   flyby-vilt-non-launch   (r(i); v(i); result:V; hi(i) ? rp(p1) +
  rp(p1); gi(i); r(i + 1); Ti(i + 1); ei(i + 1); mu(p1); safety; soilimits (p1))
    else if ni(i + 1) > 0 then
        result   flyby-with-dsm  (r(i); v(i); result:V; hi(i) ? rp(p1) +
  rp(p1); gi(i); r(i + 1); Ti(i + 1); ei(i + 1); direction; mu(p1); mu(Sun); safety)
    else
        lOutput   lambert  (r(i); r(i + 1); Ti(i + 1); mu(Sun); dir)
        if The Single-revolution Lambert is Not Feasible then
            lOutput   multi-rev-lambert   (r(i); r(i + 1); Ti(i +
  1); mu(Sun); dir)
        end if
        if Neither of Lambert solutions converged then
            return 1
        end if
        dV t   flyby  (result:V; lOutput:V1; v(i); rp(p1); mu(p1); angle-tol)
        if dV t == 1  3/F82 9.9626 Tf -43.793 -21.23 Td [(n)] TJ /F15 1 Tf 1.58 0 Td [(S)] TJ /F21 5 f 1
```

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

```
if target == Pi(m) then
    result    flyby-vilt-non-launch    (r(m); v(m); result:V; hi(m) ?
  rp(Pi(m))+ rp(Pi(m)); gi(m); r(target); Ti(m); ei(m); mu(Pi(m)); safety; soilimits (Pi(m)))
else if
     then result    flyby-with-dsm    (r(m); v(m); result:V; hi(m)?rp(Pi(m))+
  rp(Pi(m)); gi(m); r(target); Ti(m); ei(m); direction; mu(Pi(m)); mu(Sun); safety)
else
    lOutput    lambert    (r(m); r(target); Ti(m); mu(Sun); dir)
    if The Single-revolution Lambert is Not Feasible then
        lOutput    multi-rev-lambert    (r(m); r(target); Ti(m); mu(Sun); dir)
    end if
    if Neither of Lambert solutions converged then
        return 1
    end if
    dV t    flyby    (result:V; lOutput:V1; v(m); rp(Pi(m)); mu(Pi(m)); angle-tol)
    if dV t == 1 then
        return 1
    end if
    result    Result(0; dV t; lOutput:V2; 0)
end if
if result:dV == 1 then
    return 1
end if
dV(last)    result:dV
dV_total    V1 + total    V1
```

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 tness of the chromosome is not normalized. In the current algorithm, there is a possibility of the cost or tness being 1 , 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 tness value of the best parent, and the average and minimum tness values of the population.

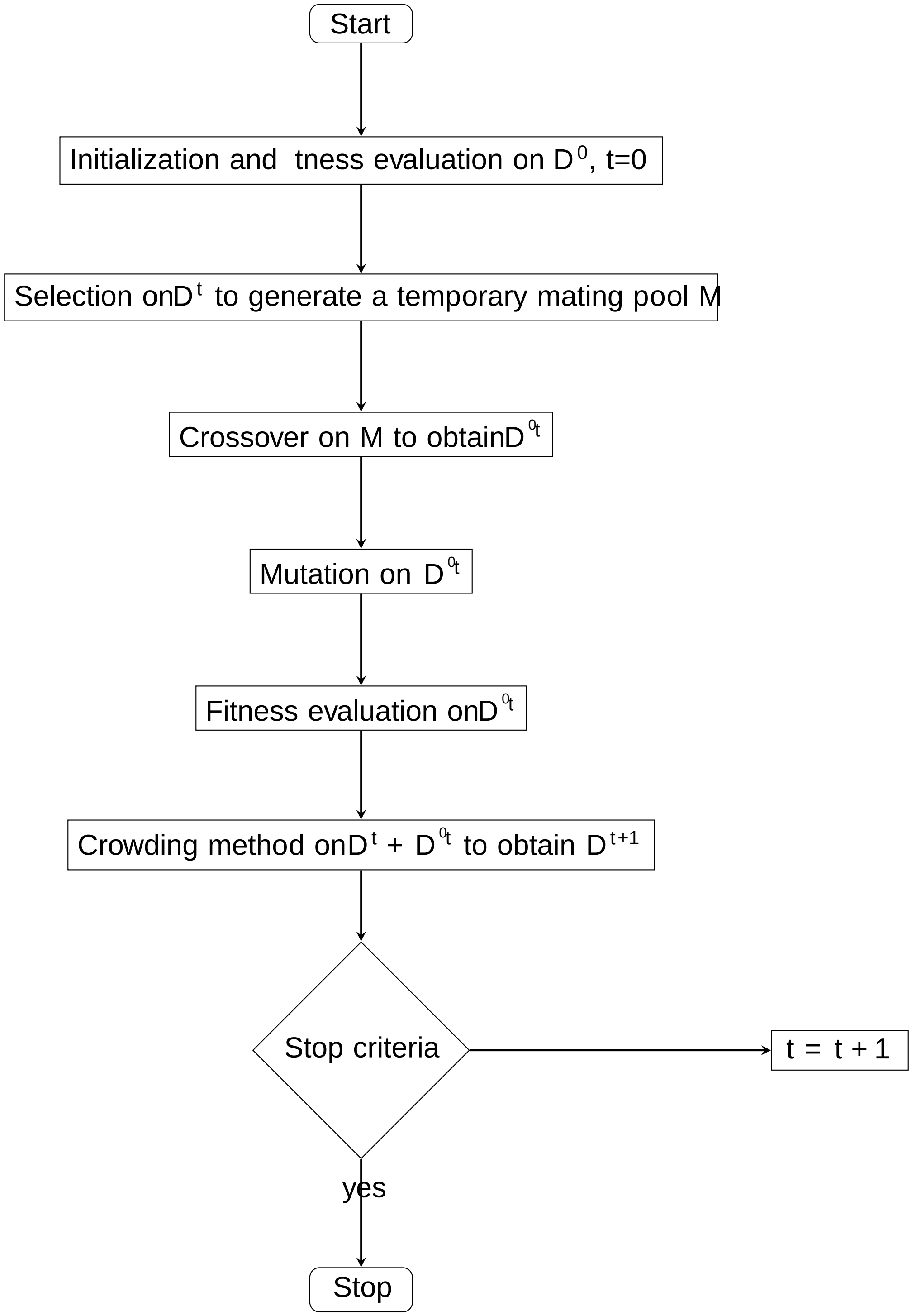
Mutation The proposed algorithm uses the Gaussian mutation scheme. In a deviation from the standard practice of each of the individual genes of the chromosome having a speci c mutation threshold or probability, the proposed algorithm uses a single mutation threshold for all genes. However, the algorithm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

4.2.4 The Diversi cation 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


Start
Initialization and tness evaluation on D0, t=0
Selection onDt to generate a temporary mating pool M
Crossover on M to obtainD0t
Mutation on D0t
Fitness evaluation onD0t
Crowding method onDt + D0t to obtain Dt+1
Stop criteria
t = t + 1
yes
Stop

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

$$p_c = k_1 \frac{f' - f_{min}}{\bar{f} - f_{min}}; f' \le \bar{f} \tag{4.5}$$

$$p_c = k_3; f' > \bar{f} \tag{4.6}$$

and

$$p_m = k_2 \frac{f' - f_{min}}{\bar{f} - f_{min}}; f' \le \bar{f} \tag{4.7}$$

$$p_m = k_4; f' > \bar{f} \tag{4.8}$$

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 tness 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- ight 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- ight speci ed in the gene.

5.1.2 Orbital Mechanics Procedures

A leg of the ight is a ight sequence between any two planets in a trajectory. There are several avors of a leg of the ight depending on the combination of various genes for that leg of the ight in the chromosome, listed as follows:

(1) Single-revolution Lambert's leg

(2) Multiple-revolution Lambert's leg

(3) V_1

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

Procedure 5.2 Procedure for V_1 -Leveraging Launch Leg

---

procedure vilt-launch ($\vec{r}_{pl}$, $\vec{v}_{pl}$, $\vec{r}_{pl_t}$, t, , , soilimits)

Input:

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

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

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

t ! Time of ight for the current leg

! Fraction of time of ight where deep-space maneuver is to be conducted

! The gravitational parameter of the Sun

soilimits ! The limits of the Sphere-of-In uence (SOI) for the launch planet

Output:

Result ! A composite object containing solution parameters

```
    min-dVp  1
    min-dVap  1
    min-err  1
    T_d   t ?
    vnorm   ‖v_pl‖_2
    dir   v_pl / vnorm
    lambert-dir   0
    for dV p   0:1 to 5:0 do
        V   dir ? (vnorm + dV p)
        result   vilt-kepler-lambert   (r_pl; V; r_pl_t; t; ; soilimits; ;  lambert-dir)
```

Procedure 5.2 Procedure for V_1 -Leveraging Launch Leg (continued)

```
      if result:error <  min-err and dV p+ result:dV < min-dVp+min-dVap
  then
          min-err     result:error
          min-dVap  j  result:dvj
          min-dVp     dV p
          ∇r     result:∇
          R̄d     result:R̄d
          ∇d     result:∇d
      end if
   end for
   return  Result(min-dVp; min-dVap; R̄d; ∇d; Td; ∇r ; min-err)
end procedure
```

<u>A Procedure for Launch Leg with Deep-Space Maneuver</u> This procedure is required when the chromosome has a deep-space maneuver speci ed between the launch planet and the target planet in the current leg of ight, and the target planet is di erent 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 ight, the use of a deep-space maneuver is optional. t5ye2l8space maneuvspeci ei(the)-327(use)-re

documents the pseudo-code for this procedure.

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

1: **procedure** launch-dsm $(\vec{r}_{pl}, \vec{v}_{pl}, t, \ , \vec{r}_{pl_t}$

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

```
      if A solution to Kepler's problem is found then
         lOutput    lambert  (rv:r; rpl_t ; t ? (1     );  ; dir )
         if Single-revolution Lambert's solution did not coverge then
            lOutput    multi-rev-lambert     (rv:r; rpl_t ; t ? (1     );  ; dir )
         end if
         if Either of Lambert's solutions coverged then
            dV     lOutput: V1    rv:v
            if min-dV-tot > vinf  + dV^2 then
               min-dV-tot    vinf + dV
               min-vinf    vinf
               min-dV    dV
               V    lOutput: V2
               R_d    rv:r
               V_d    lOutput: V1
            end if
         end if
      end if
   end for
   if Either of direct Lambert's solutions converged then
      dV l     lOutput 1: V1    v_pl  2
```

Procedure 5.4 Procedure for V_1 -Leveraging Leg from a Non-Launch Planet

procedure flyby-vilt-non-launch ($\vec{r}_{pl}$, $\vec{v}_{pl}$, $\vec{v}_{sc_{in}}$, rp, , $\vec{r}_{pl_t}$, t, , $_{pl}$, $_{S}$, safety, soilimits)

Input:

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

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

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

rp ! Periapse radius of the hyperbolic trajectory of the spacecraft around the planet at the start of the leg

! Orientation of the hyperbolic trajectory of the spacecraft around the planet at the start of the leg

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

t ! Time of ight for the current leg

! Fraction of time of ight where deep-space maneuver is to be conducted

$_{pl}$! The gravitational parameter of the planet

$_{S}$! The gravitational parameter of the Sun

safety ! 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 ! The Sphere-of-In uence (SOI) limits for the planet

Output:

Result ! A composite object containing solution parameters

$\vec{V}_{1_{in}}$ $\vec{v}_{sc_{in}}$ $\vec{v}_{pl}$

vinf-in $\vec{V}_{1_{in}}$ $_2$

$\hat{T}$ $\frac{\vec{V}_{1_{in}}}{\text{vinf-in}}$

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_1 -Leveraging Leg from a Non-Launch Planet (continued)

```
    ĵ      î    v⃗pl
    ĵ      ĵ / kĵ k2
    k̂      î    ĵ
    e      1 + rp?vinf    in^2 / pl
           2 sin 1/e
     V⃗1 out      V⃗1 in [cos( )î + sin( )sin( )ĵ + sin( )cos( )k̂]
     dir      0
     result      vilt-kepler-lambert       (r⃗pl ; V⃗1 out ; r⃗plt ; t;  ; soilimits;    S ; dir )
     if safety is set to 1 and result:dV < 1  then
        [a; e]      ae-from-rv    (result: R⃗d; result: V⃗d;   S)
        rpsc      a(1    e^2)
        if rpsc < 10% of an AU then
           return   Result(0; 1  ; empty; empty; t ?  ; empty; 1 )
        end if
     end if
     return   result
end procedure
```

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

procedure flyby-with-dsm ($\vec{r}_{pl}$, $\vec{v}_{pl}$, $\vec{v}_{sc_{in}}$, rp, , $\vec{r}_{pl_t}$, t, , dir , $_{pl}$, $_{S}$, safety)

Input:

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

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

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

rp ! Periapse radius of the hyperbolic trajectory of the spacecraft around the planet at the start of the leg

! Orientation of the hyperbolic trajectory of the spacecraft around the planet at the start of the leg

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

t ! Time of ight for the current leg

! Fraction of time of ight where deep-space maneuver is to be conducted

dir ! 0 or 1 indicating prograde or retrograde motion respectively

$_{pl}$! The gravitational parameter of the planet

$_{S}$! The gravitational parameter of the Sun

safety ! 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 ! A composite object containing solution parameters

A Procedure for determining V_1-Leveraging Trajectory This procedure is required to determine a V_1-Leveraging trajectory in a leg of ight. 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 ight. 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 V maneuver is assumed. It is the goal of this procedure to determine the minimum such V 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 No[illegible] Leg including a Deep-Space Man[illegible] (continued)

```
ĵ  î  v_pl
ĵ  ĵ / kĵk_2
k̂  î  ĵ
e  1 + (rp?vinf  in)^2 / pl
   2 sin 1/e
V1_out  V1_in [cos( )î + sin( )sin( )ĵ + sin( )cos( )k̂]
isLambertSafe  true
lOutput1  lambert (r_pl; r_pl_t; t; S; dir)
if Single-revolution Lambert trajectory does not exist then
    lOutput1  multi-rev-lambert (r_pl; r_pl_t; t; S; dir)
end if
if Either of Lambert's trajec[illegible]sts then
    dV l  lOutput1:V1  V[illegible]
    V l  lOutupt1:V2 S t dir
    [a; e]_out  ae-from-rv (r_pl; lOut[illegible]V1)
```

205 E/F20 if TT Either of f5 1(1).9525.7 11.955 [illegible] 9(1.74 0 Td [(2)]TJ/F15 11.955 [illegible] 16.495 0 Td [(l6J/F15 11.955

Procedure 5.5

Procedure 5.6 Procedure for V_1 -Leveraging Maneuver

procedure vilt-kepler-lambert ($\bar{r}$, $\bar{v}$, $\bar{r}_t$, t, , soilimits , , dir)

Input:

$\bar{r}$! Position vector of the launch planet at launch

$\bar{v}$! Velocity vector of the launch planet at launch

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

t ! Time of ight for the current leg

! Fraction of time of ight where deep-space maneuver is to be conducted

soilimits ! The limits of the Sphere-of-In uence (SOI) for the launch planet

! The gravitational parameter of the Sun

dir ! 0 or 1 indicating the prograde or retrograde motion respectively

Output:

Result ! A composite object containing solution parameters

```
rv    kepler ( r; v; t ? ;  )
if The Kepler's solution did not convergethen
    return Result(0; 1 ; empty; empty; t; empty; 1 )
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 rst 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_1 -Leveraging Maneuver (continued)

```
rvt    kepler  (rv:r; rv:v; t ? (1    );  )
if  The Kepler's solution converged then
    err  k  rvt:r    r_t k_2
    if  err > soilimits:min   and err < soilimits:max   then
        return   Result(0; 0; rv:r; rv:v; t ?  ; rvt:v; 0)
    end if
end if
lOutput1    lambert   (rv:r; r_t; t ? (1    );  ; dir  )
if  Single-revolution Lambert trajectory does not exist then
    lOutput1    multi-rev-lambert     (rv:r; r_t; t ? (1    );  ; dir  )
end if
if  Either of Lambert's trajectories exists then
    dV l     lOutput1:V1    rv:v  2
    return   Result(0; dV l; rv:r; rv:v; t ?  ; lOutput1:V2; 0)
end if
return   Result(1  ; 1  ; empty; empty; t ?  ; empty; 1  )
end procedure
```

two types of ARM Cortex™ 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

Table 6.1: Con guration 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

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 V over the generations of the Genetic Algorithm. The total V 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: Con guration 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

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_1	0.1 km/s
DSM 1 Date	01-12-2022 12:26 PM
DSM 1 V	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 V	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 V	7.561 km/s
Saturn Rendezvous Date	01-12-2035 9:33 PM
Total Mission V	11.2426 km/s
Total Mission Time	14.365 years

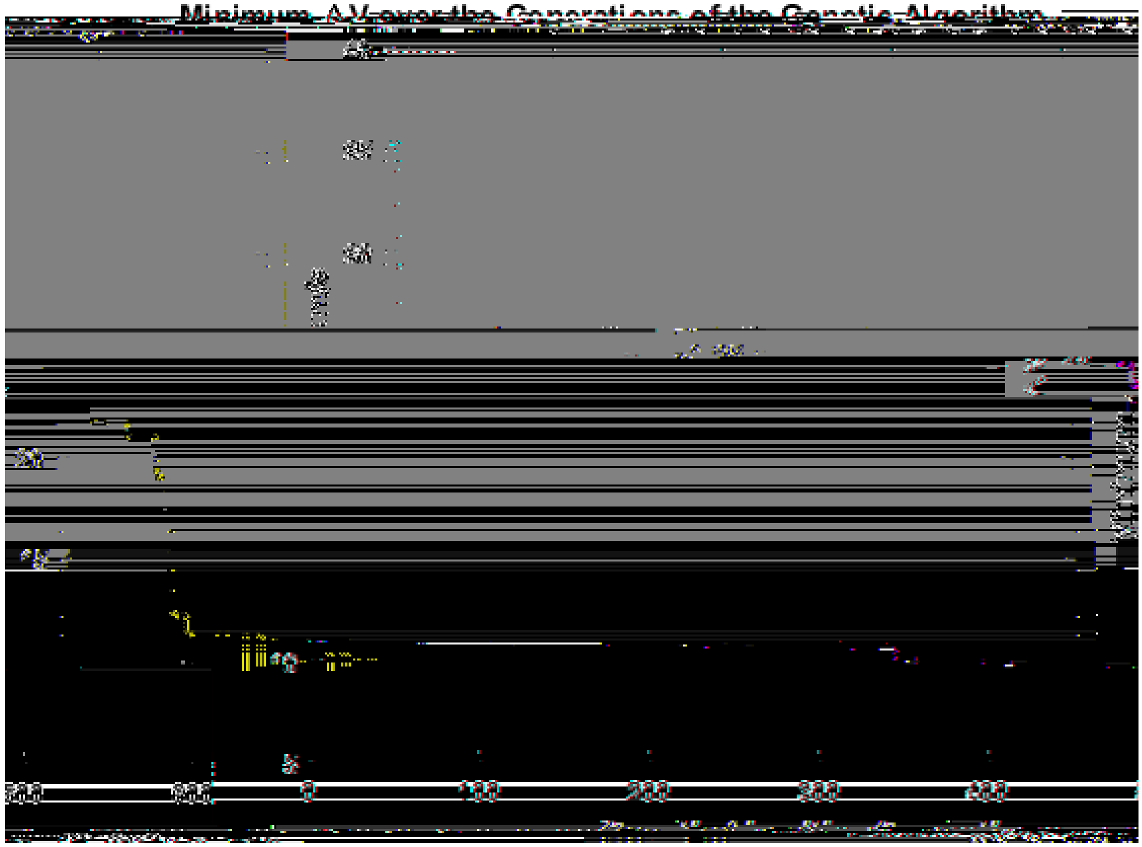


Figure 6.3: The Minimum V over the Generations of the Genetic Algorithm for 4 Gravity-Assists

6.3 Comparison of the Optimal Trajectories

The main di erence 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 V) by 1.2242 km/s. In both the trajectories, the V for Mars Gravity-Assist is very high at 7.561 km/s.

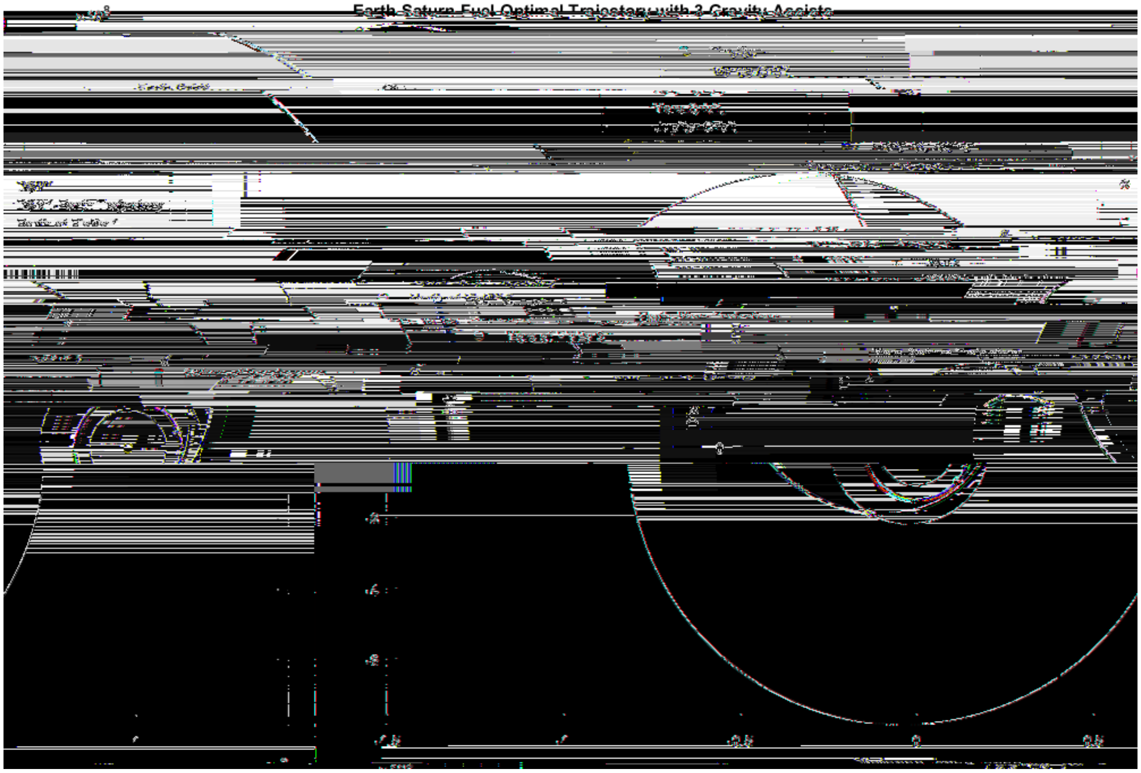


Figure 6.4: An Earth-Saturn Optimal Trajectory with 3 Gravity-Assist Maneuvers

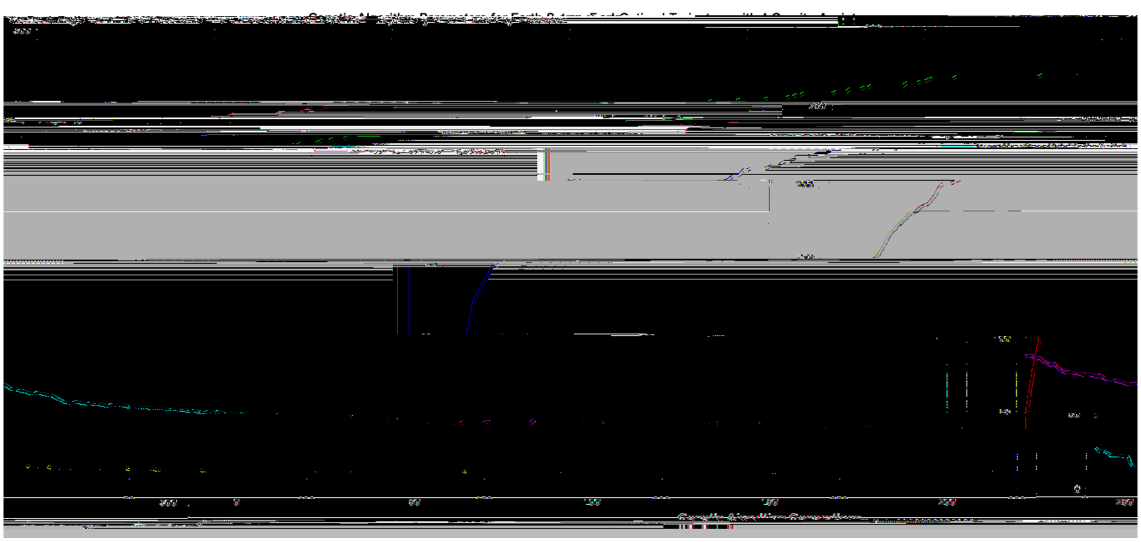


Figure 6.5: The Parameters of the Genetic Algorithm for 3 Gravity-Assists

Figure 6.6: The Minimum V 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 minimumV 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 V values also steeply decrease during this span. After the 125th generation, the number of feasible raises steadily but slowly. Accordingly, the minimum and average V 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 V values. This is due to the combination of adaptive and Twin-Space crowding techniques employed. An attempt is made to use adaptive diversi cation technique alone for the Genetic Algorithm. However, it is observed that the convergence is not as e ective as when the two diversi cation 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 e cient.

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/ tness function does not 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(in nity) 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. Brinckerho 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 ybys of low mass moons to enable an enceladus orbiter," Advances in the Astronautical Sciences, vol. 135, no. 3, 2009.23.726 Td [([10])]mIinR. P. GadN[-490(N.)-326(J.)-326-326(

[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. Bu ngton, \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.