MSc Thesis Report

Design of a Combinatorial Tool for Preliminary Space Mission Analysis

applied to the GTOC2 problem

Thomas André Leite Pinto Secretin

September 3, 2012

TU Delft Delft University of Technology

MSc Thesis Report

Design of a Combinatorial Tool for Preliminary Space Mission Analysis

applied to the GTOC2 problem

Thomas André Leite Pinto Secretin

September 3, 2012

Thomas André Leite Pinto Secretin

Delft University of Technology
Faculty: Aerospace Engineering
Department: Space Engineering
Chair: Astrodynamics and Space Missions
Building 62, Room 9.06
Kluyverweg 1, 2628 CX Delft
The Netherlands

E-mail: T.A.LeitePintoSecretin@student.tudelft.nl

Cover figure: Double the Rubble, NASA Featured Images and Galleries

---

1^{st} Printing 2012
2^{nd} Edition 2012

This document was generated with LaTeX 2_ε code used by the TeX typesetting system, including the following packages: `graphicx`, `float`, `amsmath`, `amsthm`, `longtable`, `amsfonts`, `fancyhdr`, `xcolor`, `wasysym`, `textcomp`, `url`, `color`, `listings`, `glossaries`, `natbib`, `hhline`, `epstopdf` and `hyperref`.

"Our greatest glory is not in never falling, but in rising every time we do."

- Confucius, Chinese Philosopher (551 - 479 BC)

Preface

Having completed this report, I find myself thinking about the incredible experience this last year has been. After contemplating a number of possible topics for my final thesis work, the combinatorial problem of GTOC2 immediately sparked my interest. And I initially dived into it with the confidence and enthusiasm that a new challenge brings, determined to learn from past mistakes, to accumulate as much knowledge on the topic as possible, to design the perfect approach to solve this sizeable problem. But I believe I learned the most when faced with obstacles and disappointments, requiring a positive, yet critical, attitude to overcome. The final product of this MSc Thesis work is truly the result of a learning process, with its joys and frustrations. At the end of it all, I am pleased with the work I have performed. And while the final results may not correspond to my (conceivably unreasonable) initial expectations, I believe I performed a thorough, critical and elucidative analysis.

I would also like to thank the people that contributed, in a way or another, to this project. First and foremost, I would like to thank my MSc Thesis supervisor, ir. Ron Noomen, for the regular feedback he provided me. But most of all for never letting my main goal out of my sight and allowing me to recenter my efforts where they were needed. Our weekly discussions were always well-humoured and enlightening. From an educational and personal perspective, it has been a pleasure. My gratitude also goes to Prof.dr.ir. K.I. Aardal, from the Delft Institute of Mathematics, for her precious help when I was struggling to grasp the essence of combinatorial optimization, a field little-known to me at the beginning of this project. Her knowledge and insight were of great value in a time when they were most needed. I can't help but think of all the Tudat developers: working with them was never a burden and certainly an instructive part of my studies. My appreciation also to my fellow 9^{th} floor students for the laughs we shared, be it at SpaceBar, during musical moments, or for no reason in particular. Finally, I would like to show my gratitude to all those, family, friends and more, who made my stay in Delft, in these past three years, such a memorable experience.

Thomas André Leite Pinto Secretin
Delft, September 2^{nd}, 2012

Contents

Acronyms

A&S	Astrodynamics and Space Missions
ACO	Ant Colony Optimization
ACT	Advanced Concepts Team
B&B	Branch-and-Bound
CPU	Central Processing Unit
DE	Differential Evolution
DSPP	Dynamic Shortest Path Problem
EGTSP	Exact Generalized Travelling Salesman Problem
ESA	European Space Agency
EvOps	Evolutionary Operators
GA	Genetic Algorithm
GALOMUSIT	Genetic ALgorithm Optimization of a MUltiple Swing-by Interplanetary Trajectory
GS	Grid Search
GTOC	Global Trajectory Optimization Competition
GTOC2	2^{nd} Global Trajectory Optimization Competition
GTOP	Global Trajectory Optimization Problem
ISAS	Institute of Space and Astronautical Science
JAXA	Japan Aerospace Exploration Agency
LBG	Lancaster, Blanchard & Gooding
LPTC1	Lambert Problem Test Case 1
LPTC2	Lambert Problem Test Case 2
MSc	Master of Sciences
NASA	National Aeronautics and Space Administration
NNH	Nearest Neighbour Heuristic
OPTIDUS	OPtimization Tool for Interplanetary trajectories by Delft University Students
PaGMO	Parallel Global Multi-objective Optimization
PSO	Particle Swarm Optimization
RAAN	Right Ascension of the Ascending Node
RMS	Root Mean Square
SPP	Shortest Path Problem
SPPTC1	Shortest Path Problem Test Case 1
SPPTC2	Shortest Path Problem Test Case 2
STK	Satellite ToolKit
TSP	Travelling Salesman Problem
TU Delft	Delft University of Technology
Tudat	TUDelft Astrodynamics Toolbox

USA United States of America

List of Symbols

Ω	Right ascension of the ascending node [rad]
$\Delta\Omega$	Change in Right Ascension of the Ascending Node [rad]
α	α parameter in Lagrange's expression for the time-of-flight [rad]
β	β parameter in Lagrange's expression for the time-of-flight [rad]
δ_{in}	Vertex indegree in Graph Theory
δ_{out}	Vertex outdegree in Graph Theory
δ	Declination [rad]
δ	Absolute difference
η	η parameter in the Lagrange-Gauss Equations [-]
∞	Infinity
λ	λ parameter in the Lambert Problem [-]
μ	Gravitational parameter [m^3/s^2]
ω	Argument of perigee [rad]
ϕ	ϕ parameter in the Lagrange-Gauss Equations [-]
π	Ratio of the circumference of circle to its diameter, $\pi = 3.14159265$ [-]
ψ	ψ parameter in the Lagrange-Gauss Equations [-]
σ	σ parameter in the Lagrange-Gauss Equations [-]
σ	Standard deviation
τ	Control parameter of a self-adaptive DE [-]
τ	Time instant in Dynamic Shortest Paths Problems [s]
τ	Synodic period [s]
θ	Tangential unit vector [rad]
θ	Transfer angle in the Lambert Problem [rad]
$\Delta\varepsilon$	Change in specific energy [m^2/s^2]
A	Asteroid
$\bar{A}$	Set of reverse arcs in Graph Theory
A	Set of arcs in Graph Theory
a	Semi-major axis [m]
a_m	Semi-major axis of the minimum energy ellipse in the Lambert Problem [m]
c	Chord in the Lambert Problem [m]
c	Effective exhaust velocity of the rocket engine [m/s]
C_r	Crossover probability Differential Evolution [-]
D	Directed graph in Graph Theory
$\bar{D}$	Reverse graph in Graph Theory
E	Earth
E	Eccentric anomaly [rad]
e	Eccentricity [-]

F	Scale factor in Differential Evolution [-]
f	True anomaly in the Lagrange-Gauss Equation [rad]
F_l	Lower bound of the scale factor of a self-adaptive DE [-]
F_u	Upper bound of the scale factor of a self-adaptive DE [-]
G	Undirected graph in Graph Theory
g	Generation in Differential Evolution
g_0	Standard gravitational acceleration [m^2/s]
H	Specific angular momentum [m^2/s]
Δi	Change in inclination [rad]
i	Inclination [rad]
i	Unit vector
I_{sp}	Engine specific impulse [s]
J	Cost function or objective function
j	Parameter index in Differential Evolution
j_{rand}	Randomly generated index in Differential Evolution
M	Mars
m_f	Final spacecraft mass [kg]
m	Number of asteroids in an asteroid group [-]
m	Path dimension in Graph Theory
N	Number of champion paths in the multi-path greedy algorithm [-]
n	Ratio of the final orbit radius to the initial orbit radius [-]
$n_{\text{transfers}}$	Number of transfer evaluated by the greedy algorithm [-]
p	Semi-latus rectum [m]
R	Position normalizing value in Izzo's Lambert Solver [m]
r	Radial unit vector [m]
r	Heliocentric radius [m]
s	Source vertex in the Shortest Path Problem
s	Semi-perimeter in the Lambert Problem [m]
T	Time normalizing value in Izzo's Lambert Solver [s]
T	Orbital period [s]
T	Departure window [years]
T	Time interval [s]
t	Target vertex in the Shortest Path Problem
t	Time instant [s]
t_a	Arrival Date [JD]
t_d	Departure Date [JD]
t_f	Time-of-flight [s]
t_{spec}	Specified time-of-flight in the Lambert Problem [s]
t_w	Time of wait [s]
u	Trial vector Differential Evolution
u	Vertex in Graph Theory
(u,v)	Arc in Graph Theory
V	Set of vertices in Graph Theory
V	Velocity normalizing value in Izzo's Lambert Solver [m/s]
V	Velocity [m/s]
V_c	Circular velocity [m/s]
V_∞	Hyperbolic excess velocity [m/s]
ΔV	Magnitude of impulsive maneuver [m/s]
v	Mutant vector in Differential Evolution
v	Vertex in Graph Theory

w	Weight function in graph in Graph Theory
$\bar{w}$	Weight function in reverse graph in Graph Theory
w_d	Weight function in dynamic graph in Graph Theory
w_t	Weight function in dynamic graph in Graph Theory, dependent on departure arrival time instants
x_0	x_0 parameter in Izzo's Lambert Solver [-]
x	Relative number of champion paths in the multi-path greedy algorithm [%]
x	Individual in Differential Evolution
x	x parameter in the Lagrange-Gauss Equation [-]
x	Cartesian coordinate along the X-axis
y	y parameter in Izzo's Lambert Solver
y	Cartesian coordinate along the Y-axis
z	Cartesian coordinate along the Z-axis

List of Subscripts

θ	Tangential in Izzo's Lambert Solver
1	Initial, departure
2	Final, arrival
ae	Semi-major axis and eccentricity
cor	Correction
f	Final
h	Angular momentum in Izzo's Lambert Solver
i	Index of a target individual in Differential Evolution
i	Initial
i	Iteration number
j	Body j
L	Launch in [Evertsz, 2008]
M	Mars
max	Maximum
min	Minimum
p	Pericenter
R	Rendezvous in [Evertsz, 2008]
r	Radial in Izzo's Lambert Solver
rel	Relative
RL	Relaunch in [Evertsz, 2008]
s	Spacecraft
tot	Total
x	Cartesian x-component
y	Cartesian y-component
z	Cartesian z-component

Chapter 1

Introduction

On April 15, 2010 Barack Obama, the President of the United States of America (USA), presented his views on the future of space exploration. In his speech, Obama vowed to "send humans to orbit Mars and return them safely to Earth", by 2030[1]. His address to the citizens of the USA, given at the Kennedy Space Center, Florida, was clearly a reference to the famous 1962 "Moon Speech" of John F. Kennedy[2]. In that same speech, Obama set the timeline to land an astronaut on an asteroid for the first time in history to 2025.

While Obama's speech was a strong political statement that manned missions to asteroids can be seen as a milestone towards a human landing on Mars, these celestial objects have been a focus of scientific and industrial research for more than a decade. The main reason for this is the theory according to which asteroids are remnant debris from the inner solar system formation process. In that period, the dominant gravity of Jupiter inhibited the *accretion* of *planetesimals*[3] in the region between Mars and Jupiter, now known as the Main Asteroid Belt [Yeomans, 2011]. Another reason that led to a renewed interest in studying asteroids is the belief that an asteroid impact at the end of the Cretaceous era (about 65 million years ago) caused the extinction of the dinosaurs. This raised awareness on the destructive potential of these celestial objects, as large asteroid collisions can significantly modify the Earth's biosphere [Yeomans, 2011]. Finally, asteroids are thought to be a rich source of volatiles and minerals. Many believe these resources will be instrumental in large-scale space development, namely the exploration and colonisation of our solar system [Lewis et al., 1993].

Consequently, a number of unmanned missions to asteroids have been undertaken in the last decade, by different space agencies. The most notable are the Dawn, Rosetta and Hayabusa missions. The Hayabusa mission was originally a joint project between the Institute of Space and Astronautical Science (ISAS), Japan and the National Aeronautics and Space Administration (NASA), USA. In 2003, ISAS merged with two other Japanese agencies to form the Japan Aerospace Exploration Agency (JAXA), who took the lead on the Hayabusa

[1]The full speech, entitled "Remarks by the President on Space Exploration in the 21st century", can be read at: `http://www.nasa.gov/news/media/trans/obama_ksc_trans.html`

[2]John F. Kennedy's Rice Stadium Moon Speech can be found at: `http://er.jsc.nasa.gov/seh/ricetalk.htm`

[3]The accretion of planetesimal is a theory that explains the formation of celestial bodies through the gravitational attraction of ever more massive particles [Wikipedia, 2011a].

mission. Launched in that same year, its primary scientific objective was to return to Earth a sample from the surface of 25143 Itokawa and it represented the first attempt to return asteroid surface material. The spacecraft landed on the asteroid and collected samples in November 2005. It then returned to Earth, completing its mission in June 2010. Other scientific goals, such as detailed studies of the asteroid shape, spin state, topography and composition, were completed while orbiting the asteroid [Wikipedia, 2011c]. The Rosetta mission was developed by the European Space Agency (ESA) and launched in 2004. While the primary target of the Rosetta mission is the comet 67P/Churyumov–Gerasimenko, the mission profile includes two asteroid fly-by at 2867 Steins and 21 Lutetia, achieved in September 2008 and June 2011, respectively. It is expected to reach its final target in 2014 [Wikipedia, 2011d]. Finally, the NASA-led Dawn mission was launched in 2007. It was designed to rendezvous with and orbit two of the largest asteroids of the Main Asteroid Belt: 4 Vesta and 1 Ceres. The former was reached in July 2011 and the Dawn spacecraft will make in-orbit studies of its characteristics until 2012. It will then depart towards 1 Ceres, with the rendezvous expected in 2015 [Wikipedia, 2011b]. These missions show the world-wide interest in designing mission to asteroids and comets.

This particular type of missions has introduced a new challenge in space mission analysis: target selection. Indeed, given the large number of small solar system bodies[4] and their various orbital and physical characteristics, selecting a specific target is a rather complex task. This problem of target selection was given to the participants of the 2^{nd} Global Trajectory Optimization Competition (GTOC2) in 2006. The Global Trajectory Optimization Competition (GTOC) was first initiated by the Advanced Concepts Team (ACT) of ESA as a means to foster the automation and development of novel techniques in the field of space trajectory design. For GTOC2, the problem proposed by [Petropoulos, 2006] relied on the design of a low-thrust mission performing four consecutive rendezvous with four different asteroids. The target pool consisted of over 900 asteroids, divided in four groups, one asteroids from each group needing to be included in the mission. The optimization objective was the minimization of a cost function combining both the propellant mass and the time-of-flight, while satisfying a number of constraints. For more details on the GTOC2 assignment, please refer to Chapter 2.

At the time, a Delft University of Technology (TU Delft)-led team participated in GTOC2 but was unable to deliver a solution complying with the whole set of constraints. Later on, the GTOC2 problem was tackled by students of the A&S Department of the Aerospace Faculty of the TU Delft, in the scope of their MSc Theses, with a view to improve on the previous results. The first attempt, by [Evertsz, 2008], relied on an angular-momentum-based pruning technique, a high-thrust initial guess and a Chebyshev low-thrust model. However, this analysis did not produce entirely satisfactory results as the best solution found still violated the mission constraints. Another, more sophisticated approach, was implemented by [Gorter, 2010]. The GTOC2 target selection problem was modelled as a variant of the Travelling Salesman Problem. Unfortunately, this approach did not yield a solution complying with the mission constraints. For a more detailed review of the past attempts, within the Astrodynamics and

[4]The ESA Planetary Database (`http://pdb.estec.esa.int/`) currently contains 556830 asteroid entries and 314 comet entries.

Space Missions (A&S) department, at solving the GTOC2 problem, the reader is referred to Chapter 3.

Building on the lessons learned from these previous attempts, and based on the theoretical background acquired for this purpose, the author designed an algorithm aiming at reducing the pool of GTOC2 candidate targets in a computationally fast manner. Note that the emphasis is placed on the discrete aspect of the GTOC2 problem: the goal of this MSc Thesis is not to provide a solution to the GTOC2 problem but rather a subset of the possible asteroid combinations, obtained with low computational effort, that are believed to contain a good solution to the continuous component of the problem. Ideally, this subset will contain the GTOC2 winning sequence. The proposed tool, while being designed for the GTOC2 problem, should be capable to cope with any similar target selection problem. As a result, the MSc Thesis statement was formulated as

> Design of a Combinatorial Tool for Preliminary Space Mission Analysis, applied to the GTOC2 problem.

The following chapters that constitute this MSc Thesis report aim at accurately describing the (theoretical) background behind the techniques employed, the rationale behind each design choice taken, the practical results obtained and their significance with respect to options made.

The document is divided in four parts. Part I concerns itself with the legacy of previous attempts to solve the GTOC2 problems: the GTOC2 problem and solutions are introduced in Chapter 2, while Chapter 3 gives insight into the known models used by both GTOC2 participants and A&S students when tackling this problem. Thereafter, Part II lays down the theoretical background, design choices and tuning and verification efforts for each of the building blocks of the proposed combinatorial tool: the combinatorial model (Chapter 4), the improved greedy algorithms (Chapter 5), the continuous optimizers (Chapter 6) and the Lambert routine (Chapter 7). Their assembly into the final combinatorial tool, in the framework of TUDelft Astrodynamics Toolbox (Tudat), is discussed in Chapter 8. Part III dives into the wealth of results obtained for the complete asteroid set with Grid Search (GS) and with Differential Evolution (DE) in Chapters 9 and 11, respectively, making an informative detour via the results found in a reduced asteroid pool comprised of the GTOC2 participant asteroids, in Chapter 10. Finally, Part IV brings down the curtain with conclusions (Chapter 12) and recommendations with respect to future work (Chapter 13).

Part I

Heritage

Chapter 2

GTOC2

In this chapter, the 2^{nd} Global Trajectory Optimization Competition (GTOC2) problem will be completely defined. We shall first have a look, in Section 2.1, at the history of GTOC competitions, followed by a description of the GTOC2 assignment in Section 2.2, including the problem statement, a discussion on the various types of asteroids and an extensive list of mission and engineering parameters. In Section 2.3, an overview of the results of the competition is given.

2.1 GTOC Background

The Global Trajectory Optimization Competition (GTOC) was created in 2005 by Dario Izzo, a member of the Advanced Concepts Team of ESA. The intent of this competition is to stimulate researchers to find the best solution to an interplanetary trajectory problem and thereby advance the technical developments worldwide in the field of mission analysis. The GTOC problems usually involve the need for a global optimization over a large design space (e.g. large launch window), with many local optima and unusual objective functions or constraints, such that no canned method or existing software can likely fully solve the problem [Petropoulos, 2006].

As of today, five GTOCs have been held. They are listed below, along with their respective winners:

- GTOC1 (2006) - JPL;
- GTOC2 (2007) - Politecnico di Torino;
- GTOC3 (2008) - CNES;
- GTOC4 (2009) - Moscow State University;
- GTOC5 (2010) - JPL.

This MSc Thesis will be focused on finding a satisfying solution for the discrete aspect of the GTOC2 problem, ideally outperforming the solution found by the winners, the team from the Politecnico di Torino.

2.2 GTOC2 Problem

2.2.1 Summary

The GTOC2 assignment, as designed by JPL, is a multiple asteroid rendezvous problem. The goal is to design a low-thrust trajectory for a spacecraft to be launched from Earth and subsequently to perform a rendezvous with one asteroid from each of the four defined sets of asteroids. The optimization process aims at maximizing the ratio of final spacecraft mass to flight time [Petropoulos, 2006].

This problem involves both a combinatorial and a trajectory optimization problem. The combinatorial issue here is finding the sequence of asteroids that will accommodate the optimal trajectory, while the trajectory optimization problem consists of finding the optimal rendezvous trajectory linking asteroids from the selected sequence.

2.2.2 The Four Asteroid Groups

For the GTOC2 problem, four groups of asteroid have been identified, every asteroid within each group having similar characteristics, whether regarding their composition or their orbit [Petropoulos, 2006]. The groups, the number of asteroids they contain being indicated between brackets, are as follow:

Group 1 (96): Jupiter Former Comets, i.e. asteroids which are believed to have once been comets but had their orbit modified by Jupiter's gravitational field.

Group 2 (176): C- or M-class asteroids, i.e. asteroids whose Tholen spectral type[1] is C (Carboneous) or M (Metallic).

Group 3 (300): S-class main belt asteroids, i.e. asteroids whose Tholen spectral type is S (Silicateous) and which lie in the Main Belt[2].

Group 4 (338): Aten asteroids, i.e. asteroids having a semi-major axis less than 1 AU, but apohelion radius greater than 1 AU.

The ephemeris data of each asteroid in the database was provided in a `.txt` file along with the problem description. The spatial distribution of these four asteroid groups is given in Figure 2.1: Group 1 is represented in green, Group 2 in blue, Group 3 in yellow and Group 4 in red.

From the problem description, the spacecraft trajectory must be such that the vehicle performs a rendezvous with one asteroid from each of these groups. Along with this fundamental criterion, there are other requirements that have to be met within the scope of the GTOC2 competition.

2.2.3 Mission and Engineering Parameters

The asteroids, as well as the Earth, are assumed to follow Keplerian orbits around the Sun. Regarding the spacecraft, it is said that only the Sun's gravity

[1] Asteroid classification first proposed by David J. Tholen in 1984. It was developed from broad band spectra information in combination with albedo values [Wikipedia, 2010b]

[2] Also referred to as the Asteroid Belt, this region lies between the orbits of Mars and Jupiter [Wikipedia, 2010a]

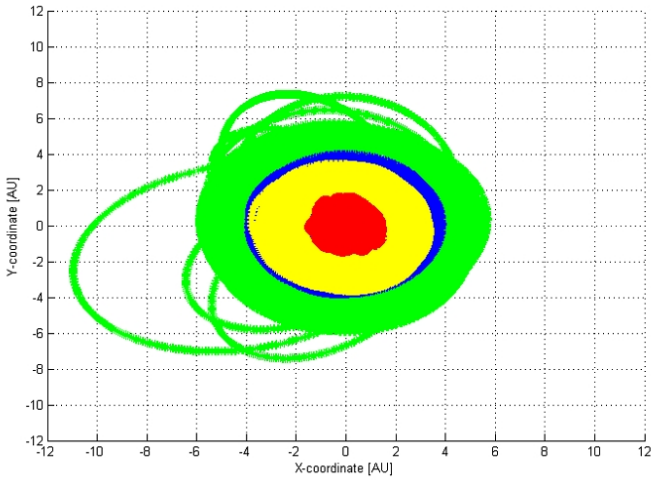

Figure 2.1: Ecliptical top view of the orbits of the asteroids from the four groups, [Evertsz, 2008]

and, when relevant, the engine thrust are to be considered. The GTOC2 assignment also contains a number of constraints, both on an engineering and on a mission level.

The mission constraints are:

- The spacecraft is to be launched from Earth with a hyperbolic excess velocity $V_\infty \leq 3.5$ km/s with no direction constraint.
- Launch must occur between 2015 and 2035, inclusive.
- A stay time of at least 90 days is required at each of the first three asteroids.
- The flight time, t_f, measured from launch to arrival at the final asteroid, must not exceed 20 years.
- Gravity assists[3] are not permitted.
- Objective function to be maximized is

$$J = m_f/t_f \tag{2.2.1}$$

  where m_f is the final spacecraft mass.
- For the rendezvous to be successful, the spacecraft must remain with 1,000 km of the asteroid and the velocity difference between both objects must not exceed 1 m/s.

The engineering constraints are:

[3]Hyperbolic flyby of a massive body for purposes of achieving a desirable course change [Petropoulos, 2006]

- The spacecraft has a fixed initial mass of 1500 kg, which does not change with V_∞.
- The initial mass includes 1000 kg of propellant.
- The propulsion is achieved by means of thruster which has a specific impulse of 4000 s and a maximum thrust level of 0.1 N.

2.3 GTOC2 Rankings

On January 2007, the results of GTOC2 were published by [Petropoulos, 2007]. Hence, the following discussion is mostly based on that work. Out of the 26 registered teams, only 15 returned their solution to the GTOC2 problem by the defined deadline. Out of these, only eleven handed in complete solutions, i.e. solutions that satisfied all of the constraints of the problem or had minor constraint violations deemed to be small enough to not induce any penalty. These 11 solutions were ranked according to the corresponding cost function value, as shown in Table 2.1. The remaining solution were either partial, had significant violation constraints (e.g. the TU Delft and Dutch Space team) and, in one case, consisted only of a proposed approach. These were therefore not ranked, as seen in Table 2.1. Tables 2.2 and 2.3 give further details on the solutions returned, in terms of objective function and asteroids and of departure and arrival times, respectively. [Petropoulos, 2007] notes that all teams selected a Group 4 asteroid as starting point and ended at a Group 1 asteroid, based on increasing orbital energy.

Table 2.1: Ranking of returned solutions [Petropoulos, 2007]

Rank	Team		J(kg/yr)
1	4	Politecnico di Torino	98.64
2	13	Moscow Aviation Institute , and Khrunichev State Research and Production Space Center	87.93
3	10	Advanced Concepts Team, ESA	87.05
4	15	Centre National d'Etudes Spatiales (CNES)	85.43
5	1	GMV Aerospace and Defence	85.28
6	2	German Aerospace Center (DLR)	84.48
7	9	Politecnico di Milano	82.48
8	19	Alcatel Alenia Space	76.37
9	14	Moscow State University	75.08
10	7	Tsinghua University	56.87
11	18	Carnegie Mellon University, J. J. Arrieta - Camacho	27.94
-	17	University of Glasgow et al.	73.87[a]
-	21	Technical University of Delft and Dutch Space	15.95[b]
-	23	Facultes Universitaires Notre-Dame de la Paix (FUNDP)	-[c]
-	26	University of Maribor, Bostjan Eferl	-[d]

[a] Significant position and velocity violations at the asteroids and Earth

[b] Significant position and velocity violations at the asteroids and Earth, and flight time limit violation

[c] Only one leg computed (Earth to Group 4)

[d] Only a proposed method described, no solution computed

Table 2.2: Asteroids visited and trajectory characteristics [Petropoulos, 2007]

Rank	Team	V_∞ (km/s)	t_f (yrs)	m_f (kg)	Asteroid sequence (SPKID) and group numbers			
1	4	3.50	9.106	898.2	3258076 (4)	2000060 (3)	2000058 (2)	2002959 (1)
2	13	3.50	10.394	913.9	3250293 (4)	2000149 (3)	2000569 (2)	2002483 (1)
3	10	2.58	9.523	829.0	3170221 (4)	2000574 (3)	2000209 (2)	2011542 (1)
4	15	2.45	9.777	835.2	3170221 (4)	2001990 (3)	2000240 (2)	2001754 (1)
5	1	2.18	10.096	861.0	3017309 (4)	2000443 (3)	2000490 (2)	2001345 (1)
6	2	3.23	10.170	859.1	3250293 (4)	2000027 (3)	2000110 (2)	2001038 (1)
7	9	3.50	10.796	890.5	3288933 (4)	2001707 (3)	2000047 (2)	2014569 (1)
8	19	3.50	10.816	826.1	3329255 (4)	2000232 (2)	2000807 (3)	2001754 (1)
9	14	2.46	11.509	864.1	3170221 (4)	2000043 (3)	2000074 (2)	2002483 (1)
10	7	3.50	12.941	735.9	3250293 (4)	2000149 (3)	2000224 (2)	2009661 (1)
11	18	3.50	19.195	536.3	3343104 (4)	2000169 (3)	2000075 (2)	2000659 (1)
-	17	-	12.991	959.6	3250293 (4)	2000443 (3)	2000058 (2)	2002959 (1)
-	21	-	32.250	514.3	3170221 (4)	2001314 (3)	2000395 (2)	2002483 (1)
-	23	-	-	-	3177202 (4)	-	-	-

Table 2.3: Dates at the various bodies [Petropoulos, 2007]

Rank	Team	Earth launch, and asteroid arrival and departure dates (MJD)							
1	4	59870	60283	60373	61979	62069	62647	62737	63196
2	13	62866	63028	63118	64907	64997	65712	65802	66662
3	10	57372	57747	57849	59485	59587	60034	60139	60851
4	15	59574	60104	60194	61749	61839	62306	62396	63145
5	1	61073	61258	61348	63178	63268	64011	64101	64761
6	2	58021	58379	58469	60236	60326	60872	60963	61735
7	9	62201	62454	62544	64444	64534	65394	65484	66144
8	19	59418	59610	59700	61603	61693	62288	62378	63369
9	14	57561	57987	58106	59627	59717	60935	61025	61764
10	7	58448	58752	58846	60826	61048	61991	62232	63175
11	18	58246	59125	59215	61731	61821	62552	62642	65257
-	17	58460	58794	58884	60623	60714	62303	62393	63204
-	21	57755	58659	58749	61861	62190	64925	65200	69534
-	23	57052	59226	-	-	-	-	-	-

Throughout this report, we will refer to the different GTOC2 participant solutions by the term *GTOC2 #* followed by their rank. Teams with no official ranking are numbered according to their position in Table 2.1. Therefore *GTOC2 #1* refers to the solution from the Politecnico di Milano and *GTOC2 #12* to the sequence from the University of Glasgow.

Chapter 3

Earlier Models and Methods

This chapter aims at providing a brief overview of the different known models applied by different parties to solve the GTOC2 problem and highlight a number of lessons learned which were of importance to the design of the combinatorial tool. We start by looking into the GTOC2 participant models in Section 3.1, where special relevance is given to the different space pruning techniques applied. We then proceed, in Section 3.2, to describing the models and methods developed at the TUDelft, both during the competition and in the following efforts to improve the results from this competition. Lastly, a number of relevant conclusions are drawn in Section 3.3.

3.1 GTOC2 Participant Models

The methods employed by the GTOC2 participants to obtain their final solutions are rather diverse. From the initial pruning step to the final local optimization techniques, different approaches were selected. An overview of these is given in Section 3.1.1. An analysis of the effectiveness of the pruning techniques is summarized in Section 3.1.2.

3.1.1 Methods Overview

Let us now take a look at the methods employed by the different participants. The explanation of all the techniques employed falls outside of the scope of this report but the reader is referred to relevant sections of the report, when applicable. Note that the final objective function value is given between brackets following the team identification. An asterisk indicates a solution with considerable constraint violations. This section is, unless mentioned otherwise, based on [Petropoulos, 2007].

GTOC2 #1 (98.64 kg/yr)

One of the key aspects in the quality of the solution of Team 4 lies with the observation that, of the Group 1 asteroids, those with low energy and low in-

clination pass through their perihelia within two-year windows, repeated about every 8 years. Intercepting the last asteroid just after its perihelion passage is time-efficient as it is the closest the asteroid gets to the Group 2 and Group 3 asteroids. This observation significantly reduced the search space, namely in terms of arrival dates at the last asteroid. Based on several performance indices, a database of optimal trajectories to the first asteroid group (Group 4) was obtained. Group 2 and Group 3 asteroids were then selected using a modified Edelbaum approximation and taking phasing into consideration. The final candidate trajectories obtained in this manner were optimized using the indirect formulation solved by classical shooting.

Note that during the optimization process, Team 4 found various trajectories with a larger cost function than that found by the runner-up GTOC2 #2, including five trajectories with $J > 90$ kg/yr [Casalino et al., 2007].

GTOC2 #2 (87.93 kg/yr)

Team 13 used Lambert solutions (cf. Chapter 7) between asteroids to screen the global search space. The resulting candidates were then optimized using the Maximum Principle, with continuation with respect to the boundary conditions, with respect to the gravity parameter and from a power-limited engine model to the problem's constant exhaust-velocity engine model.

GTOC2 #3 (87.05 kg/yr)

An initial pruning of the asteroid sequence space was performed by Team 10, using for that purpose a branch-and-prune algorithm based on the three-impulsive ΔV of Hohmann-like transfers (cf. Equation (3.2.19)). This procedure resulted in 13132 asteroid sequences, a number further reduced taking asteroid phasing into account, namely assuming Lambert arcs (cf. Chapter 7) between the asteroids with additional time allowed for "spiralling". A second approach was also implemented, substituting the Lambert arcs between the first and the second asteroids by exponential sinusoids (cf. [Secretin, 2011]). The free parameters, i.e. the various departure and arrival times, were then optimized using Differential Evolution (cf. Section 6.2.2). Finally, the optimal solution was found through a non-linear programming method, using a trajectory constructed by exponential sinusoids as an initial guess.

GTOC2 #4 (85.43 kg/yr)

Team 15 reduced the potential asteroid sequences by selecting 22 asteroids based on continuously increasing semi-major axes, minimising inclination corrections and the transfer time from the first to the second asteroid, and selecting reasonable phasing between Group 3, 2 and 1 asteroids. The resulting candidate sequences were then assessed using Lambert arcs (cf. Chapter 7) and impulsive ΔVs, enforcing constraints on arrival, departure and deep-space maneuvers. The assessment was based on a Nelder-Mead simplex (direct, derivative-free) method (cf. [Secretin, 2011]). Finally, the low-thrust problem was tackled using Pontryagin's Maximum Principle and a decomposition-coordination technique.

GTOC2 #5 (85.28 kg/yr)

The search space pruning technique used by Team 1 was done by defining upper bounds in the variation of orbital elements per leg, and on the cost of propellant-optimal two-impulsive phase-free transfers between asteroid pairs. The phasing was then included to further prune the search space, along the replacement of the high-thrust transfer between the first and second asteroid by a low-thrust arc. An augmented objective function was then defined, integrating equality and inequality constraints as penalty functions, and used in Lawden's implicit guidance scheme. Optimization was performed using a Genetic Algorithm (cf. [Secretin, 2011]) and a simplex Nelder-Mead derivative-free optimizer (cf. [Secretin, 2011]).

GTOC2 #6 (84.48 kg/yr)

The solution presented by Team 2 was obtained through a global search on a per-leg low-thrust analysis, performed using neural networks driven by an evolutionary algorithm. The four legs were built up in an iterative loop with a local optimizer, based on a non-linear-programming formulation solved using the optimization package SNOPT.

GTOC2 #7 (82.48 kg/yr)

The global search performed by the team from the Politecnico di Milano resembles that employed by the ESA/ACT Team: Lambert arcs (cf. Chapter 7) were initially assumed for all legs, except for the leg from the first to the second asteroid, taken to be an exponential sinusoid arc[1] (cf. [Secretin, 2011]). Optimization of this simplified model was performed using three different optimizers: Genetic Algorithm (cf. [Secretin, 2011]), Particle Swarm Optimization (cf. [Secretin, 2011]) and Multi-level Coordinate Search. A direct method with either multiple shooting or collocation with Lagrange polynomial interpolation was then applied to optimize the resulting candidates in the full model.

GTOC2 #8 (79.37 kg/yr)

A fourfold approach was employed by Team 19. The first step consisted of screening the asteroids using dynamic programming and an augmented cost function with penalties for large angular momentum changes between asteroids and for large orbital periods of asteroids in the sequence. This led to an asteroid group sequence: Group 4 - Group 2 - Group 3 - Group 1. The second step pruned the departure date, with impulsive trajectories between asteroids ranked by ΔV and duration. Dynamic programming was used in the third step to solve the best candidates as a minimum time problem. The final step solved the results from the previous step as a maximum final mass problem, meaning that the mass and time-of-flight optimization were decoupled.

[1] This exception steemed from the belief that there is little correlation between high- and low-thrust solutions for the second leg.

GTOC2 #9 (75.08 kg/yr)

Team 14 did a preliminary selection of asteroids based on Lambert solutions (cf. Chapter 7) for optimal two-impulsive transfers. The resulting best asteroid sequences were then optimized based on Pontryagin's Maximum Principle, using a shooting method and a continuation method.

GTOC2 #10 (56.87 kg/yr)

After selecting an asteroid sequence, Team 7 divided the trajectory legs into segments during which the thrust magnitude is held fixed, and the direction varies linearly between initial and final cone and clock angles for the segment. In the following step, a Genetic Algorithm (cf. [Secretin, 2011]) was used to optimize the thrust variables, the dates and other problem parameters. An augmented objective function was employed, adding penalty functions to impose constraints. The final accuracy requirements were met through a final local optimization performed by MATLAB's `fminsearch` function.

GTOC2 #11 (27.94 kg/yr)

The initial pruning carried out by Team 18 was based on at first excluding asteroids requiring large plane changes or large rotation of the line of the nodes, and secondly excluding those with large changes in semi-major axes. Based on the resulting subset, the candidate departure dates were found based on when the asteroids were least separated in space. At last, local optimization was performed by means of a direct transcription method in modified equinoctial elements.

GTOC2 #12 (73.87* kg/yr)

The solution found by Team 17 was not ranked due to major constraint violations. Their approach started with the pruning of asteroids based on the orbital elements. Two different optimization approaches were then implemented. In the first one, the dynamical models were coded as "black-box" functions to be optimized by several global and local optimization packages. In the second one, which produced the final solution, candidate solutions were found using Lambert (cf. Chapter 7) and shape-based low-thrust arcs instead, selected by an evolutionary branching algorithm. The team then proceeded to the refinement of the resulting candidate solutions using a direct optimization method. Due to the strict deadline of the competition, Team 17 was unable to completely refine the solutions and therefore large constraint violations were still present in their final solution.

GTOC2 #13 (15.95* kg/yr)

The method employed by the TU Delft-led team is described in Section 3.2.1.

GTOC2 #14

Team 23 had no previous experience with astrodynamics prior to the competition. Their approach was based on Gauss' variational equations. Taking a

pseudo-inverse of these equations in a least-square sense, they were able to determine the thrust profile to reach any given asteroid by choosing suitable dates. The short time available for the competition prevented Team 23 to compute a complete solution, only a trajectory to a Group 4 asteroid was determined.

GTOC2 #15

Team 26 proposed a graph theory approach to the problem, without handing in any solution. Based on the assumption that methods are available to easily compute trajectories from one point in the four-dimensional space-time to a neighbouring point, the proposed graph theory method would scan these points in polynomial time, resulting in the sequence of points that best joins an initial point with a desired final point in the graph.

Observations Regarding GTOC2 Methods

Let us now formulate some general observations regarding the different approaches to the GTOC2 problem. Firstly, it is evident that most teams divided the problem into two parts: the asteroid selection and sequencing, mostly via high-thrust approximation, and trajectory computation, calculus-based methods being preferred by the participants for this part. In other words, the discrete aspect of GTOC2 is generally solved under the high-thrust assumption, which, due to its simplicity, proves to be less computationally demanding. On the other hand, the trajectory is, in most cases, obtained with powerful and accurate calculus-based methods to ensure compliance with the stringent constraint set by the GTOC2 assignment. Also, note that only one team (GTOC2 #8 - Alcatel Alenia Space) opted for a 4 - 2 - 3 - 1 asteroid group sequence; all the remaining participants identified the sequence 4 - 3 - 2 - 1 as being optimal. While it is tempting to attribute the 4 - 3 - 2 - 1 option to the use of initial Lambert solutions, the variety of techniques which resulted in this sequence (including non-high-thrust approaches) leads to the conclusion that the 4 - 3 - 2 - 1 asteroid group sequence is more likely to entail the problem optimum.

3.1.2 Analysis of the GTOC2 Search Space Pruning Techniques

In the wake of the GTOC2 competition, [Alemany and Braun, 2007] performed a review of the search space pruning techniques adopted by the different contenders. The pruning techniques aimed at discarding asteroids and combinations thereof, as well as portions of the launch date and time-of-flight domain. It is emphasized that most teams cited one of their major weaknesses as being their pruning approach, feeling that it had removed some of the best solutions from the search space. As a result, the validity of the pruning methods employed is analysed based on a subset of the full GTOC2 problem, containing 22 representative asteroids. Three major categories of pruning techniques were identified in the scope of GTOC2: ephemeris-based techniques, phase-free high-thrust approximations and phasing considerations. This paragraph and the remainder of this section are, unless mentioned otherwise, based on the work from [Alemany and Braun, 2007]. Please refer to that reference for a more detailed description of the approach implemented to analyse the pruning techniques.

Ephemeris-Based Pruning Techniques

The majority of GTOC2 participants used an ephemeris-based pruning approach consisting of visiting the asteroids in order of increasing semi-major axis: Group 4 - Group 2/3 - Group2/3 - Group 1. Groups 2 and 3 were combined due to the similarities in their semi-major axis values (see Section 2.2.2). This design decision is based on the fact that visiting the asteroids in order of increasing semi-major axis considerably reduces the total mission time-of-flight, which is included in the objective function. This effectively reduces the number of total asteroid combinations by one order of magnitude: from over 41 billion possible combinations to 3.4 billion. This number is still very large and hence other methods were employed to further reduce the search space.

Another popular ephemeris-based pruning method was to filter asteroid combinations based on their inclination changes, rooted in the conjecture that large inclination changes require significant amounts of propellant, as is the case in high-thrust trajectories. Inclination changes and propellant mass proved to be highly correlated in the transfers from Earth to Group 4 (from here on referred to as *Leg 1*) and from Group 4 to Group 2/3 (*Leg 2*). It was also shown that for these two legs there is a maximum inclination change value above which no feasible transfers were found within the time interval considered. On the other hand, there is no apparent correlation between the two parameters for *Leg 3* (transfer from Group 2/3 to Group 1). As a result, if inclination-change pruning is applied to eliminate Group 1 asteroids, it is expected that some of the best solutions would be ruled out at the same time. Figure 3.1 plots the maximum final mass as a function of the inclination change. Note that additional Group 4 asteroids were randomly added to the asteroid subset to yield a representative analysis for Leg 1.

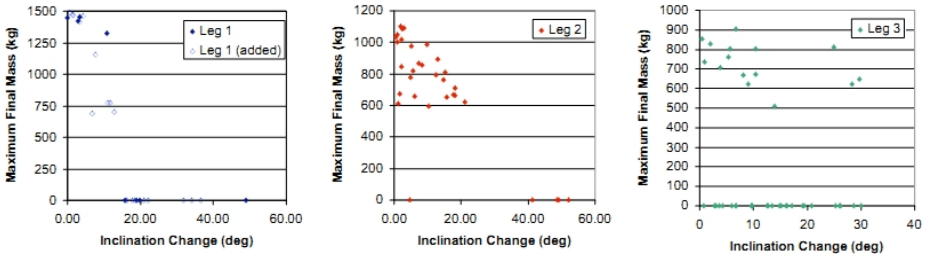

Figure 3.1: Maximum final mass for each trajectory leg as a function of inclination change [Alemany and Braun, 2007].

Another recurrent technique was to combine the inclination change and the change in longitude of the ascending node and use the resulting value to rule out particular asteroid combinations. The premise behind this approach is that if the change in right ascension of the ascending node is small, the inclined orbits are more closely aligned and the resulting transfer requires less propellant. However, testing this hypothesis revealed that it is not sound as it excludes feasible solutions from the design space.

A number of other ephemeris-based pruning techniques were employed but proved not to be reliable. Among these, we will mention excluding Group 1 asteroids with high energy, i.e. large semi-major axis, and filtering out asteroid

combinations with the largest distance between departure asteroid apoapsis and arrival asteroid periapsis.

Phase-Free High-Thrust Approximations

Besides ephemeris-based pruning techniques, most GTOC2 teams approximated the low-thrust problem using Lambert high-thrust solutions, usually without phasing considerations. The optimal transfers found in such manner provide a fast way to determine the most reachable asteroids. As the approximation conducted was phase-free, there is no guarantee that the optimal transfers found will occur within the time frame of the problem. The maximum low-thrust final mass was plotted against the optimal two-impulse ΔV for each leg in Figure 3.2. As can be seen, there is a strong correlation between increasing ΔV and decreasing low-thrust final mass for Leg 1. The same affirmation is valid for Leg 2, while there is no apparent correlation between both parameters for Leg 3, despite a possible threshold value for the optimal two-impulse ΔV above which no feasible low-thrust solution is found. As with ephemeris-based techniques, the high-thrust approximation seems to fail to provide a useful initial guess when applied to the final leg, as indicated by the lack of visible correlation for Leg 3 in Figure 3.2. In addition, the study also shows that increasing the number of revolutions allowed in the solution of the two-point orbital boundary-value problem effectively decreases the correlation between the approximation and the actual low-thrust propellant mass consumption.

Phasing Considerations

If in the previous step the phasing of the different transfers was not taken into account, most teams did reduce the search space further through phasing considerations.

The most common approach was to use the Lambert high-thrust approximation to include the phasing aspect of the problem. However this approach did not appear to consistently represent the real low-thrust phasing, neither in terms of departure dates or times-of-flight. As a result, the bi-impulsive approximation to allow for phasing study is not expected to reliably identify regions of the search space that contain the best solutions.

Another approach adopted by the contenders to address phasing issues was to determine when Group 1 asteroids were at their perihelion, based on the premise that it is most interesting to rendezvous with the final asteroid just after its perihelion passage. The idea behind this premise is that Group 1 asteroids are the closest to Group 2 and 3 asteroids at their perihelion. This allows for considerable savings, namely in terms of time-of-flight, especially if the eccentricity is relatively large. The analysis of the validity of this hypothesis revealed that although the largest values for the final mass do not occur directly after perihelion passage, they definitely lie in its vicinity. Aiming for the (near) perihelion passage of the final asteroid may therefore be used as an effective pruning technique.

Conclusions

Despite the pruning rules generally working for most asteroid combinations and date ranges, exceptions were always found that would have resulted in very

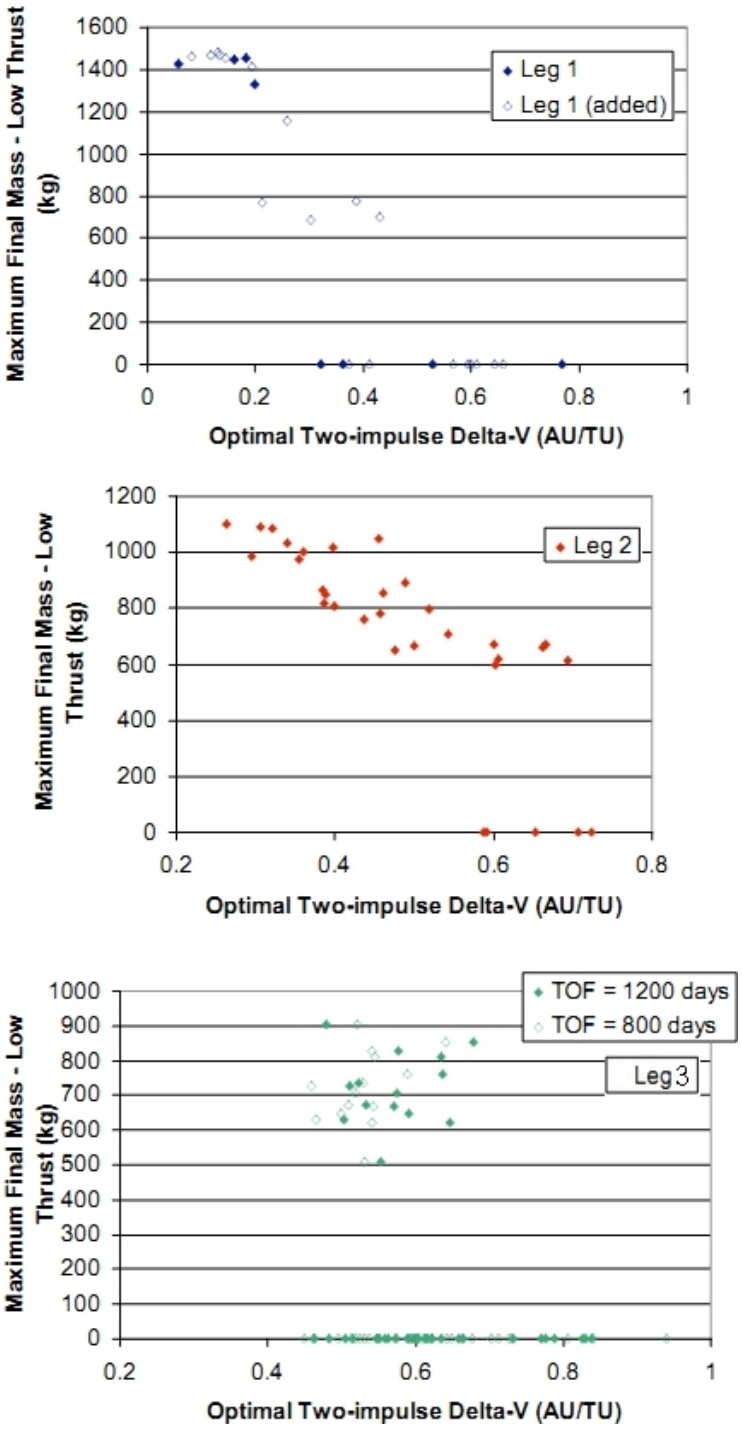

Figure 3.2: Maximum final mass for each trajectory leg as a function of optimal two-impulsive ΔV. Adapted from [Alemany and Braun, 2007].

attractive solutions. These exceptions are likely to be eliminated in a pruning process, hence depriving the search space from potentially (near-)optimal solutions.

The inclination change and the optimal phase-free bi-impulsive ΔV pruning metrics were the most effective in screening asteroid combinations for Leg 1 and Leg 2 but would most likely eliminate attractive solutions if applied to the final leg. Again, as a general rule, the final asteroid was intercepted near perihelion and this pruning metric seems to be quite accurate, despite a few non-optimal outliers. Using the high-thrust model to carry out a phasing analysis does not yield satisfactory results.

[Alemany and Braun, 2007] proposes a probabilistic approach to the definition of threshold values, namely to fit a distribution to each metric. This is presented as a more objective method to determine upper bound values than simple visual inspection and perceived intuition. Note however that this approach is not foolproof and still requires a certain degree of subjectivity. Finally, it is noted that an incremental approach, i.e. solving the individual leg subproblems separately, appears to be appropriate for the pruning phase.

These considerations are taken into account when designing the model and method used to solve the GTOC2 combinatorial aspect, in the scope of the author's MSc Thesis.

3.2 TU Delft Models

The main challenge in finding good solutions for the GTOC2 problem stems from the vast search space and stringent constraints set out in the problem description. A TU Delft-led team of students participated in GTOC2. Since then, there has been a concerted effort within the A&S research group to further investigate feasible solutions and develop new methods to achieve better results. An overview of the models and methods developed at the TU Delft to solve the GTOC2 combinatorial problem is given in this section.

3.2.1 High-Thrust, Ephemeris-Based Screening

During the GTOC2 competition, the team led by TU Delft, Team 21, initially screened the asteroids based on the estimated ΔV needed to change only the ascending node, or only the inclination, or to perform a Hohmann transfer.

It can be shown that in order to produce the desired Right Ascension of the Ascending Node (RAAN) change, $\Delta\Omega$, without affecting the remaining orbital parameters an impulsive maneuver has to be applied at the points of the orbit where the declination is maximum, i.e. where $\delta = \pm i_1$, i_1 being the inclination of the initial orbit. Moreover, it can be shown that the magnitude of that impulse is equal to [Wakker, 2007b]:

$$\frac{\Delta V}{V_1} = 2 \sin i_1 \sin \frac{1}{2} \Delta\Omega \tag{3.2.1}$$

where V_1 is the velocity in the initial orbit at the point where the maneuver is executed. With respect to the inclination, a maneuver aiming at modifying the inclination without affecting other orbital parameters has to be executed at either the ascending or descending node. The relation between impulse magnitude

and inclination change, Δi, is simply [Wakker, 2007b]:

$$\frac{\Delta V}{V_1} = 2 \sin \frac{1}{2} \Delta i \tag{3.2.2}$$

The Hohmann transfer is the minimum propellant consumption, outward[2] transfer orbit between two circular, co-planar orbits. The total ΔV needed to perform a Hohmann transfer is given by [Wakker, 2007b]:

$$\frac{\Delta V_{tot}}{V_{c1}} = (n-1)\sqrt{\frac{2}{n(n+1)}} + \sqrt{\frac{1}{n}} - 1 \tag{3.2.3}$$

where V_{c_1} is the circular velocity in the initial orbit and n is the ratio of the final radius, r_2, to the initial radius, r_1: $n = r_2/r_1$. Using these three metrics, the number of asteroid combinations was reduced.

Promising resulting sequences were then assessed, first using exponential sinusoids (cf. [Secretin, 2011]). Then, due to the large constraint violations yielded by the previous method, an evolutionary algorithm was employed to optimize the low-thrust transfers. The independent variables of the problem were taken to be the various times, the launch V_∞ and the parameters of a low-dimensioned parametrisation of the thrust profile [Petropoulos, 2007].

3.2.2 High-Thrust, Angular Momentum-Based Screening

In the aftermath of the GTOC2 competition, a number of Master of Sciences (MSc) students from the A&S group worked on improving the results obtained. One approach [Evertsz, 2008] relied on pruning the pool of possible targets and subsequently performing a high-thrust optimization, the result of which would serve as an initial guess for the low-thrust optimization.

Based on the ephemeris data of the asteroid groups, it was noticed that the most promising asteroid group sequence was either 4 - 3 - 2 - 1 or 4 - 2 - 3 - 1, which is a result in coherence with various other sources (cf. e.g. Section 3.1.2). Noting that it is a time-consuming process to approach the problem as a whole, the total set of asteroids was then screened via a specific angular momentum-based filter. The specific angular momentum is given by:

$$\mathbf{H} = \mathbf{r} \times \mathbf{V} \tag{3.2.4}$$

which decomposed in Cartesian coordinates results in:

$$\begin{aligned} H_x &= y \cdot V_z - z \cdot V_y \\ H_y &= z \cdot V_x - x \cdot V_z \\ H_z &= x \cdot V_y - y \cdot V_x \end{aligned} \tag{3.2.5}$$

Combining the total specific angular momentum, $\mathbf{H}$, and its z-component, H_z, the deviation of an asteroid orbital plane with respect to the ecliptic, β, can be determined:

$$\cos \beta = \left(\frac{H_z}{|\mathbf{H}|} \right) \tag{3.2.6}$$

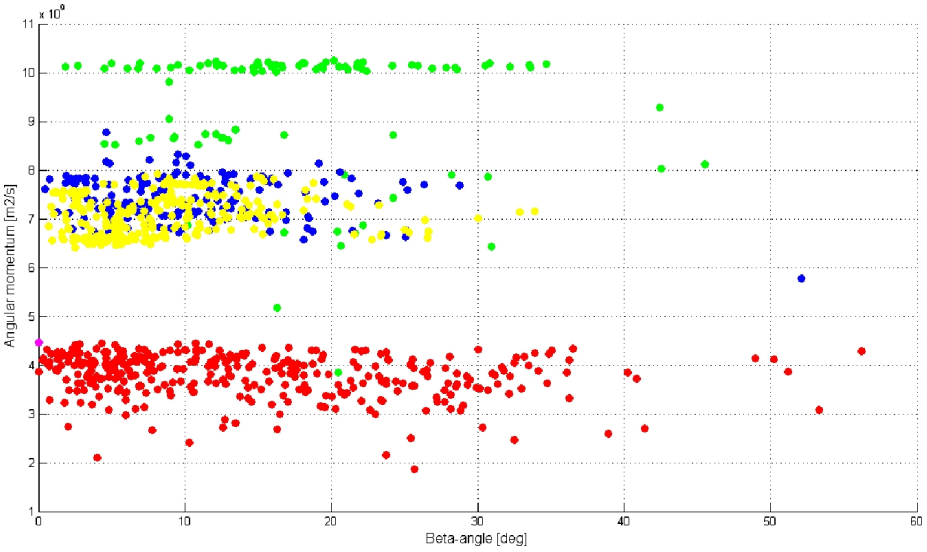

Figure 3.3: Specific angular momentum of the asteroids plotted against the β-angle [Evertsz, 2008].

The angular momentum as a function of the angle β for the whole asteroid set is shown in Figure 3.3. The colour of the data points correspond to each asteroid group: Group 1 is green, Group 2 is blue, Group 3 is yellow and Group 4 is red.

Based on the premise that out-of-plane maneuvers are very demanding in terms of propellant (cf. Equation (3.2.2)), only asteroids with a β-angle lower than 5° were considered feasible. This value reflects proximity in the orbital planes of the asteroids and that of the Earth: $\beta_{Earth} = 8.854 \times 10^{-4}$ rad. However, this pruning criterion yielded still a large number of asteroids and thus the decision was made to contemplate only 5 to 10 asteroids per group, preferably the ones with the lowest β-angle. This led to a final 28 feasible asteroids, listed in Table 3.1, alongside the threshold criterion for each asteroid group. Note that none of the asteroids from the GTOC2 winning sequence were selected through this pruning technique.

Once a reduced pool of asteroids was obtained, the optimization of the problem followed. The optimization phase was divided in two parts: an initial optimization assuming high-thrust propulsion, followed by a low-thrust optimization. The high-thrust approximation aimed at defining the best asteroid sequence while the low-thrust optimization aspired to fully optimize, within the GTOC2 parameters, the resulting sequence.

The high-thrust optimization was performed with Genetic ALgorithm Optimization of a MUltiple Swing-by Interplanetary Trajectory (GALOMUSIT), a software developed and maintained by Aerospace Engineering students at the TUDelft. The GTOC2 high-thrust model, as defined in the thesis, included some of the mission constraints from the original GTOC2 assignment. The optimizable launch maneuver from Earth discounted the allowed

[2]Here, *outward* designates the motion away from the center of mass: the Sun.

Table 3.1: The feasible asteroids. Adapted from [Evertsz, 2008].

$\beta \leq 5.0°$		$\beta \leq 1.5°$		
Group 1	2000659 (Nestor)	Group 3	2000020 (Massalia)	2000277 (Elvira)
	2002483 (Guinevere)		2000149 (Medusa)	2000644 (Cosima)
	2002674 (Pandarus)		2000158 (Koronis)	2001442 (Corvina)
	2005012 (Eurymedon)		2000180 (Garumna)	
	2002357 (Phereclos)		2000243 (Ida)	
$\beta \leq 2.0°$		$\beta \leq 1.0°$		
Group 2	2000024 (Themis)	Group 4	2101329 (SZ162)	2001873 (FH)
	2000147 (Protogeneia)		2011011 (UK11)	2021542 (QA22)
	2000569 (Misa)		3127401 (LT24)	3293923 (TH50)
	2001082 (Pirola)		3156302 (LN6)	2096237 (VL1)
	2002379 (Heiskanen)		3167367 (WT153)	

hyperbolic excess velocity:

$$\mathbf{\Delta V}_{launch} = \mathbf{V}_{\infty}(t_L) - \mathbf{V}_{\infty L_{max}} \tag{3.2.7}$$

where $|\mathbf{V}_{\infty L_{\max}}| = 3.5$ km/s and t_L is the Earth launch date. The rendezvous conditions are such that both the velocity and position of the spacecraft match that of asteroid. As a result the rendezvous maneuver is equal to the hyperbolic excess velocity needed to match these conditions:

$$\mathbf{\Delta V}_{rendezvous} = \mathbf{V}_{\infty}(t_{R_j}) = \mathbf{V}_s(t_{R_j}) - \mathbf{V}_{A_j}(t_{R_j}) \tag{3.2.8}$$

where the subscripts s and A_j indicate the spacecraft and the jth asteroid, respectively, and t_{R_j} the time of rendezvous at that asteroid. Similarly, the re-launch maneuver, i.e. the launch from an asteroid, is given by the corresponding hyperbolic excess velocity:

$$\mathbf{\Delta V}_{relaunch} = \mathbf{V}_{\infty}(t_{RL_j}) = \mathbf{V}_s(t_{RL_j}) - \mathbf{V}_{A_j}(t_{RL_j}) \tag{3.2.9}$$

with t_{RL_j} the time of re-launch from the jth asteroid. During the asteroid stayover, the constraint in position and velocity are maintained:

$$\mathbf{r}_s(t_{w_j}) = \mathbf{r}_{A_j}(t_{w_j}) \tag{3.2.10}$$

$$\mathbf{V}_s(t_{w_j}) = \mathbf{V}_{A_j}(t_{w_j}) \tag{3.2.11}$$

with $t_{wj} \in [0, t_{w_j,f}]$ and $t_{w_j,f} \geq 90$ days. Despite the objective function being different than that of GTOC2 (cf. Section 2.2.3), and the high-thrust model, GALOMUSIT was used to provide a first estimate on the optimal asteroid sequence out of the reduced asteroid pool from Table 3.1.

The low-thrust optimization was performed using another internal tool, OPTIDUS, using an augmented cost function. The thrust profile, which allowed for coast arcs, was approximated by 3^{rd} order Chebyshev polynomials (cf. [Secretin, 2011]). For a more in depth discussion of this approach, please refer to [Evertsz, 2008]. The results of this model are briefly detailed in Section 3.2.4.

3.2.3 Travelling Salesman Problem Model

Another approach [Gorter, 2010] identified three separate stages in solving the GTOC2 problem: the asteroid selection and sequencing, the phasing assessment of the candidate sequences, and the determination of the constraint-satisfying trajectory. The latter part was not considered due to time constraints.

The asteroid selection and sequencing was identified as being an Exact Generalized Travelling Salesman Problem (EGTSP), a variant of the well-known Travelling Salesman Problem (cf. [Secretin, 2011]) where the salesman is to visit each *city cluster* (or *state*) exactly once and that exactly one city from each cluster is visited. Replacing the salesman by a spacecraft, the cities by the asteroids and the states by the asteroid groups, the analogy between the EGTSP and the GTOC2 becomes evident. In order to solve the problem of the cyclicity of the solution[3], the cost of returning to Earth from any asteroid is set to zero. This effectively removes the final returning leg from the optimization process hence providing an optimal solution to the asteroid sequencing problem. The resulting cost matrix can be seen in Figure 3.4. The value of each non-zero non-infinity entry in the matrix is given by the cost function. Note that the ordering of groups was reversed with respect to that defined in [Petropoulos, 2006]: Group 1 contains the asteroids closest to Earth, i.e. Group 4 asteroids according to the problem definition of Section 2.2.2.

The cost matrix shown in Figure 3.4 is a 911×911 square matrix, leading to a difficult problem to solve due to its large size. The complexity of the problem is reduced by selecting the four best transfers from each block, i.e. asteroid group intersections, in the matrix. This pruning method, despite its simplicity, reduces the size of the problem to about 60 candidate asteroids, fruit of the selection criterion, of overlaps between asteroids selected in each block and of the unit-sized Group 0.

The distance matrix from one city to another is determined via three different cost functions, relying mostly on orbital parameters. The first cost function was referred to as the *energy* cost function and represented the sum of the energy required to transfer the spacecraft from the departure to the arrival plane, $\Delta\varepsilon_{plane}$ and the difference in orbital energy between the departure and arrival orbit, $\Delta\varepsilon_{orbit}$. Mathematically:

$$\Delta\varepsilon = |\Delta\varepsilon_{plane}| + |\Delta\varepsilon_{orbit}| \tag{3.2.12}$$

where

$$\Delta\varepsilon_{orbit} = -\frac{\mu}{2}\left(\frac{a_2 - a_1}{a_1 a_2}\right) \tag{3.2.13}$$

$$\Delta\varepsilon_{plane} = \begin{cases} \frac{1}{2}\left(V_{p,1} + \Delta V_i + \Delta V_\Omega\right)^2 - \frac{1}{2}(V_{p,1})^2 & \text{if } a_2 > a_1 \\ \frac{1}{2}(V_{p,1})^2 - \frac{1}{2}\left(V_{p,1} - \Delta V_i - \Delta V_\Omega\right)^2 & \text{if } a_2 < a_1 \end{cases} \tag{3.2.14}$$

The second cost function, named ΔV, is based on the maneuvers required to match the orbital elements, excluding the true anomaly, of the departure with the arrival orbit. It is a sum of various individual maneuvers: the change in inclination, ΔV_i, in right ascension, ΔV_Ω, in pericenter, ΔV_ω, in semi-major axis

[3]The optimal solution to the EGTSP requires the salesman (spacecraft) to return home (Earth).

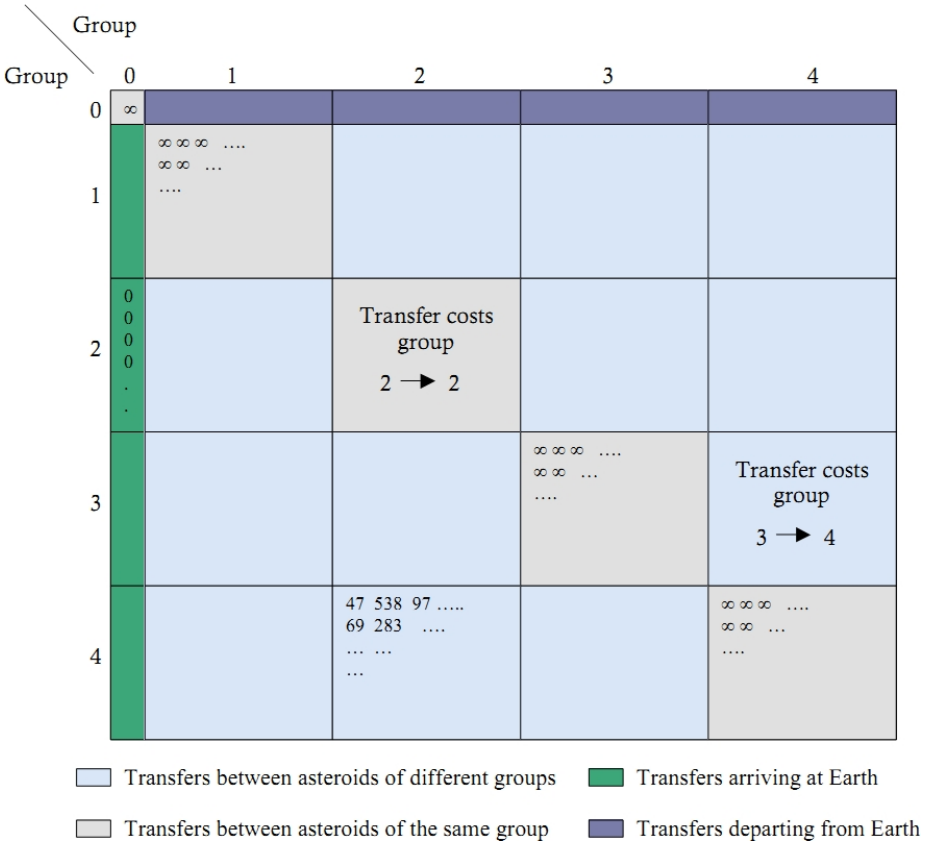

Figure 3.4: Cost matrix of the EGTSP [Gorter, 2010].

and eccentricity, ΔV_{ae}, and a possible correction maneuver, ΔV_{cor}, to adjust the apocenter of the departure orbit. This is mathematically expressed as:

$$\Delta V_{tot} = \Delta V_i + \Delta V_\Omega + \Delta V_\omega + \Delta V_{ae} + \Delta V_{cor} \tag{3.2.15}$$

where

$$\Delta V_\omega = 2\sqrt{\frac{\mu}{a(1-e^2)}e\sin\left(\frac{\alpha}{2}\right)} \tag{3.2.16}$$

$$\Delta V_{ae} = \sqrt{(V_2\cos\beta - V_{a,1})^2 + V_2^2\sin^2\beta} \tag{3.2.17}$$

$$\Delta V_{cor} = \begin{cases} \sqrt{2\left(\frac{\mu}{r_{p,1}} - \frac{\mu}{r_{p,1}+r_{p,2}}\right)} - \sqrt{2\left(\frac{\mu}{r_{p,1}} - \frac{\mu}{r_{p,1}+r_{a,1}}\right)} & \text{if } r_{a,1} < r_{p,2} \\ \sqrt{2\left(\frac{\mu}{r_{p,1}} - \frac{\mu}{r_{p,1}+r_{a,1}}\right)} - \sqrt{2\left(\frac{\mu}{r_{p,1}} - \frac{\mu}{r_{p,1}+r_{a,2}}\right)} & \text{if } r_{p,1} > r_{a,2} \end{cases} \tag{3.2.18}$$

Note that it is probably more efficient (propellant-wise) to design one single maneuver rather than a summation of individual maneuvers. Finally, the last cost function, is the one used by the ESA/ACT team in GTOC2 and is hence named after its creator: the ESA cost function. It consists of the sum of the ΔV needed at pericenter of the lower orbit to raise the apocenter and of the ΔV needed at apocenter to achieve a pericenter raise and an inclination change. It is defined as:

$$\Delta V_{tot} = \Delta V_1 + \Delta V_2 \tag{3.2.19}$$

where

$$\Delta V_1 = \sqrt{\mu}\left(\sqrt{\frac{2}{r_{p,1}} - \frac{2}{r_{p,1}+r_{a,1}}} - \sqrt{\frac{2}{r_{p,1}} - \frac{2}{r_{p,1}+r_{a,2}}}\right) \tag{3.2.20}$$

$$\Delta V_2 = \sqrt{V_i^2 + V_f^2 - 2V_iV_f\cos i_{rel}} \tag{3.2.21}$$

$$\tag{3.2.22}$$

and

$$V_i = \sqrt{\mu}\left(\sqrt{\frac{2}{r_{a,2}} - \frac{2}{r_{p,1}+r_{a,2}}}\right) \tag{3.2.23}$$

$$V_f = \sqrt{\mu}\sqrt{\frac{2}{r_{a,2}} - \frac{1}{a_2}} \tag{3.2.24}$$

$$\begin{aligned}\cos i_{rel} = {} & \cos i_1\cos i_2 + \sin i_1\sin i_2\cos\Omega_1\cos\Omega_2 \\ & + \sin i_1\sin i_2\sin\Omega_1\sin\Omega_2\end{aligned} \tag{3.2.25}$$

Of the three cost functions, this is the only symmetric one. A detailed description of the parameters in the equations above can be found in [Gorter, 2010].

The EGTSP was then solved using two distinct methods: a Branch-and-Bound (B&B) algorithm and the Nearest Neighbour Heuristic (NNH). In order to solve the EGTSP with the B&B, the cost matrix is reduced and the EGTSP transformed into a Travelling Salesman Problem (TSP). The TSP is then solved using a B&B method, namely a modified Hungarian Algorithm [Munkres, 1957].

The solution of the TSP is then converted back into an EGTSP solution. The other implemented approach was a multistart NNH. The difference between a traditional NNH (as described in Section 5.1) and the multistart variant is pretty straightforward: instead of randomly selecting the first node, all nodes are consecutively considered as the starting point. Given the nature of the multi-start NNH, the EGTSP does not need to be transformed into a TSP, nor does the cost matrix need to be reduced.

The phasing was considered posterior to the obtainment of the candidate asteroid sequences. The phasing assessment was carried out based on a continuous model using a shape-based method, namely *exponential sinusoids* (cf. [Secretin, 2011]). The optimal solutions within that model were obtained by consecutively applying a Monte-Carlo search, a Genetic Algorithm (GA) and an Interior Point Method. For a more detailed description of this approach, please refer to [Gorter, 2010]. The results with this method are compared, in Section 3.2.4, with those obtained with the models described in the previous sections.

3.2.4 Results

The solution found during the GTOC2 competition by the TU Delft team had a low objective function value (15 kg/yr) and large constraint violations in terms of position, velocity and time-of-flight.

The angular momentum-based pruning discarded all the asteroids used by GTOC2 participants from the resulting, reduced target pool. After the high- and low-thrust optimizations, the final solution shows large constraint violations both in terms of position and velocity. The final objective function value is however larger (50.77 kg/yr).

The cost matrix reduction for the EGTSP Model leads to a reduced set of targets, of which only up to 20% correspond to GTOC2 participant asteroids. The B&B performs better than the multi-start NNH on the reduced set but is unable to deal with the complete cost matrix. The continuous model yielded large objective function values (at times, above a 100 kg/yr) but coupled to large constraint violations in position and/or velocity.

The design of an accurate low-thrust model and of a powerful local optimizer, capable of handling the stringent GTOC2 constraints, is currently being undertaken at the A&S research group. With respect to the discrete aspect, there is hope that another approach will lead to a subset of asteroids more closely correlated to the GTOC2 solutions.

3.3 Lessons Learned

The topic of the author's MSc Thesis is part of the lineage of the aforementioned heritage, the A&S Department of Aerospace Engineering at the TU Delft wishing to add an efficient combinatorial tool for preliminary space mission design to its software toolbox. Here the possible contribution of each of the approaches discussed above to the author's own MSc Thesis project is discussed.

During the GTOC2 competition, the discrete aspect was generally solved under the high-thrust assumption, which, due to its simplicity, proves to be less computationally demanding. With respect to pruning procedures, the angular

momentum-based filter does not seem to be sufficiently reliable. Due to the similarities with an inclination-based filter, it is expected that the latter will not lead to a successful pruning. The pruning approaches based on energy, on the ΔV needed to match orbital elements and on a Hohmann-like bi-impulsive transfer lead to a reduced pool of asteroids but with little similarities with the GTOC2 solutions. The optimal phase-free bi-impulsive ΔV pruning metric was one of the most effective in screening asteroid combinations for the majority of the trajectory legs. However, using the high-thrust model to carry out a phasing analysis does not yield satisfactory results.

An incremental approach, i.e. solving the individual leg subproblems separately, appears to be appropriate for the pruning phase. This incremental approach should follow the Group 4 - Group 3 - Group 2 - Group 1 asteroid group sequence order, as it is the most likely to entail the problem optimum.

A very powerful B&B algorithm is needed to solve the discrete aspect of the GTOC2 assignment for the entire asteroid pool but may yield satisfying results on an efficiently-pruned set. Modelling the GTOC2 problem as an EGTSP results in numerous approximations which may jeopardize the quality of the solutions. With respect to the NNH, a multi-start implementation brings little to the quality of the solution.

These lessons learned will be a valuable asset in the design of an efficient combinatorial tool to solve the discrete aspect of the GTOC2 problem.

Part II

Combinatorial Tool

Chapter 4

GTOC2 Combinatorial Model

Identifying the most promising asteroid combinations for the GTOC2 can be seen as a combinatorial problem, most probably a so-called NP-hard problem[1] due to the dynamical aspect of the problem [Aardal, 2011]. Out of the classical combinatorial problems, the Dynamic Shortest Path Problem (DSPP) appears to be the most suited to model the GTOC2 problem [Secretin, 2011].

After a short introduction to the concepts and terms of graph theory in Section 4.1, we proceed to the description of the static Shortest Path Problem (SPP) in Section 4.2, along with how it can be used to model the GTOC2 problem. The dynamic variant of the SPP is explained in Section 4.3 and the rationale for prefering the static model over the DSPP is presented. Section 4.4 describes how the dynamic GTOC2 problem is turned into a static SPP by decoupling the continuous from the integer optimization.

4.1 Basic Graph Theory

Combinatorial optimization problems are often modelled according to *Graph Theory*. This introductory section is based on [Hartmann and Weigt, 2005].

An *undirected graph*, $G = (V, A)$, is given by a set of *vertices*, $u \in V$, and a set of undirected *arcs*, $(u, v) \in A$. A *weighted* graph includes a function, $w : A \to \mathbb{R}$ that gives the cost of travelling along any given arc. A graph is said to be a *directed graph*, $D = (V, A)$, when the arcs are ordered. In other words, in a directed graph, (u, v) and (v, u) represent different arcs. The *indegree* $\delta_{in}(u)$ is the number of ingoing arcs (v, u) while the *outdegree* $\delta_{out}(u)$ is the number of outgoing arcs (u, v). These concepts are illustrated in Figure 4.1.

Figure 4.1 shows a directed, weighted graph. The circles represent the vertices and the arrows the directed arcs of the graph. The weights of each arc are given next to the arrows: e.g. $w(a, e) = 2$. The "directed" nature of the graph can be seen from two different perspectives: notice, first, that $w(e, f) \neq w(f, e)$, and second, that while $(c, b) \in A$, $(b, c) \notin A$. This situation is sometimes described, in a weighted graph, as $w(b, c) = \infty$. For the sake of clarity, these

[1] NP-hard (Non-deterministic Polynomial-time hard) problems, in computational theory, are the set of yes-no problems that can be verified in polynomial time [Aardal, 2011].

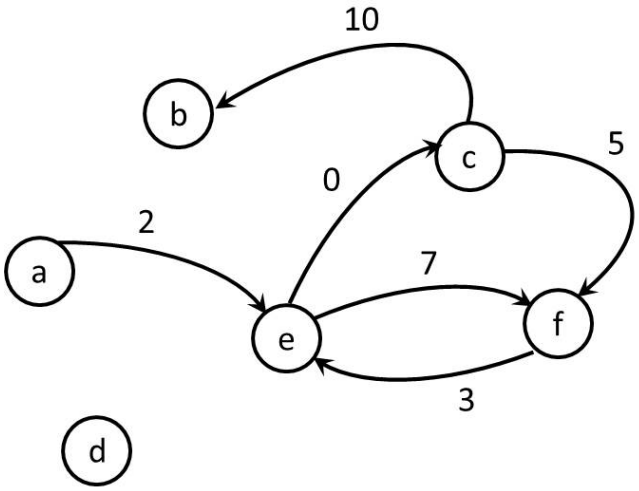

Figure 4.1: Example of a graph.

infinite arcs are left out in the graphical representation. The vertex f has indegree 2 and outdegree 1: $\delta_{in}(f) = 2$, $\delta_{out}(f) = 1$. The indegree and outdegree of vertex d are equal to zero, i.e. $\delta_{in}(d) = \delta_{out}(d) = 0$: the vertex d is said to be isolated.

4.2 The Shortest Path Problem

The SPP is one of the most fundamental network optimization problems and has been studied at length. See e.g. [Bellman, 1958], [Dial, 1969], [Schrijver, 2010]. It is the problem of finding a path, i.e. a sequence of nodes, between two vertices such that the sum of the costs of its inherent arcs is minimized. Mathematically, given the directed graph $D = (V, A)$ and the cost function w, the SPP relies on finding the path $P = (v_0, v_1, ..., v_{m-1}, v_m)$, with $v_i \in V$ and $(v_i, v_{i+1}) \in A$ for $i = 1, ..., m$, that minimizes

$$w(P) := \sum_{i=0}^{m} w(v_i, v_{i+1}) \tag{4.2.1}$$

SPPs can be classified in four major categories, based namely on the number and nature of the shortest paths sought:

Single-source The shortest paths from one source vertex, s, to all the other vertices in the graph are computed;

Single-destination The shortest paths from all nodes to a destination, or target, node, t, are computed;

All-pairs The shortest paths between all possible pairs, (u, v), in the graph are computed;

Point-to-Point Only the shortest path from a specific source vertex, s, to a specific target vertex, t, is computed.

Each category has a different complexity and therefore different approaches are available to solve them.

The problem of selecting the most promising asteroid combinations from the GTOC2 database can be seen as single-source, or *one-to-all*, dynamic shortest path problem. From such a perspective, the nodes represent the asteroids, the directed arcs model the possible connections between asteroids from different groups, while the dynamic arc costs represent the individual contributions to the cost function. This model is illustrated in Figure 4.2. Note that, in the SPP model of the GTOC2 problem in Figure 4.2, the arcs are directed which ensures that no backward path cut, i.e. from Group 3 back to Group 4, is considered.

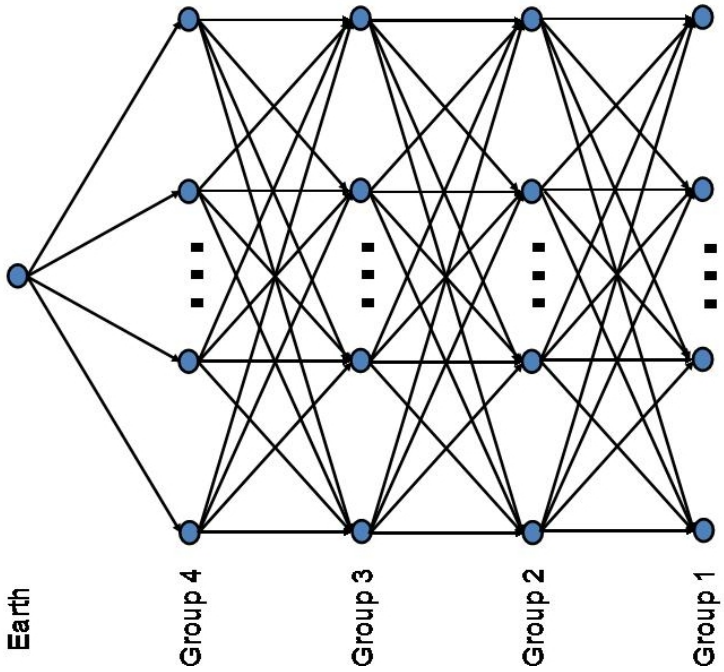

Figure 4.2: The GTOC2 asteroid selection problem modelled as an SPP.

According to [Alemany and Braun, 2007], the best asteroid group sequence is: Group 4 - Group 2/3 - Group 3/2 - Group 1. However, in an SPP implementation, one must adopt a clear order, i.e. opt for either the 4-3-2-1 or the 4-2-3-1 asteroid group sequence, as shown in Figures 4.3(a) and 4.3(b) respectively, to avoid possible shortcuts in the algorithm. More specifically, if Group 4 asteroids are connected to both Group 2 and Group 3 asteroids, as shown in Figure 4.3(c), the situation might arise where the algorithm short-circuits one of the asteroid groups and finds a (wrong) solution with only three arcs instead of four: e.g. Group 4 - Group 2 - Group 1 (red path in Figure 4.3). The implementation of a protection mechanism to prevent this type of short-circuit is not desirable in virtue of the expected added complexity, as the condition for

a total of four asteroids must be checked for each solution. Besides the raised number of computations due to the implementation of such a protection mechanism in the algorithm source code, a clear asteroid group sequence decreases the number of arcs in the graph and hence the computational difficulty of the problem. More specifically, setting a clear asteroid group sequence reduces the number of possible asteroid combinations from 41 billion to 1.7 billion, i.e. by one order of magnitude. We adopt the 4-3-2-1 sequence for it is the combination most commonly used by GTOC2 participants (see Chapter 2.3). Note that this is the sequence illustrated in Figure 4.2.

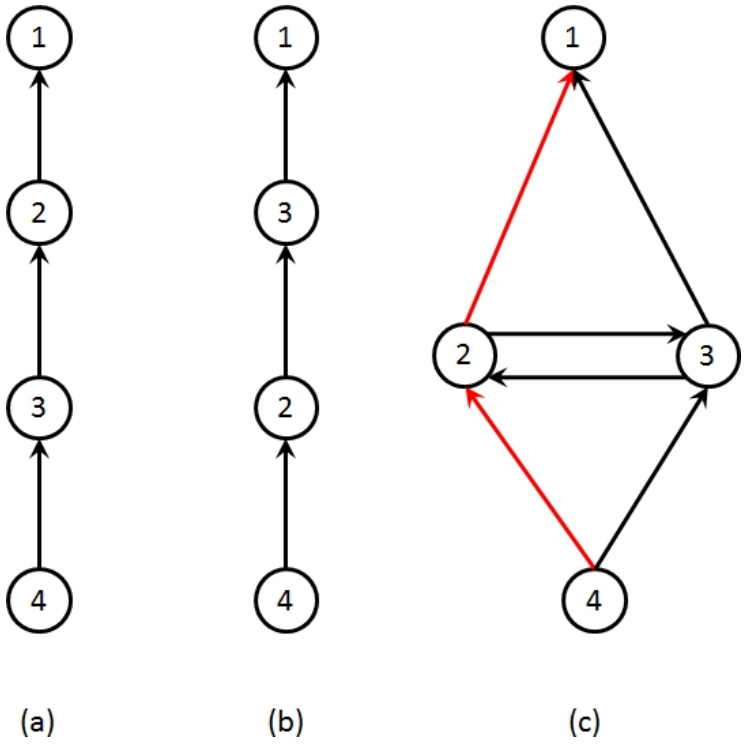

Figure 4.3: Asteroid group sequences: (a) 4-3-2-1, (b) 4-2-3-1, and (c) 4-2/3-3/2-1 with a possible shortcut highlighted in red.

4.3 The Dynamic Shortest Path Problem

A graph is said to be *dynamic* when some of the entities (vertices, arcs, weights) change with time. We can distinguish two types of DSPP. In the first variant, usually called *time-dependent*, the cost of an arc is given by a pre-determined function of time, i.e. the cost of an arc (u, v) depends on the starting time along the path and on the time already spent to reach u. The time-dependent DSPP can be mathematically described as finding the path P, that minimizes

$$w_d(P) := \sum_{i=0}^{m} w_d(v_i, v_{i+1}, t_i) \tag{4.3.1}$$

with $t_i \in T$ for $i = 1, ..., m$ and T a predefined time interval, and where $w_d(v_i, v_{i+1}, t_i)$ is the travelling cost along the arc (v_i, v_{i+1}) starting from v_i at time t_i.

The second variant is referred to as *the cost-update* variant. It concerns itself with graphs where the cost function changes or is updated at regular time intervals but the graph is static in-between updates. It can be seen as a discretized form of the time-dependent variant. Note however, that the first variant is more adequate to model the GTOC2 asteroid selection problem.

The DSPP, independently of the variant, is an NP-hard problem. In literature, one finds several references ([Ahuja et al., 2003], [Ziliaskopoulos and Wardell, 2000], [Chabini, 1998]) to the First-In-First-Out (FIFO) property, as an assumption to simplify time-dependent graph problems. The FIFO property states that for each pair of time instants τ, τ' with $\tau < \tau'$:

$$\forall (u, v) \in A, w_d(u, v, \tau) + \tau \leq w_d(u, v, \tau') + \tau' \tag{4.3.2}$$

Put into words, an object departing from u along the arc (u, v) at time τ' cannot arrive at v before an object travelling along the same arc but that departed at time $\tau < \tau'$. As a result, the FIFO property is sometimes referred to as the *non-overtaking property*. It reduces the complexity of the DSPP as it becomes polynomially solvable while the SPP in a non-FIFO time-dependent graph is NP-hard [Nannicini and Liberti, 2008].

The GTOC2 problem is hardly a FIFO DSPP, as the time-of-flight is a variable of the problem and not the result of a cost function. This means that we would have to model the GTOC2 discrete problem as a non-FIFO DSPP which has severe consequences in terms of computational effort. Therefore, we make the choice to adopt a static SPP model.

4.4 Dynamic to Static Cost Function

As stated in Section 4.2, in the SPP model of the GTOC2 problem, the nodes represent the celestial bodies and the directed arcs model the possible connections between asteroids. We now define the cost function, w, that gives the cost of travelling from a given body to another. Since the dynamic model is discarded, the dynamic arc costs must be replaced by static ones. The static arc costs, $w(u, v)$, are defined as:

$$w(v_i, v_{i+1}) \equiv w'_t(v_i, \bar{t}_i, v_{i+1}, \bar{t}_{i+1}) \tag{4.4.1}$$

where $w_t(v_i, t_i, v_{i+1}, t_{i+1})$ is the cost of travelling along (v_i, v_{i+1}) starting at time t_i and arriving at time t_{i+1}, and $\bar{t}_i$ and $\bar{t}_{i+1}$ are the departure and arrival times, respectively, that minimize w_t. Phrased differently, the dynamic arc costs are replaced by their optimal value, obtained in an a-priori optimization step. The optimization techniques used are described, in Chapter 6.

To reduce the computational effort of this optimization procedure, and bearing in mind that only a preliminary analysis of the GTOC2 problem is sought, we define the dynamic cost function, w_t, under the assumption of a high-thrust propulsion system. It has been shown that there is correlation between increasing high-thrust ΔV and decreasing low-thrust final mass for the majority of the GTOC2 trajectory legs (see Section 3.1.2). Moreover, a large number of

GTOC2 participants used a high-thrust first estimate when solving the problem (see Section 2.3). Therefore, it is expected that this approximation will provide a fast way to determine the most reachable asteroids, as well as provide a sound basis for comparison with GTOC2 results.

It is proposed to solve the SPP model with two different cost functions. The first one, $w'_{t,1}$, is the high-thrust equivalent of the original GTOC2 cost function (see Equation (2.2.1)), adapted to a single, high-thrust leg:

$$w'_{t,1}(v_i, t_i, v_{i+1}, t_{i+1}) = \frac{m_{i+1}(v_i, t_i, v_{i+1}, t_{i+1})}{t_{i+1} - t_i} \tag{4.4.2}$$

where m_{i+1} is the mass at v_{i+1}, obtained with Tsiolkowski's law:

$$m_{i+1}(v_i, t_i, v_{i+1}, t_{i+1}) = m_i \exp(-\frac{\Delta V(v_i, t_i, v_{i+1}, t_{i+1})}{c}) \tag{4.4.3}$$

with m_i the mass at v_i, c the exhaust velocity of the propulsion system and ΔV the total magnitude of the impulsive shots needed to travel from v_i at t_i to v_{i+1} at t_{i+1}. We take the trajectory between v_i and v_{i+1} to be a Lambert arc (see Chapter 7), as it was shown (see Section 3.1.2) that the optimal phase-free bi-impulsive pruning metric is one among the most efficient.

Equation (4.4.2) explicitly takes into account the time-of-flight. However, doing so is expected to worsen the quality of the results obtained, due to the large differences in terms of time-of-flight between optimal high- and low-thrust trajectories. Therefore a separate cost function, $w'_{t,2}$, is defined, that removes the travel time from the equation:

$$w'_{t,2}(v_i, t_i, v_{i+1}, t_{i+1}) = m_{i+1}(v_i, t_i, v_{i+1}, t_{i+1}) \tag{4.4.4}$$

with m_{i+1} as defined in Equation (4.4.3). Note that the optimal solutions correspond to those that maximize the cost functions $w'_{t,1}$ and $w'_{t,2}$. Since the SPP is a minimization problem, we solve this discrepancy by taking the inverse of Equations (4.4.2) and (4.4.4):

$$w_{t,1} = 1/w'_{t,1} \tag{4.4.5}$$
$$w_{t,2} = 1/w'_{t,2} \tag{4.4.6}$$

We have mentioned earlier that the minimum phase-free high-thrust bi-impulsive ΔV metric is one of the most effective pruning methods. Given the large differences in time-of-flight between high- and low-thrust trajectories, adding the time-of-flight to the cost function will most probably lead to a degradation in the correlation between the sequences obtained and those submitted by GTOC2 participants. This was confirmed when observing that carrying a high-thrust phasing analysis did not yield satisfying results. Given the belief that using Equation (4.4.4), which does not consider the time-of-flight, will provide better results, this variant of the SPP model will take precedence over the one with Equation (4.4.2).

Chapter 5

Greedy Algorithms

When considering an NP-hard combinatorial optimization problem, there are roughly three approaches: one chooses a (potentially) computationally expensive enumerative method that is guaranteed to find the optimal solution (1 in Figure 5.1), one leans towards an approximation algorithm that runs in polynomial time (2 in Figure 5.1), or one opts for some type of heuristic technique with no definitive guarantee on the quality of the outcome but with lesser computational cost (3 in Figure 5.1) [Aarts and Lenstra, 1997]. These three options are shown in Figure 5.1, where the general relation between computational cost and result fidelity is illustrated.

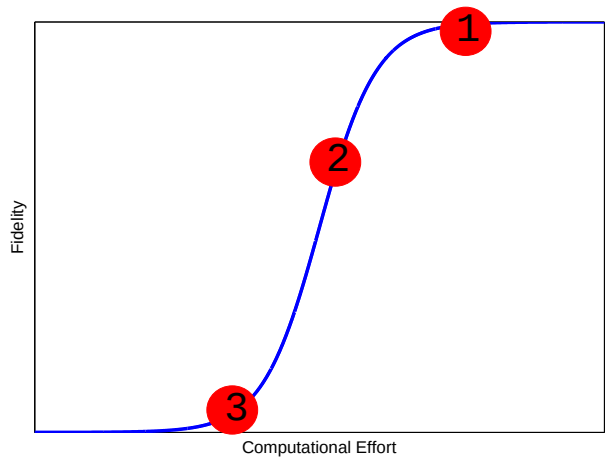

Figure 5.1: Computational cost and fidelity of different approaches to solve a combinatorial problem.

There are a number of methods available to solve the SPP, with different characteristics in terms of computational cost and solution quality, as seen in Figure 5.1. The algorithms that provide the best guarantee on the quality of the solution (see e.g. [Dijkstra, 1959]) tend to consider the search space in its

entirety. Applying such algorithms to the selected combinatorial model would defeat the purpose of designing a computationally efficient tool. At the other end of the spectrum of available techniques, greedy algorithms promise a fast generation of solutions and allow for a leg-per-leg reduction of the search space. The choice is therefore made to improve the greedy algorithms such that they provide a more accurate solution while preserving their attractive computational speed.

Section 5.1 lays down the bases of the NNH, while Section 5.2 extensively explores two different avenues for improving the accuracy of the results provided by the NNH: the multi-path (Section 5.2.1) and the bi-directional (Section 5.2.2) variants.

5.1 The Nearest Neighbour Heuristic

The term "greedy algorithms" refers to the set of optimization algorithms that make the choice that looks best at that instant. Of these, the NNH is probably the most well-known. This algorithm mimics a traveller whose rule-of-thumb is to visit the next closest unvisited city (vertex). In a graph $G = (V, A)$ as defined in Chapter 4.1, starting at the source vertex, v_0, the next vertex, v_1 is the vertex $v_i \in V$ that minimizes $w(v_0, v_i)$ with $(v_0, v_i) \in A$. This approach is then repeated until an end vertex is reached. This algorithm is relatively fast but there is no guarantee that the solution is indeed the optimal one [Aarts and Lenstra, 1997]. This is mostly due to the fact that the algorithm looks at the immediate neighbouring solution instead of trying to handle the problem as a whole. Figure 5.2, where a simple SPP is given, along with the optimal solution and the sub-optimal solution found by the NNH, illustrates this situation.

5.2 Improving the Nearest Neighbour Heuristic

As mentioned earlier, the choice is made to improve the accuracy of greedy algorithms. In the following sections, the most promising leads (see [Secretin, 2011]) to do so are detailed.

5.2.1 Multi-path Nearest Neighbour Heuristic

A possible solution to improve the accuracy of NNH is to allow it to pursue not only one thread (or path) but N threads. This increases the probability of finding the optimal solution, as shown in Figure 5.3, where the same problem as that from Figure 5.2 is solved but with an NNH pursuing two threads (NNH_1 and NNH_2) instead of just one. Moreover, bear in mind that the goal of the tool is not to provide the optimal solution to the GTOC2 asteroid sequence but rather a subset of possible combinations that are likely to contain said optimal solution. Therefore, allowing the NNH to pursue more than one thread is in line with the proposed goal. In Figure 5.3, note that despite the increased quality of the solutions found, none of them corresponds to the optimum.

At each stage of the optimization, i.e. after each leg is optimized, the cost of the (partial) paths, $P_{\text{partial}} = (v_0, v_1, ..., v_j)$ with $v_i \in V$ and $(v_i, v_{i+1}) \in A$ for $i = 0, ..., j-1;\ j \leq 4$, are computed according to Equation (4.2.1). The partial

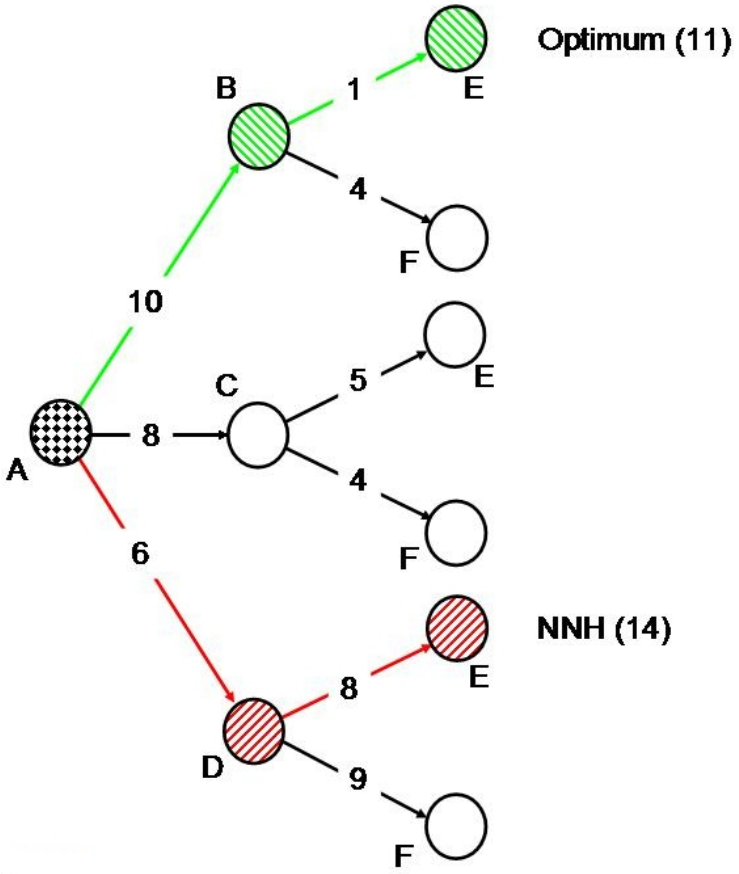

Figure 5.2: Sub-optimal solution to an SPP as found by a greedy algorithm.

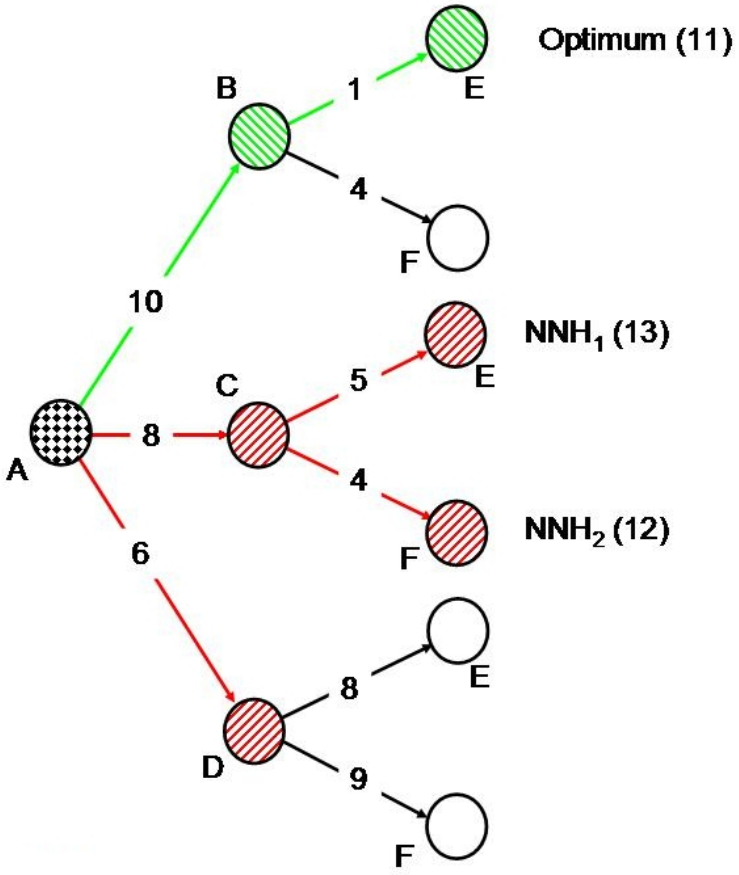

Figure 5.3: Sub-optimal solutions to an SPP as found by a multi-path greedy algorithm with N = 2.

paths are then ranked by increasing cost (the SPP is a minimization problem) and the N best paths (or champion paths) are selected to be pursued in the next stage. In Figure 5.3, all arcs neighbouring the start vertex (with the chess pattern) are evaluated and the two best, i.e. the two with the lowest partial path cost, indicated in red, are selected. The remaining arc (and corresponding vertex) is discarded. In the second stage, the process is repeated, starting at the vertices that were preserved.

Since the individual contributions to the path cost are computed separately, the phasing of the resulting trajectories is not explicitely taken into account. Given the generally large discrepancies in terms of time-of-flight between high-thrust and low-thrust trajectories, using a high-thrust model to carry out a phasing analysis of GTOC2 trajectories does not yield satisfactory results (see Section 3.1.2).

The advantage of solving the GTOC2 discrete aspect with a multi-path NNH is that this technique approaches the problem on a leg-per-leg basis which allows to discard, after each leg is selected, large areas of the search space, as shown in Figure 5.4. This incremental approach, which reduces the complexity of the problem, has proven to be an appropriate method to solve the GTOC2 asteroid sequence selection problem (see Section 3.1.2). In Figure 5.4, where only a subset of the total GTOC2 graph is shown for sake of readability, we can see that after analysing the arc weights for the first leg, a number of asteroids are cast aside, effectively reducing the number of possible combinations. As a result, there is no need to compute the cost of the outgoing arcs from the discarded asteroids, leading to significant computational time savings. Even more so given that the arc costs are the result of an optimization procedure.

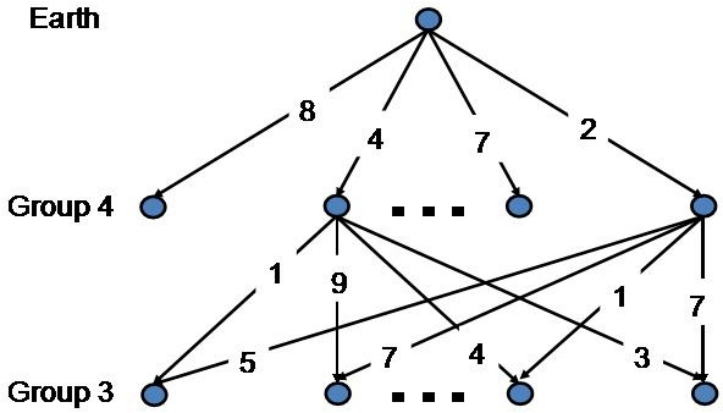

Figure 5.4: The multi-path NNH applied to the first two legs of the GTOC2 problem. Arc costs do not represent realistic values.

Decision Metrics

Implementing a multi-path NNH introduces the problem of the definition of the number of champions to be selected after each stage computation. We consider a number of different decision-making methods with respect to the number of

paths to be pursued.

One possible approach relies on a user-defined absolute number of paths. This is the most direct approach as the user directly sets the number of champions. In order to obtain a first estimate of the impact of the number of champions on the computational effort required to solve the GTOC2 problem with the multi-path NNH, Figure 5.5 plots the total number of transfers computed as a function of the number of champions, as well as the individual contributions from each leg.

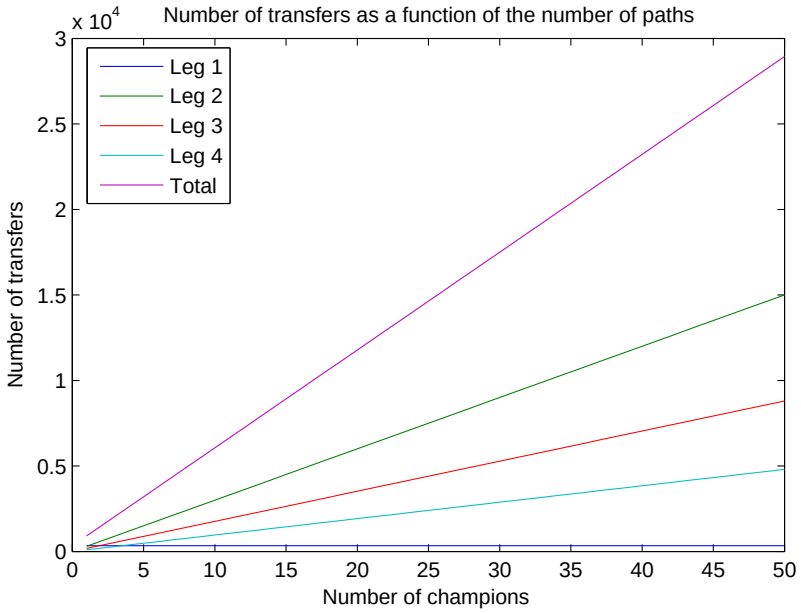

Figure 5.5: Total number of transfers evaluated as a function of the number of champions.

In Figure 5.5, it is evident that the total number of transfers grows linearly with the number of champion paths, assuming the latter is constant for each leg. The number of Leg 1 transfers computed is constant and therefore independent of the number of champions because the problem is single-sourced: the Earth is the only point of origin. The largest contribution to the total number of transfers computed comes from Leg 2: Group 3 is the largest asteroid group affected by the number of champions. All these observations are coherent with the formula that governs the total number of transfers, $n_{\text{transfers}}$, evaluated:

$$n_{\text{transfers}} = m_4 + N_1 \cdot m_3 + N_2 \cdot m_2 + N_3 \cdot m_1 \tag{5.2.1}$$

where m_4, m_3, m_2, and m_1 correspond to the number of asteroids in the asteroid group indicated by the index, and N the number of champions from the leg indicated by the index. Assuming that $N_1 = N_2 = N_3 = N$, Equation (5.2.1)

becomes:

$$\begin{aligned} n_{\text{transfers}} &= m_4 + N(m_3 + m_2 + m_1) \\ &= 338 + N(300 + 176 + 96) \\ &= 338 + 572N \end{aligned} \tag{5.2.2}$$

which gives a linear relation between $n_{\text{transfers}}$ and N. One can therefore expect that the computational time needed to execute the multi-path NNH will increase linearly with a user-defined, constant absolute number of champions.

A user-defined relative number of paths, expressed as a fraction, is also possible. The idea here is to have a number of champion paths that reflects the effort put into obtaining them: instead of selecting a fixed number of champions independently of the total amount transfers computed at each leg, the number of champions grows with the number of transfers computed. Using a relative number of champions, the number of champions after each leg computation is given by:

$$N_i = n_{\text{transfers,i}} \cdot x_i \ : \ i = 1, 2, 3 \tag{5.2.3}$$

where x_i is the relative number of champion paths from Leg i and $n_{\text{transfers,i}}$ the number of Leg i transfers computed:

$$n_{\text{transfers,i}} = \begin{cases} m_4 & : \ i = 1 \\ N_{i-1} \cdot m_{5-i} & : \ i = 2, 3, 4 \end{cases} \tag{5.2.4}$$

Assuming the same relative number of champions for every leg, x, and substituting Equations (5.2.3) and (5.2.4) into Equation (5.2.1) yields:

$$\begin{aligned} n_{\text{transfers}} &= m_4 \\ &\quad + m_4 * x * m_3 \\ &\quad + m_4 * x * m_3 * x * m_2 \\ &\quad + m_4 * x * m_3 * x * m_2 * x * m_1 \\ &= 338 + 1.01 \cdot 10^5 x + 1.78 \cdot 10^7 x^2 + 1.71 \cdot 10^9 x^3 \end{aligned} \tag{5.2.5}$$

which is a polynomial expression. Since all coefficients are positive, it is expected that the computational time needed to complete the multi-path NNH will increase exponentially with a constant relative number of champions, defined by the user. Each term in Equation (5.2.5) corresponds to the number of transfers evaluated at each leg: the number of Leg 1 transfers computed is constant, that of Leg 2 transfers increases linearly, that of Leg 3 transfers grows parabolically, and, finally, that of Leg 4 transfers increases hyperbolically. Mathematically:

$$n_{transfers,1} = 338 \tag{5.2.6}$$

$$n_{transfers,2} = 1.01 \cdot 10^5 x \tag{5.2.7}$$

$$n_{transfers,3} = 1.78 \cdot 10^7 x^2 \tag{5.2.8}$$

$$n_{transfers,4} = 1.71 \cdot 10^9 x^3 \tag{5.2.9}$$

Since $x \leq 1$, one could expect that Leg 2 transfers would contribute to the total number of transfers more than Leg 3 transfers, which in turn would be in larger numbers than Leg 4 transfers. However, due to the large coefficient of

x^3, the number of Leg 4 computations will rapidly dominate the total number of transfers evaluated. To visualize this, take a look at Table 5.1 where the absolute number of transfers evaluated at each leg and in total is given. For each leg, the percentage of the total number of transfers evaluated is also given.

Table 5.1: Contribution of each leg to the total number of transfers evaluated, as a function of the relative number of champion paths.

Relartive Number of Paths [%]	$n_{\text{transfers},1}$		$n_{\text{transfers},2}$		$n_{\text{transfers},3}$		$n_{\text{transfers},4}$		$n_{\text{transfers}}$
	[-]	[%]	[-]	[%]	[-]	[%]	[-]	[%]	[-]
1	338	7.76	900	20.65	1584	36.35	1536	35.25	4358
2	338	1.41	2100	8.74	7392	30.75	14208	59.11	24038
3	338	0.52	3000	4.63	15840	24.45	45600	70.39	64778
4	338	0.23	4200	2.84	29568	20.02	113568	76.90	147674
5	338	0.13	5100	1.92	44880	16.89	215424	81.07	265742

As shown in Table 5.1, the Leg 4 evaluations rapidly dominate the contributions to the total number of transfers computed: for a relative number of champions of 2%, these already represent more than half the total number of evaluations. The share of Leg 4 computations continues to increase as the relative number of champions increases. This phenomenon, as well as the large dominance of Leg 4 evaluations, is illustrated in Figure 5.6.

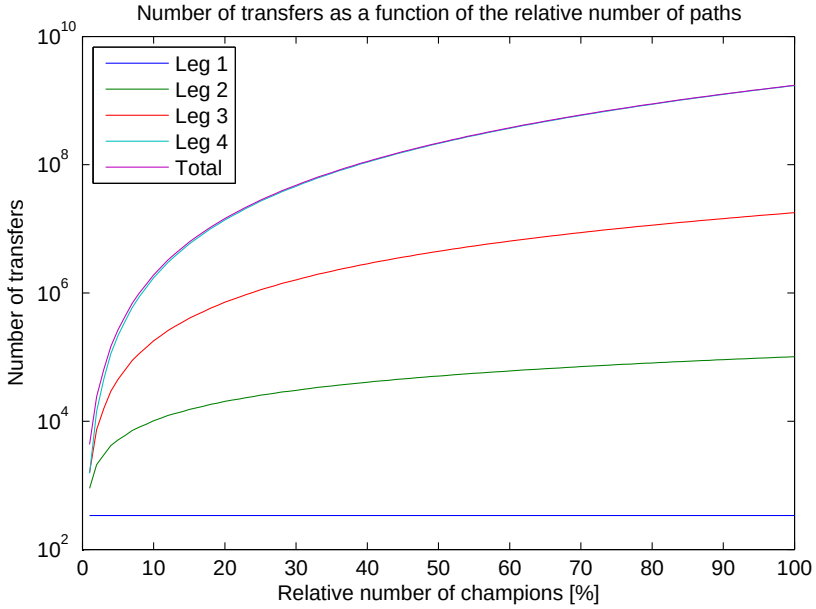

Figure 5.6: Total number of transfers evaluated as a function of the relative number of champions.

If we focus on smaller values for the relative number of champions, as depicted in Figure 5.7, a different pattern emerges. The discontinuities in the

curves are due to the `round` function[1], coupled to the requirement that there be at least one champion in each leg. In this fashion, a complete solution is returned, independently of the value of the relative number of champions. Nonetheless, despite being a multi-path NNH, the algorithm will find a single solution to the problem for relative numbers of champions less than approximately 0.45%. The minimum number of transfers computed, which corresponds to $x < 0.45\%$, is less than a thousand, more specifically 910. The relative contribution of each leg to the total number of transfers evaluated varies within the interval considered.

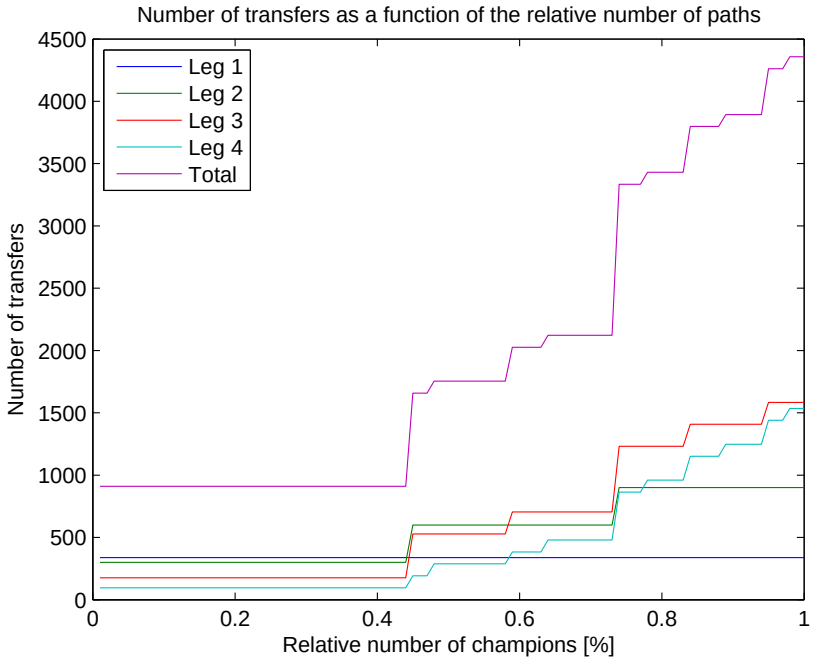

Figure 5.7: Total number of transfers evaluated as a function of the relative number of champions, for small values.

Using a constant, relative number of champions instead of an absolute number seems to lead to a less sensitive algorithm due to the real-to-integer conversions neede: indeed, two different real valued relative numbers of champions may lead to the same number of transfers being evaluated and therefore to the same solution. While this property is interesting for the user, the absolute number of champions decision metric will be privileged to conduct the analysis of the combinatorial tool. This decision is consolidated by the fact that all decision metrics eventually boil down to an integer number of champions.

Other decision metrics could be implemented, such as a user-defined threshold value for the path costs at each stage. Or even a probabilistic approach to the definition of said threshold values, by fitting a distribution to the (partial) path costs, as suggested by [Alemany and Braun, 2007]. This is expected to be

[1]This function, $\mathbb{R} \to \mathbb{Z}$, returns the closest integer value to the input value.

a more objective method to determine threshold values than simple visual inspection and perceived intuition. Note however that this approach still requires a certain degree of subjectivity. However, the absolute number of champions metric will be investigated in priority in the scope of this MSc Thesis. The other metrics might be also studied, depending on the time constraints of project.

Verification

In order to test the implemented multi-path NNH algorithm, it is applied to a simple test case: the SPP depicted in Figure 5.3, to which we shall refer as Shortest Path Problem Test Case 1 (SPPTC1). The starting vertex, A, is connected to three different nodes, B, C, and D, which in turn are each connected to the two end vertices, E and F. The graph is directed, as implied by the arrows representing the arcs. The arc costs, which are indicated on top of the arrows, are listed in Table 5.2.

Table 5.2: Arc costs in SPPTC1.

	v_{i+1}					
v_i	A	B	C	D	E	F
A	∞	10	8	6	∞	∞
B	∞	∞	∞	∞	1	4
C	∞	∞	∞	∞	5	4
D	∞	∞	∞	∞	8	9
E	∞	∞	∞	∞	∞	∞
F	∞	∞	∞	∞	∞	∞

The properties of the SPPTC1 can be reconstructed from Table 5.2. The graph is directed since it is not symmetric. E and F are end vertices given that no arc departs from them (arc costs are set to ∞). By contrast, A is the starting vertex but is only (forward) connected to B, C, and D, which in turn are only (forward) connected to E and F.

The multi-path NNH, with a varying number of champions, is applied to the SPPTC1. The results are given in Table 5.3.

Table 5.3: Solutions to SPPTC1 found by the multi-path NNH.

N	Sequence	Path Cost [-]
1	ADE	14
2	ACF	12
	ACE	13
3	ABE	11
	ACF	12
	ACE	13

Looking at Table 5.3, we can see that the quality of the solutions found improves when the number of champions increases. For a single champion,

$N = 1$, the multi-path NNH acts as a traditional NNH. The solution found, $w(A, D, E) = 14$, corresponds to the sequence illustrated (in red) in Figure 5.2. With $N = 2$, the two solutions found, $w(A, C, F) = 12$ and $w(A, C, E) = 13$, have an increased quality. They correspond to the paths shown in red in Figure 5.3. Finally, with three champions, the multi-path NNH finds the problem optimum, $w(A, B, E) = 11$, which is illustrated in green in both Figures 5.2 and 5.3. The other two solutions return correspond to the solutions found with $N = 2$. Note however that for SPPTC1, setting the number of champions to three is equivalent to probing the entire search space. It is therefore not surprising that the optimum was found.

Using SPPTC1, it was shown that the implemented multi-path NNH is functional and outperforms the original NNH in terms of solution fidelity and quality.

5.2.2 Bi-directional Nearest Neighbour Heuristic

Another improvement to the NNH comes from the concept of bi-directional search, a technique that has been succesfully applied to Dijkstra's algorithm in point-to-point SPPs (see e.g. [Nannicini and Liberti, 2008]). In a bi-directional search, the construction of the optimal path starts simultaneously at the source vertex, s, and at the end vertex, t, in the reverse graph, $\bar{D} = (V, \bar{A})$ where $(v, u) \in \bar{A} \Leftrightarrow (u, v) \in A$. For the searches starting at end vertices (backward searches), the arc weights, $\bar{w}$, are taken to be the opposite of those of the forward search:

$$\bar{w}(v_{i+1}, v_i) = w(v_i, v_{i+1}) \tag{5.2.10}$$

where w is as defined in Equation (4.4.1).

If we think of the bi-directional search as exploring nodes in circles centred at the source vertex s with increasing radius until t is reached, the bi-directional variant is intuitively faster as it explores vertices in two circles centred at both s and t until the two circles meet. This is graphically illustrated in Figure 5.8. We can see that the sum of the areas of the two bi-directional search circles is smaller than that of the unidirectional search circle, up to a factor of two for Dijkstra's algorithm [Nannicini and Liberti, 2008].

Bi-directional search can be adapted to single-source SPPs, with the expected caveat that the computational effort grows with the number of end vertices. Moreover, it should be noted that bi-directional search leads to time savings with respect to a traditional, forward Dijkstra search, which is a combinatorial optimization technique with high fidelity. For a heuristic search, namely NNH, a bi-directional search is expected to have benefits in terms of solution quality rather than in terms of computational effort, especially for single-source SPPs. In Figure 5.9, such a problem is illustrated. The problem, to which we shall refer from this point forward as Shortest Path Problem Test Case 2 (SPPTC2), has a one source vertex, A, and two end vertices, H and I. The optimal solution ($w(A, B, E, H) = 14$) is highlighted in green. Different variants of bi-directional NNH are investigated hereafter, based on SPPTC2.

Free Bi-directional Search

The term *free* was coined by the author based on the property that the forward and backward searches, while running simultaneously, are independent. Figure 5.10 shows a free, bi-directional NNH applied to SPPTC2. Greedy searches

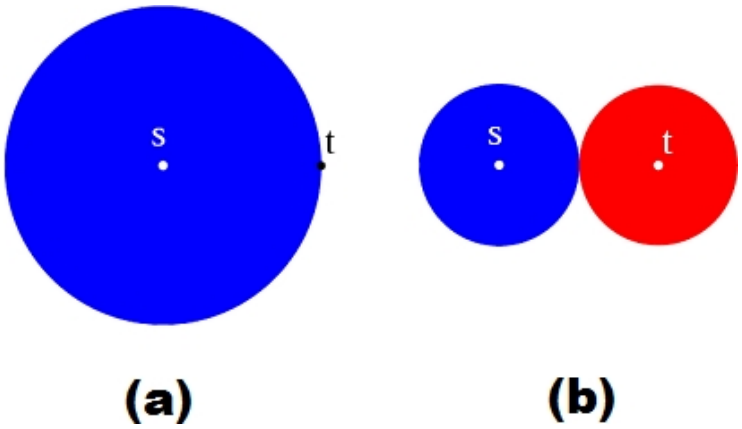

Figure 5.8: Schematic representation of a (a) unidirectional and a (b) bidirectional search. Adapted from [Nannicini and Liberti, 2008].

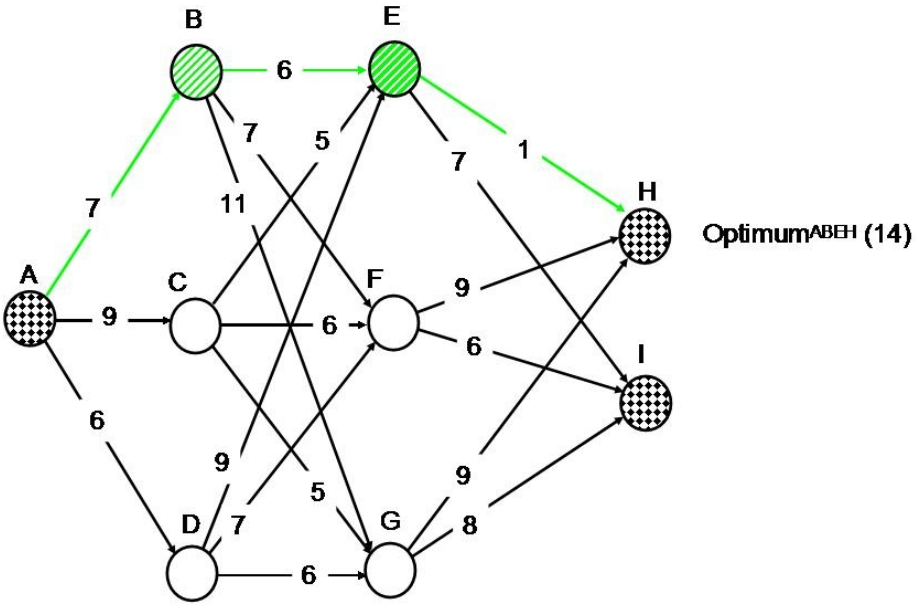

Figure 5.9: Optimal solution to SPPTC2.

are launched from both start and end vertices. This results in three different solutions ($w(A, D, G, I) = 20$, $w(A, C, F, I) = 21$, and $w(A, C, E, H) = 15$), highlighted in red, none of which correspond to the optimal solution of SPPTC2 ($w(A, B, E, H) = 14$). While one of the backward solutions, namely $w(A, C, E, H) = 15$, has a lower path cost than the one found by the forward search ($w(A, D, G, I) = 20$), the computational effort needed to handle the problem more than doubled (see Table 5.4), as three independent searches were run. Even if the arcs only need to be evaluated once, all the arcs of the problem were evaluated. Note at this point, each of the three greedy searches only pursues one thread. Table 5.4 indicates the number of SPPTC2 arcs evaluated by the forward, uni-directional NNH, by the free, bi-directional NNH, and by other variants of the NNH which will be described hereafter.

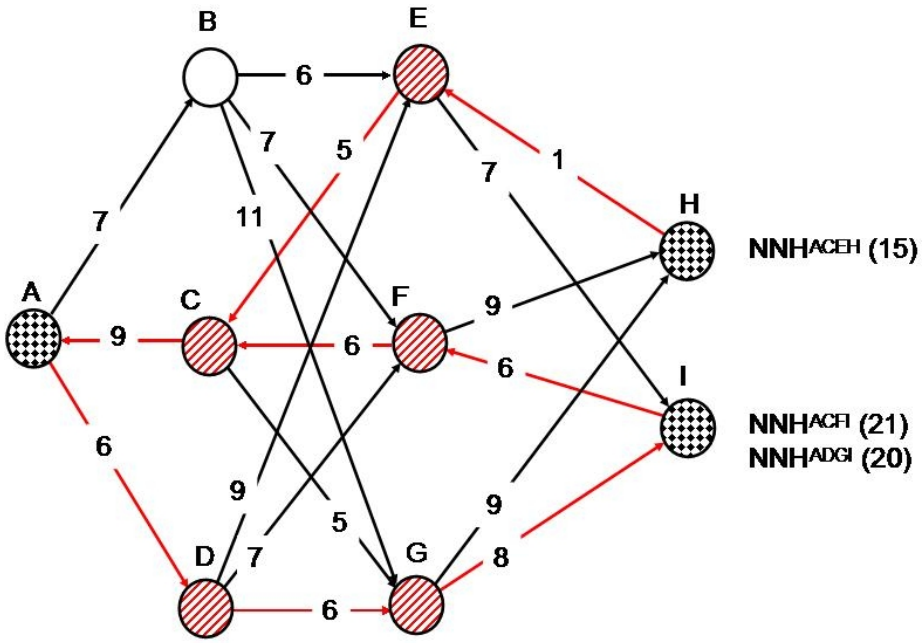

Figure 5.10: Solutions to SPPTC2 as found by a free, bi-directional NNH.

Table 5.4: Number of SPPTC2 arcs evaluated by the different variants of the NNH.

NNH Variant	# Arcs Evaluated
Uni-directional	8
Free bi-directional	18
Driven bi-directiona	11
Aggregated bi-directional	10
Multi-path bi-directional	13

Driven Bi-directional Search

In order to benefit from the computational time savings from a bi-directional search, it is possible to force the forward and backward searches to encounter each other. Otherwise, the greedy searches may never share arcs (as is the case in Figure 5.10) and the algorithm running time increases considerably, for all searches run from start to end (or end to start). We shall refer to this variant as a driven, bi-directional NNH. In Figure 5.11, the searches are forced to meet in the second arc (or leg): the only (second) arcs evaluated are combinations of D, E, and F. These nodes are the result of the backward and forward greedy searches up until the second arc. Arcs that are not evaluated are not represented in Figure 5.11.

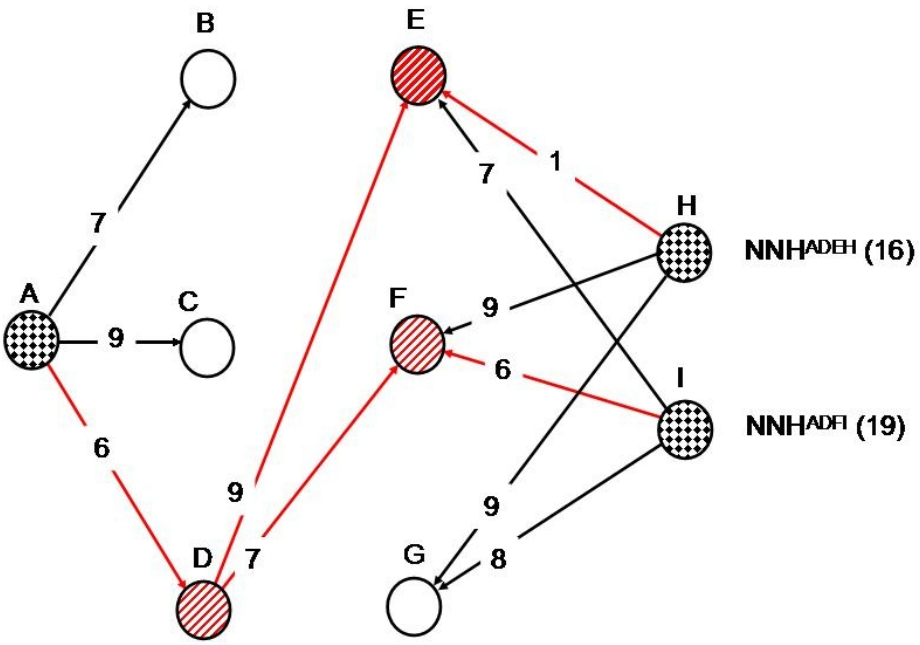

Figure 5.11: Solutions to SPPTC2 as found by a driven, bi-directional NNH.

Akin the free variant, the driven, bi-directional NNH fails to provide the optimal path of SPPTC2. Moreover, there is a degradation of the best solution found ($w(A, D, E, H) = 16$) with respect to the free bi-directional search ($w(A, C, E, H) = 15$). On the other hand, the number of arcs evaluated was significantly reduced, as shown in Table 5.4.

Aggregated Bi-directional Search

Further time savings can be accomplished by considering all backward searches as a single backward search: starting at all end vertices, all neighbouring arcs are evaluated and only the best arc, across all evaluated arcs, is pursued. Hence, the number of paths being pursued in the bi-directional search is limited, independently on the number of end vertices. Note however that, the algorithm

still requires that all end-neighbouring arcs be computed. This situation is illustrated in Figure 5.12, where the forward and backward searches are patched in the second leg. As in Figure 5.11, only evaluated arcs are represented in Figure 5.12.

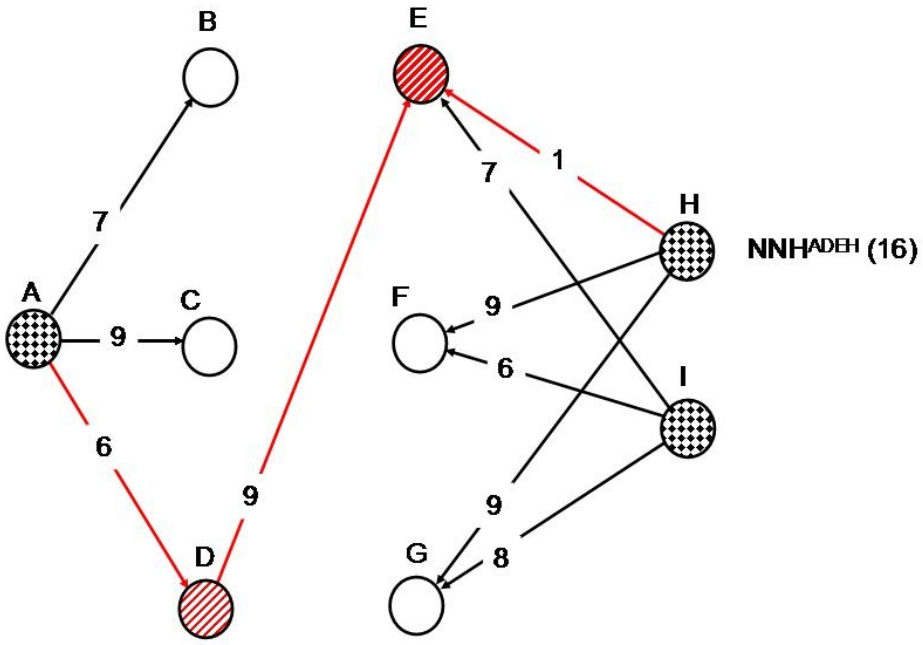

Figure 5.12: Solutions to SPPTC2 as found by an aggregated, bi-directional NNH.

Similarly to the previous variants, the aggregated bi-directional NNH is incapable of returning the optimal solution of SPPTC2. There is no degradation of the quality of the best solution with respect to the driven variant (the best solution is the same in both: $w(ADEH) = 16$) and only a slight reduction of the number of arcs evaluated (see Table 5.4). However, should the number of end vertices increase, the difference in number of arcs evaluated is expected to increase.

Multi-path Bi-directional Search

Given an aggregated bi-directional search of SPPTC2, one can implement the equivalent of a uni-directional multi-path NNH (see Section 5.2.1). In order to achieve this, all backward neighbouring arcs are evaluated simultaneously and only the N champions are pursued. The same is done for the forward search. As with the previous examples, the patch between forward and backward threads occurs, in Figure 5.13, in the second leg.

Applying the multi-path bi-directional NNH with $N = 2$ to SPPTC2 yields the optimal solution: $w(ABEH) = 14$. The second solution, $w(ADEH) = 16$, has a near-optimal path cost. These improved result come at the cost of an increased number of arc evaluation, as seen in Table 5.4.

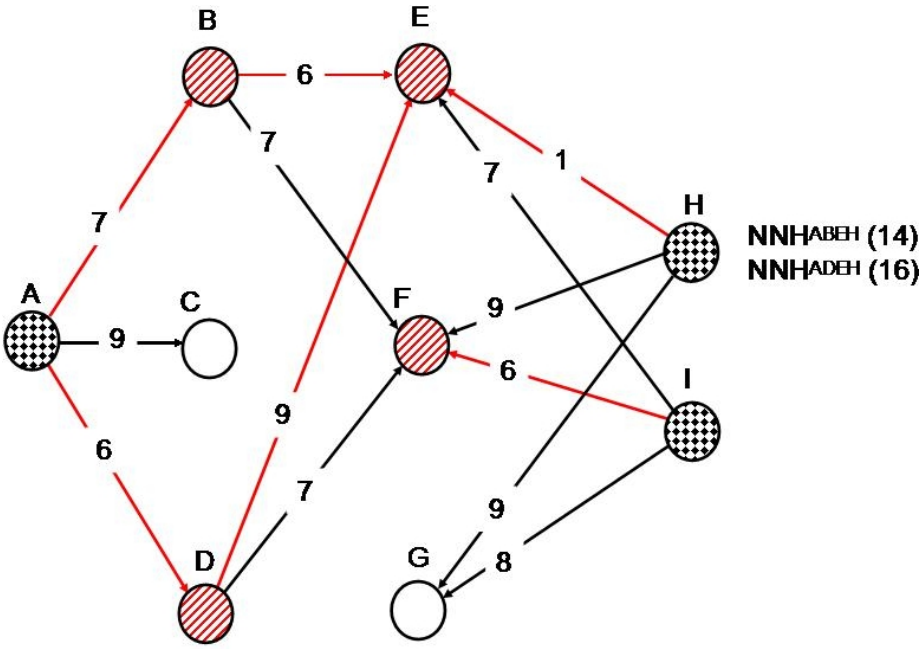

Figure 5.13: Solutions to SPPTC2 as found by a multi-path bi-directional NNH with N=2.

Figure 5.14 shows a Pareto front of solution quality versus number of arcs evaluated in SPPTC2 for five types of NNH: the simple forward search and the free, driven, aggregated and multi-path bi-directional searches. One can see that the original, forward NNH is the algorithm that evaluates the smallest number of paths but it is also the one that yields the worst solution. The best solution, which incidentally corresponds to the problem optimum, is found by the multi-path bi-directional search. It is however not the variant that requires the most number of evaluations. That distinction falls upon the free bi-directional NNH. Note that these results are problem dependent, as other SPPs may be accurately solved already with a simple forward NNH search, or another variant with less arc evaluations. However, it is not expected that this is the case for the GTOC2 problem. Since the multi-path search appears to be the most reliable variant of the bi-directional NNH, it is decided to implement it.

Verification

In order to conduct test the correctness of the implemented multi-path bi-directional NNH algorithm is applied to SPPTC2. The cost matrix of SPPTC2 is as given in Table 5.5. The ∞ symbol indicates the absence of an arc connecting the two nodes.

Table 5.5 resumes the properties of SPPTC2. The assymetry of the matrix indicates that the graph is directed. H and I have an outdegree equal to zero: they are end vertices. A, the starting vertex, neighbours B, C, and D, which in turn are connected to E, F and G. The latter are the only nodes neighbouring the

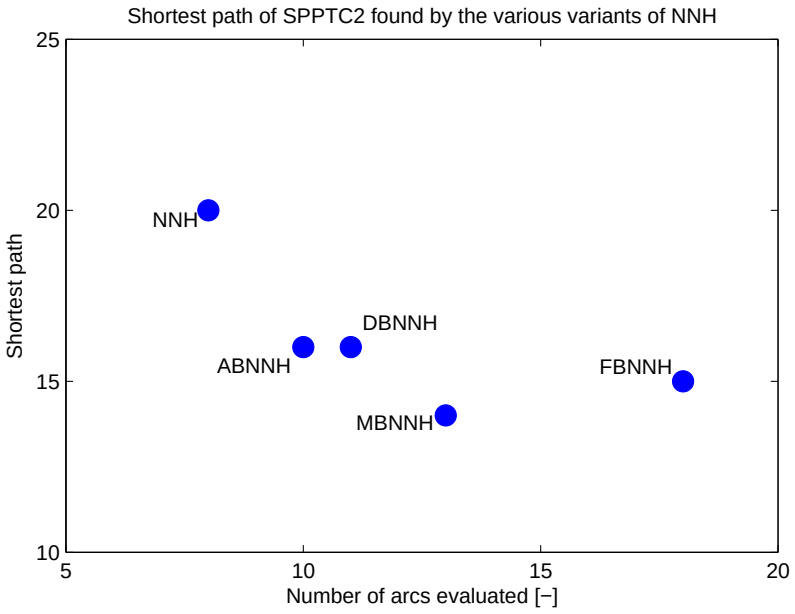

Figure 5.14: Shortest Path in SPPTC2 as a function of the number of arcs evaluated, found by a forward NNH (NNH), a free bi-directional NNH (FBNNH), a driven bi-directional NNH (DBNNH), an aggregated bi-directional NNH (ABNNH), and a multi-path bi-directional NNH (MBNHH).

Table 5.5: Arc costs in SPPTC2.

	v_{i+1}								
v_i	A	B	C	D	E	F	G	H	I
A	∞	7	9	6	∞	∞	∞	∞	∞
B	∞	∞	∞	∞	6	7	11	∞	∞
C	∞	∞	∞	∞	5	6	5	∞	∞
D	∞	∞	∞	∞	9	7	6	∞	∞
E	∞	∞	∞	∞	∞	∞	∞	1	7
F	∞	∞	∞	∞	∞	∞	∞	9	6
G	∞	∞	∞	∞	∞	∞	∞	9	8
H	∞	∞	∞	∞	∞	∞	∞	∞	∞
I	∞	∞	∞	∞	∞	∞	∞	∞	∞

end vertices. In Table 5.6, the outcome of applying the multi-path bi-directional NNH, with a varying number of champions, to SPPTC2 is indicated.

Table 5.6: Solutions to SPPTC2 found by the multi-path bi-directional NNH.

N	Sequence	Path Cost [-]
1	ADEH	16
2	ABEH	14
	ADEH	16
3	ABEH	14
	ACEH	15
	ADEH	16

Table 5.6 shows that increasing the number of champions, N, improves the quality of solutions found, although the optimal solution is found with $N = 2$ already. With a single champion, the multi-path bi-directional NNH is equivalent to the aggregated bi-directional NNH and the solution returned is the one highlighted in red in Figure 5.12: $w(A, D, E, H) = 16$. As discussed earlier, with $N = 2$, the multi-path bi-directional NNH finds the optimal sequence in SPPTC2: $w(A, D, E, H) = 14$. The second solution returned is the same as the one yielded by the aggregated bi-directional NNH. As expected, these are the paths highlighted in red in Figure 5.13. Increasing once more the number of champions, to $N = 3$, preserves the ADEH sequence and more importantly the optimal path. The third solution, $w(A, C, E, H) = 15$, is a shorter path than ADEH, and corresponds to the solution found by the free bi-directional NNH.

Solving the SPPTC2 allowed to verify the performance, in terms of solution quality, of the multi-path bi-directional NNH.

Patch Leg

In order to solve the GTOC2 model from Chapter 4 with the multi-path bi-directional NNH, one must solve the issue of the patch leg, i.e. one must decide in which leg are the forward and backward searches joined. One aspect to take into consideration is the number of transfers evaluated, as this will impact the computational effort needed to solve the problem. The number of transfers computed as a function of the number of champions, N, taken to be the same for the backward and forward search, for each patch leg, can be determined with the following equations:

$$n^1_{\text{transfers}} = N + m_4 \cdot N + m_3 \cdot N + m_2 \cdot m_1 = 16896 + 639 \cdot N \tag{5.2.11}$$

$$n^2_{\text{transfers}} = m_4 + N \cdot N + m_3 \cdot N + m_2 \cdot m_1 = 17234 + 300 \cdot N + N^2 \tag{5.2.12}$$

$$n^3_{\text{transfers}} = m_4 + m_3 \cdot N + N \cdot N + m_2 \cdot m_1 = 17234 + 300 \cdot N + N^2 \tag{5.2.13}$$

$$n^4_{\text{transfers}} = m_4 + m_3 \cdot N + m_2 \cdot N + m_1 \cdot N = 338 + 572 \cdot N \tag{5.2.14}$$

where $n^i_{\text{transfers}}$ indicates the number of transfers evaluated if the patch leg is taken to be Leg i, and m_j is the number of asteroids in the asteroid Group j. Note that selecting the first leg as patch leg corresponds to conducting a

complete backward multi-path search. Similarly, patch the searches in the last leg is equivalent to a forward multi-path search. Another interesting observation is that patching the searches in the second or third leg leads to the same number of transfers evaluated. This is due to the even number of legs. The equations above are illustrated in Figure 5.15.

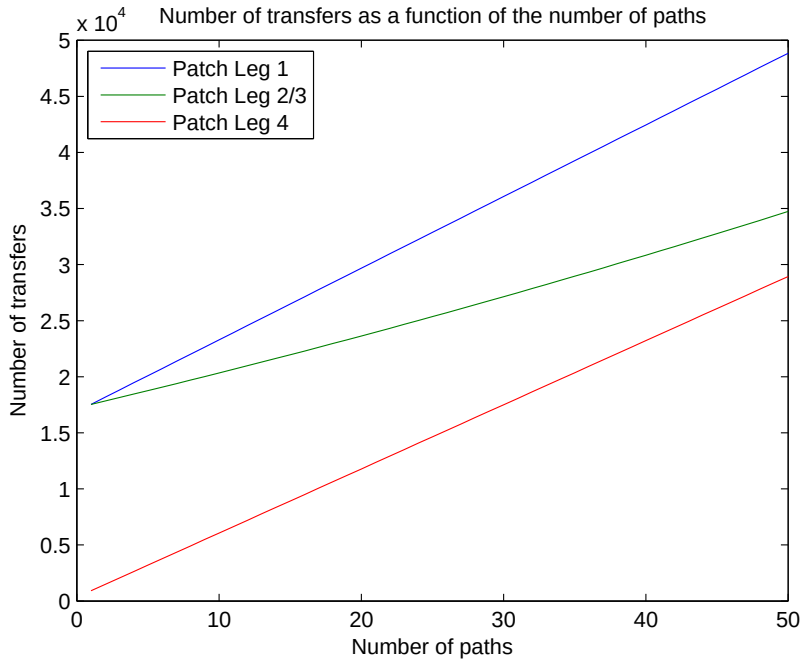

Figure 5.15: Number of GTOC2 transfers evaluated by the multi-path bi-directional NNH as a function of the number of champions and the patch leg.

As expected, performing the backward multi-path search leads to a larger number of transfers evaluated with respect to the forward multi-path search. This can be seen in Figure 5.15 in the form of an offset between the Patch Leg 1 (backward) and Patch Leg 4 (search) curves. The increase in number of computations is mostly due to the fact that all Leg 4 transfers, which are more numerous than all Leg 1 transfers, are evaluated. Moreover, the slope of Patch Leg 1 is slightly larger than that of Patch Leg 4, as seen in Equations (5.2.11) and (5.2.14). Unlike the other two curves, the relation between number of champions and number of computed transfers is polynomial. This is visible in Figure 5.15, as the plot of Patch Leg 2/3 describes a curve instead of a straight line. This curve is consistently below the Patch Leg 1 curve, meaning that patching the backward and forward searches is more computationally attractive than the lone backward search. As previously established, performing the lone forward search, i.e. patching the searches in Leg 4, is more computationally attractive that patching the searchs in Leg 2 or 3 but the latter is expected to yield more accurate results.

Since it is equivalent, from a computational effort perspective, to patch the

forward and backward searches in the second or third leg, the choice between these two possibilities should be based on a preliminary assessment of the quality of the partial solutions found in each direction. This is done in Chapter 9.

Chapter 6

Arc Cost Optimization

As mentioned in Section 4.4, the dynamic GTOC2 problem is transformed into a static SPP by replacing the dynamic arc costs by their optimal value, as obtained in an a-priori optimization step.

After formulating the continuous optimization problem in Section 6.1, we discuss the two optimization techniques used in the scope of the MSc Thesis in Section 6.2: Grid Search (GS) and Differential Evolution (DE). Note that, formally, these are not optimization techniques, given that they do not guarantee the optimality of the solution. However, we will refer to them as such in the sense that they return the best solution evaluated.

6.1 Problem Formulation

Optimization is a recurrent subject in most engineering fields. It can be defined as the process of finding the values for a given number of variables that minimize (or maximize) one or more objective functions while satisfying a given set of constraints. Here, we aim at minimizing the arc costs between celestial bodies. As a reminder, in Section 4.4, the cost functions were defined as:

$$w_{t,1}(v_i, t_i, v_{i+1}, t_{i+1}) = \frac{t_{i+1} - t_i}{m_{i+1}(v_i, t_i, v_{i+1}, t_{i+1})} \tag{6.1.1}$$

$$w_{t,2}(v_i, t_i, v_{i+1}, t_{i+1}) = \frac{1}{m_{i+1}(v_i, t_i, v_{i+1}, t_{i+1})} \tag{6.1.2}$$

with v_i and v_{i+1} the departure and arrival bodies, respectively, t_i and t_{i+1} the departure and arrival dates, respectively, and m_{i+1} the mass at arrival, obtained with Equation (4.4.3). The departure and arrival bodies can be seen as integer parameters which are constant in the scope of the arc cost optimization. The departure and arrival dates are bound within certain intervals, which may overlap. In order to avoid placing a constraint on the problem, namely that $t_{i+1} > t_i$, the arrival date is replaced by another variable, the time-of-flight, t_f:

$$t_f = t_{i+1} - t_i \tag{6.1.3}$$

Based on these considerations, new cost functions must be defined. Given the departure and arrival bodies, the cost functions J to be optimized are defined

as:

$$J_1(t_i, t_f) = \frac{t_f}{m_{i+1}(v_i, t_i, v_{i+1}, t_i + t_f)} \tag{6.1.4}$$

$$J_2(t_i, t_f) = \frac{1}{m_{i+1}(v_i, t_i, v_{i+1}, t_i + t_f)} \tag{6.1.5}$$

and the optimization problem, for a single leg, can be formulated as:

$$\min J_k(t_i, t_f) \tag{6.1.6}$$

with $k = \{1, 2\}$ and subject to:

$$t_i \in [t_{i,\min}, t_{i,\max}] \tag{6.1.7}$$

$$t_f \in [t_{f,\min}, t_{f,\max}] \tag{6.1.8}$$

where $[t_{i,\min}, t_{i,\max}]$ is the departure window, and $t_{f,\min}$ and $t_{f,\max}$ are the minimum and maximum times-of-flight, respectively. It can be shown that the cost functions from Equations (6.1.4) and (6.1.5) are continuous. The optimization problem (6.1.6) can therefore be described as a box-constrained, continuous problem. This has implications on the sort of optimization techniques needed to solve it.

6.2 Optimization Techniques

The field of optimization is rather large and, due to developments to date in computational mathematics and engineering, a very large number of optimization techniques, with specific characteristics, have been developed. There are two main categories of optimization techniques: *analytical* and *numerical* methods. Analytical methods are exact but may only be applied to relatively simple and small problems [Gorter, 2009]. This makes analytical methods non-practical when optimizing a complex problem such as GTOC2. Numerical methods on the other hand are more powerful, despite the fact that they discretize the problem and, as a result, are less accurate than analytical methods. The discretization of the problems makes them ideally tailored for computer analysis. Given that only a preliminary analysis of the GTOC2 problem is sought, numerical methods seem to be well suited for the purpose of optimizing the arc costs.

Two different numerical optimization techniques were selected to perform the arc cost minimization: GS and DE. GS was chosen due to its simplicity, to serve as a milestone before implementing the second algorithm, DE. The latter was selected among other evolutionary algorithms based on the results of several benchmarking efforts (see [Secretin, 2011]) and bearing in mind the requirements in terms of accuracy and computational cost.

6.2.1 Grid Search

GS is best described as an enumerative technique. The premise of enumerative methods is rather simple: the search space is discretized to form a mesh and the objective function is evaluated at each node of the mesh, i.e. at every possible combination of the discrete variables. The accuracy of the result and computational effort of the algorithm depends on the density of the mesh:

typically, the denser the mesh, the more expensive and accurate the solution. The dependence on the grid density also makes it impossible to guarantee the optimality of the result as steep basins of attraction may be missed in between mesh nodes. Although very thorough, enumerative techniques are not the most efficient methods due to the cost of evaluating the entire search space. The pseudo-code for GS, is shown in Figure 6.1.

```
-Input: number of variables and their bounds, stepsize
-Do for each node of variable 1
   -Do for each node of variable 2
      -Do for each node of variable i
            Evaluate objective function
      -End do
   -End do
-End do
-Return best node
```

Figure 6.1: Grid Search function generator [Gorter, 2009].

It is apparent in Figure 6.1 that GS is a very simple algorithm. This simplicity is the major argument behind the choice to implement GS as a landmark, precursor to the implementation of DE. However, due to difficulties in integrating DE within a reasonable timeframe, GS ended up having a major role in the MSc Thesis, as the reader will see in Chapter 9.

Verification

GS was implemented in Tudat (see Chapter 8.1). To proceed with the verification of the algorithm, it is applied to a common benchmark function, namely Branin's function [Myatt et al., 2004]:

$$f(x_1, x_2) = h + a(x_2 - bx_1^2 + cx_1 - d)^2 + h(1 - e)\cos x_1 \tag{6.2.1}$$

where the parameters were set to a = 1, b = $\frac{5.1}{4\pi^2}$, c = $\frac{5}{\pi}$, d = 6, h = 10 and e = $\frac{1}{8\pi}$. The boundary constraints are $x_1 \in [-5, 10]$, $x_2 \in [0, 15]$. In these intervals, Branin's test function contains 3 global optima:

$$f(-\pi, 12.275) = f(\pi, 2.275) = f(9.42478, 2.475) = 0.397887$$

Branin's function is illustrated in Figure 6.2.

In order to avoid the GS returning different optima as a function of the mesh density, the bounds of x_1 are restrained to [5,10] such that the only optimum solution is: $f(9.42478, 2.475) = 0.397887$. The solutions found by GS, as a function of the step-size of the mesh, are given in Table 6.1. The relative error, ϵ, with respect to the true optimum is also given.

As expected, there is a general trend visible in Table 6.1: the denser the mesh, the better the solution found. There are however a few exceptions. First, for a step-size of 0.3 the function value is larger than the value found with step-sizes of 0.1 and 0.5:

$$f_{0.1}(9.40, 2.50) < f_{0.5}(9.50, 2.50) < f_{0.3}(9.50, 2.40)$$

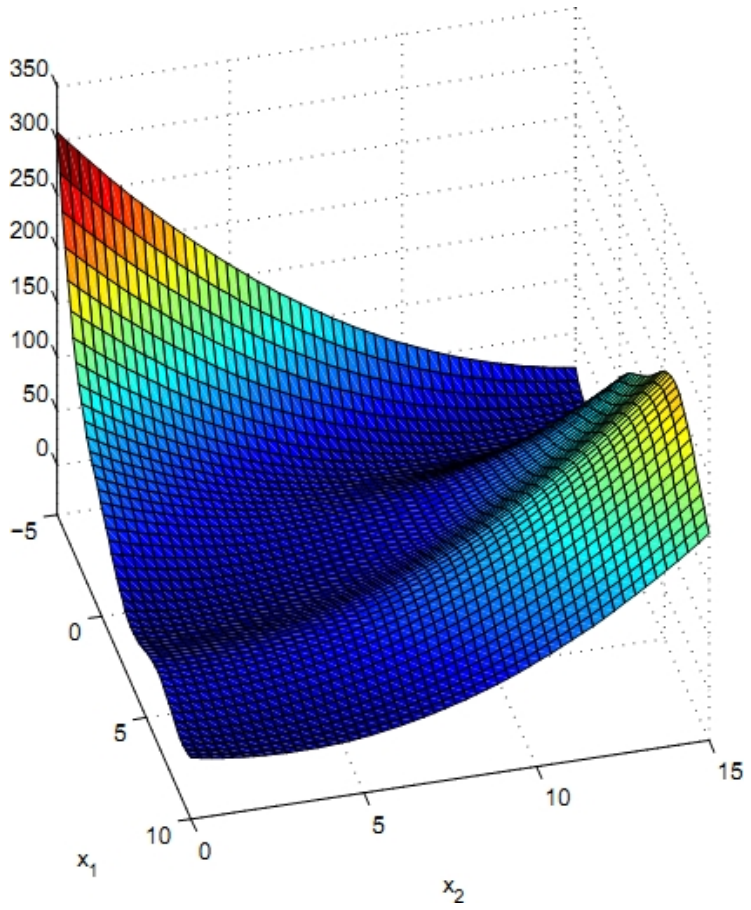

Figure 6.2: Branin's global optimization test function [Myatt et al., 2004].

Table 6.1: Optimal solution of the restricted Branin function, obtained with GS.

Step-size	Number of nodes	x_1	x_2	f	ϵ [%]
0.01	752001	9.42	2.47	0.397998	0.028
0.03	84168	9.41	2.46	0.398942	0.265
0.05	30401	9.40	2.45	0.400852	0.745
0.10	7701	9.40	2.50	0.402934	1.269
0.30	918	9.50	2.40	0.444412	11.693
0.50	341	9.50	2.50	0.426576	7.210
1.00	96	9.00	2.00	1.270825	219.393
2.00	36	9.00	2.00	1.270825	219.393

This is due to the location of the grid points, which varies with the step-size: with a step-size of 0.5, there is a grid point which is closer to the optimum than with a step-size of 0.3. Second, the solution found with step-sizes of 1 and 2 is the same:

$$f_1(9,2) = f_2(9,2)$$

The reason is that the node that is closest to the optimum is present in both meshes. Note that for both step-sizes (1 and 2) the mesh is not dense enough and leads to a solution with a relative error over 200 %. Opposite, a grid step-size of 0.01 leads to a very accurate solution, as the relative error is about 0.02%. Figure 6.3 allows to graphically visualize the impact of the accuracy achieved on the algorithm complexity, as it plots the relative error of the solutions as a function of number of grid points.

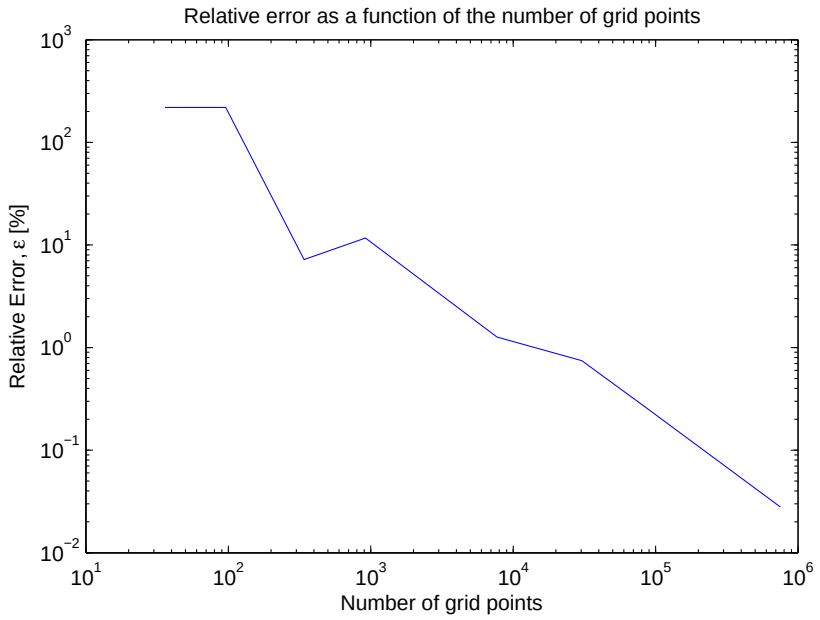

Figure 6.3: Relative error of the Branin function value as a function of the number grid points in a GS.

Based on Figure 6.3 and Table 6.1, an interesting compromise seems to be, for the Branin function, a step-size of 0.05: while the relative error is still under 1%, the dimension of the problem is reduced by one order of magnitude with respect to a step-size of 0.01. In this case, the ratio of the step-size to the length of the largest variable interval, here $x_2 \in [0, 15]$, is 0.003. The ratio to the length of the smallest interval, $x_1 \in [5, 10]$, is 0.01.

Using Branin's function, it was shown that the implemented GS is functional and capable of finding solutions with high accuracy, provided the mesh is sufficiently dense. A sensitivity analysis is carried out hereafter to obtain relevant step-sizes for optimizing GTOC2 transfers.

Sensitivity Analysis

In order to determine the adequate GS parameters when optimizing the GTOC2 transfers, a sensitivity analysis is performed on the step-sizes for the departure date and time-of-flight. The test case is the `Earth - 2002062` transfer. The cost function is the ΔV of the corresponding Lambert arc (see Chapter 7). The departure window is the original GTOC2 departure window: 2015 - 2035. The minimum and maximum times-of-flight are set to 10 and 610 days. The following departure date step-sizes were analyzed: 5, 10, 15, 20, 25, and 30 days. For the time-of-flight, integer values between 1 and 20 days were considered. The best solution found, as a function of the mesh density, is illustrated in Figure 6.4, in the form of a heat map. The number of grid points as a function of the step-size of both problem parameters is given in Figure 6.5.

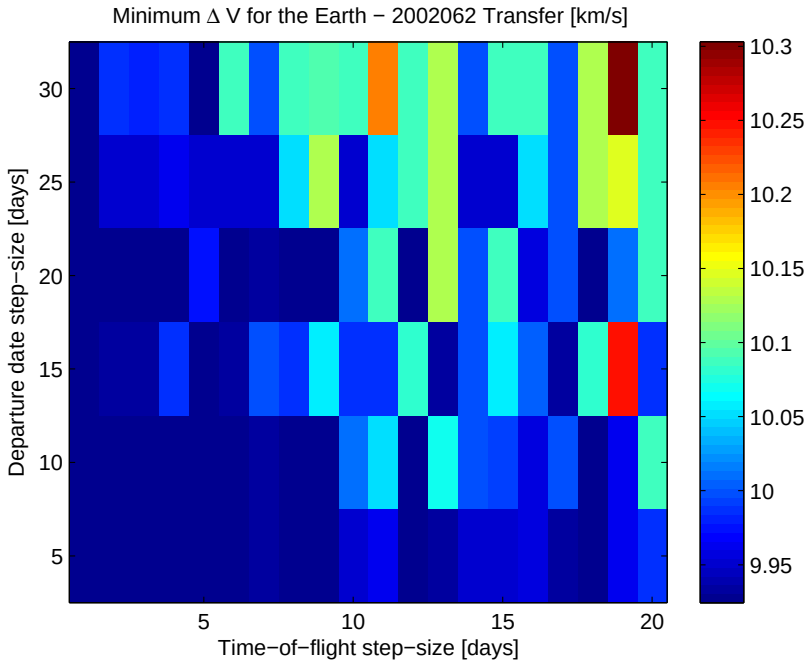

Figure 6.4: Best `Earth - 2002062` transfer found as a function of the GS mesh density.

Figure 6.4 shows that the solution quality is quite sensitive to the step-size of both parameters: while a combination of small step-sizes seems to consistently produce good results, situations occur, for larger step-sizes, where increasing either step-size leads to a better solution. This situation was already observed in Table 6.1, where increasing the step-size from 0.3 to 0.5 led to a lower Branin function value. The epitome of the lack of correlation between decreasing mesh density and decreasing solution quality is the solution found with a departure step-size of 15 days and a time-of-flight step-size of 19 days: increasing either

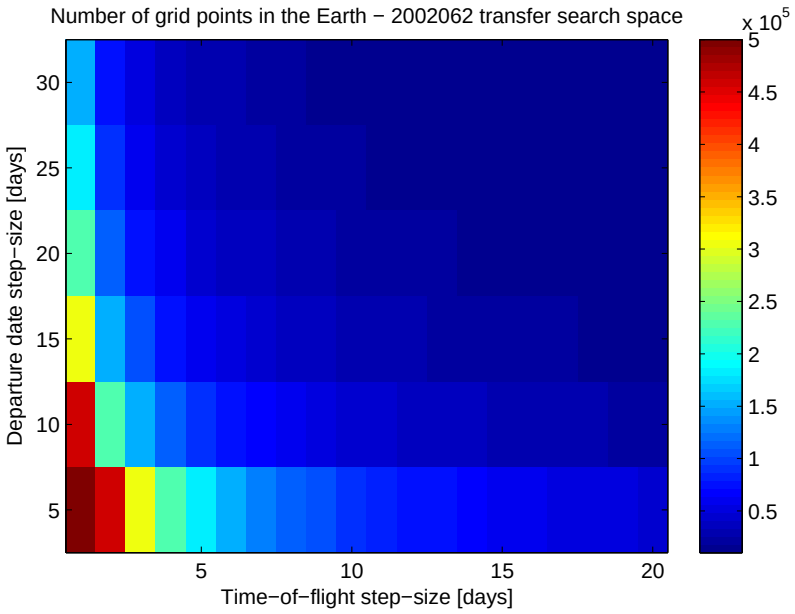

Figure 6.5: Number of transfers evaluated as a function of the GS mesh density.

(or both) step-sizes leads to a better solution[1]. Another example is the fact that the worst solution found (19,30) is not located at the bounds of both intervals.

It is therefore interesting to focus on the area, with small step-sizes, where the solution quality is nearly constant. The bounds of said region are [1,9] and [5,10] for the time-of-flight and departure date step-sizes, respectively. Let us refer to this area as the stable area. Looking at Figure 6.5, one can see that there is, as expected, a correlation between the density of the mesh and computational effort needed by GS to solve a problem. Therefore, the step-sizes of GS should be located within the stable area but at the intersection which yields the less number of grid points. Looking at Figure 6.5, this corresponds to a 9 days step-size for the time-of-flight and a 10 days step-size for the departure date.

In practice, the results from Chapter 9 were obtained with a step-size of 10 days for both parameters. For the `Earth - 2002062` transfer, this corresponds to a degradation of less than 1% in the minimum ΔV found and a decrease in computational effort of 95% with respect to the best solution found. Note that, although it was not implemented in the scope of the MSc Thesis, it is common practice to restart GS after each iteration, on a smaller search space centered around the best solution, until a given accuracy in the real-values variables is reached.

[1] With the exception of (19,30).

6.2.2 Differential Evolution

Differential Evolution (DE) is a relatively recent algorithm proposed in the mid-1990s by [Storn and Price, 1995]. DE is an evolutionary algorithm, designed to solve optimization problems over continuous domains [Coello et al., 2006]. In evolutionary algorithms, a solution to the objective function, i.e. a set of objective function variables, is referred to as an individual. A set of individuals constitutes a population.

The first step of the DE is the random initialization of the population within the possible variable bounds. Each individual is a vector containing real values corresponding to the objective function variables. After initialization of the population, three Evolutionary Operators (EvOps) are applied. The first, referred to as *mutation*, is in fact the combination of three randomly chosen individuals: the first individual, $\mathbf{x}_{0,g}$, is the base individual to which the vector difference of the remaining individuals, $\mathbf{x}_{1,g}$ and $\mathbf{x}_{2,g}$, is added to create a mutant vector, $\mathbf{v}_{i,g}$. The process is depicted in Figure 6.6 and can be mathematically expressed as [Price et al., 2005]:

$$\mathbf{v}_{i,g} = \mathbf{x}_{0,g} + F \cdot (\mathbf{x}_{1,g} - \mathbf{x}_{2,g}) \tag{6.2.2}$$

where F is a scale factor that controls the rate at which the population evolves and the index g indicates the generation. A mutant vector is created for every target individual, $\mathbf{x}_{i,g}$.

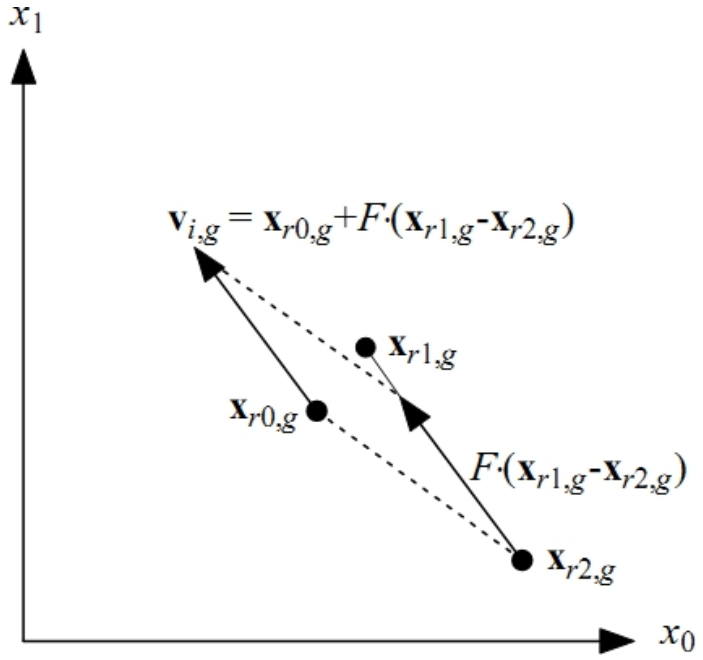

Figure 6.6: Differential mutation [Price et al., 2005].

Following the creation of a mutant vector, DE employs a technique referred to as *uniform crossover*. Here, the target individual and mutant vector are

crossed to create a trial vector, $\mathbf{u}_{i,g}$. The trial vector is built according to [Price et al., 2005]:

$$\mathbf{u}_{i,g} = u_{j,i,g} = \begin{cases} v_{j,i,g} \text{ if } \text{rand}_j(0,1) \leq C_r \text{ or } j = j_{rand} \\ x_{j,i,g} \text{ otherwise} \end{cases} \tag{6.2.3}$$

where the crossover probability, C_r, is user-defined, the index j denotes the different parameters within the individuals and j_{rand} is a randomly generated index to ensure that the trial vector is not a replica of the target vector. The difference between the target vector, $\mathbf{x}_{i,g}$, the mutant vector, $\mathbf{v}_{i,g}$, and the trial vector, $\mathbf{u}_{i,g}$, can be seen, with a slightly different notation, in Figure 6.7.

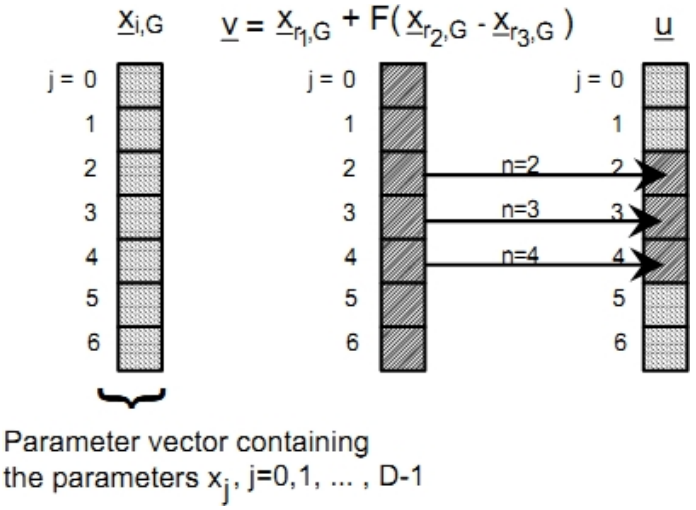

Figure 6.7: The crossover process [Storn and Price, 1995].

Once the trial vector has been generated, its objective function value, $f(\mathbf{u}_{i,g})$ is compared against that of the target vector and the individual yielding the best objective function value is retained for the next generation [Price et al., 2005]:

$$\mathbf{x}_{i,g+1} = \begin{cases} \mathbf{u}_{i,g} \text{ if } f(\mathbf{u}_{i,g}) \leq f(\mathbf{x}_{i,g}) \\ \mathbf{x}_{i,g} \text{ otherwise} \end{cases} \tag{6.2.4}$$

This step is referred to as *selection*. Once a complete new population is installed, the mutation, recombination, selection process is repeated until a convergence criterion is met or a maximum user-defined number of generations is reached. Note that there are different variants of the DE, namely in terms of the type of mutation and recombination EvOps. The one described above is the original DE1 defined by [Storn and Price, 1995].

One should note that increasing the size of the DE population or the number of pairs in the mutation process increases the diversity of possible movements, promoting a more extensive exploration of the space search. On the other hand, the probability of finding the correct search directions decreases considerably. The equilibrium between the population size and the number of vector differences therefore determines the efficiency of the algorithm [Coello et al., 2006].

Self-Adaptive Control Parameters

One significant improvement to DE comes from the concept of self-adaptive control parameters [Izzo, 2011]. The idea behind self-adaptive control parameters for DE is that the variables that control the EvOps, namely the cross-over probability, C_r, and the scale factor, F, are encoded into the individuals and are subjected to the actions of the EvOps. In this fashion, the better values of the control parameters lead to better individuals, which are more likely to survive and hence propagate these attractive parameter values. The equations and parameter values given below are, unless mentioned otherwise, taken from [Brest et al., 2006].

The scale factor, which governs the mutation process, is updated, for each target individual, $\mathbf{x}_{i,g}$, according to:

$$F_{i,g+1} = \begin{cases} F_l + rand(0,1) \cdot F_u, & \text{if } rand(0,1) < \tau_1 \\ F_{i,g}, & \text{otherwise} \end{cases} \tag{6.2.5}$$

while the crossover probability is modified according to:

$$C_{r,i,g+1} = \begin{cases} rand(0,1), & \text{if } rand(0,1) < \tau_2 \\ C_{r,i,g}, & \text{otherwise} \end{cases} \tag{6.2.6}$$

Equations (6.2.5) and (6.2.6) introduce four new parameters: F_l, F_u, τ_1 and τ_2. The last two can be set to $\tau_1 = \tau_2 = 0.1$. Defining $F_l = 0.1$ and $F_u = 0.9$, allows the scale factor to take a real value from $[0.1, 1.0]$. The crossover probability is bound within $[0, 1]$.

The new control parameters are computed before mutation occurs and therefore influence the three EvOps: mutation, crossover and selection. In a DE algorithm with self-adaptive control parameters, Equations (6.2.2) and (6.2.3) take the form:

$$\mathbf{v}_{i,g} = \mathbf{x}_{0,g} + F_{i,g+1} \cdot (\mathbf{x}_{1,g} - \mathbf{x}_{2,g}) \tag{6.2.7}$$

$$\mathbf{u}_{i,g} = u_{j,i,g} = \begin{cases} v_{j,i,g} \text{ if } \mathrm{rand}_j(0,1) \leq C_{r,i,g+1} \text{ or } j = j_{rand} \\ x_{j,i,g} \text{ otherwise} \end{cases} \tag{6.2.8}$$

With this self-adaptive variant of DE, there is no need to guess good values for the control parameters, which are usually problem dependent, making the optimization process more robust. The modifications necessary are rather simple and therefore there is no increase in the time complexity of the algorithm.

Verification

Given that DE is a rather well-known optimization heuristic, there are a number of readily available implementations. The choice is therefore made to make use of an external software package rather than implementing DE from scratch. The software package selected is Parallel Global Multi-objective Optimization (PaGMO) [Biscani et al., 2010]. This choice is driven by three important factors. First, this implementation has been tested and validated for a number of space mission scenarios. Second, making use of the PaGMO library allows to (almost) effortlessly switch between the other available optimization algorithms. Third,

the same choice was made by the Tudat Management Team (see Chapter 8.1) and therefore the efforts of integration of PaGMO with Tudat were divided among several students rather than relying on the shoulders of the author alone.

However, numerous and obscure[2] obstacles were encountered when attempting to interface Tudat and PaGMO. These difficulties, coupled with time constraints imposed by a conference paper ([Secretin and Noomen, 2012]), led to the late introduction of DE into the MSc Thesis. At that point, a wealth of results had already been obtained with GS. As a result, the verification of the DE algorithm from PaGMO was carried out by comparison with said results. More specifically, the results obtained for the first leg of the GTOC2 problem.

The benchmark data consists of the best `Earth` - Group 4 asteroids (Leg 1) transfers obtained with GS. The departure window from Earth was taken to be the original GTOC2 interval: January 1^{st} 2015 to December 31^{st} 2035. The minimum and maximum times-of-flight were set to 10 and 610 days, respectively. The step-size of the grid search was taken to 10 days for both variables. With respect to DE, use was made of the self-adaptive variant, `de_self_adaptive`, available in PaGMO. The population of a 100 individuals was evolved for 50 generations. The top-10 Leg 1 transfers found with both DE and GS are given in Table 6.2.

Table 6.2 shows that the solutions found by the self-adaptive DE are rather similar to the benchmark data obtained with GS. The ranking of the top-ten transfers is preserved, with the exception of asteroid `3167367` which climbs one place from the GS (8^{th}) to the DE ranking (7^{th}). This can be explained by the fact that the ΔV returned by GS for the `Earth - 3167367` transfer is larger than the one found by DE. Table 6.3 lists the difference in parameter and objective function values between the best solution found by both algorithms for each of the top-ten transfers.

Table 6.3 shows that the differences between the parameters of the best solutions found are all, with one (noteworthy) exception, within 10 days (the step-size of GS[3]) of each other. One can therefore state that the best solutions found by DE are very similar to those found with GS. This similarity is also observable in the objective function value, the largest difference being in ΔV being 22.79 m/s for the `Earth - 3167367` transfer, which corresponds to an improvement of merely 0.7%. Note that this is the transfer that rose one place in the DE ranking.

With respect to the previously mentioned exception, the departure date of the best `Earth - 3042555` transfer found by DE is about 7 years earlier than the best GS solution. However, given that the transfer duration and ΔV budget of both solutions are relatively similar, it is hypothesized that this corresponds to the synodic period of both bodies. The synodic period is defined as the time interval between the repetition of a particular relative orbit geometry between celestial bodies [Kemble, 2010]. Assuming circular, co-planar orbits[4], the synodic period, τ, is computed according to [Kemble, 2010]:

$$\tau = \frac{360}{\frac{360}{T_1} - \frac{360}{T_2}} \tag{6.2.9}$$

[2] To the author, that is.

[3] Bear in mind that higher accuracy for GS could have been achieved by "zooming in" on the grid point of lowest cost function value.

[4] The eccentricity of asteroid `3042555` is $e_{3042555} = 0.11212565$ and its inclination is $i_{3042555} = 2.8299469°$.

Table 6.2: The top-ten Leg 1 transfers found by GS and self-adaptive DE.

Grid Search				
Rank	Asteroid	Departure Date [JD]	Time-of-flight [days]	ΔV [m/s]
1	3042555	2461944.0	310.0	2187.94
2	3250293	2462194.0	280.0	2290.85
3	3017309	2461074.0	170.0	2390.14
4	3024030	2461824.0	270.0	2621.51
5	3054338	2463974.0	250.0	2692.80
6	3293922	2459474.0	210.0	2707.18
7	3261681	2460744.0	280.0	3051.47
8	3167367	2460604.0	170.0	3074.55
9	3070801	2459084.0	270.0	3092.74
10	3156302	2460954.0	240.0	3226.06
Self-adaptive DE				
Rank	Asteroid	Departure Date [JD]	Time-of-flight [days]	ΔV [m/s]
1	3042555	2459379.4	311.4	2175.87
2	3250293	2462197.5	277.6	2287.21
3	3017309	2461073.8	175.1	2383.57
4	3024030	2461817.7	273.3	2613.28
5	3054338	2463972.3	252.2	2697.10
6	3293922	2459470.9	207.1	2697.53
7	3167367	2460601.1	171.1	3051.76
8	3261681	2460743.0	284.4	3056.91
9	3070801	2459080.7	269.2	3076.09
10	3156302	2460958.6	236.0	3229.73

Table 6.3: Difference in parameter and objective function values between GS and DE top-ten Leg 1 transfers. The departure date is denoted t_d.

Asteroid	$t_{d,DE} - t_{d,GS}$ [days]	$t_{f,DE} - t_{f,GS}$ [days]	$\Delta V_{DE} - \Delta V_{GS}$ [m/s]
3042555	-2564.6	1.4	-12.07
3250293	3.5	-2.4	-3.64
3017309	-0.2	5.1	-6.57
3024030	-6.3	3.3	-8.23
3054338	-1.7	2.2	4.3
3293922	-3.1	-2.9	-9.65
3261681	-1.0	4.4	5.44
3167367	-2.9	1.1	-22.79
3070801	-3.3	-0.8	-16.65
3156302	4.6	-4.0	3.67

where T_1 and T_2 are the orbital periods of bodies, given by [Wakker, 2007a]:

$$T = 2\pi\sqrt{\frac{a^3}{\mu}} \tag{6.2.10}$$

with a denoting the semi-major axis and μ the central body gravitational parameter. Using Equations (6.2.9) and (6.2.10), the synodic period of the Earth and asteroid `3042555` is 2600 days which is about the same value as the difference in departure dates seen in Table 6.3. Hence, the solutions returned by DE and GS for the best `Earth - 3042555` are in fact (nearly) the same.

Table 6.3 suggests that the solutions found by DE are generally better, i.e. have a lower ΔV, than those found with GS. This is however not the case if one looks at Figure 6.8, where the difference in cost function, computed as $\delta = \Delta V_{DE} - \Delta V_{GS}$, is shown for all 338 Leg 1 transfers. Similar plots, for the departure date and time-of-flight, can be found in Appendix B.1.

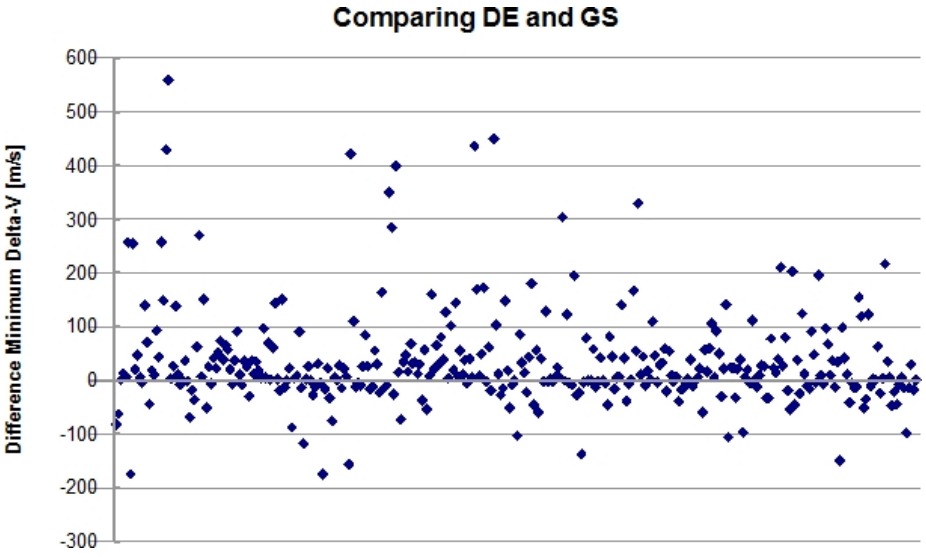

Figure 6.8: Difference in ΔV between the best Leg 1 transfers found by DE (with 100 individuals and 50 generations) and GS.

In Figure 6.8, the positive part of the difference spectrum is more densely populated than the negative part, meaning that a majority of transfers have a better cost function when optimized with GS. Coherently, the largest outliers can be found on the positive side of the ordinates axis: while the maximum improvement in cost function is about 200 m/s, the largest degradation is approximately 550 m/s. Note however that a majority of data points are within $\pm$50 m/s, meaning that, overall, there is a relatively small degradation of the results when DE is used instead of GS. This is confirmed when looking at the Root Mean Square (RMS) value of all Leg 1 transfers for each optimizer:

$$\Delta V^{GS}_{Leg1,RMS} = 11821.35 \text{ m/s} \tag{6.2.11}$$

$$\Delta V^{DE,50,100}_{Leg1,RMS} = 11863.29 \text{ m/s} \tag{6.2.12}$$

Taking the difference between both values, the objective function values found with DE are on average 41.94 m/s larger than with GS. This seems to hint at the fact that the DE parameters used are not stringent enough to yield overall better results than GS.

Besides the quality of the solutions returned, the computational time needed by both algorithms to optimize all 338 Leg 1 transfers can also be compared. Doing so reveals that, with the control parameters described earlier[5], DE is 7 times faster than GS: GS needed 184 seconds to optimize all Leg 1 transfers while it took DE only 26 seconds.

Optimizing the Leg 1 transfers with the selected DE implementation shows that the algorithm is functional. Moreover, the analysis above reveals that the selected GS step-size results in relatively good solutions when compared with DE. With these control parameters, replacing GS with a self-adaptive DE allows to considerably reduce the computational effort needed to solve the benchmark problem. Despite the considerable reduction in computational effort, there is only a relatively small degradation of the solution quality.

Sensitivity Analysis

We have seen in the previous section that with a population of 100 individuals updated for 50 generations, the results found with DE are, overall, worse than with GS. In order to identify the most interesting combination of DE parameters, a sensitivity analysis is performed on the influence of the numbers of individuals and generations on the quality of the solutions. The same test case as for GS is taken, namely the `Earth - 2002062` transfer with the 2015-2035 departure window and the time-of-flight bound between 10 and 610 days. The cost function is also the same, i.e. the ΔV of the corresponding Lambert arc. The numbers of individuals and generations take a value from $\{50, 100, 150, 200, 250, 300, 350, 400, 450, 500\}$. Figure 6.9 gives the minimum ΔV found as a function of both parameters as a heat map. In similar format, the number of transfers evaluated, which is a metric of the computational effort, for the different combinations of the DE parameters is given in Figure 6.10.

It is interesting to note that, for the `Earth - 2002062` transfer illustrated in Figure 6.9, the quality of the solution is independent of the parameters, as long as the number of generations is larger than 100. In these scenarii, the minimum ΔV is the same: 9922.62 m/s. This suggests that this value is very close or equal to the problem optimum: the solution does not improve past this value, either because it represents the true optimum or because the convergence criterion regarding the maximum number of generations without fitness improvement is reached. For 100 generations, the solution quality is the same for populations with more than 50 individuals and is equal to the solution found with larger numbers of generation. For 50 generations, the solution quality is highly sensitive to the number of individuals, although a larger number of individuals does not necessarily guarantee a better solution. This fact can be attributed to the population initialization, whose initial average fitness is random. The number

[5] As a reminder, DE was executed for 50 generations of a population with 100 individuals, while the step-size of the GS mesh was set to 10 days for both the departure date and time-of-flight.

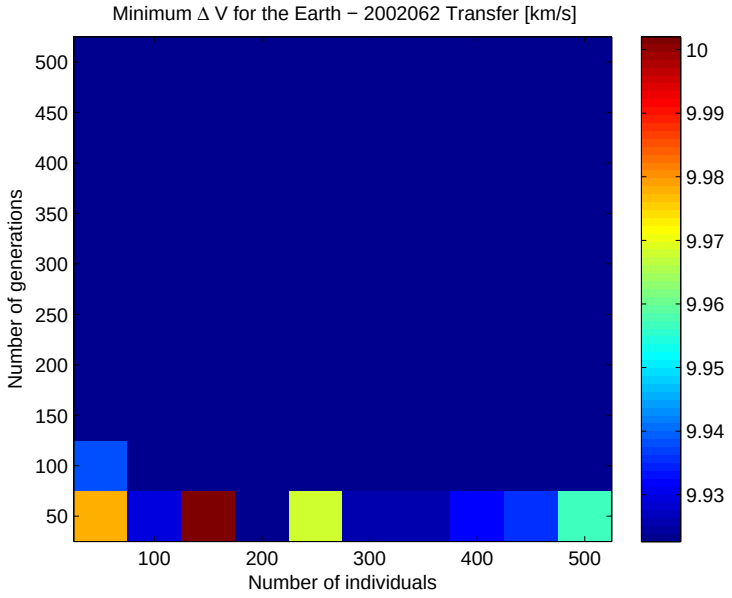

Figure 6.9: Best `Earth - 2002062` transfer found as a function of the DE parameters.

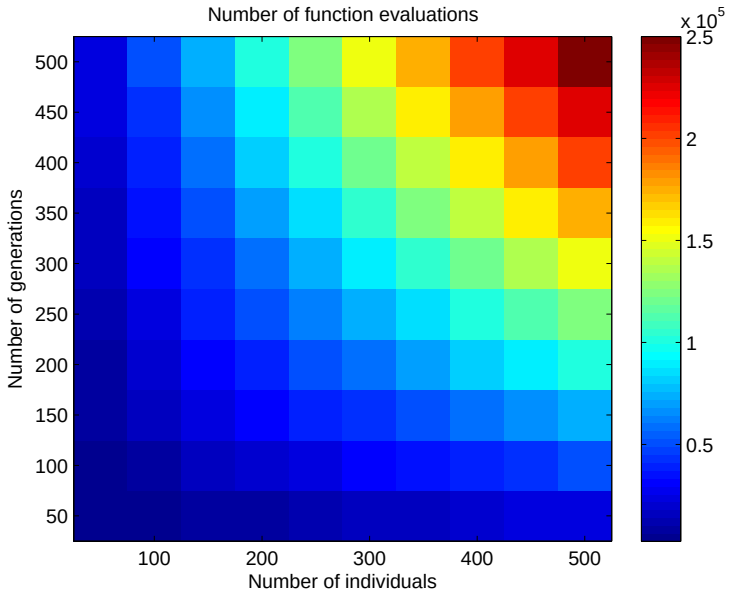

Figure 6.10: Number of transfers evaluated as a function of the DE parameters.

of generations (50) is too limited to allow the algorithm to consistently converge towards the optimum.

Figure 6.10 shows that increasing the number of generations and/or individuals leads to an increase in the (theoretical) computational effort needed to solve the problem. Combining this information with the one extracted from Figure 6.9, it would seem that the best trade-off between computational effort and solution fidelity comes from having a 50 individuals population evolve for 150 generations. With these parameters, all Leg 1 transfers are computed within 35 seconds, which is 5 times faster than GS. The difference in cost function, computed in the same fashion as in Figure 6.8, is shown for all the 338 Leg 1 transfers in Figure 6.11. Again, similar plots, for the departure date and time-of-flight, can be found in Appendix B.1.

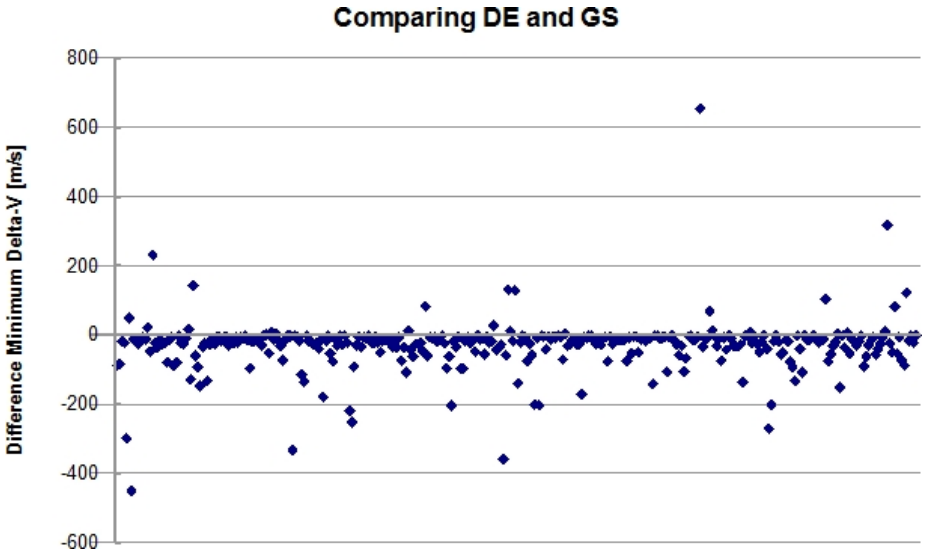

Figure 6.11: Difference in ΔV between the best Leg 1 transfers found by DE (with 50 individuals and 150 generations) and GS.

The difference between Figures 6.8 and 6.11 is striking. Despite a larger positive outlier (+650 m/s), the "center of mass" of the graph is now in the negative part of the difference spectrum, meaning that, overall, the objective function values found by DE are now better than those found with GS. The number of positive differences (which represent transfers that yielded lower function values with GS than DE) has considerably diminished and only three of those are larger than +200 m/s. Finally, the magnitude of the largest objective function improvement reaches -450 m/s. This progress in solution quality can also be seen when comparing the RMS value of all Leg 1 transfers:

$$\Delta V^{GS}_{Leg1,RMS} = 11821.35 \text{ m/s} \tag{6.2.13}$$

$$\Delta V^{DE,150,50}_{Leg1,RMS} = 11790.78 \text{ m/s} \tag{6.2.14}$$

With the selected parameters, the objective function values found with DE are on average 30.56 m/s better than with GS.

The sensitivity analysis shows that replacing GS with a self-adaptive DE evolving a population of 50 individuals for 150 generations leads to an overall improvement of the solutions while being approximately five times faster.

Chapter 7

Lambert Problem

Having formulated the continuous optimization problem in the previous chapter, we now turn towards the mathematics behind the cost functions J_1 and J_2. Based on Tsiolkowski's law, maximization of the final mass is equivalent to minimization of the total ΔV. The way in which this ΔV is obtained can vary as seen in Chapter 3. For the purposes of the MSc Thesis, we take the ΔV budget to be the result of a bi-impulsive, high-thrust transfer modelled as a Lambert arc.

Following the description of the generic Lambert problem in Section 7.1, the Lagrange-Gauss approach to solving it is detailed in Section 7.2. Thereafter we elaborate, in Section 7.3, on the modifications proposed by Dr. Dario Izzo from ESA/ACT. Finally, the newly implemented algorithm is tested and compared against another Lambert routine in Section 7.4.

7.1 General Description

The *Lambert Problem*, also referred to in literature as the *orbital boundary-value problem*, is named after the first mathematician that found its solution. The problem can be described as determining the conic section connecting two points in space, described by their position vectors $\mathbf{r}_1$ and $\mathbf{r}_2$, in a specified time-of-flight, t_{spec}.

Introducing the time parameter into the equation allows to solve an essential aspect of rendezvous missions: the phasing. It not enough to reach the orbit of a given body; that body must be at certain point along its trajectory, namely at the point of insertion, i.e. the point where the transfer and target orbits intersect. Another phasing aspect which must be taken into consideration when solving consecutive Lambert Problems is that of the sequence of asteroids. The departure time from one asteroid must occur after the arrival time at that asteroid. More specifically, when considering the asteroid-to-asteroid trajectories in the GTOC2 problem, the departure must occur at least 90 days after arrival.

The geometry of the Lambert Problem, assuming an elliptical transfer orbit, is shown in Figure 7.1. The initial and final positions are defined by their respective position vectors, $\mathbf{r}_1$ and $\mathbf{r}_2$, the angle between them is denoted θ and the Euclidean distance between them is named the chord, c. Lambert showed that the solution to the Lambert Problem is only a function of the semi-major

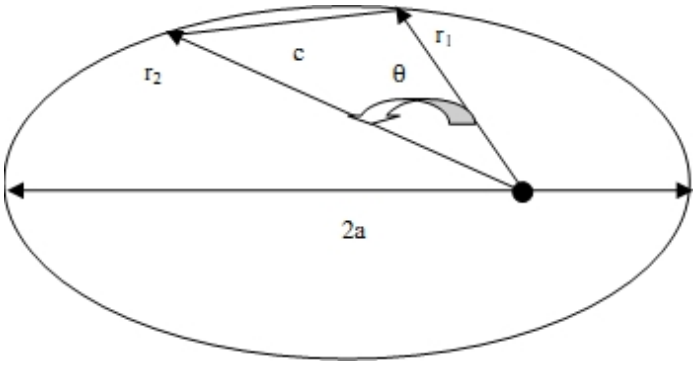

Figure 7.1: Geometry of the Lambert Problem. [Kemble, 2010]

axis, a, the sum of the distances, $r_1 + r_2$, and the chord, c, in what is called the Lambert Theorem:

$$\sqrt{\mu}(t_2 - t_1) = F(a, r_1 + r_2, c) \tag{7.1.1}$$

Since the radii and time-of-flight are given, and that the chord can be easily derived from the Law of Cosines:

$$c^2 = r_1^2 + r_2^2 - 2r_1 r_2 \cos\theta \tag{7.1.2}$$

the Lambert problem is that of finding a satisfying Equation (7.1.1).

Note that the angle θ in Figure 7.1 depicts the smallest angle between both position vectors. In any geometry, the two position vectors define two complementary transfer angles: θ and $\theta' = 2\pi - \theta$. These two angles yield two different transfers: the long-way (that corresponds to the largest transfer angle) and the short-way (associated with the smallest transfer angle). It is important to differentiate both situations as they will yield different conics due to the time-of-flight constraint. Also, bear in mind that the situation depicted in Figure 7.1 represents an elliptical solution; very short time constraints may yield a hyperbolic transfer orbit instead. Finally, multi-revolution scenarios, where the spacecraft completes one revolution or more along the transfer orbit before intercepting the target, can be considered, although this situation will likely not be considered in the subsequent MSc Thesis.

More than the conic section itself, we are interested in the velocity vectors along that orbit at the initial and final positions. For a complete discussion on the Lambert Problem, its different solutions and their characteristics, please refer to [Battin, 1999]. Here we will focus on the method developed by Dr. Dario Izzo from ESA/ACT, which can be seen as an improvement upon the Lagrange-Gauss method.

7.2 The Lagrange-Gauss Equations

In this section, the method upon which Izzo's algorithm is based is described. This method uses both Lagrange and Gauss equations and makes extensive use of different trigonometric identities, which are gathered in Appendix A. This

section is, unless mentioned otherwise, based on [Battin, 1999] and the reader is referred to that literature for a more comprehensive discussion of this method.

The Time-Of-Flight Equation

It can be shown that the time-of-flight, t_f, along the ellipse between the initial and final position, defined by $\mathbf{r}_1$ and $\mathbf{r}_2$, respectively, in Figure 7.1, is given by:

$$t_f = t_2 - t_1 = \sqrt{\frac{a^3}{\mu}} \left[E_2 - E_1 - e(\sin E_2 - \sin E_1) \right] \tag{7.2.1}$$

where a is the semi-major axis of the ellipse, μ the gravitational parameter of the central body, e the eccentricity of the ellipse and E the eccentric anomaly of either the departure or the arrival body, according to the index. Using the trigonometric identity (A.6.2), the previous equation can be re-written as:

$$t_f = 2\sqrt{\frac{a^3}{\mu}} \left[\frac{1}{2}(E_2 - E_1) - e \sin \frac{1}{2}(E_2 - E_1) \cos \frac{1}{2}(E_2 + E_1) \right]$$

Introducing the parameters ψ and ϕ, defined as:

$$\psi = \frac{1}{2}(E_2 - E_1) \qquad \cos \phi = e \cos \frac{1}{2}(E_2 + E_1) \tag{7.2.2}$$

leads to

$$t_f = 2\sqrt{\frac{a^3}{\mu}} (\psi - \sin \psi \cos \phi) \tag{7.2.3}$$

From the properties of a conic, we obtain:

$$\begin{aligned} r \cos f &= a \cos E - ae = a(\cos E - e) \\ r \sin f &= a\sqrt{1 - e^2} \sin E \end{aligned}$$

where f is the true anomaly. A different notation for the true anomaly was adopted to avoid confusions with the transfer angle, θ: $\theta = f_2 - f_1$. The new notation is illustrated in Figure 7.2.

These two equations combined yield:

$$r = a(1 - e \cos E) \tag{7.2.4}$$

This equation can be used to compute the sum of radii:

$$r_1 + r_2 = a(1 - e \cos E_1) + a(1 - e \cos E_2) = 2a - ae(\cos E_1 + \cos E_2)$$

Using the trigonometric identity (A.6.3) and substituting for ψ and ϕ:

$$r_1 + r_2 = 2a(1 - \cos \phi \cos \psi) \tag{7.2.5}$$

Let us now take a look at the chord, which was defined in Equation (7.1.2):

$$c^2 = r_1^2 + r_2^2 - 2r_1 r_2 \cos \theta = (r_1 + r_2)^2 - 2r_1 r_2 (1 + \cos \theta)$$

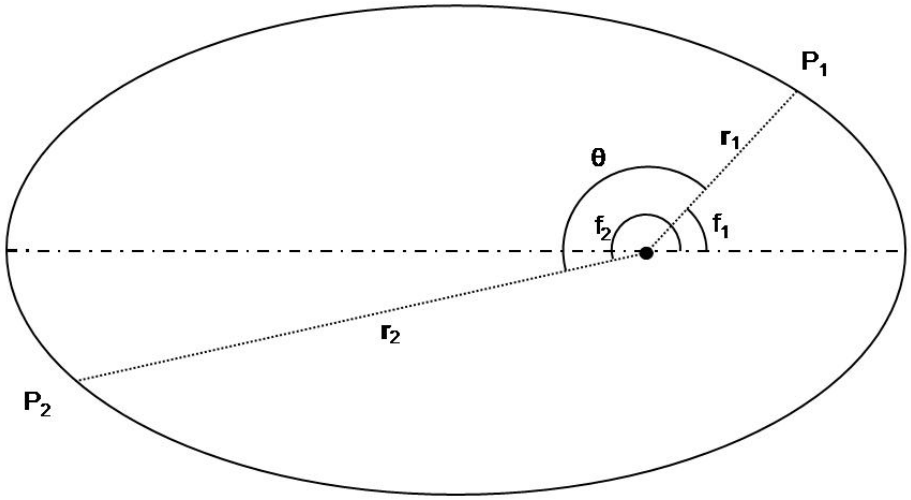

Figure 7.2: The new angular notation.

Trigonometrical manipulation, by the use of the trigonometric identity (A.4.1), results in the following expression:

$$c^2 = (r_1 + r_2)^2 - 4r_1 r_2 \cos^2 \frac{1}{2}\theta \tag{7.2.6}$$

Now, it can be shown that the following holds:

$$\sqrt{r_1 r_2} \cos \frac{1}{2}\theta = a(\cos \psi - \cos \phi) \tag{7.2.7}$$

Substituting Equations (7.2.5) and (7.2.7) into Equation (7.2.6) yields:

$$c^2 = 4a^2 \sin^2 \phi \sin^2 \psi$$

The equation for the chord as a function of ψ and ϕ simply follows:

$$c = 2a \sin \phi \sin \psi \tag{7.2.8}$$

Let s be the semi-perimeter of the triangle defined by the primary focus of the ellipse and the initial and final positions, visible in Figure 7.1. Intuitively, we derive the following equation for the semi-perimeter:

$$s = \frac{r_1 + r_2 + c}{2} \tag{7.2.9}$$

Combining Equations (7.2.5) and (7.2.8) with Equation (7.2.9) and making use of the trigonometric identities (A.3.3) and (A.3.4), we can derive the following quantities:

$$\begin{aligned} 2s = r_1 + r_2 + c &= 2a\,[1 - \cos(\phi + \psi)] \\ 2(s - c) = r_1 + r_2 - c &= 2a\,[1 - \cos(\phi - \psi)] \end{aligned}$$

Introducing two auxiliary variables, α and β, defined as:

$$\alpha = \phi + \psi \qquad \beta = \phi - \psi \tag{7.2.10}$$

these quantities can be expressed as:

$$\frac{s}{2a} = \frac{1}{2}(1 - \cos\alpha)$$
$$\frac{s-c}{2a} = \frac{1}{2}(1 - \cos\beta)$$

Using the trigonometric identity (A.4.2), we obtain:

$$\frac{s}{2a} = \sin^2\frac{1}{2}\alpha \tag{7.2.11}$$
$$\frac{s-c}{2a} = \sin^2\frac{1}{2}\beta \tag{7.2.12}$$

Taking a step back to the time-of-flight formulation in Equation (7.2.3), we can replace ψ and ϕ by the new auxiliary variables, α and β. From Equation (7.2.10), the expressions for ψ and ϕ can be derived :

$$\psi = \frac{1}{2}(\alpha - \beta) \qquad \phi = \frac{1}{2}(\alpha + \beta) \tag{7.2.13}$$

These expressions lead to the following equation:

$$t_f = 2\sqrt{\frac{a^3}{\mu}}\left[\frac{1}{2}(\alpha - \beta) - \sin\frac{1}{2}(\alpha - \beta)\cos\frac{1}{2}(\alpha + \beta)\right]$$

Using yet another trigonometric identity, namely (A.5.1), we obtain what is commonly referred to as the Lagrange expression for the time-of-flight equation:

$$t_f = \sqrt{\frac{a^3}{\mu}}\left[(\alpha - \sin\alpha) - (\beta - \sin\beta)\right] \tag{7.2.14}$$

The derivation of the Lagrange expression for the time-of-flight for the hyperbola is quite similar, but use is made of hyperbolic trigonometric functions instead. Note that the only unknown, for a given geometry, in Equation (7.2.14), is the semi-major axis, a. Indeed, the parameters α and β can be expressed in terms of a, c and s (Equations (7.2.11) and (7.2.12), respectively), the last two being constant for a given geometry (Equations (7.2.8) and (7.2.9), respectively). However, it can be shown that a is not convenient variable to solve the time-of-flight equation, for two reasons. First, the time-of-flight equation is a double-valued function of a, meaning that a single value for the semi-major axis yields two different solutions for the time-of-flight. Second, the derivative of the time-of-flight with respect to the semi-major axis is a rather complex function with a singularity. It is therefore more appropriate to express Equation (7.2.14) as a function of another variable, x:

$$x = \cos\frac{1}{2}\alpha \tag{7.2.15}$$

Combining Equations (7.2.15) and (7.2.11) and making use of the trigonometric identity (A.2.1) leads to a more insightful expression for x:

$$x^2 = 1 - \frac{s}{2a} = 1 - \frac{a_m}{a} \tag{7.2.16}$$

where

$$a_m = s/2 \tag{7.2.17}$$

is the semi-major axis of the minimum energy ellipse. Given Equation (7.2.16), we can determine the shape of the orbit based on the value of x, as we would have done with the semi-major axis:

$$\begin{array}{ccc} a > 0 & \text{elliptic orbit} & -1 < x < 1 \\ a = \infty & \text{parabolic orbit} & x = 1 \\ a < 0 & \text{hyperbolic orbit} & x > 1 \end{array}$$

For a given x, the time-of-flight can be determined by reconstructing its different parameters. The semi-major axis, a, can be expressed as:

$$a = \frac{a_m}{1 - x^2} \tag{7.2.18}$$

From this definition, which derives from Equation (7.2.16), one can see that the sign of a, and therefore the shape of the conic, depends on the value of x. If $x < 1$, the orbit is an ellipse and the shape-dependent parameters of Equation (7.2.14) are:

$$\alpha = 2 \arccos x \tag{7.2.19}$$

$$\beta = \pm 2 \arcsin \left(\sqrt{\frac{s - c}{2a}} \right) \tag{7.2.20}$$

Equations (7.2.19) and (7.2.20) derive from Equations (7.2.15) and (7.2.12), respectively. In case the value of x is larger than 1, the orbit is a hyperbola and the α and β parameters are obtained via the hyperbolic trigonometric functions:

$$\alpha = 2 \cosh^{-1} x \tag{7.2.21}$$

$$\beta = \pm 2 \sinh^{-1} \left(\sqrt{\frac{s - c}{-2a}} \right) \tag{7.2.22}$$

and the time-of-flight equation must be modified to accommodate these hyperbolic trigonometric functions:

$$t_f = \sqrt{\frac{-a^3}{\mu}} \left[(\sinh \alpha - \alpha) - (\sinh \beta - \beta) \right] \tag{7.2.23}$$

Note that the $\pm$ signs in the definition of β account for either the short or the long-way solutions: the + sign is used for the short-way and the - sign for the long-way solution. Given that spacecraft motion typically occurs in the counter-clockwise direction[1], both short and long-way solutions are valid options, depending on the relative position of the initial and final points.

[1] Traveling in the counter-clockwise direction allows to benefit from the motion of the Earth at launch.

Terminal Velocities

As mentioned earlier, beyond the ellipse connecting the departure and arrival points in the give time-of-flight, we are interested in knowing the velocity vectors along that ellipse at those two points, $\mathbf{V}_1$ and $\mathbf{V}_2$ respectively. Let us start by establishing the expression for $\mathbf{V}_2$ as function of $\mathbf{V}_1$.

Note that any velocity vector, $\mathbf{V}$, can be decomposed in its radial, V_r, and tangential, V_θ, components:

$$\mathbf{V} = V_r \cdot \mathbf{i}_r + V_\theta \cdot \mathbf{i}_\theta \tag{7.2.24}$$

where $\mathbf{i}_r$ and $\mathbf{i}_\theta$ are the radial and tangential unit vectors, respectively. The velocity decomposition in radial and tangential components for both the departure and arrival velocities is illustrated in Figure 7.3.

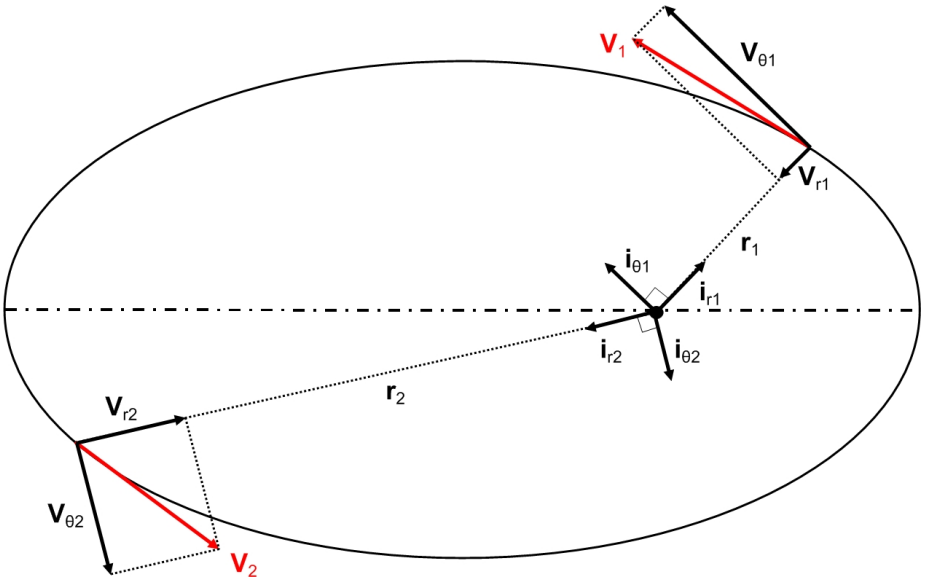

Figure 7.3: Radial and tangential components of the velocity vectors along an ellipse. Note that the radial components are negative in the situation depicted above.

When the velocity formulation assumes this form, the equation for the angular momentum, $\mathbf{H}$, which is constant along a given conic, becomes:

$$\mathbf{H} = \mathbf{r} \times \mathbf{V} = r \cdot \mathbf{i}_r \times (V_r \cdot \mathbf{i}_r + V_\theta \cdot \mathbf{i}_\theta) = rV_\theta \cdot \mathbf{i}_h = \text{constant}$$

where $\mathbf{i}_h$ is the angular momentum unit vector. This leads to the following, simple relation between the initial and final tangential velocities:

$$H = \text{constant} = r_1 V_{\theta 1} = r_2 V_{\theta 2} \tag{7.2.25}$$

which is equivalent to:

$$V_{\theta 2} = \frac{r_1 V_{\theta 1}}{r_2} \tag{7.2.26}$$

The relation between the initial and final radial velocity components is more complex to obtain. The hodograph analysis of a Keplerian orbit reveals that:

$$V_r = \frac{dr}{dt} = \frac{\mathbf{r}\cdot\mathbf{V}}{r} = \frac{\sqrt{\mu}\sigma}{r} = \frac{He}{p}\sin f \tag{7.2.27}$$

where p is the semi-latus rectum and the parameter $\sigma = \mathbf{r}\cdot\mathbf{V}/\sqrt{\mu}$ is introduced. It can be shown that

$$H^2/\mu = p \tag{7.2.28}$$

Bearing in mind Equation (7.2.28) and that, with the notation at hand, $\theta = f_2 - f_1$, the sum of the radial velocity components is:

$$V_{r1} + V_{r2} = \frac{He}{p}\left[\sin f_1 + \sin(f_1+\theta)\right]$$

Applying the trigonometric identities (A.5.1) and (A.1.2)

$$V_{r1} + V_{r2} = \frac{2He}{p}\sin(f_1+\frac{1}{2}\theta)\cos\frac{1}{2}\theta = \frac{2He}{p}\sin(f_1+\frac{1}{2}\theta)\sin\frac{1}{2}\theta\cot\frac{1}{2}\theta$$

followed by a third trigonometric identity, namely (A.5.4), leads to

$$V_{r1} + V_{r2} = \frac{H}{p}\left[e\cos f_1 - e\cos(f_1+\theta)\right]\cot\frac{1}{2}\theta$$

Combining this equation with the equation of the conic

$$r = \frac{p}{1+e\cos f} \tag{7.2.29}$$

yields

$$V_{r1} + V_{r2} = \left[\frac{H}{r_1} - \frac{H}{r_2}\right]\cot\frac{1}{2}\theta$$

Finally, substitution of Equation (7.2.25) gives

$$V_{r1} + V_{r2} = \frac{V_{\theta 1} - V_{\theta 2}}{\tan\frac{1}{2}\theta}$$

Therefore, the radial and tangential components of $\mathbf{V}_2$ can be expressed as functions of the initial velocity components:

$$V_{r2} = \frac{V_{\theta 1} - V_{\theta 2}}{\tan\frac{1}{2}\theta} - V_{r1} \tag{7.2.30}$$

$$V_{\theta 2} = \frac{r_1 V_{\theta 1}}{r_2} \tag{7.2.31}$$

We now must derive a formulation for the initial velocity vector as a function of the solution x. Substituting Equations (7.2.27), (7.2.25) and (7.2.28) into Equation (7.2.24), we obtain the following expression:

$$\mathbf{V}_1 = \frac{\sqrt{\mu}\sigma_1}{r_1}\cdot\mathbf{i}_{r1} + \frac{\sqrt{\mu p}}{r_1}\cdot\mathbf{i}_{\theta 1} = \frac{\sqrt{\mu}}{r_1}\left[\sigma_1\cdot\mathbf{i}_{r1} + \sqrt{p}\cdot\mathbf{i}_{\theta 1}\right] \tag{7.2.32}$$

We must therefore find expressions for the semi-latus rectum, p, and for the σ parameter. It can be shown that:

$$p = \frac{2r_1 r_2 \sin^2 \frac{1}{2}\theta}{r_1 + r_2 - 2\sqrt{r_1 r_2}\cos\frac{1}{2}\theta\cos\psi}$$

Substituting Equations (7.2.5) and (7.2.7) and applying the trigonometric identity (A.2.1) results in the more convenient form:

$$p = \frac{r_1 r_2 \sin^2 \frac{1}{2}\theta}{a\sin^2\psi}$$

Introducing the parameter η, defined as:

$$\eta = \sqrt{\frac{\pm 2a\sin^2\psi}{s}} \tag{7.2.33}$$

where the same rule applies to the $\pm$ sign as that of the definition of β, and keeping in mind the definition of a_m from Equation (7.2.17), the equation for the semi-latus rectum becomes:

$$p = \frac{r_1 r_2}{a_m \eta^2}\sin^2\frac{1}{2}\theta \tag{7.2.34}$$

With respect to the parameter σ, it can be shown that:

$$\frac{\sigma_1}{\sqrt{p}}\sin\frac{1}{2}\theta = \cos\frac{1}{2}\theta - \sqrt{\frac{r_1}{r_2}}\cos\psi$$

which, combined with Equation (7.2.34), is equivalent to:

$$\sigma_1 = \frac{\sqrt{r_1 r_2}}{\sqrt{a_m}\eta}\left[\cos\frac{1}{2}\theta - \sqrt{\frac{r_1}{r_2}}\cos\psi\right] \tag{7.2.35}$$

Let λ be defined as:

$$\lambda s = \sqrt{r_1 r_2}\cos\frac{1}{2}\theta \tag{7.2.36}$$

It can be shown that:

$$\cos\psi = \lambda + x\eta \tag{7.2.37}$$

Substitution of Equations (7.2.36) and (7.2.37) into Equation (7.2.35) yields:

$$\sigma_1 = \frac{1}{\eta\sqrt{a_m}}\left[2\lambda a_m - r_1(\lambda + x\eta)\right] \tag{7.2.38}$$

Therefore, the radial and tangential components can be expressed as:

$$V_{r1} = \frac{1}{\eta}\sqrt{\frac{\mu}{a_m}}2\lambda\frac{a_m}{r_1} - (\lambda + x\eta) \tag{7.2.39}$$

$$V_{\theta 1} = \frac{1}{\eta}\sqrt{\frac{\mu}{a_m}}\sqrt{\frac{r_2}{r_1}}\sin\frac{1}{2}\theta \tag{7.2.40}$$

and the initial velocity along the orbit corresponding to the solution x can be written as:

$$\mathbf{V}_1 = \frac{1}{\eta}\sqrt{\frac{\mu}{a_m}}\left\{\left[2\lambda\frac{a_m}{r_1} - (\lambda + x\eta)\right]\cdot\mathbf{i}_{r1} + \sqrt{\frac{r_2}{r_1}}\sin\frac{1}{2}\theta\cdot\mathbf{i}_{\theta 1}\right\} \tag{7.2.41}$$

The derivations above are the mathematical description of the Lambert problem according to the Lagrange-Gauss equations. We will now proceed to describing an efficient method to solve these equations, more specifically Equations (7.2.14) and (7.2.23).

7.3 Izzo's Algorithm

In the scope of the Global Trajectory Optimization Problem (GTOP) database [ACT, 2011], Izzo developed a computationally fast Lambert Solver. This solver is based upon the Lagrange-Gauss solution to the problem, as described in the previous section. His approach has two major advantages with respect to other methods. First, the algorithm describes the general solution of the orbital boundary-value problem in terms of the variable

$$x_0 = \log\left(1 + \cos(\frac{\alpha}{2})\right) \tag{7.3.1}$$

which turns the graph of the time-of-flight into a straight line defined over the entire real space. As a result, convergence is granted in only a few iterations, independent of the problem configuration. This can be seen in Figure 7.4. Note the offset in the x-transformation, motivated by the fact that the logarithmic function is only defined for positive values. Also, note that the reduced number of iterations comes at the "price" of applying the logarithmic function, log. Second, once the shape of the orbit is determined, the velocities along that orbit at the initial and final points are computed in such a way that they do not show singularities for transfer angles approaching π. Furthermore, Izzo designed his algorithm to work with non-dimensional units, for computational purposes. Although no paper explaining this method was found, the algorithm described below is publicly available in its C++ and MATLAB implementations at [ACT, 2011].

7.3.1 Method

Given the vectors of the initial and final positions, $\mathbf{r}_1$ and $\mathbf{r}_2$ respectively, the specified time of flight, t, and the gravitational constant of the central body, μ, one can determine the solution to the Lambert problem, as seen in the previous section. Note that an additional parameter is needed to specify which of the two configurations, short or long-way, to consider. Izzo's method relies on non-dimensional parameters:

$$t = t/T \tag{7.3.2}$$
$$\mathbf{r}_1 = \mathbf{r}_1/R \tag{7.3.3}$$
$$\mathbf{r}_2 = \mathbf{r}_2/R \tag{7.3.4}$$

where the input parameters units are consistent and the normalizing values are:

$$R = \|\mathbf{r}_1\| \quad V = \sqrt{\frac{\mu}{R}} \quad T = R/V = \sqrt{\frac{R^3}{\mu}} \tag{7.3.5}$$

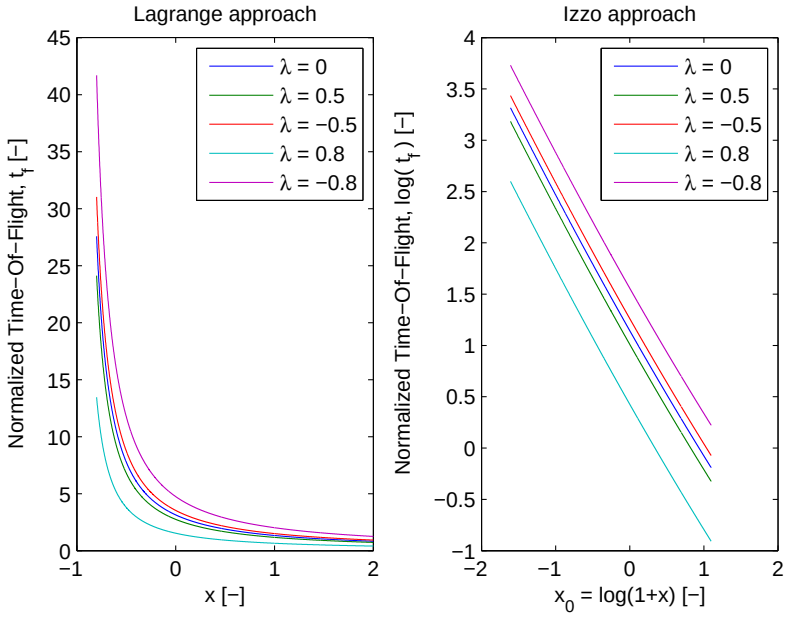

Figure 7.4: Time-of-flight graph for both Lagrange's and Izzo's method.

Note that according to these definitions, for the normalized initial radius, $\|\mathbf{r}_1\| = 1$. The short-way transfer angle[2] is computed by default:

$$\theta = \arccos\left(\frac{\mathbf{r_1} \cdot \mathbf{r_2}}{\|\mathbf{r_2}\|}\right) \tag{7.3.6}$$

Other necessary normalized constants are the chord, c, the semi-perimeter, s, the minimum energy ellipse semi-major axis, a_m, and the parameter λ. These constants can be computed according to:

$$c = \sqrt{1 + r_2^2 - 2r_2 \cos\theta} \tag{7.3.7}$$

$$s = (1 + r_2 + c)/2 \tag{7.3.8}$$

$$a_m = s/2 \tag{7.3.9}$$

$$\lambda = \sqrt{r_2}\cos(\theta/2)/s \tag{7.3.10}$$

which are the normalized versions of Equations (7.1.2), (7.2.9), (7.2.17) and (7.2.36).

The Izzo algorithm relies on iteratively solving the time-of-flight equation until it matches the specified value, t_{spec}, within a certain tolerance. The time-of-flight computations are based on Lagrange's expression, more precisely on its normalized counterpart. Bearing in mind the non-dimensional units defined in the set of Equations (7.3.5), the normalized time-of-flight equation is:

$$t_f = \sqrt{a^3}\left[(\alpha - \sin\alpha) - (\beta - \sin\beta)\right] \tag{7.3.11}$$

[2]For the long-way transfer angle, simply take the complementary angle.

As in the Lagrange-Gauss method, the parameters depend on whether the solution describes an ellipse or a hyperbola. The Lagrangian x-parameter can be simply derived from Equations (7.3.1) and (7.2.15):

$$x = \exp x_0 - 1$$

The semi-major axis of the solution, a, is computed with Equation (7.2.18). The equations for the parameters α and β for the ellipse, respectively Equations (7.2.19) and (7.2.20), as well as for the hyperbola, respectively Equations (7.2.21) and (7.2.22), still hold. The normalized time-of-flight equation for the hyperbola is:

$$t_f = \sqrt{-a^3}\left[(\sinh\alpha - \alpha) - (\sinh\beta - \beta)\right] \tag{7.3.12}$$

Finally we take the logarithm of Equation (7.3.11), such that the iterative process aims at solving:

$$y(x_0) = \log\left\{t_f(x_0)\right\} - \log t_{spec} = 0 \tag{7.3.13}$$

The root-finding process is done using the finite differences method, also known as the secant method.

7.3.2 The Secant Method

The secant method is quite simple and therefore computationally fast. It is an approximation of the well-known Newton-Raphson technique that does not require an analytical expression for the first function derivative:

$$\begin{aligned} x_{03} &= x_{02} - \frac{f(x_{02})}{f'(x_{02})} \approx x_{02} - f(x_{02})\left[\frac{f(x_{02}) - f(x_{01})}{x_{02} - x_{01}}\right]^{-1} \\ &= \frac{x_{01}f(x_{02}) - x_{02}f(x_{01})}{f(x_{02}) - f(x_{01})} \equiv \frac{x_{01}y_{02} - x_{02}y_{01}}{y_{02} - y_{01}} \end{aligned} \tag{7.3.14}$$

As we can see, this method requires two initial guesses for the variable x_0 (x_{01} and x_{02}) to compensate for the absence of an analytical derivative. These initial guesses are hard-coded to ensure that the initial guess is an ellipse, hence getting rid of the shape ambiguity for the first iteration. The values for these initial guesses are:

$$\begin{aligned} x_{01} &= \log(1 - .5233) \\ x_{02} &= \log(1 + .5233) \end{aligned}$$

Remember that applying the exponential logarithm to the time-of-flight function effectively smoothens the shape of the function and hence the secant approximation is valid, even with these relatively distant initial guesses. After each iteration the oldest values (x_{01},y_{01}) are discarded, the new values (x_{03},y_{03}) integrated for the next iteration and the middle values (x_{02},y_{02}) become the oldest values of the next iteration. This update method, where the old values are automatically outcasted, aims at improving the computational speed of the algorithm, as no evaluation of the quality of the old solution is made. Nonetheless, due to the smooth shape of the function, the algorithm will converge rapidly.

7.3.3 Velocity Computations

Once the root-finding process has converged, the next step consists of determining the associated conic and corresponding velocities at the initial and final points. Based on the final x_0 value, the semi-major axis and the parameters α and β are computed according to the shape of the transfer. The normalized radial and tangential components of the velocity at the point of departure are given by:

$$V_{r1} = \frac{2\lambda a_m - \lambda - x\eta}{\eta\sqrt{a_m}} \tag{7.3.15}$$

$$V_{\theta 1} = \sqrt{\frac{r_2}{\eta^2 a_m}\sin(\theta/2)} \tag{7.3.16}$$

where η is computed from Equation (7.2.33). The final velocity components can then be obtained according to Equations (7.2.30) and (7.2.31). Finally, the dimensional velocity vectors, along the conic at the initial and final points, $\mathbf{V}_1$ and $\mathbf{V}_2$ respectively, are intuitively obtained from the following equations:

$$\mathbf{V}_1 = (V_{r1} \cdot \mathbf{i}_{r1} + V_{\theta 1} \cdot \mathbf{i}_{\theta 1}) * V \tag{7.3.17}$$

$$\mathbf{V}_2 = (V_{r2} \cdot \mathbf{i}_{r2} + V_{\theta 2} \cdot \mathbf{i}_{\theta 2}) * V \tag{7.3.18}$$

where $\mathbf{i}_{r1,2}$ and $\mathbf{i}_{\theta 1,2}$ are the radial and tangential, respectively, unit vectors at the point of departure and arrival, and V reverts the velocities into dimensional vectors. Note that if the transfer angle is equal to π, the tangent directions cannot be determined and the velocity vectors cannot be computed, although the shape of the transfer orbit is known.

This Lambert solver, as implemented in the GTOP toolbox by Izzo, is available at [ACT, 2011].

7.4 Implementation in Tudat

The TUDelft Astrodynamics Toolbox (Tudat) is a C++ library developed and maintained by staff and students of the Astrodynamics and Satellite missions department at the TU Delft (see Chapter 8). Tudat already contains a Lambert solver, based on the algorithm developed by [Lancaster and Blanchard, 1969] with further improvements by [Gooding, 1990]. However, it has been suggested that Izzo's algorithm is extremely fast ([Oldenhuis, 2011], [Izzo, 2011]) although no documentation regarding benchmarking tests could be found. Izzo's algorithm was implemented in Tudat and the computational tests carried out, in terms of accuracy, robustness and speed, are reported in this Section.

7.4.1 Test cases

In order to validate the implemented Izzo algorithm, two test cases were set up. The first test case, Lambert Problem Test Case 1 (LPTC1), consists of finding the solution to the Lambert Problem with randomly generated values for the radii, time-of-flight and gravitational parameter. The reference values are those produced by the same values using the GTOP implementation. The second test case, Lambert Problem Test Case 2 (LPTC2), consists of computing the solution

to the Lambert Problem for a circular, co-planar Earth-Mars transfer under varying orbital configuration and time-of-flight. For LPTC2, the reference values are computed with the Lambert targeting routine in PyKEP, a scientific library providing basic tools for astrodynamics research, also developed at ESA/ACT. This routine is another (more robust) implementation of Izzo's algorithm.

Test Case 1

As mentioned earlier, Lambert Problem Test Case 1 (LPTC1) consists of solving Lambert Problems for which the variables are randomly obtained. The values for the Cartesian velocity components and for the gravitational parameter are allowed to vary in the interval $[0, 100]$, while the value for the time-of-flight can vary within the interval $[0, 1000]$. For all these parameters, a uniform real distribution is assumed. Note that the intervals defined above do not relate to any realistic transfer, and therefore it is assumed that the dimensions across the different variables are consistent. For each run, 100 random problems are solved.

This test case was setup to rapidly assess the robustness and accuracy of the implemented algorithm. The solutions to these random Lambert Problems were compared against those obtained with the GTOP Lambert targeting routine. For each run, the absolute difference in the Cartesian components of the initial and final velocities as well as in the norm of those vectors is computed. The maximum and mean absolute differences for each quantity were computed for three consecutive runs of LPTC1. The mean of each type of error was taken and is listed in Table 7.1 below, where V_x, V_y and V_z, refer to the x-, y- and z-components, respectively, of the Cartesian velocity vectors, and the indices *1* and *2* refer to the initial and final velocities, respectively. The errors for the individual runs are listed in Table B.1.

Quantity		Average
$V_{1,x}$	Maximum	$3.066 \cdot 10^{-7}$
	Mean	$1.769 \cdot 10^{-8}$
$V_{1,y}$	Maximum	$7.693 \cdot 10^{-7}$
	Mean	$2.404 \cdot 10^{-8}$
$V_{1,z}$	Maximum	$5.350 \cdot 10^{-7}$
	Mean	$2.031 \cdot 10^{-8}$
V_1	Maximum	$1.001 \cdot 10^{-6}$
	Mean	$4.005 \cdot 10^{-8}$
$V_{2,x}$	Maximum	$7.075 \cdot 10^{-7}$
	Mean	$2.470 \cdot 10^{-8}$
$V_{2,y}$	Maximum	$4.854 \cdot 10^{-7}$
	Mean	$2.025 \cdot 10^{-8}$
$V_{2,z}$	Maximum	$4.451 \cdot 10^{-7}$
	Mean	$1.793 \cdot 10^{-8}$
V_2	Maximum	$9.993 \cdot 10^{-7}$
	Mean	$4.032 \cdot 10^{-8}$

Table 7.1: Average, of three independent runs, of the maximum and mean absolute velocity errors for LPTC1.

Looking at Table 7.1, one can see that the results obtained with the algorithm implemented in Tudat are very close to those found with the GTOP implementation. Roughly, the maximum errors have an order of magnitude of 10^{-7} while the mean errors have an order of magnitude of 10^{-8}. Note that the largest errors occur for the maximum errors in the norm of the velocity vectors. This is due to the fact that the velocity norm reflects the errors from all three Cartesian components. For interplanetary Lambert problems, the dimension of the norm for the initial and final velocities is about 10^3 m/s, which means that the errors between both implementations would be at most in the order of mm/s. Given the low order of magnitude of both the maximum and mean errors, one can state that, at first glance, the proposed implementation is quite accurate and robust.

These preliminary results are quite encouraging but a more realistic test case, with variables of physical meaning, needs to be defined and tested.

Test Case 2

As previously stated, Lambert Problem Test Case 2 (LPTC2) consists of a circular, co-planar Earth-Mars transfer under varying celestial geometry and time-of-flight. The circular, co-planar transfer indicates that the eccentricity and the inclination of both Earth's and Mars' orbits are taken to be zero. This implies that the velocity components along the z-axis are constant and null. Therefore, no analysis on the accuracy of said component is given in this report. In this case, the celestial geometry, or configuration, can be boiled down to the transfer angle, θ. In order to vary the transfer angle, the mean anomaly of Earth was set to zero, while that of Mars varies from 1 to 360 degrees, with a step-size of 1 degree. This approach is illustrated in Figure 7.5, where it can be seen that Earth, denoted E, remains static while the mean anomaly, θ, of Mars, denoted M, varies. Note that $\theta_M = 180°$, corresponding to a π-transfer, was replaced by a near π-transfer, namely $\theta_M = 179.999...°$. The value for the time-of-flight is made to vary from 200 to 1000 days with a step-size of 50 days. All the relevant parameters for LPTC2 can be seen in Table B.2. Due to the varying parameters, LPTC2 entails a total of 6120 different problems. A prograde motion is assumed in all of them, meaning that both short- and long-way transfers are contemplated in LPTC2.

For LPTC2, the reference values are obtained with the Lambert routine in PyKEP[3], which is another, more sophisticated implementation of Izzo's algorithm. This algorithm has been thoroughly verified and validated [Izzo, 2011]. LPTC2 is the reference test case for the following sections.

7.4.2 Accuracy of the Implemented Algorithm

The first step taken towards the validation of the implemented Lambert routine, is to verify its accuracy. In order to do so, the algorithm is tested in the scope of LPTC2. The absolute errors, in meters per second, in terms of the x- and y-components of the velocity vectors, as well as in terms of their norm, are given in Figures 7.6 to 7.11. In those figures, we can see that the areas of the problem space where the largest discrepancies appear follow a clear and consistent

[3]PyKEP documentation website: `http://keptoolbox.sourceforge.net/`

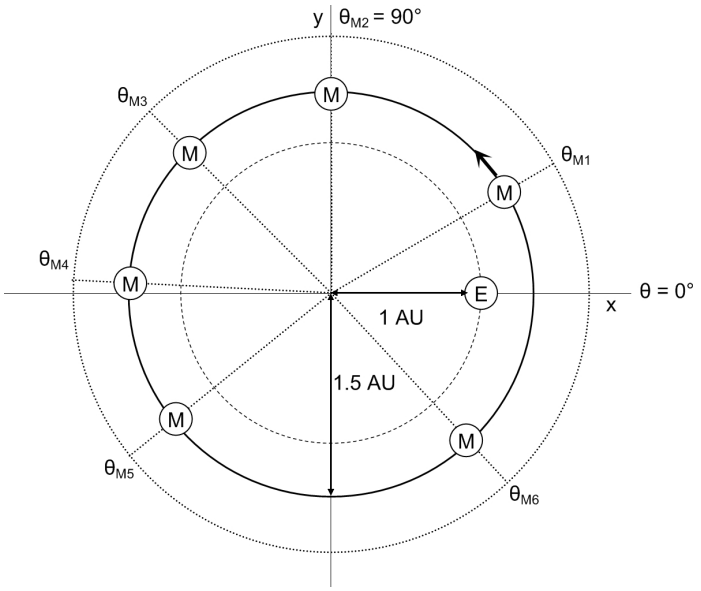

Figure 7.5: The varying heliocentric configuration in LPTC2.

pattern across all analyzed quantities. These seem to indicate Lambert problems that the implemented algorithm struggles to accurately solve. In order to gain more insight into these problems, let us take a look at Figure 7.12, where the number of iterations needed by the secant method to solve the Lambert problems is depicted.

Looking at Figure 7.12, it is apparent that the maximum number of root-finding iterations needed to solve each of the Lambert problems from LPTC2 is six, while the minimum is three. Moreover, there is little variation in terms of the number of iterations, since the minimum number of variations seldom occurs and the majority of the problems is solved within 5 iterations. Cross referencing this figure with the previous ones, the same pattern as the one seen in the previous plots emerges. It appears that the largest errors occur near the transitions to a higher number of iterations, namely from 4 (in yellow in Figure 7.12) to 5 (in orange) iterations and from 5 to 6 (in red) iterations. These transitions seem to highlight thresholds in terms of the difficulty of the problems, meaning that at these thresholds, the algorithm converged too rapidly. It is hypothesized that these regions correspond to the time-of-flight graphs with a large gradient, meaning that a relatively small difference in abscissae corresponds to a relatively large difference in ordinate, therefore driving the root-finding process to a premature end. Increasing the convergence tolerance does not yield any significant improvement in the quality of the results, as we will show in Section 7.4.3.

Nonetheless, we can see that the majority of the problem space is solved with (very) high accuracy. Even for the problem with the largest errors, the maximum error committed is in the order of 10^{-5} meters per second, which is certainly accurate enough for the purposes of a Lambert solver. To illustrate the overall accuracy of the implemented algorithm, Table 7.2 gathers the maximum

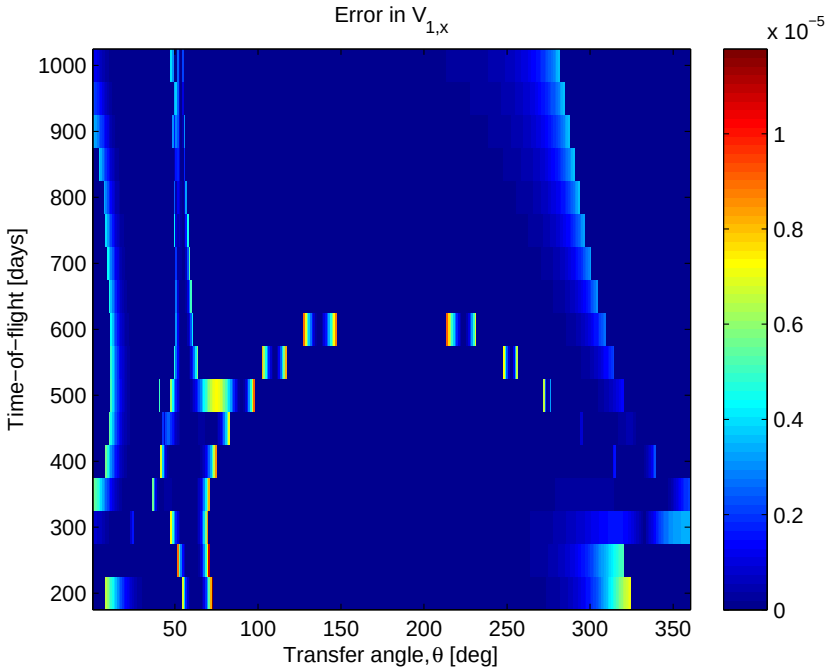

Figure 7.6: LPTC2: The absolute error in $V_{1,x}$ in meters per second.

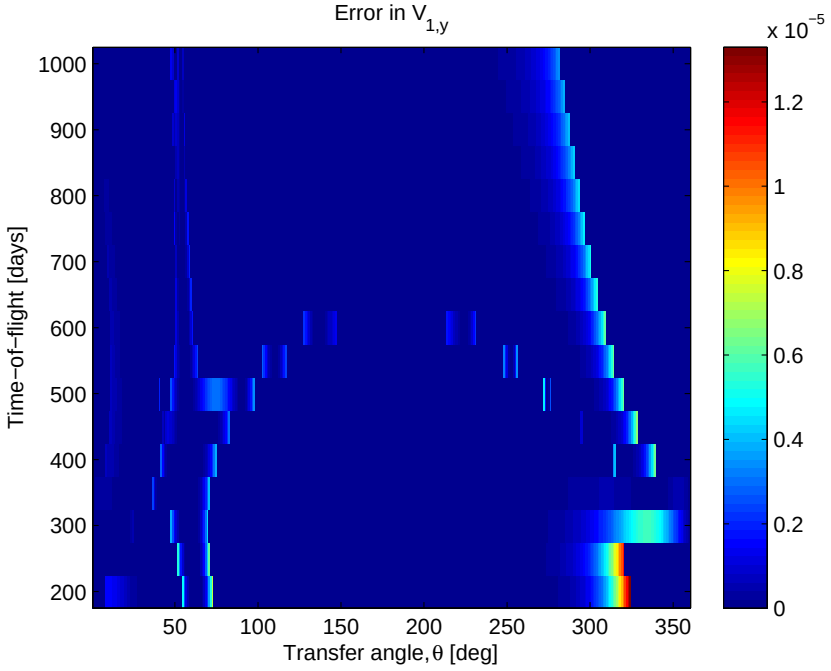

Figure 7.7: Test Case 2: The absolute error in $V_{1,y}$ in meters per second.

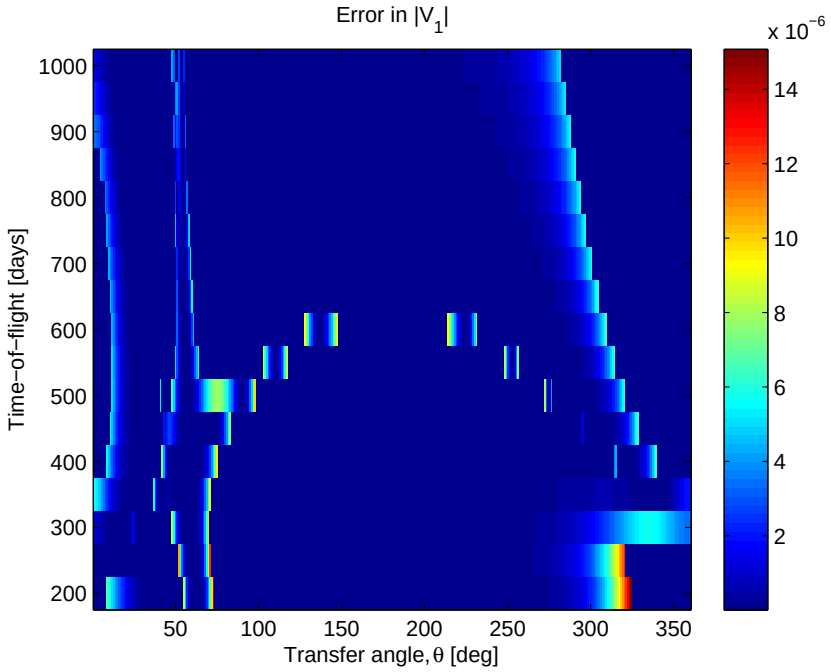

Figure 7.8: Test Case 2: The absolute error in $\|V_1\|$ in meters per second.

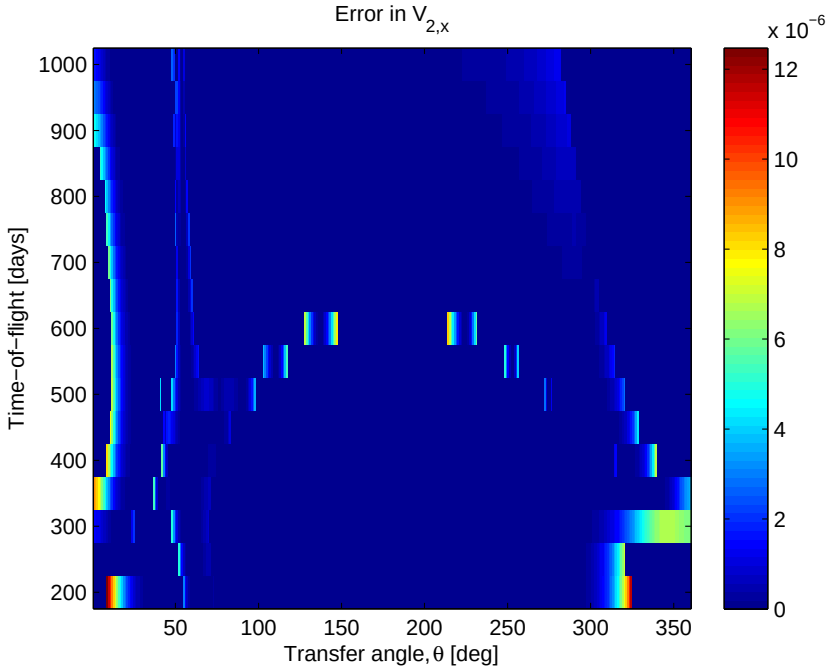

Figure 7.9: Test Case 2: The absolute error in $V_{2,x}$ in meters per second.

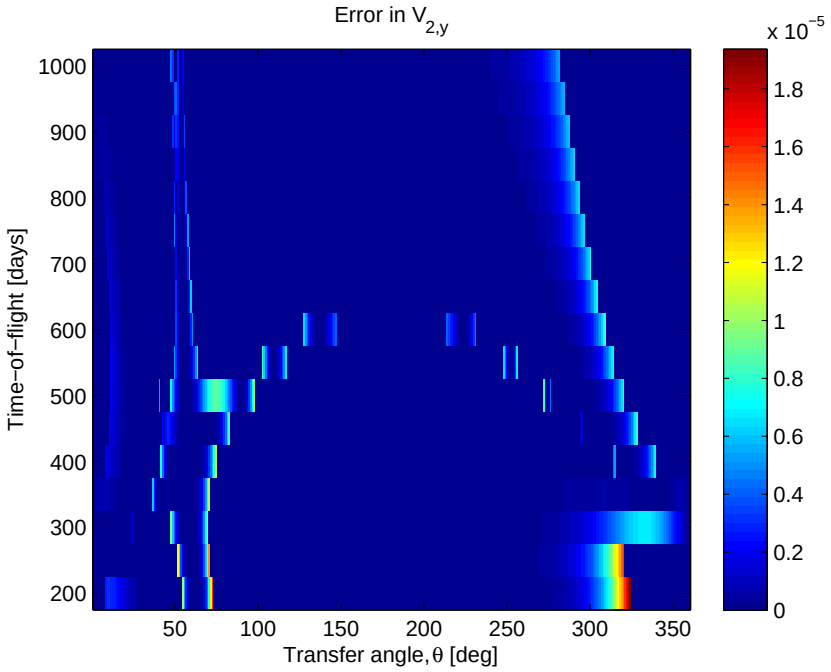

Figure 7.10: Test Case 2: The absolute error in $V_{2,y}$ in meters per second.

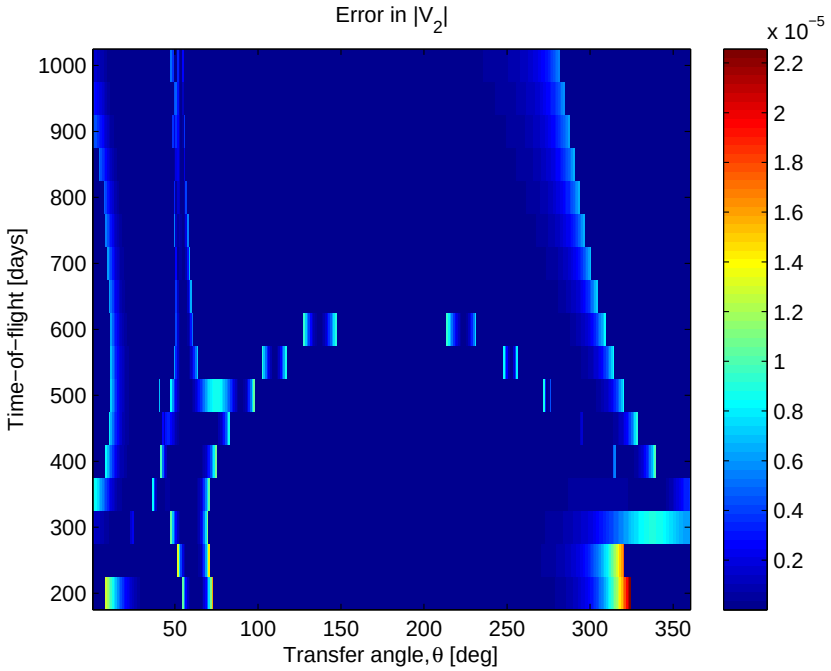

Figure 7.11: Test Case 2: The absolute error in $\|V_2\|$ in meters per second.

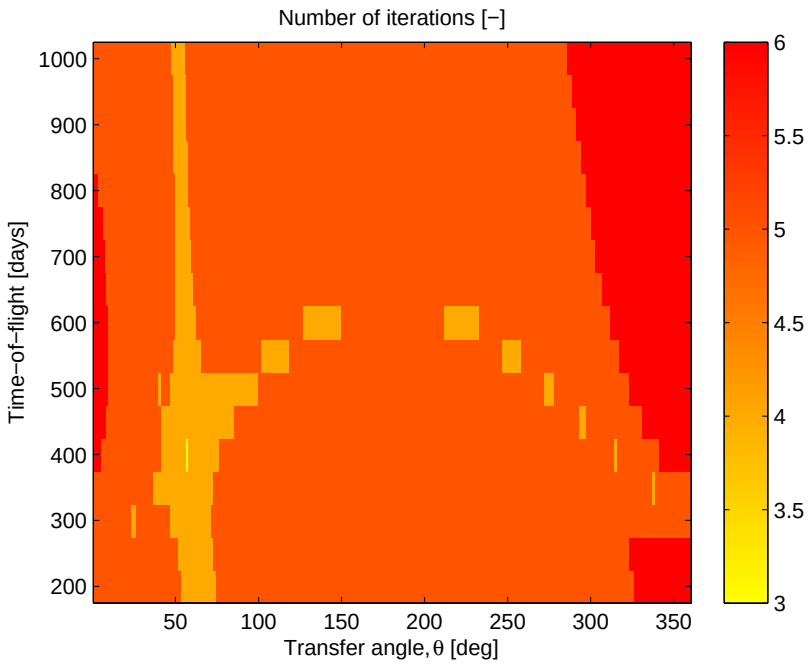

Figure 7.12: Test Case 2: The number of root-finding iterations.

and the RMS errors in all quantities analyzed, namely the norm and the x- and y-components of the initial and final velocities. The maximum and RMS number of root-finding iterations is also given. The notation is the same as in Table 7.1.

Quantity	Maximum [m/s]	RMS [m/s]
$V_{1,x}$	$1.178 \cdot 10^{-5}$	$1.058 \cdot 10^{-6}$
$V_{1,y}$	$1.329 \cdot 10^{-5}$	$9.248 \cdot 10^{-7}$
V_1	$1.508 \cdot 10^{-5}$	$1.405 \cdot 10^{-6}$
$V_{2,x}$	$1.247 \cdot 10^{-5}$	$9.984 \cdot 10^{-7}$
$V_{2,y}$	$1.937 \cdot 10^{-5}$	$1.506 \cdot 10^{-6}$
V_2	$2.256 \cdot 10^{-5}$	$1.807 \cdot 10^{-6}$
# Iterations	6	5.087

Table 7.2: LPTC2: Maximum and RMS errors and number of iterations.

While, as previously mentioned, the maximum errors produced by the implemented algorithm are in the order of 10 micrometers per second, the RMS errors are closer to 1 μm/s, which is a very accurate result. Akin the results from Table 7.1, the largest errors occur for the norm of the velocity vectors. The explanation for this phenomenon is the same, namely the fact that the velocity norm error reflects the errors in all its separate components. With respect to the number of root-finding iterations, the algorithm needs on average 5 iterations

to solve a problem, as hinted by the dominance of the colour orange in Figure 7.12.

We can therefore conclude that the implemented Lambert solver is very accurate and solves a wide range of problems within a fairly constant number of iterations.

7.4.3 Sensitivity Analysis

It can be argued that, for the purposes of the MSc thesis, a micrometer-per-second-accurate Lambert solver by far exceeds the accuracy requirements. Therefore, a sensitivity analysis was conducted to determine the influence of the root-finding convergence tolerance on both the number of iterations and the accuracy of the results.

The convergence tolerance is a criterion that can be defined as a threshold for the difference between two succesive solutions, $\epsilon = |x_{i+1} - x_i|$, beyond which the root-finding process terminates. In the analysis conducted in the previous section, the convergence tolerance was set to $\epsilon = 10^{-9}$. Note that this is the same convergence tolerance as in the PyKEP implementation of the algorithm. For the sensitivity analysis, LPTC2 was run with convergence tolerances of 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 10^{-9} and 10^{-12}. The results of the sensitivity analysis are summarized below. The extensive results can be found in Appendix B.2.3.

With respect to the accuracy of the results, Figure 7.13 details the maximum and RMS errors in the velocity components and norms as a function of the convergence tolerance. We can see that decreasing the convergence tolerance beyond 10^{-6} does not lead to any significant increase in the quality of the results. This value either represents the maximum accuracy achievable with the implemented algorithm or indicates some sort of machine limitation due to, perhaps, the machine precision. Either way, the maximum accuracy achievable is of the order of 10 μm/s for the maximum error and 1 μm/s for the RMS error. These values are the same as those found in the previous section since the 10^{-9} convergence tolerance is evidently smaller than the 10^{-6} convergence threshold. For convergence tolerance larger than 10^{-6}, the accuracy of the results decreases. For the largest convergence tolerance tested, the maximum errors reach 1 m/s while the RMS errors have an order of magnitude of about 0.1 m/s. These values are still acceptable given that Lambert problems usually involve velocities with magnitudes of km/s.

In Figure 7.14, the maximum and RMS number of iterations as a function of the convergence tolerance are plotted. It is clear that the RMS number of iterations continuously decreases with the increase in convergence tolerance. The same, however, cannot be said for the maximum number of iterations as we can see a step at 5 iterations for 10^{-6} through 10^{-4} convergence tolerances[4]. Note however that the maximum number of iterations is a discrete function, unlike its RMS counterpart, and that the step occurs in a more densely sampled region of the abscissae. Despite not decreasing continuously with the increase of the convergence tolerance, a decreasing trend can be distinguished for the maximum number of iterations. Based on Figure 7.14, we can state that increasing the convergence tolerance leads to a decrease in the RMS number of iterations up to approximately 2 iterations. It is expected that decreasing the number of

[4]Note the logarithmic scale of the abcissae.

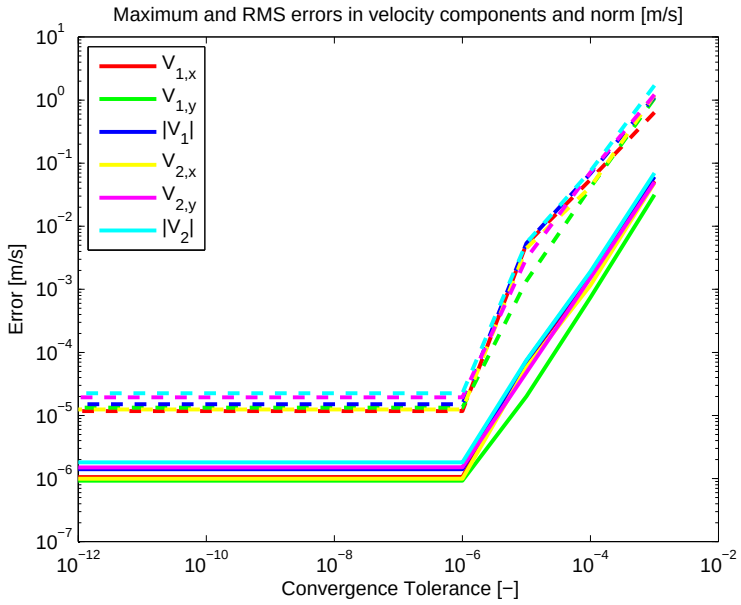

Figure 7.13: RMS (continuous lines) and maximum (dashed) errors as a function of the convergence tolerance.

iterations required to solve the Lambert problem will lead to a decrease in the computational time needed to achieve results, specially in cases where a large number of solutions needs to be determined.

Based on the sensitivity analysis, it is decided to set the convergence tolerance of the implemented Lambert routine to $\epsilon = 10^{-5}$. At this convergence tolerance, the RMS errors in the velocity results are within 10 μm/s, while the maximum errors level at 0.01 m/s. These values instill confidence that the velocity results will be very accurate under any circumstance. Moreover, both the maximum and the RMS number of root-finding iterations decrease by 1 iteration with respect to the nominal algorithm (i.e. the implemented algorithm with a convergence tolerance of $\epsilon = 10^{-9}$).

7.4.4 Comparison with the Lancaster & Blanchard Algorithm

As mentioned at the beginning of Section 7.4, Tudat already encompasses a Lambert routine, based on the algorithm developed [Lancaster and Blanchard, 1969] and further improved by [Gooding, 1990], from here on referred to as LBG. For a description of this algorithm, please refer to those references. It is expected however that the implemented Izzo algorithm, from here on referred to as IZZO, will be more efficient than the native LBG. In order to verify this assumption, both algorithms are compared in terms of accuracy, number of iterations and computational speed, based on LPTC2.

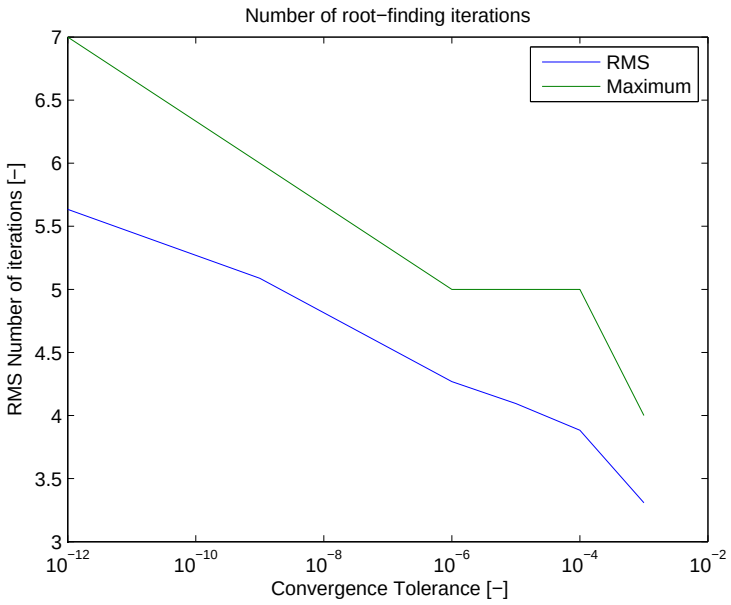

Figure 7.14: Maximum and RMS number of iterations as a function of the convergence tolerance.

Accuracy

In order to compare the accuracy of IZZO and LBG, both algorithms are applied to LPTC2. The results are illustrated in Figures 7.15 through 7.20. The maximum and RMS errors are detailed in Table 7.3. Bear in mind that for IZZO, the convergence tolerance is set to $\epsilon = 10^{-5}$. A sensitivity analysis for LBG was performed and the tolerance of its root-finding routine tuned to produce results with an accuracy in the same order of magnitude as IZZO with $\epsilon = 10^{-5}$.

Based on Figures 7.15 to 7.20, we can state that, in general, LBG has a better accuracy than IZZO. It is interesting to note that the error pattern is similar for both LBG and IZZO, signaling Lambert problems that are difficult to solve for both algorithms. This indicates that the errors discussed in Section 7.4.2 are not implementation-specific. Also of note, is the red line at $\theta = 180°$[5], visible in all LBG plots, which indicates that the LBG algorithm suffers from a near π-transfer singularity. IZZO, in contrast, does not seem to be afflicted by that particular singularity.

Table 7.3 confirms the previous statement regarding the accuracy of both algorithms. With the exception of the y-component of the initial velocity vector, all maximum and RMS errors are larger for IZZO than LBG. However, the RMS errors for the norm of the velocity vectors have the same order of magnitude for both algorithms. Therefore, despite being more accurate, LBG is not significantly more precise than IZZO. To support this statement, note that the

[5] As discussed in Section 7.4.1, the π-transfer case is not analyzed as there is no analytical solution. It is replaced by a near π-transfer.

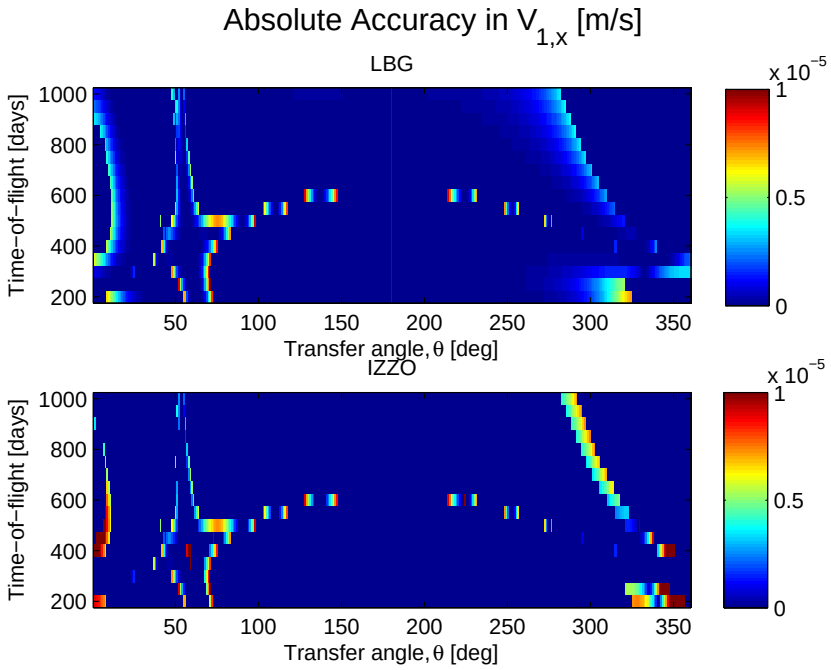

Figure 7.15: LPTC2: Comparison between the absolute errors in $V_{1,x}$.

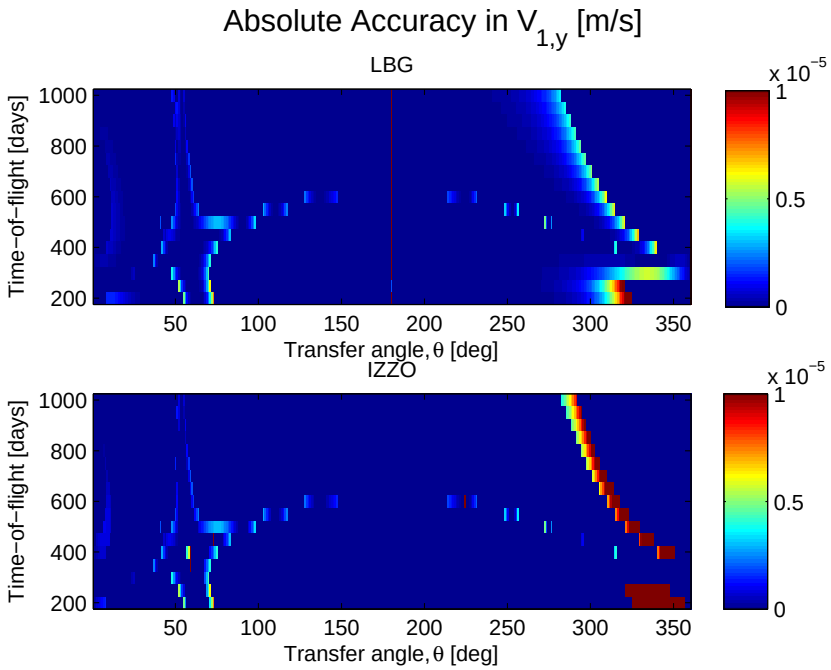

Figure 7.16: LPTC2: Comparison between the absolute errors in $V_{1,y}$.

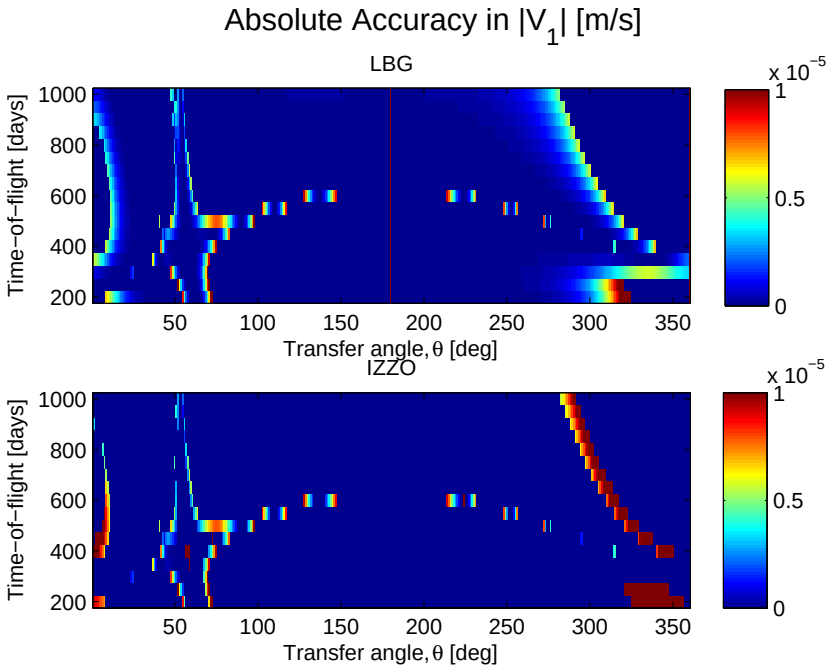

Figure 7.17: LPTC2: Comparison between the absolute errors in $\|V_1\|$.

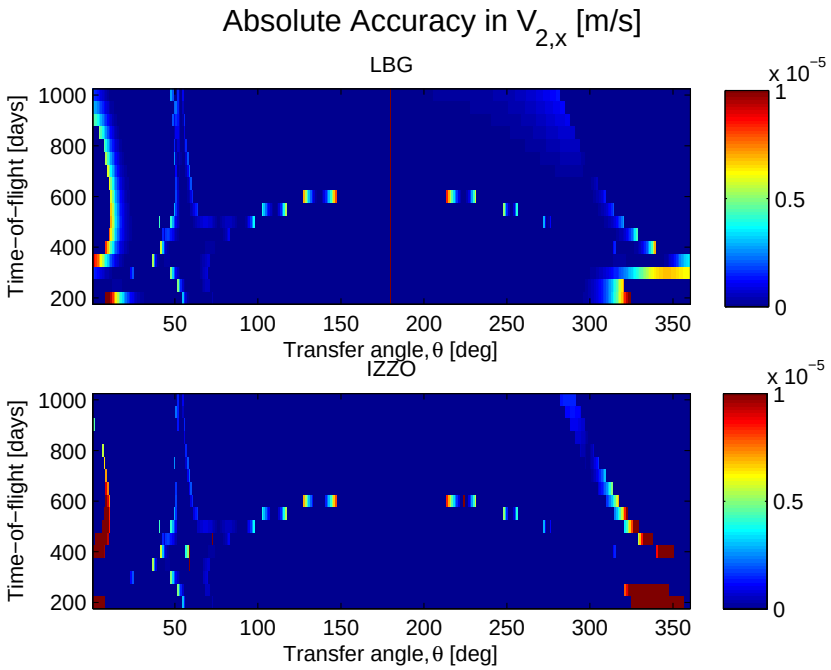

Figure 7.18: LPTC2: Comparison between the absolute errors in $V_{2,x}$.

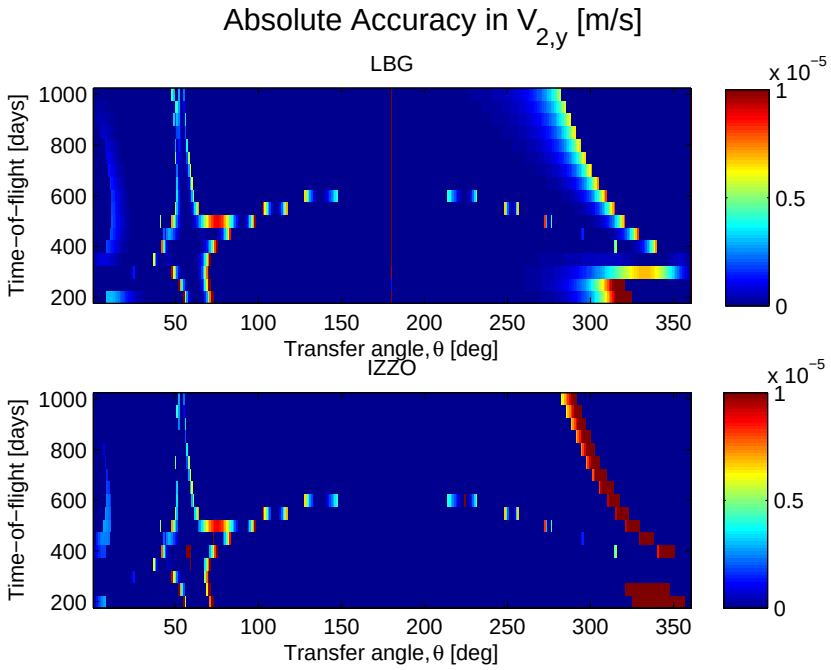

Figure 7.19: LPTC2: Comparison between the absolute errors in $V_{2,y}$.

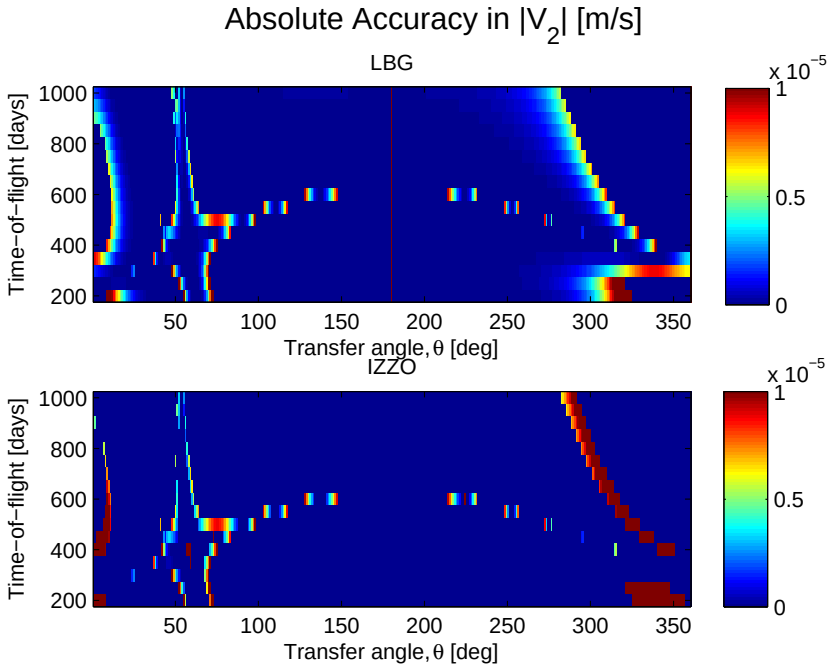

Figure 7.20: LPTC2: Comparison between the absolute errors in $\|V_2\|$.

Quantity	Maximum Error [m/s]		RMS Error [m/s]	
	IZZO	LBG	IZZO	LBG
$V_{1,x}$	$5.209 \cdot 10^{-3}$	$1.458 \cdot 10^{-4}$	$7.019 \cdot 10^{-5}$	$7.759 \cdot 10^{-6}$
$V_{1,y}$	$1.358 \cdot 10^{-3}$	$1.422 \cdot 10^{-3}$	$2.017 \cdot 10^{-5}$	$6.549 \cdot 10^{-5}$
V_1	$5.384 \cdot 10^{-3}$	$1.422 \cdot 10^{-3}$	$7.303 \cdot 10^{-5}$	$6.594 \cdot 10^{-5}$
$V_{2,x}$	$4.508 \cdot 10^{-3}$	$6.486 \cdot 10^{-5}$	$5.785 \cdot 10^{-5}$	$3.558 \cdot 10^{-6}$
$V_{2,y}$	$3.094 \cdot 10^{-3}$	$9.480 \cdot 10^{-4}$	$4.821 \cdot 10^{-5}$	$4.368 \cdot 10^{-5}$
V_2	$5.468 \cdot 10^{-3}$	$9.480 \cdot 10^{-4}$	$7.531 \cdot 10^{-5}$	$4.382 \cdot 10^{-5}$

Table 7.3: LPTC2: Comparison between the maximum and RMS errors.

maximum errors for IZZO have a magnitude of milimeters per second. Although LBG maximum errors are lower, its improved accuracy has little consequences, in practice.

Number of Iterations

Now that we have determined that the differences in accuracy between LBG and IZZO have no significant practical impact, let us take a look at the number of iterations. Bear in mind that one of the benefits of IZZO is the linear time-of-flight graph which is supposed to reduce the number of iterations needed for the root-finding method to converge. The number of iterations needed by both algorithms to solve all problems in LPTC2 are depicted in Figure 7.21.

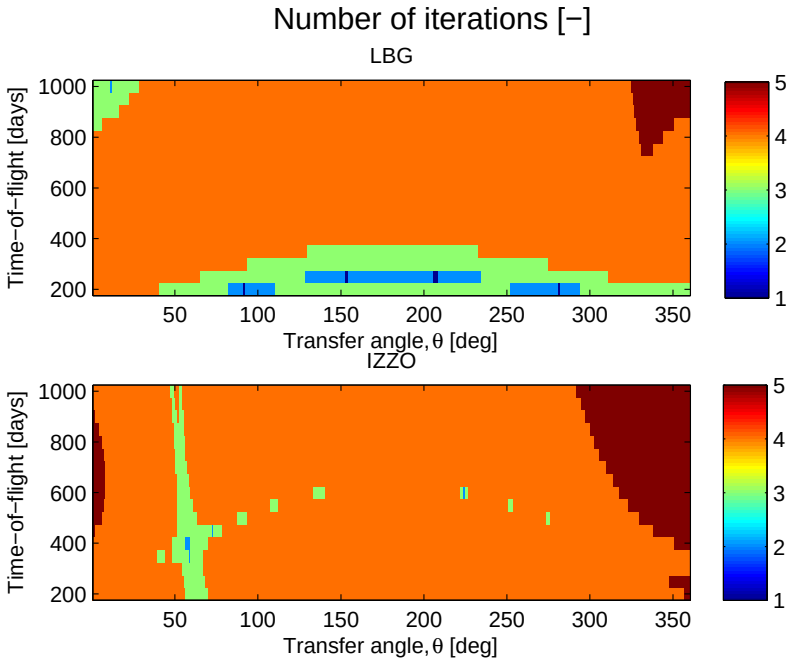

Figure 7.21: LPTC2: Comparison between the number of iterations.

The first observation we can make when looking at Figure 7.21 is that, overall, IZZO and LBG require roughly the same number of root-finding iterations. However, this is not true for the entire search space. We note that, for short transfer times (between 200 and 300 days) in combination with transfer angles between 90 and 290 degrees, LBG is capable of solving some Lambert problems within 1 or 2 iterations. However, the majority of the problems is solved in 4 iterations, as is the case for IZZO. Looking at the IZZO plot, it becomes apparent that the algorithm converges quickly (within 3 iterations) for transfer angles around 60°. These observations are confirmed when looking at Table 7.4. The maximum number of iterations is the same for both algorithms, the RMS number of iterations for IZZO being slightly larger than for LBG. Both RMS values are about 4 iterations.

	Maximum [-]		RMS [-]	
Algorithm	IZZO	LBG	IZZO	LBG
# of Iterations	5	5	4.0941	3.8753

Table 7.4: LPTC2: Comparison between the maximum and RMS number of iterations.

Computational Speed

Having established that the IZZO and LBG algorithms have roughly the same performance in terms of accuracy and number of iterations (with a slight advantage for LBG), let us take a look at the computational speed required to solve LPTC2. In Figure 7.22, the Central Processing Unit (CPU) time, in seconds, required by each algorithm to solve each problem is given. In order to circumvent the limited precision of the clock used to time the algorithms, each combination of time-of-flight and transfer angles was repeated a thousand times. After that, the total amount time is divided by 1000 to reflect the computational time needed to solve a single problem. The computational times correspond to the elapsed time needed to solve a particular Lambert problem on the author's personal laptop.

Figure 7.22 shows that IZZO is faster than LBG for a vast majority of the problems, up to twice as fast apparently. It seems that the average time needed to solve a given problem is about 5 μs for IZZO while it is approximately 10 μs for LBG. Moreover, we notice that LBG seems to have a wider range of values for the time needed to solve the problems over all the LPTC2 problem space than IZZO. More detailed information can be obtained by inspecting Table 7.5, where the maximum, RMS and standard deviation, σ, for the CPU times of both algorithms are detailed.

	Maximum [μs]		RMS [μs]		σ [μs]	
Algorithm	IZZO	LBG	IZZO	LBG	IZZO	LBG
CPU time	35	18	6.477	11.352	1.087	1.402

Table 7.5: LPTC2: Comparison between the maximum, RMS and standard deviation CPU times.

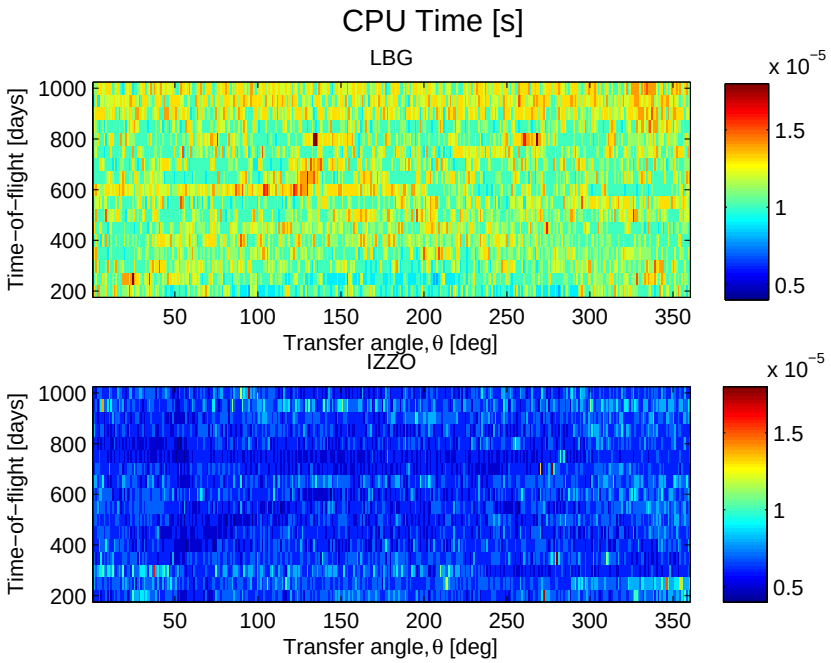

Figure 7.22: LPTC2: Comparison between the CPU times.

The maximum CPU time needed to solve a Lambert problem is larger for IZZO than for LBG. Nonetheless, the RMS values show that IZZO is, on average, about twice as fast as LBG, meaning that the larger maximum value probably represents a rather isolated case. This is confirmed when looking at the standard deviation, σ: the standard deviation is smaller for IZZO, meaning that over the entire problem space, the CPU time needed to solve a particular problem fluctuates less for IZZO than for LBG. Therefore, we can state that IZZO is typically faster over the entire problem space.

In order to represent these computational times in a more familiar order of magnitude, let us take a look at Table 7.6. Remember that, in order to circumvent the limited precision of the clock used to time the algorithms, each of the 6120 unique Lambert problems in LPTC2 was repeated a thousand times in each run. Table 7.6 shows the CPU time for three of those runs for both algorithms, as well as the corresponding RMS values. Additionally, the resulting computational speed, in problems per second, is shown. Finally, the last column shows the relative computational speed.

This table confirms the conclusions drawn previously, namely that IZZO is about twice as fast as LBG. Looking at the RMS values, IZZO is 1.86 times faster than LBG. On average, IZZO solves about 180,000 Lambert problems per second.

Based on the results of the comparison between IZZO and LBG, IZZO seems to be the most appropriate Lambert routine to be employed in the scope of the MSc Thesis, mainly due to the fact that it is considerably faster.

Algorithm	CPU Time [s]		CPU Speed [problems/s]		
	IZZO	LBG	IZZO	LBG	IZZO/LBG [-]
Run #1	36.542	65.416	167478	93555	1.79
Run #2	33.824	63.908	180936	95762	1.89
Run #2	32.607	62.367	187689	98128	1.91
RMS	34.364	63.909	178899	95833	1.86

Table 7.6: LPTC2: Comparison between the CPU times and speeds.

Chapter 8

Implementation

The building blocks described in the previous chapters (multi-path NNH, GS, and Lambert Targeter) were developed, implemented and assembled in the framework of TUDelft Astrodynamics Toolbox (Tudat), a C++ library developed internally at the TU Delft.

We take a look, in Section 8.1, at Tudat, its design philosophy and its project setup. We then present the block diagram of the assembled tool in Section 8.2, highlighting the code blocks that were developed, integrated or simply used.

8.1 Tudat

Tudat is a C++ library for astrodynamics simulations, developed and maintained by staff and students of the Astrodynamics and Space Missions (A&S) department of the Aerospace Engineering faculty at the TU Delft. The genesis of Tudat rises from the need, within the A&S group, for a generic, collaboratively developed astrodynamics toolbox.

Figure 8.1: The Tudat logo [Kumar, 2011].

The currently available software (e.g. STK, GALOMUSIT, OPTIDUS) has a steep learning curve and is rather fragmented, each software being tailored to a specific scope and a specific set of applications. Furthermore, some of these software require licenses which often limit the accessibility. The wide range of

applications within Tudat (interplanetary trajectories, launcher ascent trajectories, vehicle re-entry, exoplanet orbits, amongst others) offers the potential for conducting large-scale projects with a single tool.

As mentioned above, Tudat is collaboratively developed by MSc students within the scope of their respective theses, under the supervision of A&S PhD students and staff members. This collaborative aspect ensures that MSc students working with Tudat need not to reinvent the wheel, as they can integrate readily available features and focus on the novel features they wish to implement. As a result, Tudat is not a static software as each MSc student adds new features to the toolbox. Moreover, feature development for Tudat promotes the communication, cooperation and coordination within the A&S group members, both students and staff, and encourages the development of programming skills.

The high-level requirements of Tudat translate into technical and management requirements, namely in terms of project setup and software architecture. These are briefly described hereafter, along with a short description of the available features. For more details, please refer to [Kumar et al., 2012].

8.1.1 Design Philosophy

Different applications often possess common elements. For these features to be easily applied in different context, the software architecture must be modular and written in a generic fashion. To represent this modular aspect, Tudat can be thought of as a series of blocks that can be arranged in several configurations, depending on the respective application. Therefore, developers can gather the already available, independent building blocks and combine them with the new necessary blocks to set up their respective application. Specifically, Tudat code blocks correspond to generic elements of astrodynamic simulations such as bodies, environment and force models, amongst others.

In practice, this modularity in the software architecture is difficult to reach. To achieve it, Tudat makes use of two important characteristic of the object-oriented programming languages C++: *polymorphism* and *inheritance*. Inheritance can be seen as the top-down propagation of functionality between derived classes: child classes derive a number of properties from their parent class. In practice, the interaction between the different code blocks in Tudat is done on the overall parent class level, i.e. the interface is generic. The child classes inherit this interface while implementing specific functionalities.

Other software architecture options derive from the need to maintain the coding readability, coherence and robustness: a conservative stance is taken towards external libraries, there is a clear categorization of the code (e.g. mathematics, astrodynamics), standard file formats exist for input and output handling and the standardization of units and reference systems occurs internally. The ability of the child classes to have a unique interface while taking different forms is referred to as polymorphism and is an important aspect of Tudat's software architecture.

8.1.2 Project Setup

Tudat is split into two libraries: Tudat and Tudat Core. The latter is a stand-alone library, while the former is allowed to have dependencies on Tudat

Core. An additional layer, called Tudat Applications, works as a personal workspace for developers.

Tudat Core

As hinted by its name, Tudat Core contains the core functionalities of the Tudat project. It was set up with purpose of freezing the interfaces of widely used code blocks and as a result provide a stable and reliable base for the state-of-the-art functionalities in Tudat. Code to be added to Tudat Core needs to comply with a number of constraints which ensure that the code is well-tested, well-documented and bug-free. Tudat Core undergoes slow evolution and backwards compatibility between successive releases is guaranteed.

At the time of writing, Tudat Core contains a number of robust code blocks in three major categories: Astrodynamics, Input/Output and Mathematics. The most notable Tudat Core Astrodynamics routines are an extensive collection of unit and orbital elements conversions. The Tudat Core Mathematics functionality are linear algebra functions, coordinate conversions, as well as a generic base class for numerical integrators. Other notable Tudat Core routines are basic input/output handling functions and test macros.

Tudat

Unlike Tudat Core, the interfaces of Tudat are allowed to change frequently, as this library represents the state-of-the-art of the Tudat project. In technical terms, Tudat undergoes fast evolution. Features can, after a thorough and well-established code-checking process, be readily added to Tudat such that other developers can immediately benefit from them. The code-checking process ensures that code blocks comply with Tudat protocols, do not break existing functionality and have been written in collaboration or consultation with at least one other developer.

At the time of writing, Tudat, in its 1.1 version, already contains a considerable number of features, from environment models to celestial bodies and vehicles, from propagators to orbital elements conversion mechanisms. The Tudat features be categorized into three major categories: astrodynamics, mathematics, and input/ouput. These three are in turn broken down into several sub-categories, as shown in Table 8.1. The complete list of features, which is rather extensive, can be found on the Tudat website.

The division of the Tudat features into major categories reflects the dependencies of the routines within. Tudat code blocks should have as little dependencies as possible on code from another category, as to limit the propagation of code changes through the library.

Tudat Applications

Tudat Applications is the development workspace where simulations, making use of Tudat and/or Tudat Core, are set up by Tudat developers. These applications are documented and are accessible to all team members, enabling developers to easily share their code. Intuitively, Tudat and Tudat Core are not allowed to have dependencies on Tudat Applications. Figure 8.2 is aschematic representation of the three Tudat layers.

Table 8.1: Tudat 1.1 features

Tudat	Astrodynamics	Aerodynamics
		Basic Astrodynamics
		Bodies
		Ephemerides
		Gravitation
		Mission Segments
		Reference Frames
		States
	Input/Output	Basic Input/Output
		Parsers
	Mathematics	Basic Mathematics
		Geometric Shapes
		Interpolators
		Numerical Integrators
		Root-finding Methods
		Statistics

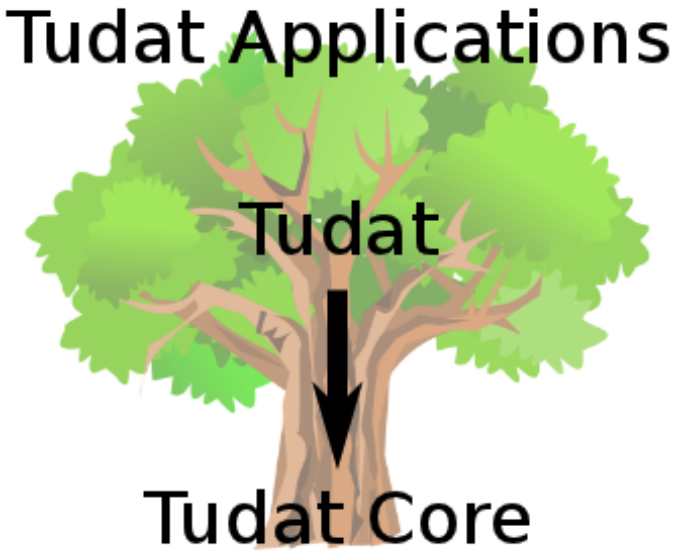

Figure 8.2: The setup of Tudat: the rapidly evolving TUDAT is rooted in the stable TUDAT CORE, with TUDAT APPLICATIONS sitting on top of the tree [Tudat, 2012].

8.1.3 Management Setup

As previously discussed, Tudat is mainly developed by MSc students, which often abandon the university, and thus the Tudat project, after completion of their studies. This introduces several challenges such as ensuring the longevity of the toolbox despite a transient group of developers, guaranteeing the modularity of the code such that each user can easily integrate the implemented features and enforcing the robustness of the features over different applications. The way in which Tudat was set up was thought as to answer these challenges.

Organization

Tudat members assume several responsibilities within the project. The majority of the members are *code developers* as well as *code checkers*. This dual responsibility allows for a faster code-checking process, an even distribution of the workload and promotes the communication between developers. At the top of the food chain, one finds the Tudat *managers*, i.e. A&S staff members actively involved in the Tudat project. Other responsibilities arise from the need to maintain the different project management tools. The organogram of Tudat is illustrated in Figure 8.3.

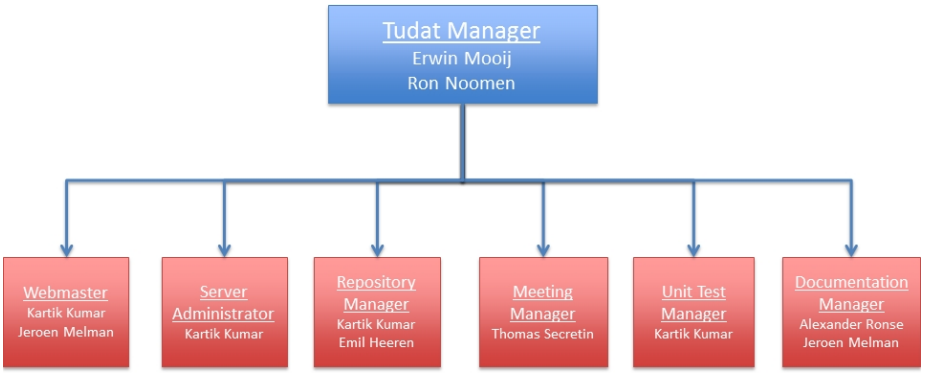

Figure 8.3: The Tudat organogram [Tudat, 2012].

Roughly three different types of formal Tudat meetings are set up, with different frequencies. The *Management Team*, comprising all members with duties other than code development, meets, at the time of writing, once a month and is responsible to maintain oversight and ensure the longevity of the project. The *Working Group* is formed by all Tudat developers and gathers twice a month to discuss current and future code development. Specialized meetings can be held, on a need basis, between developers working on common functionalities. These meetings are referred to as *Taskforce Meetings*.

Coding Standards and Protocols

As stated from the start, Tudat is a C++ library which means that all the underlying code is written in that programming language as to ensure uniformity and coherence. Moreover, it is a rather recent programming language which has

proven to be efficient and flexible. Having a single tool written in a single programming language facilitates the new developers' learning process as they avoid juggling between different languages. With readability, coherence and robustness requirements in mind, the code is standardized based on common industry standards and Tudat group choices. These standards are gathered in the "The N Commandments" document [Melman, 2011].

Documentation Standards and Protocols

Proper documentation is a vital part of Tudat. It allows Tudat developers to efficiently gain insight into another developer's code and therefore facilitates the understanding and integration of any piece of code. To automate and standardize the documentation process, the Tudat project relies on an external tool, Doxygen[1]. Besides the code documentation, tutorials, examples and manuals are provided, via the website *wiki*, to reduce the learning curve of new developers and users.

Code Robustness

To ensure the code robustness of Tudat, a thorough code-checking process was implemented. Every piece of source code is independently checked by at least one other developer, both for coding standards compliance and for mathematical and physical correctness. A unit test framework was established, where each developer is required to provide a simple test case to facilitate debugging and code-checking. The code-checking process is often iterative and therefore somewhat long. Once code-checking process is completed, the revised code is submitted to the repository manager that commits it to the repository. This code-checking process is illustrated in Figure 8.4.

File Repository

The file repository contains all the (checked) pieces of code of Tudat, as well as a series of relevant documents such as manuals, meeting minutes and presentations. It enables a decentralized code development as any developer can access the repository and review the code development history, from anywhere in the world, using the code repository application of his or her choice. To ensure quality, readability and robustness, commits to the repository are controlled by the repository manager, and are accompanied by a small description of changes.

Website

The Tudat website (`tudat.tudelft.nl`) is an essential tool of the Tudat project as it provides the central point of communication for the entire project. It also operates as a project management tool, where meetings are set up, issues are discussed and several resources are made available for users, developers and managers. The website also provides access to the repository.

[1] More information about Doxygen can be found at: `http://www.stack.nl/~dimitri/doxygen/`

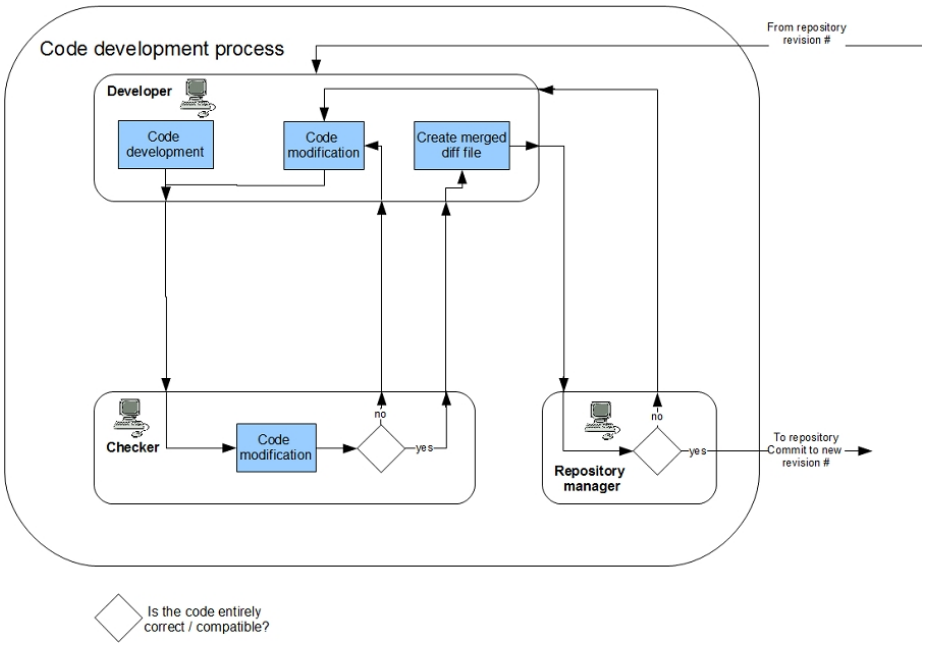

Figure 8.4: The Tudat code development process [Tudat, 2011].

8.2 Tool Implementation

In order to implement the proposed tool to solve the combinatorial problem of GTOC2, use was made of several existing Tudat features. While the complete tool was developed based on the Tudat library, only certain code-blocks were added to the library. The entirety of the code developed in the scope of the MSc Thesis could not be added to Tudat due to time restrictions. It will however be made available via the Tudat Applications layer. The distinction between the code blocks that were already available in Tudat, code blocks that were added to library and code blocks that will remain in the Tudat Applications layer, is made in the following section.

8.2.1 Tool Block-diagram

The block-diagram of the implemented tool can be seen in Figure 8.5. The input consists of three elements: the ephemeris data of the celestial bodies[2], the cost function of the continuous optimization problem and parameters of the NNH, namely the number of champions, the departure windows and the maximum time-of-flight of the transfers. The first step is the pre-processing of the ephemeris data: the text file is parsed[3] and relevant data transformed and extracted to a data container. The dimensions of the data container are used by the NNH to determine the number of asteroid groups, their order and

[2]The ephemeris data is provided via a text file.

[3]Parsing refers to the process of tokenizing a text, i.e. dividing it into individuals segments (e.g. words) based on a given structure, e.g. comma-separated file.

their dimension. The NNH then runs, calling the arc cost optimization block as needed. This block receives, independently of the specific optimizer (GS or DE), the initial and final bodies, as well as the bounds for the departure date and time-of-flight. During the optimization, a Kepler propagator is invoked to determine the position of the bodies at the given departure and arrival dates. This information is then passed on to the Lambert targeter, along with the corresponding time-of-flight, which returns the ΔV of the corresponding Lambert problem. This value is used to compute the cost function value according to the specified cost function. The best solution found at the end of the continuous optimization procedure is returned to the NNH. Using these static arc cost values, the NNH solves the combinatorial problem and returns the N best asteroid sequences.

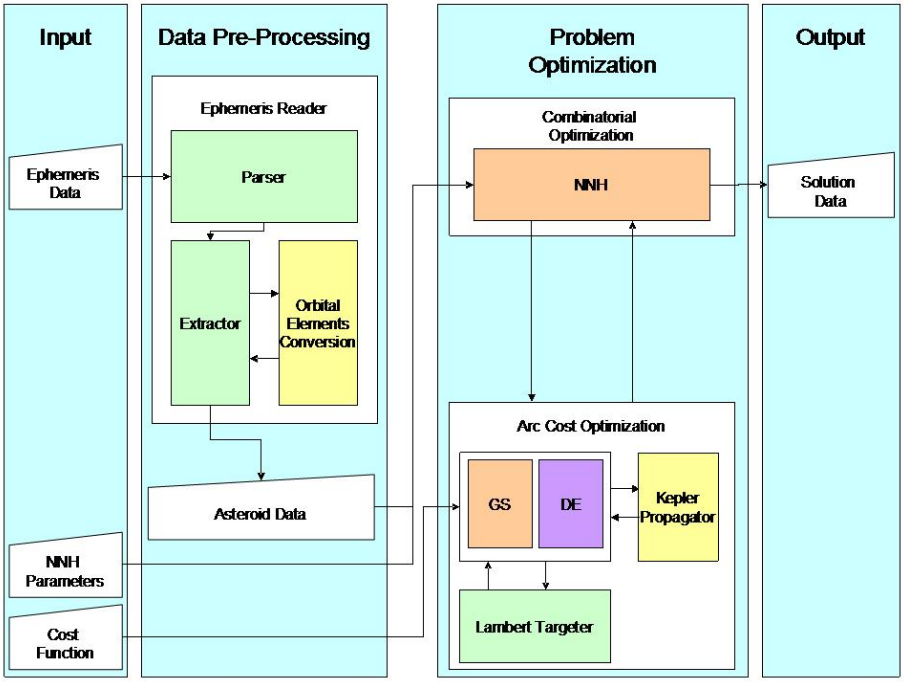

Figure 8.5: The flow-chart of the assembled tool.

Assembling several building blocks, as shown in Figure 8.5, confers a certain modularity to the tool. The most notable illustration of said modularity is the flexibility in the choice of the continuous optimizer: in the scope of the MSc thesis, the GTOC2 combinatorial problem was solved using either GS or DE. Thanks to the interface with PaGMO, these can readily be replaced with another algorithm, e.g. Particle Swarm Optimization (PSO) ([Kennedy and Eberhart, 1995]). Writing specific parsers and extractors would allow to solve similar problems but with a different data set. Finally, one could replace the IZZO Lambert routine with its Lancaster, Blanchard & Gooding (LBG) counterpart or even with a different trajectory model, e.g. a shape-based low-thrust model such as exponential sinusoids ([Petropoulos and Longuski, 2004]).

8.2.2 Contributions to Tudat

In Figure 8.5, code blocks that were added to the Tudat library are shown in green, while yellow indicates existing Tudat code blocks that were used to build the tool. Code blocks that were not added to Tudat, but that can be found in Tudat Applications, are highlighted in red. Finally, the purple code block (DE) refers to an external routine.

Following this colour code, one can see that the two main code blocks were added to Tudat: the ephemeris reader (which makes use of the previously available orbital elements conversions) and the Lambert targeter.

Ephemeris Reader

In the scope of the MSc Thesis, the author was an active member of the Ephemeris Reader Taskforce within Tudat. This taskforce was mandated to review and update the ephemeris reader available at the time in the library.

A new architecture was designed by the taskforce with the overarching principles of robustness, modularity and flexibility, as ephemeris data files come in a variety of formats. The main characteristic of the proposed architecture is the separation between data acquisition (parsing) and data interpretation (extraction). This allows to have the same routines to parse files with different data in the same format (e.g. comma-separated value files) and the same routines to extact the same data (e.g. Keplerian elements) in different formats. Parsers and extractors can be combined as needed to create the adequate ephemeris reader. In between the parsing and extraction steps, the data is stored in a generic container.

The taskforce designed, implemented and tested a number of parsers and extractors which have been added to the Tudat library. This new architecture was used in the scope of the author's MSc thesis, as seen in Figure 8.5. A specific parser for the GTOC2 ephemeris file, `GTOC2AsteroidEph.txt`, was coded.

Lambert Targeter

The Lambert targeter described in Chapter 7 was implemented in Tudat. This was done in the scope of the Mission Segments Taskforce efforts to review the mission segments routines in Tudat.

Previous to the taskforce update, all Mission Segments code blocks (Lambert targeter, gravity assist function, deep space maneuvers, and escape and capture maneuvers) inherited from a single parent class. The taskforce noted however that these routines did not share enough functionalities, nor did they share a common interface, to justify the inheritance from a common base class. All code blocks were therefore re-written as free functions. The obsolete base class and the empty deep space maneuver derived classes were removed. A number of different approaches to gravity assist computations (powered/unpowered, propagated/computed) were added. With respect to the Lambert targeters, LBG and IZZO, these functions are intuitively similar with respect to their input (departure and arrival positions, time-of-flight, gravitational parameter of the central body) and output (ΔV inherent to the trajectory solution). These commonalities led to the implementation of a Lambert targeter base class with derived classes wrapping the free functions.

The reviewed Mission Segments routines were extensively tested and collaboratively code-checked. They can now be found in the Tudat repository. For the purposes of the author's MSc Thesis, the free function version of the IZZO Lambert Targeter was used.

Part III

Results

Chapter 9

Complete Asteroid Pool, using GS

Here, the results obtained with the multi-path NNH are described. The NNH was applied with different cost functions as well as different search directions: forward and backward. The a priori optimization of the individual transfers is done using an enumerative method, namely a Grid Search (GS), rather than an optimization heuristic such as Differential Evolution (DE). GS was originally intended as a preliminary approach but due to timeliness considerations, DE was only implemented in the final part of the MSc Thesis work. The results obtained with DE can be found in Chapter 11. GS relies on a discretized mesh for both the departure dates and the times-of-flight, each grid point being 10 days apart.

For the sake of simplicity, we will refer to the transfers from Earth to Group 4 asteroids as Leg 1, to transfers from Group 4 to Group 3 asteroids as Leg 2, to transfers from Group 3 to Group 2 asteroids as Leg 3, and finally to transfers from Group 2 to Group 1 asteroids as Leg 4. Moreover, we will refer to asteroid combinations corresponding to one of the solutions handed in by GTOC2 participants as GTOC2 sequences. Given that, at the time of writing, no accurate low-thrust model is implemented in Tudat, the quality of the sequences is assessed through comparison with the GTOC2 participant sequences.

Three different variants of the multi-path NNH search are performed. The forward search is applied using both the optimal final mass cost function, J_2, in Section 9.1 and the optimal final mass to time-of-flight ratio cost function, J_1, in Section 9.3. In between, we take a look at the backward search using J_2, in Section 9.2.

9.1 Forward Search, Optimal Final Mass

We start by solving the asteroid sequence selection problem with a forward multi-path NNH, meaning that the algorithm starts by computing Leg 1 transfers and ends at the Leg 4 transfers. The continuous arc cost function to be optimized by GS is Equation (6.1.5), meaning that the time-of-flight is not taken into consideration and that the maximization of final mass is sought. As per Tsiolkowski's Law (Equation (4.4.3)), maximizing the final mass is equivalent to

minimizing the total ΔV. The latter is often used as a metric in the following sections.

The departure window for Leg 1 transfers is taken to be the same as that of the original problem description (see Section 2.2), i.e. the spacecraft can depart anywhere between January 1^{st} 2015 and December 31^{st} 2035. For the subsequent legs, different departure windows, starting at the time of arrival at the asteroid, are investigated. We also vary the number of champion paths. The time-of-flight for all legs is bounded within 10 and 610 days.

9.1.1 Leg 1

Let us first look at Leg 1 transfer results. These are constant over the different cases since they are not affected neither by the number of champion paths (in a forward search, all Leg 1 transfers are computed) nor by the departure window, which is kept constant for the first leg.

In Figure 9.1, the histogram for the final mass fraction of Leg 1 transfers is given. The final mass fraction is computed according to Equation (4.4.3), where the ΔV is the minimum value found during the a priori optimization and the effective exhaust velocity, c, is obtained with:

$$c = I_{sp} g_0 \tag{9.1.1}$$

where g_0 is the standard gravitational acceleration and the specific impulse, I_{sp}, is assumed to be 300 s. Note that the final mass fraction, and not the J_2 value, is shown. Therefore, larger values in Figure 9.1, which correlate to lower J_2 values, correspond to attractive transfers.

Figure 9.1 shows that there is a relatively small number of transfers with a larger final mass fraction, meaning that the high-thrust approximation leads to a fast selection of promising Group 4 asteroids. To compare the Leg 1 results with the GTOC2 solutions, Table 9.1 lists the 40 best transfers, according to decreasing ΔV. In order to include all GTOC2 Group 4 asteroids (8 different asteroids in total), one must consider all the 40 best trajectories. This means that when considering the 40 best Leg 1 transfers found, only 20% correspond to GTOC2 solutions. On the other hand, if one limits the analysis to the top asteroids, two of the three best transfers correspond to GTOC2 solutions. Therefore, the number and ratio of GTOC2 Group 4 asteroids considered for the subsequent legs is influenced by the number of paths that will be pursued by the multi-path NNH. The number and the ratio[1] of GTOC2 Group 4 asteroids as function of the number of paths to pursue are illustrated in Figures 9.2 and 9.3, respectively.

Another interesting observation can be made from Table 9.1. The Group 4 asteroid corresponding to the GTOC2 winner, `3258076`, ranks merely 37^{th} with the proposed approach. This is due to the fact that when selecting the first asteroid, the team from Politecnico di Torino took into account the "free" 3.5 km/s hyperbolic excess velocity. As a result, they are able to reach a Group 4 asteroid further away from Earth (hence more expensive in terms of ΔV) but closer to Group 3 and 2 asteroids. An attempt to reverse-engineer the multi-path

[1]The ratio is defined as being the number of GTOC2 Group 4 asteroids divided by the total number of asteroids considered.

Table 9.1: The 40 best Leg 1 transfers, according to J_2 (ΔV), using GS.

Rank	Group 4 Asteroid	Minimum ΔV [m/s]	GTOC2 Rank
1	3042555	2187.94	
2	3250293	2290.85	#2,#6,#10,#12
3	3017309	2390.14	#5
4	3024030	2621.51	
5	3054338	2692.80	
6	3293922	2707.18	
7	3261681	3051.47	
8	3167367	3074.55	
9	3070801	3092.74	
10	3156302	3226.06	
11	3170221	3249.37	#3,#4,#9,#13
12	3071939	3257.78	
13	3329255	3261.25	#8
14	3072273	3398.61	
15	3297379	3772.55	
16	3297629	3788.24	
17	3278402	3865.22	
18	3293923	3992.41	
19	3288933	3993.45	#7
20	3054373	4057.22	
21	3299721	4264.83	
22	3160723	4292.23	
23	3067492	4313.04	
24	2099942	4353.46	
25	3144155	4383.47	
26	3114023	4392.09	
27	3147579	4433.71	
28	3297182	4484.71	
29	3068066	4524.96	
30	2065679	4569.38	
31	3167353	4645.76	
32	3172322	4671.53	
33	3072291	4709.23	
34	3177202	4782.00	#14
35	3153530	4921.74	
36	3102787	4974.27	
37	3258076	4990.94	#1
38	3263232	5001.39	
39	3341199	5123.89	
40	3343104	5138.33	#11

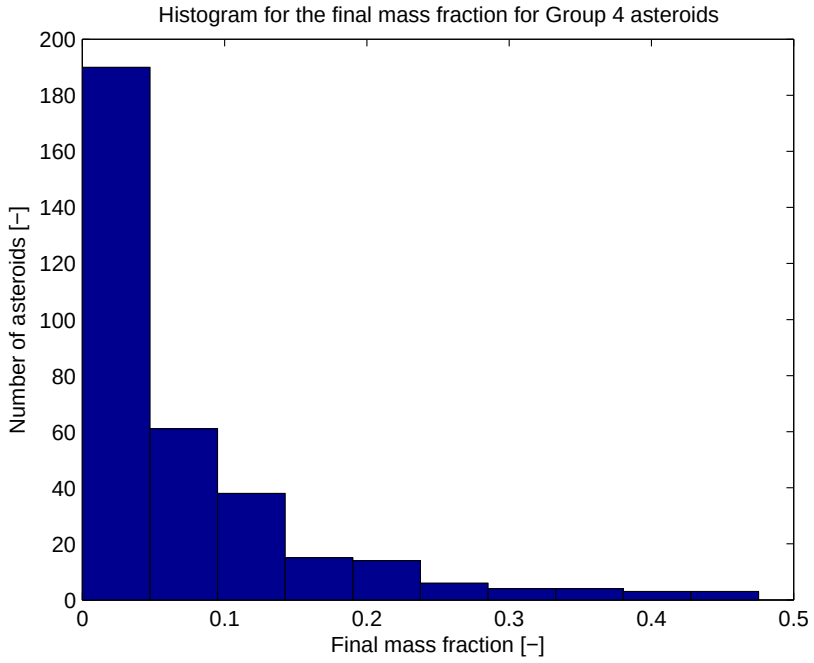

Figure 9.1: Histogram for the final mass fraction of Leg 1 transfers.

NNH by tuning the number of champions at each leg[2] to output the GTOC2 winner's sequence resulted in prohibitive computational times (the simulation had to be interrupted after 20 hours of computations...) and was therefore abandoned. In the analysis of the subsequent legs, we set the maximum number of paths to pursue to a more tractable value, namely 20. At this threshold, the ratio of GTOC2 Group 4 asteroids is 25% (see Figure 9.3).

9.1.2 Remaining Legs

Unlike the first leg, the remaining legs computations are influenced by the number of champion paths pursued by the multi-path NNH. Based on the Leg 1 results, we investigate different number of champion paths: 3, 4, 5, 10, 13, 15, and 20. Similarly, the departure windows can be tuned. We opt to investigate three different departure windows: 10, 15 and 20 years. These departure windows start from the date of arrival at the current departure asteroid. Note that if the current transfer has already been computed, the algorithm uses the corresponding result rather than re-starting the grid search. It is assumed that the selected departure windows are wide enough to allow geometric repeatability, at least up to a certain extent. Based on this premise, the best individual transfer found are cached. This caching feature means that, should a particular asteroid pair be present in more than one sequence, the corresponding arc is optimized

[2] i.e., for Leg 1 select 37 champions, for Leg 2 a number of champions corresponding to the rank of the partial GTOC2 #1, and so on.

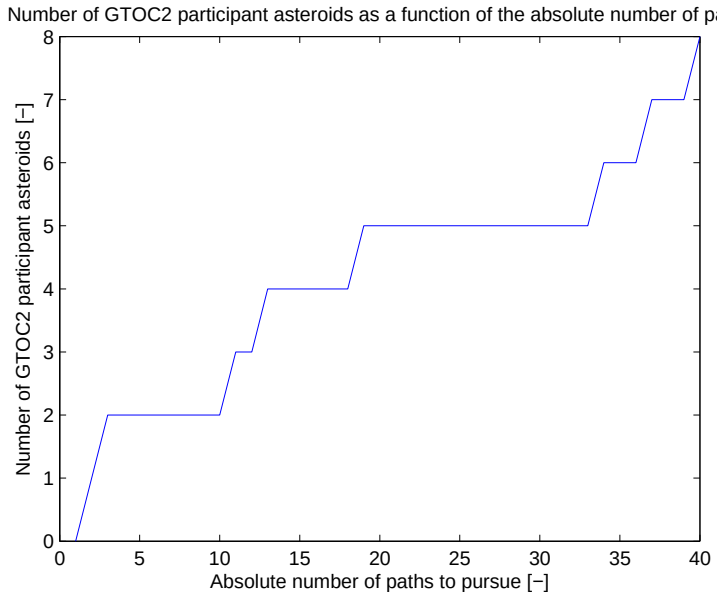

Figure 9.2: Number of GTOC2 Group 4 asteroids as a function of the number of paths to pursue, using GS.

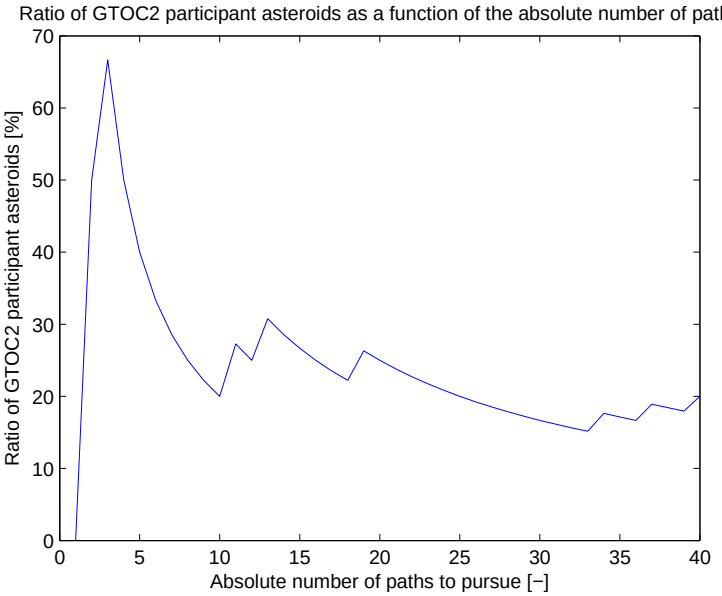

Figure 9.3: Ratio of GTOC2 Group 4 asteroids as a function of the number of paths to pursue, using GS.

only once and the resulting value re-used if this arc needs to be evaluated again. The caching of optimal values for individual transfers is expected to reduce the complexity of the problem.

10-Years Departure Window

Independently of the number of champion paths selected, none of the GTOC2 solutions appear in the results yielded by the analysis with a 10 year departure window. This seems to indicate that the departure window is too narrow: the GTOC2 sequences rank below the path threshold in the intermediate legs and are therefore discarded along the way. The 10 best sequences, and corresponding total ΔV, can be found in Appendix C.

While the quality of the results obtained with a 10 year departure window are not satisfactory, at least when compared to GTOC2 results, we can extract some information regarding the computational time needed by the multi-path NNH to produce results, as a function of the number of champion paths. The computational time needed for each run of the multi-path NNH is given in Table 9.2. The number of different transfers (arcs) optimized, according to Equation (5.2.2), is also listed. We can see that, with departure windows of 10 years, the multi-path NNH never takes more than one hour to complete: with 20 champion paths, the algorithm takes 46 minutes to terminate. At the other extreme, taking 3 champion paths results in computational times of merely 10 minutes. As expected, the computational time increases with the number of champion paths. Indeed, the larger the number of champion paths, the larger the number of transfers that need to be computed.

Table 9.2: The computational time needed by the multi-path NNH with 10-years departure windows, using GS.

Number of champion paths	3	4	5	10	13	15	20
Number of arcs evaluated	2054	2626	3198	6058	7774	8918	11778
Computational Time [h]	0.17	0.21	0.25	0.44	0.56	0.61	0.77

15-Years Departure Window

When setting the departure window to 15 years, the only GTOC2 solution present in the last leg computations is the runner-up: the solution from the Moscow Aviation Institute and Khrunichev State Research and Production Space Center, to which we shall refer, from here on, as GTOC2 #2.

Table 9.3 lists the rank of the (partial) GTOC2 #2 asteroid sequence at each of the consecutive leg computations, with the departure windows for the multi-path NNH set to 15 years. Note that *n.a.* represents (partial) sequences that are discarded before reaching a given leg computation.

The first observation we can make when looking at Table 9.3 is that the rank of the GTOC2 #2 sequence is greatly degraded when the last leg is reached.

Table 9.3: Rank of the GTOC2 #2 (partial) sequence as found by the multi-path NNH with 15-years departure windows, using GS.

Number of champion paths	Leg 1	Leg 2	Leg 3	Leg 4
3	2	4	n.a.	n.a.
4	2	4	3	39
5	2	4	3	44
10	2	4	6	81
13	2	4	6	103
15	2	4	7	115
20	2	4	10	148

While the partial sequences (sequences up to Leg 1, up to Leg 2 and up to Leg 3) rank in the top-ten independently of the number of champion paths (except for the third leg with three champion paths), the complete sequence ranks between 39^{th} and 148^{th}, depending on the number of champion paths. This degradation of the rank of the GTOC2 #2 sequence in Leg 4 is coherent with the study conducted by [Alemany and Braun, 2007], which showed that the correlation between the optimal high-thrust and low-thrust trajectories decreases for the last leg of the GTOC2 problem. Also worthy of noting, the GTOC2 #2 final ranking improves when reducing the number of champion paths. This is true while the number of champion paths is larger than three: with three champion paths, the GTOC2 #2 partial sequence ranks 4^{th} in Leg 2 and is therefore discarded. Hence the *n.a.* notes for the remaining legs. The improvement of the GTOC2 #2 rank with the downsizing of the number of champions can be explained by the fact that a more restrictive value will discard, at a given leg, more partial sequences which may prove to be more attractive further down the line. Reducing the number of champion paths effectively decreases the number of rival sequences and therefore the GTOC2 #2 sequences ranks better.

With respect to the computational times involved in solving the GTOC2 asteroid selection problem with a multi-path NNH with 15-years departure windows, Table 9.4 list the elapsed time as function of the number of champion paths. As with the 10-years departure windows variant, the computational time needed for algorithm completion increases with the number of champion paths. The explanation for this phenomenon is identical: the larger the number of paths that transit from one leg to the other, the larger the number of transfers that need to be computed at the following step. Note also that the computational time needed by the multi-path NNH with 15 years is larger than with 10 years: with 20 champions, the multi-path NNH takes about one hour and half instead of 46 minutes, i.e. close to twice as long. The same explanation applies: the larger the departure windows, the larger the number of grid points inspected by the grid search and therefore the larger the computational effort.

20-Years Departure Window

For departure windows of 20 years, the GTOC2 #2 sequence is the only GTOC2 solution to make it to the final leg computations for every number of champions

Table 9.4: The computational time needed by the multi-path NNH with 15-years departure windows, using GS.

Number of champion paths	3	4	5	10	13	15	20
Number of arcs evaluated	2054	2626	3198	6058	7774	8918	11778
Computational Time [h]	0.26	0.32	0.43	0.76	0.93	1.04	1.54

considered, including three champion paths.

The rank of the (partial) GTOC2 #2 asteroid sequence at each of the consecutive leg computations with 20-years departure windows is given in Table 9.5.

Table 9.5: Rank of the GTOC2 #2 (partial) sequence as found by the multi-path NNH with 20-years departure windows, using GS.

Number of champion paths	Leg 1	Leg 2	Leg 3	Leg 4
3	2	3	3	36
4	2	3	4	43
5	2	3	4	55
10	2	3	6	96
13	2	3	7	122
15	2	3	7	145
20	2	3	8	186

Akin the results with 15-years departure windows, the rank of the GTOC2 #2 sequence deteriorates for the final leg, when compared with the rank of the same sequence truncated at Leg 3. Another similarity with the previous case arises when noticing that the rank of the GTOC2 #2 solution improves when reducing the number of champion paths: the 186^{th} position with 20 champions evolves into a 36^{th} position with 3 champions. But unlike what we could see with 15-years departure windows, selecting three champion paths does not discard the GTOC2 #2 solution from the final trajectories. This means that allowing for larger departure windows leads to a relatively less expensive partial GTOC2 #2 sequence, namely up to Leg 3. However, it also leads to a relatively more expensive final GTOC2 #2 sequence as infered by the GTOC2 #2 rank drop in the complete sequences, for the same number of champion paths, from 15 (Table 9.3) to 20 (Table 9.5) years departure windows.

For a number of champions larger than 10, another GTOC2 solution is present in the last leg computations: the sequence from the University of Glasgow et al., to which we shall refer to as GTOC2 #12[3]. Bear in mind that

[3]Although the solution from the University of Glasgow et al. was not ranked due to large constraint violations, it is the 12^{th} listed team in [Petropoulos, 2007].

GTOC2 #12 was disqualified from the competition due to significant position and velocity violations which could not be resolved within the timeframe of the competition. The rank of the GTOC2 #12 as a function of the number of champion paths is given in Table 9.6, where the same notation for discarded sequences as in Table 9.3 is used.

Table 9.6: Rank of the GTOC2 #12 (partial) sequence as found by the multi-path NNH with 20-years departure windows, using GS.

Number of champion paths	Leg 1	Leg 2	Leg 3	Leg 4
3	2	11	n.a.	n.a.
4	2	11	n.a.	n.a.
5	2	11	n.a.	n.a.
10	2	11	n.a.	n.a.
13	2	12	8	32
15	2	12	8	38
20	2	12	9	45

As with GTOC2 #2, there is a deterioration of the GTOC2 #12 rank in Leg 4. However, this degradation is not as severe, given that GTOC2 #12 ranks considerably better than GTOC2 #2, for the same number of champion paths. For the larger numbers of champions, GTOC2 #12 is between 80 and 141 places ahead of GTOC2 #2, for the complete sequences. Given that the GTOC2 #12 was not ranked in the competition due to large constraint violations, one should be careful when drawing conclusions. If we assume that a constraint-compliant GTOC2 #12 complete trajectory would have a lower objective function value than GTOC2 #2[4], the improved final rank of GTOC2 #12 over GTOC2 #2 is in tune with the observations from [Alemany and Braun, 2007] regarding the degradation of the correlation between high- and low-thrust optimal trajectories for Leg 4. As with the GTOC2 #2 sequence, the larger the number of champions (starting at 13), the lower the final rank of GTOC2 #12.

Finally, let us take a look at the computational times inherent to a multi-path NNH with 20-years departure windows, as shown in Table 9.7.

Table 9.7 exhibits the same trend as Tables 9.2 and 9.4: increasing number of champions leads to increasing computational times. Moreover, there is also an increase in computational time, for the same number of champions, from 15- to 20-years departure windows. These increases are due to the larger complexity inherent to larger numbers of transfers and to larger departure windows, respectively. Note however that the relative increase from 15 to 20 years (approximately 20% for 20 champions) is quite lower than from 10- to 15-years departure windows (approximately 100%).

[4]The author believes this is a fair assumption given that the constraint-violating solution as a lower objective function value and that forcing the trajectory to respect the constraints will most probably come at the cost of a loss in objective function value.

Table 9.7: The computational time needed by the multi-path NNH with 20-years departure windows, using GS.

Number of champion paths	3	4	5	10	13	15	20
Number of arcs evaluated	2054	2626	3198	6058	7774	8918	11778
Computational Time [h]	0.32	0.41	0.55	0.83	1.09	1.16	1.88

9.1.3 Conclusions

In order to compare the performance of the multi-path NNH with the different departure windows, the GTOC2 #2 rank and the elapsed time are plotted in Figures 9.4 and 9.5, respectively.

In Figure 9.4, a rank of 500 is equivalent to a discarded sequence. As discussed earlier, the 10 year departure windows are too narrow to allow the GTOC2 #2 sequence to be present in the final asteroid sequences, independently of the number of champion paths. When considering 15-years departure windows, selecting merely three champion paths to pursue is too strict of a requirement to not discard the GTOC2 #2 solution. For the rest of the possible numbers of champions, the GTOC2 #2 rank exhibits a negative linear trend: the rank of the GTOC2 #2 sequence decreases linearly with increasing number of champion paths. With 20-years departure windows, a similar trend can be observed with exception of the GTOC2 #2 sequence not being discarded with three champions. When not discarded, the GTOC2 #2 ranks better with the 15-years departure windows than with the 20-years ones. Nonetheless, the overall best rank obtained by the GTOC2 #2 sequence is 36^{th}, for 20-years departure windows and only three champion paths.

With respect to the computational effort needed by the three variants of the forward multi-path NNH, based on Figure 9.5, these display roughly linear trends: the larger the number of champions, the longer the algorithm needs to complete. The difference in relative computational effort increase from 10- to 15-years departure windows and from 15 to 20-years departure windows is clearly visible. Note that even the most complex variant (20-years departure windows and 20 champions) of the problem is handled in less than two hours.

The relative quality of the sequences found is rather disappointing. The algorithm cannot return the GTOC2 winning sequence without a considerable computational effort, and even then, we believe it would be poorly ranked. On the other hand, the GTOC2 runner-up sequence can be found in most of the cases, namely with large departure windows, a restricted number of champions on the first three legs and considering a larger number (about 50) of complete sequence. Such parameters are in tune with the findings of [Alemany and Braun, 2007]: a more constraining number of champions is applied to the legs with a high correlation between optimal high- and low-thrust solutions (Legs 1, 2 and 3) and more leeway for the final leg, where the correlation is weaker. In a few special cases (very large departure windows and large number of champions), a second GTOC2 solution is returned by the multi-path NNH. While, in these

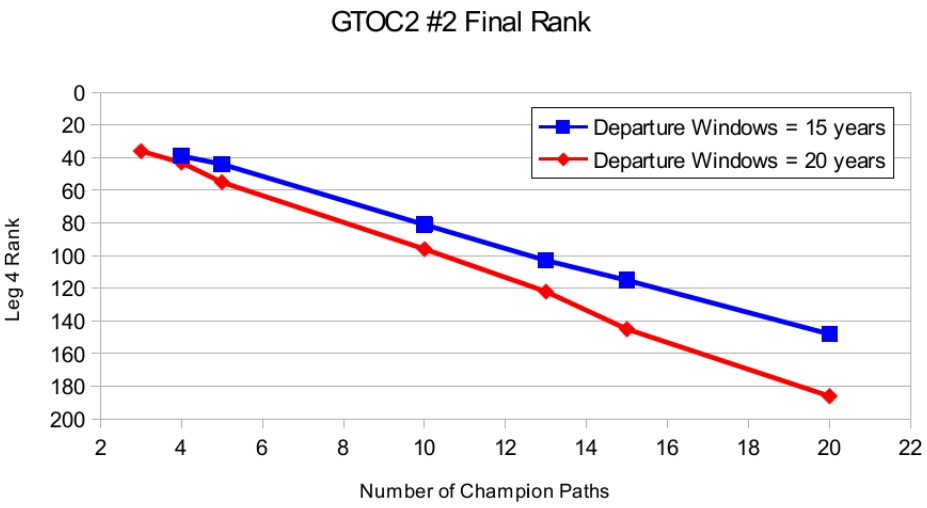

Figure 9.4: Rank of the GTOC2 runner-up in the forward search as a function of the number of champion paths, using GS.

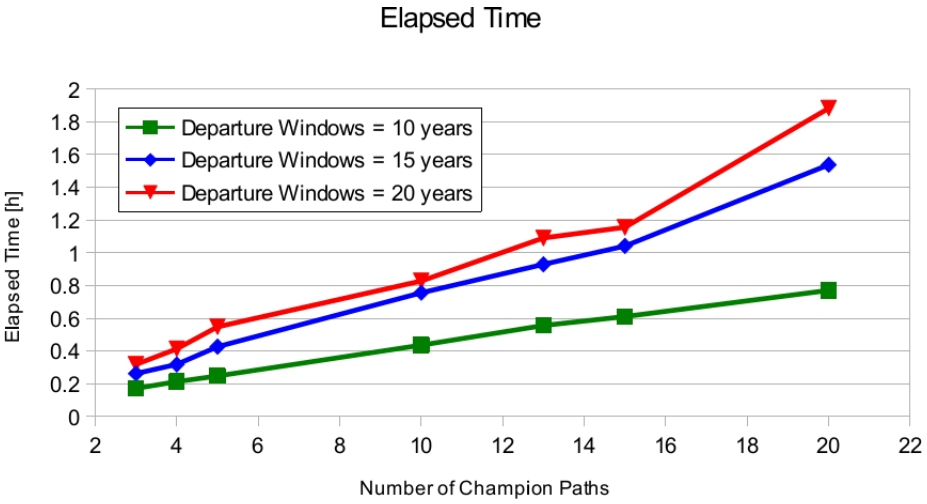

Figure 9.5: Computational time of the forward search as a function of the number of champion paths, using GS.

cases this solution outranks the GTOC2 runner-up, it was disqualified from the GTOC2 competition due to constraint violations. Considering the limit on the maximum number of champion paths (20), the forward search does not yield any of the other GTOC2 solutions. However, the relatively small computation times involved in obtaining said results provide confidence that the multi-path NNH can handle slightly more complex models. Therefore, we attempt a backward search.

9.2 Backward Search, Optimal Final Mass

Given that, for the forward search, Leg 4 results proved to be the less reliable, we decide to pursue a backward search with multi-path NNH. In other words, we will start the selection process by identifying promising transfers for Leg 4 and finish with Leg 1. The continuous arc cost function to be optimized by GS is the same as in the previous section, namely Equation (6.1.5).

Given that most GTOC2 solutions correspond to mission times-of-flight of approximately 10 years (see Section 2.3), we take the arrival window at Group 1 asteroids to be 10 years later than the original 2015 - 2035 departure window, i.e. the spacecraft can arrive at a Group 1 asteroid anywhere between January 1^{st} 2025 and December 31^{st} 2045. Since we are constructing the different mission scenarios backwards, this involves some gymnastic with respect to the departure dates for the following transfers. To explain this, let us consider the Leg 3 computations. Given the departure date of the Leg 4 transfer, we determine the arrival date interval. The upper limit of said interval is the departure date of the next transfer, to avoid a mission scenario where the spacecraft would leave an asteroid before getting there. The lower limit is variable, based on a variable arrival window, similar to the departure window used for the forward search. The departure date is then derived from the time-of-flight, which can vary between 10 and 610 days. These computations are illustrated in Figure 9.6, where overlined values are constant, t_d and t_a stands for departure and arrival date, respectively, the arrival window is indicated by T, the numerical index indicates the leg number, and $t_{f,\min}$ and $t_{f,\max}$ denote the minimum and maximum times-of-flight, respectively.

9.2.1 Leg 4

As with the forward search, the results of the first leg investigated, here Leg 4, are constant over the different parameter variations since all the possible Leg 4 transfers are computed, therefore overrulling the effect of the number of champions, and since the arrival window is fixed. Table 9.8 lists the 35 best Leg 4 transfers according to minimum J_2 value (more specifically according to the equivalent minimum ΔV), as well as the GTOC2 partial sequences that fall outside the top-35 transfers.

The degradation of the correlation between optimal high- and low-thrust transfers for Leg 4 is visible when comparing Tables 9.1 and 9.8. While for Leg 1 all GTOC2 Group 4 asteroids are contained in the top-40 high-thrust transfers, only half the GTOC2 Group 1 - Group 2 asteroid combinations are present in the top-35 high-thrust Leg 4 transfers. The transfers corresponding to the GTOC2 second, third and fourth best sequences are ranked below 100. At

Table 9.8: The 35 best Leg 4 transfers, according to minimum ΔV, and GTOC2 solutions ranking, using GS.

Rank	Group 1 Asteroid	Group 2 Asteroid	Minimum ΔV [m/s]	GTOC2 Rank
1	2001902	2000022	2465.18	
2	2014569	2000325	2553.29	
3	2015278	2000120	2672.94	
4	2001754	2000490	2687.16	
5	2002760	2000325	2730.82	
6	2002483	2000168	2806.76	
7	2002959	2000414	2858.49	
8	2002959	2000334	2892.60	
9	2000361	2000508	2904.82	
10	2011542	2000086	3041.75	
11	2003134	2000338	3111.54	
12	2002959	2000104	3113.69	
13	2002959	2002407	3178.96	
14	2002959	2001015	3188.20	
15	2002760	2001625	3195.10	
16	2001038	2000022	3258.48	
17	2001038	2000481	3269.42	
18	2000225	2000105	3280.42	
19	2003134	2000713	3295.03	
20	2011542	2001445	3295.24	
21	2002959	2000058	3301.07	#1, #12
22	2009661	2000508	3309.66	
23	2001345	2000173	3317.23	
24	2001038	2000110	3377.06	#6
25	2001902	2000977	3441.63	
26	2014569	2000356	3460.41	
27	2009661	2000566	3479.23	
28	2003134	2000798	3519.53	
29	2001038	2001028	3519.84	
30	2014569	2000047	3542.20	#7
31	2002760	2000120	3548.37	
32	2002483	2000074	3555.60	#9
33	2001038	2000494	3588.34	
34	2001345	2000490	3662.57	
35	2001902	2000145	3679.52	#5
...				
105	2011542	2000209	4447.89	#3
117	2002483	2000395	4513.97	#13
124	2002483	2000569	4569.23	#2
348	2009661	2000224	5637.77	#10
365	2001754	2000240	5676.23	#4
5966	2000659	2000075	11866.00	#11

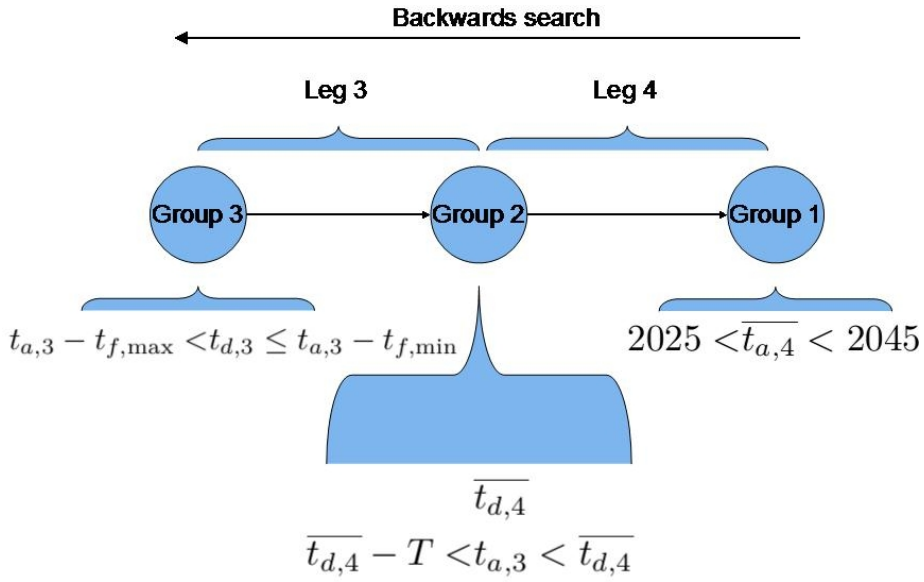

Figure 9.6: Departure dates determination for backward search.

the bottom of the list, the transfer corresponding to 11^{th} best GTOC2 sequence ranks rather badly (5966^{th} out of 16896 transfers) when compared to the other GTOC2 sequences, which is in conformity with the relatively poor GTOC2 objective function value corresponding to said sequence. The GTOC2 sequence that ranks best (21^{st}) corresponds to the GTOC2 winner.

In order to include as many GTOC2 solutions as possible while preserving the computational tractability of the problem, we set the maximum number of champions for the following leg, Leg 3, to 35. A sensitivity analysis on this parameter is conducted hereafter.

9.2.2 Leg 3

While the Leg 4 results are not influenced by the number of champion paths, the Leg 3 computations will depend on that parameter. Based on the results of Leg 4, we select the following numbers of champions for investigation: 21, 25, 30, 35. The maximum value, 35, was previously justified and the minimum value, 21, corresponds to the position of the best ranked GTOC2 partial sequence. The arrival windows can also be tuned and here we analyze three variants: 10, 15 and 20 years wide arrival windows.

Looking back on Table 9.8, we can see that the GTOC2 partial sequence that ranks best corresponds to both the GTOC2 winner (Politecnico di Torino, referred to as GTOC2 #1 from here on) and the GTOC2 #12 solutions. These are the only two GTOC2 sequences that will consistently be present in Leg 3 calculations, throughout the selected spectrum of number of champions. We therefore focus, in the following sections, on both sequences to assess the quality of the Leg 3 calculations.

10-Years Arrival Window

Adopting a 10-years arrival window for the backward search multi-path NNH does not yield very positive results, as confirmed by looking at Table 9.9 where the Leg 3 rank of the GTOC2 solutions that ranked higher than 35 in Leg 4 is given. Two GTOC2 partial sequences, GTOC2 #9 and #5, are only present in Leg 3 computations for a number of champions equal to 35 (cf. Table 9.8). In this case, the GTOC2 sequences #9 and #5 rank 1889^{th} and 4537^{th}, respectively, which are rather poor standings. The GTOC2 #6 and #7 sequences change relative positions from Leg 4 to Leg 3, meaning that the asteroid `2000047` is a more promising arrival asteroid for Leg 3 than asteroid `2000110`. Despite ranking better than GTOC2 #6 sequence (200^{th} versus 604^{th} with 35 champion paths), GTOC2 #7 sequence is discarded from Leg 3 computations at an earlier stage due to its weaker rank in Leg 4. By design, the GTOC2 #1 and #12 sequences transit to the Leg 3 computations independently of the number of champions. Figure 9.7 plots the rank of both sequences as a function of the number of champions. Note in Table 9.9 that, as long as they are not discarded, the ranking of the GTOC2 solutions improves when the number of champion paths is reduced, meaning that some lower-ranked sequences in Leg 4 end up being more attractive when the Leg 3 computations are added, and therefore outrank GTOC2 sequences in that leg.

Table 9.9: Leg 3 rank of the GTOC2 (partial) sequences as found by the backward multi-path NNH with 10-years arrival windows, using GS.

Number of	GTOC2 Rank					
champion paths	#1	#12	#6	#7	#9	#5
21	80	76	n.a.	n.a.	n.a.	n.a.
25	90	86	472	n.a.	n.a.	n.a.
30	98	94	537	189	n.a.	n.a.
35	101	97	604	200	1899	4537

The decreasing trend in ranking as the number of champions grows is clear visible in Figure 9.7. We can see that both sequences rank between approximately 80^{th} and 100^{th}, independently of the number of champions. Also noteworthy, the GTOC2 #12 partial sequence constantly outperforms the GTOC#1 partial sequence. The difference in their ranking is constant and equal to 4 (see Table 9.9) throughout the analysis.

Finally, let us turn to the computational cost of the backward search. The computational time needed up to Leg 3 with 10-years arrival windows is given in Table 9.10. The number of individual transfers (arcs) optimized is also listed. The algorithm takes between roughly two and three hours, depending on the number of champion paths. Despite the fact that only two out of the four legs are computed, this is a large increase with respect to the complete forward search from Section 9.1.2. This increase is due to two factors. First, the Leg 4 computations, not being single-sourced (i.e. not having a single departure body), take considerably longer than Leg 1 computations. Second, the magnitude of the numbers of champion paths is larger (the minimum number of champions considered for the backward search is larger than the maximum

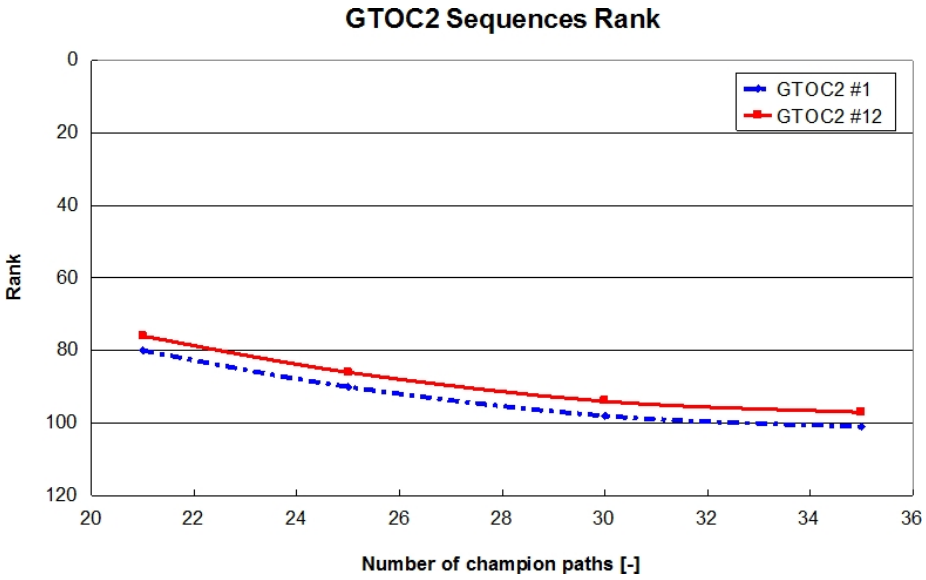

Figure 9.7: Rank of the GTOC2 #1 and #12 partial sequences for Leg 3 of the backward search with 10-years arrival windows, using GS.

number of champions in the forward search analysis), leading to an increased number of transfers needing to be computed.

Table 9.10: The computational time needed up to Leg 3 by the backward multi-path NNH with 10-years arrival windows, using GS.

Number of champion paths	21	25	30	35
Number of arcs evaluated	23196	24396	25896	27396
Computational Time [h]	1.99	2.27	2.84	3.22

Should the number of champion paths be kept constant from Leg 3 to Leg 2, none of the GTOC2 sequences would transit to the next phase since their ranking (cf. Table 9.9) is always below the number of champions. Including them in Leg 2 computations would require considerably broadening the number of champions and add a large amount of time to the already lengthy computations. Moreover, there would be no guarantee on the quality of the results. Therefore, the Leg 2 computations with 10-years arrival windows will not be pursued.

15-Years Arrival Window

Independently of the number of champions, the backward search with the multi-path NNH for 15-years arrival windows yields poor results, as seen in Table 9.11

which indicates the Leg 3 rank of the GTOC2 solutions that ranked above 35 in Leg 4. The GTOC2 #9 and #5 sequences are only present when considering 35 champion paths but rank very badly: 2559^{th} and 3107^{th}, respectively. As with 10-years arrival windows, the GTOC2 #6 and #7 sequences swap relative positions, indicating a relatively less costly Leg 3 transfer for the latter. The GTOC2 #1 and #12 sequences are the only ones not to be discarded throughout the analysis. The evolution of their ranking is illustrated in Figure 9.8. Table 9.11 displays the same trends as Table 9.9, namely with respect to the improvement of the rank of all GTOC2 solutions, when reducing the number of champion paths.

Table 9.11: Leg 3 rank of the GTOC2 (partial) sequences as found by the backward multi-path NNH with 15-years arrival windows, using GS.

Number of	GTOC2 Rank					
champion paths	#1	#12	#6	#7	#9	#5
21	110	104	n.a.	n.a.	n.a.	n.a.
25	125	119	664	n.a.	n.a.	n.a.
30	136	130	757	270	n.a.	n.a.
35	143	137	846	286	2559	3107

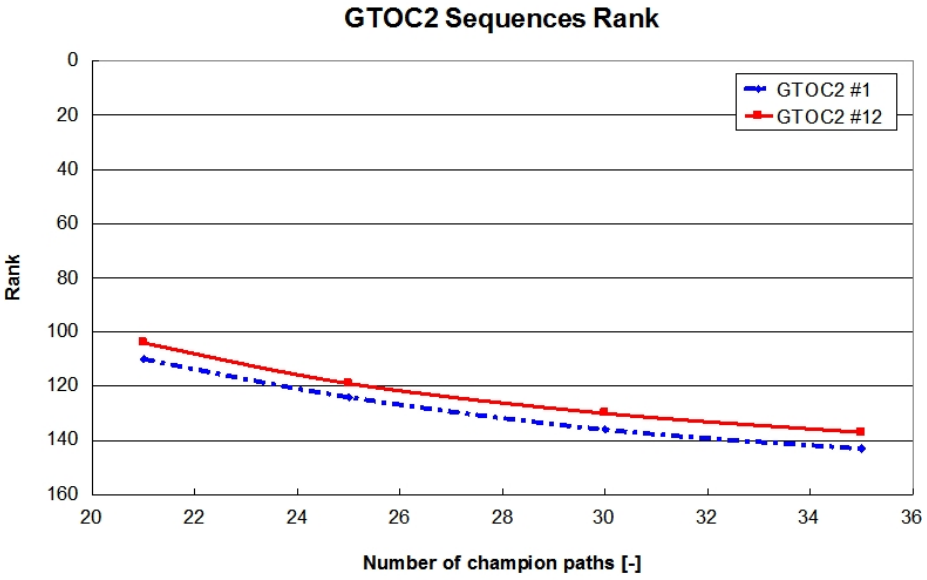

Figure 9.8: Rank of the GTOC2 #1 and #12 partial sequences for Leg 3 of the backward search with 15-years arrival windows, using GS.

Looking at Figure 9.8, both the GTOC2 #1 and #12 partial sequences rank below a 100^{th} at all times. The rank of both partial sequences decreases with the increase in the number of paths, akin what we observed in Figure 9.4. The

GTOC2 #12 partial sequence is consistently better ranked than the GTOC2 #1 partial sequence, albeit by less than 10 places.

Let us take a look at the computational times involved in the obtention of these results, as given in Table 9.12. With an arrival window of 15 years, the backward multi-path NNH needs between two hours and a half and four hours, depending on the number of champion paths, to complete the Leg 4 and Leg 3 computations. The larger the number of champion paths, the larger the number of Leg 3 transfers needing to be computed and therefore the longer the duration needed to complete those computations.

Given that the best GTOC2 solutions rank very poorly, the number of champions needed to include them in the Leg 2 computations would need to be increased. This would lead to even larger computational times, greatly hindering the computational efficiency of the algorithm. We therefore decide not proceed with the Leg 2 computations.

Table 9.12: The computational time needed up to Leg 3 by the backward multi-path NNH with 15-years arrival windows, using GS.

Number of champion paths	21	25	30	35
Number of arcs evaluated	23196	24396	25896	27396
Computational Time [h]	2.65	2.95	3.50	3.92

20-Years Arrival Window

With 20-years arrival windows, the Leg 3 results found with the backward search for Leg 3 are worse when compared to the 15-years arrival windows results (see Table 9.13). The exception is the GTOC2 #6 partial sequence, which ranks slightly better with 20-years arrival windows: 810^{th} instead of 846^{th} for 15-years windows, with 35 champion paths. The GTOC2 #9 and #5 partial sequences are discarded from Leg 3 computations for all the numbers of champions considered, except for 35 champion paths. For that value, their rank is quite poor: the GTOC2 #9 sequence ranks 3045^{th} while the GTOC2 #5 sequence ranks 3661^{th}. Akin what we observed for 15-years arrival windows, the Group 2 asteroid from GTOC2 #7 sequence is a more interesting arrival target than that of GTOC2 #6 sequence, as reflected by the change in their relative ranks from Leg 4 to Leg 3. As expected, the GTOC2 #1 and #12 sequences are preserved throughout the analysis. The evolution of their ranking as a function of the number of champion paths is plotted in Figure 9.9. Similarly to what happened with 15-years arrival windows, the rank of all GTOC2 partial sequences improves when we restrict the number of champions, as long as they are not discarded.

Based on Figure 9.9, the GTOC2 #1 and #12 sequences never rank above a 100^{th}, as was the case for 15-years arrival windows. Note however that the difference between their ranks is larger than previously. They are up to 26 places appart, the GTOC2 #12 partial sequence continuing to overrank the GTOC2

Table 9.13: Leg 3 rank of the GTOC2 (partial) sequences as found by the backward multi-path NNH with 20-years arrival windows, using GS.

Number of champion paths	GTOC2 Rank					
	#1	#12	#6	#7	#9	#5
21	132	112	n.a.	n.a.	n.a.	n.a.
25	152	129	645	n.a.	n.a.	n.a.
30	167	144	734	321	n.a.	n.a.
35	177	151	810	344	3045	3661

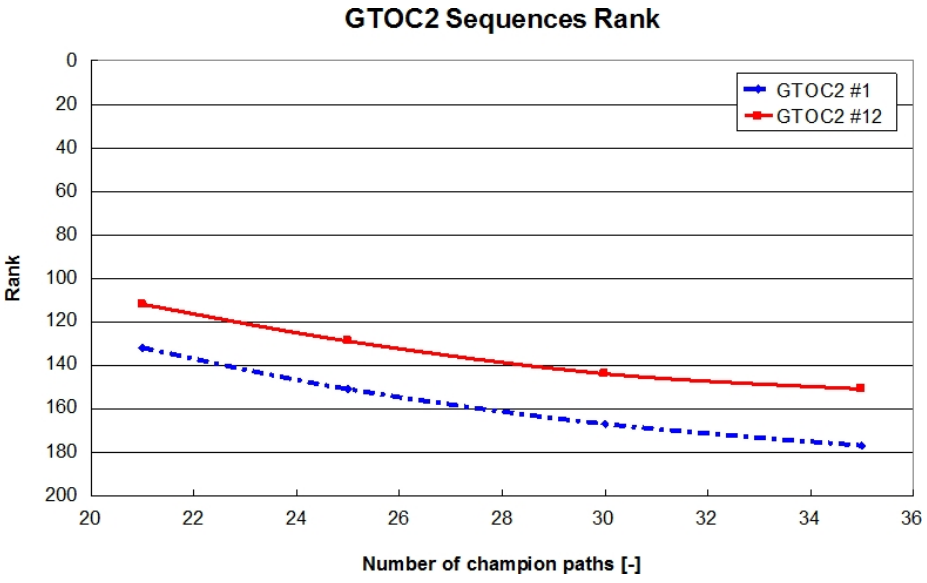

Figure 9.9: Rank of the GTOC2 #1 and #12 partial sequences for Leg 3 of the backward search with 20-years arrival windows.

#1 one, for all numbers of champions. Both ranks decrease with the increase in the number of champion paths.

In terms of computational effort, the larger arrival windows lead to an increase, with respect to the previous results with 15-years arrival windows, in the computational time needed. This quantity is listed in Table 9.14 for each of the number of champions investigated. The backward search completes in three to four and half hours, depending on the number of champions. Intuitively, the larger the number of champions, the larger the computational effort needed to compute Leg 3 results.

Table 9.14: The computational time needed up to Leg 3 by the backward multi-path NNH with 20-years arrival windows, using GS.

Number of champion paths	21	25	30	35
Number of arcs evaluated	23196	24396	25896	27396
Computational Time [h]	2.94	3.25	3.82	4.26

We reach the same conclusion as for 15-years arrival windows: with the GTOC2 sequences ranking very poorly and the computational effort being considerable, attempting to enlarge the number of champions for Leg 2 will probably lead to increased complexity, increased computational cost and no guarantee on the improvement of the GTOC2 partial sequences ranking. We therefore do not proceed to Leg 2 computations.

9.2.3 Conclusions

In order to compare the backward search results across the three different arrival windows, we plot the ranking of GTOC2 #1 and the computational time up to Leg 3 as a function of the number of champion paths in Figures 9.10 and 9.11, respectively.

If we focus on the GTOC2 winner sequence ranking, whose evolution is shown in Figure 9.10, it varies from about 80^{th} (for small arrival windows and a small number of champions) to approximately 180^{th} (with large arrival windows and a large number of champions). The ranking of the GTOC2 winner deteriorates when we increase the arrival windows and/or the number of champion paths. The larger the arrival window, the more considerable is the deterioration of the GTOC2 winner ranking. This translates, in Figure 9.10, into the increasingly steeper slopes as the arrival windows becomes lengthier.

Concerning the computational effort inherent to the different arrival windows, as depicted in Figure 9.11, the computational cost increases almost linearly with the number of champions. Interestingly, the slopes describing the computational effort appear to be roughly the same, which was not the case in Figure 9.5. On the other hand, the gap between the slopes from 10 to 15 is larger than from 15 to 20, a situation previously observed in Figure 9.5. Throughout the analysis, the computational time is bound between two hours and four and

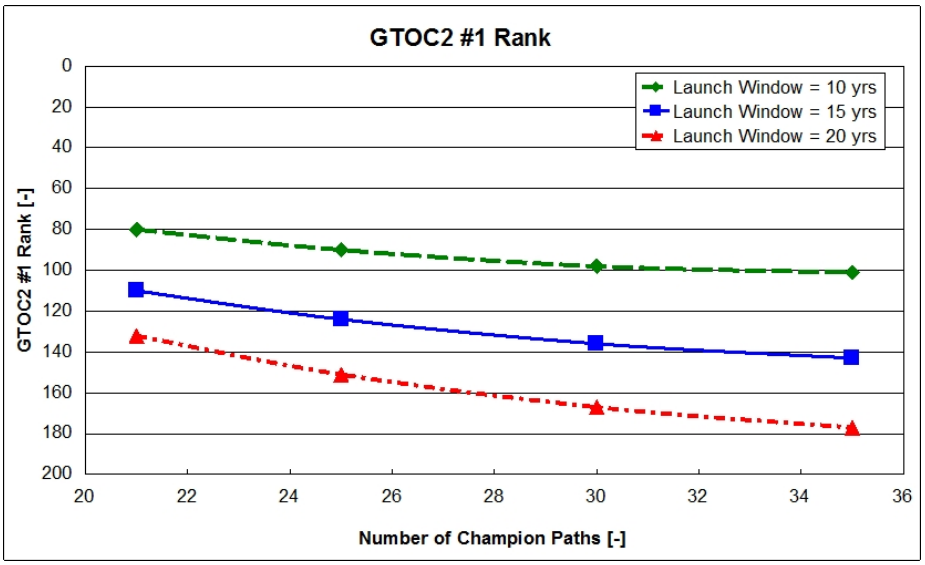

Figure 9.10: Rank of the GTOC2 winner in the backward search up to Leg 3 as a function of the number of paths, using GS.

half hours, which is considerably larger than the time needed by the forward search, especially considering that these values only correspond to half the total number of legs.

While the Leg 4 results seemed promising, namely due to the relatively high rank of the GTOC2 winning partial sequence, Leg 3 computations did not produce satisfactory results. The GTOC2 solution that, up to Leg 3, ranks best (GTOC2 #12, which was disqualified from the competition due to constraint violations) is below the 75^{th} place at all times. The GTOC2 winner partial sequence follows shortly behind. The computational times involved are quite large, considering that only the second half of the mission is analysed. In fact, they are too long to contemplate pursuing the computations for Legs 1 and 2, since doing so would mean increasing the number of champions. Extrapolating from Figure 9.11, this would lead to even larger computational efforts, which would compromise the efficiency of the search beyond usefulness.

Given that the results of the backward search are promising for Leg 4 but disappointing for Leg 3, it is suggested to select Leg 3 as patch leg when performing a multi-path bi-directional NNH search. Due to the timeframe of the MSc Thesis, such search was not performed. Combining the results above with those from Section 9.1 suggests that the only GTOC2 sequence to come out of such search, within a reasonable computational time, would be GTOC2 #12, provided the number of champions is larger than 20 and the departure windows length set to 20 years. This sequence, or more precisely its corresponding GTOC2 solution, was disqualified due to constraint violations but led to a larger objective function value than the equally constraint-violating solution submitted by the TU Delft-led team.

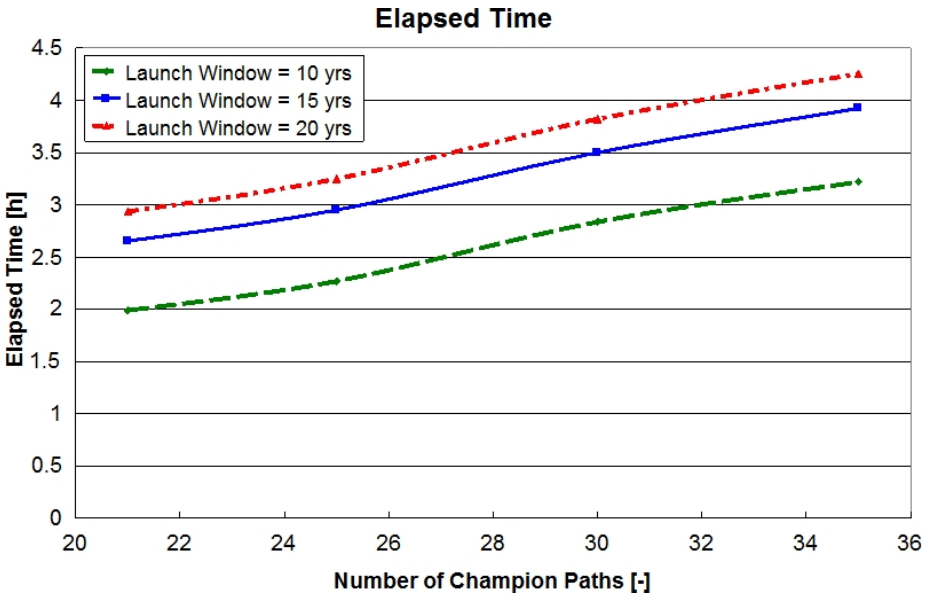

Figure 9.11: Computational time of the backward search up to Leg 3 as a function of the number of paths, using GS.

9.3 Forward Search, Optimal Final Mass to Time-of-Flight Ratio

We now attempt a forward search with the multi-path NNH where the arc costs to be optimized by GS are given by Equation (6.1.4), where the initial mass is taken to be $m_0 = 1500$ kg. Unlike the two previous sections, the time-of-flight is now taken into account. However, since the arc costs of the different legs are computed separately, only the time-of-flight for the transfers is evaluated. Waiting times and total mission time are therefore not evaluated here.

The parameters for the first leg are the same as in Section 9.1: 2015-2035 departure window, time-of-flight between 10 and 610 days, both parameters discretized in 10 days segments.

9.3.1 Leg 1

As discussed in Section 9.1, the Leg 1 transfer computations are not affected by the number of champions since all transfers need to be computed. Moreover, the departure window (2015-2035) is kept constant. Table 9.15 lists the 30 best Leg 1 transfers found by minimum J_1 value, as well as the GTOC2 Leg 1 transfers that rank below 30^{th}.

Table 9.15 reveals that 7 GTOC2 partial sequences, including the GTOC2 winner, belong the in the top-30 of best Leg 1 transfers evaluated based on J_1. Refining the top transfers shows that 6 GTOC2 partial sequences are ranked in the top-ten. One should note however that the `Earth - 3250293` transfer (8^{th}) alone accounts for four different GTOC2 solutions. The GTOC2 winner (28^{th}) does not figure in the top-ten transfers. The remaining 6 GTOC2 partial

Table 9.15: The 30 best Leg 1 transfers according to J_1 and GTOC2 solutions ranking, using GS.

Rank	Group 4 Asteroid	Minimum J_1 [yrs/kg]	GTOC2 Rank
1	3156302	0.000071	
2	3297182	0.000092	
3	3068066	0.000113	
4	3024030	0.000272	
5	3177193	0.000275	
6	3329255	0.000280	#8
7	3017309	0.000283	#5
8	3250293	0.000322	#2, #6, #10, #12
9	3293923	0.000371	
10	3071939	0.000394	
11	3167367	0.000400	
12	3297379	0.000462	
13	3293922	0.000477	
14	3297629	0.000519	
15	3120863	0.000519	
16	3160723	0.000562	
17	3124996	0.000571	
18	3262569	0.000574	
19	3054338	0.000575	
20	3144155	0.000590	
21	3278402	0.000604	
22	3263232	0.000626	
23	2099942	0.000643	
24	3016523	0.000677	
25	3324656	0.000689	
26	3338368	0.000710	
27	3070801	0.000728	
28	3258076	0.000743	#1
29	3114023	0.000747	
30	3153530	0.000754	
...			
38	3177202	0.000939	#14
44	3288933	0.001007	#7
59	3343104	0.001318	#11
68	3170221	0.001589	#3,#4,#9,#13

sequences rank below 40^{th}, with the particularity of the GTOC2 #11 solution not ranking last (unlike what was observed in Tables 9.1 and 9.8) and therefore outclassing solutions with higher GTOC2 cost function values, e.g. GTOC2 #3.

Comparing Tables 9.15 and 9.1 allows to establish a few observations regarding the suitability of cost function J_1. First note that, as a whole, GTOC2 sequences have a worse ranking: the GTOC2 sequence that ranks best in Table 9.15, GTOC2 #8, is 6^{th}, while in Table 9.1 the best GTOC2 sequence, GTOC2 #2 among others, ranks 2^{nd}. Moreover, with J_1, the last GTOC2 sequence, GTOC2 #3 among others, ranks 68^{th}, which is 28 positions below the last GTOC2 sequence with J_2, GTOC2 #11. Which brings into focus the fact that 2 of the 5 best GTOC2 solutions, namely GTOC2 #3 and #4, are the worst classified GTOC2 sequences according to J_1. The GTOC2 runner-up drops a few places from Table 9.1 to Table 9.15. On the other hand, the GTOC2 #1 sequence ranks better with J_1 than J_2. This suggests that using J_1 as cost function may allow to reproduce the GTOC2 winning sequence.

Figure 9.12 is a histogram of the J_1 values of Leg 1 transfers. Unlike Figure 9.1, where the histogram for the final mass fraction was plotted, a lower J_1 value represents a better solution. Figure 9.12 seems to indicate that the J_1 cost function does not allow to clearly identify promising candidates, as the vast majority of the transfers lies around $J_1 = 0.005$ years/kg. Note that four transfers have a J_1 value larger than 0.1 years/kg and are therefore not depicted in Figure 9.12.

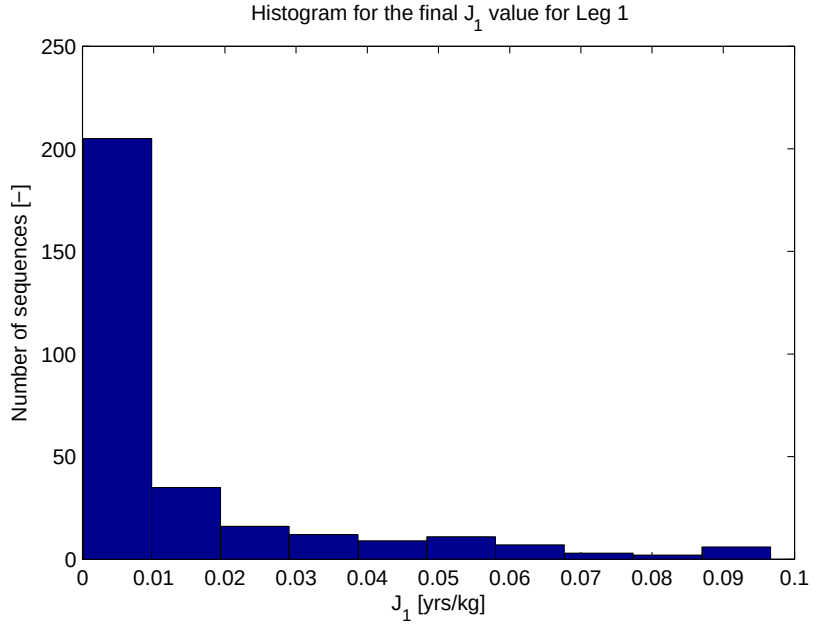

Figure 9.12: Histogram of J_1 values for Leg 1 transfers, using GS.

The histogram in Figure 9.13 allows to have a closer look at the distribution of J_1 for the top transfers found. Doing so contradicts the idea transmitted by Figure 9.12 regarding the suitability of cost function J_1 to allow a clear and

rapid identification of the top-transfers. There are 43 transfers with J_1 values below 0.001 years/kg. The most noteworthy feature of this histogram is the gap between the three leading transfers (around $J_1 = 0.0001$ years/kg) and the remaining solutions found. It seems to indicate that these three transfers will tend to be present in the top transfers of the following legs due to their path cost being relatively better than the rest. Unfortunately, none of these three transfers correspond to GTOC2 solutions.

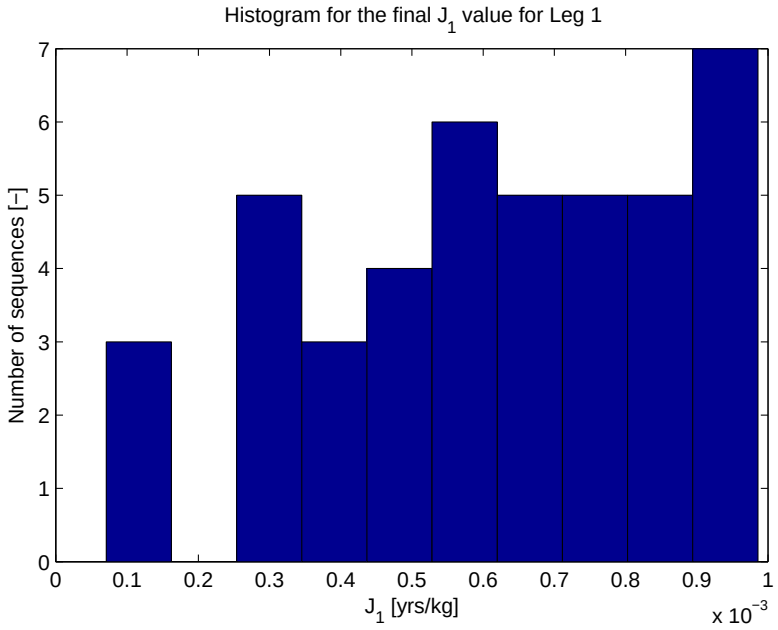

Figure 9.13: Zoomed-in histogram of J_1 values for Leg 1 transfers, using GS.

9.3.2 Leg 2

After the analysis of the initial leg, the multi-path NNH proceeds to Leg 2 computations. Akin what was done in the previous sections, Leg 2 results are obtained under varying conditions, namely in terms of departure window (10, 15 and 20 years) and number of champions (10, 15 and 20 champions).

10-Years Departure Windows

Analyzing Leg 2 results for 10-years departure windows, and namely the ranking of the partial GTOC2 sequences given in Table 9.16, reveals that no partial GTOC2 combination would progress to Leg 3 computations. Indeed, the partial sequence that ranks best, GTOC2 #2[5], ranks below the champion threshold, independently of the value of that threshold. The other GTOC2 partial sequences that transit from Leg 1, i.e. GTOC2 #5, #6 and #12, consistently rank below 100^{th}. The GTOC2 winner ranks below 20^{th} in Leg 1 and therefore

[5]Up to and including Leg 2, the GTOC2 #2 and #10 partial sequences are identical.

is not present in Leg 2 computations. Also, GTOC2 #8 does not follow the 4 - 3 - 2 - 1 asteroid group sequence and was therefore not considered in Leg 2 evaluations.

Table 9.16: Leg 2 rank of the GTOC2 (partial) sequences as found by the forward multi-path NNH with 10-years arrival windows, using GS.

Number of champion paths	GTOC2 Rank			
	#2, #10	#12	#6	#5
10	14	110	119	216
15	29	168	178	326
20	34	209	221	407

Increasing the number of champions leads to a drop in ranking for all partial GTOC2 sequences. This signifies that there is a shift in power from Leg 1 to Leg 2: a certain number of transfers that ranked lower in the initial leg lead to better transfers in the second leg and end up overtaking the sequences which were ahead in Leg 1. Therefore, increasing the number of champions does not increase the probability of allowing a partial GTOC2 solution to transit to the next leg as it simply leads to a larger number of transfers that outrank the GTOC2 combinations. It does however, as expected, lead to an increase in computational time, as shown in Table 9.17. This table also lists the number of asteroid pairs (arcs) evaluated.

Table 9.17: The computational time needed up to Leg 2 by the forward multi-path NNH with 10-years arrival windows, using GS.

Number of champion paths	10	15	20
Number of arcs evaluated	3338	4838	6338
Computational Time [min]	15.58	21.23	27.22

15-Years Departure Windows

Table 9.18 gives the Leg 2 ranking of the partial GTOC2 sequences, as obtained based on the minimum J_1 value for departure windows of 15 years. As with 10-years departure windows, none of GTOC2 partial sequences would transit to Leg 3 computations as none rank within the established number of champions. The relative ranking of the partial GTOC2 sequences is the same for all numbers of champions: GTOC2 #2, #12, #6 and finally #5. This is, incidentally, the same order as with 10-years departure windows. Note that the latter ranks above the three other GTOC2 sequences in Leg 1 computations, which means that asteroid `3250293` is a more interesting Leg 2 departure asteroid than asteroid `3017309`, with respect to J_1 values.

Table 9.18: Leg 2 rank of the GTOC2 (partial) sequences as found by the forward multi-path NNH with 15-years arrival windows, using GS.

Number of champion paths	GTOC2 Rank			
	#2, #10	#12	#6	#5
10	16	137	146	255
15	33	206	217	386
20	40	255	268	476

As in the previous scenario, increasing the number of champions leads to the GTOC2 sequences falling in the Leg 2 ranking, as well as to an increase in execution time (see Table 9.19). Also, notice the drop in ranking for all GTOC2 sequences and for all numbers of champions, from 10- to 15-years departure windows. This indicates that increasing the length of the departure windows allows the cost of other sequences to decrease in a more pronounced fashion than that of GTOC2 partial sequences. In other words, partial GTOC2 sequences profit less than other sequences from increasing the departure windows.

Table 9.19: The computational time needed up to Leg 2 by the forward multi-path NNH with 15-years arrival windows, using GS.

Number of champion paths	10	15	20
Number of arcs evaluated	3338	4838	6338
Computational Time [min]	21.17	32.80	39.59

20-Years Departure Windows

Increasing the departure windows further to 20 years does not yield better results. As seen in Table 9.20, none of the partial GTOC2 combinations are part of the different numbers of champions. While their order in the classification is unchanged, all GTOC2 sequences drop a bit further down the ranking, with respect to the scenario with 15-years departure windows. Increasing the number of champions only leads to an increase in the computational times needed to solve the problem, as listed in Table 9.21.

9.3.3 Conclusions

Focusing on the Leg 2 rank of GTOC2 #2 (Figure 9.14) and on the computational times needed to solve the problems (Figure 9.15), one can draw certain conclusions.

With respect to the ranking of GTOC2 #2, the GTOC2 solution that systematically ranks highest in Leg 2, it never falls within the specified number of champions, independently of the departure window. Moreover, increasing the

Table 9.20: Leg 2 rank of the GTOC2 (partial) sequences as found by the forward multi-path NNH with 20-years arrival windows, using GS.

Number of champion paths	GTOC2 Rank			
	#2, #10	#12	#6	#5
10	25	147	156	273
15	43	224	236	415
20	50	278	293	517

Table 9.21: The computational time needed up to Leg 2 by the forward multi-path NNH with 20-years arrival windows, using GS.

Number of champion paths	10	15	20
Number of arcs evaluated	3338	4838	6338
Computational Time [min]	29.86	41.42	51.81

number of champions leads to a degradation of the Leg 2 ranking, since it yields a larger number of sequences that overtake GTOC2 #2 from Leg 1 to Leg 2. The minimum number of champions that allow GTOC2 #2 to be part of Leg 2 computations is 8, its classification among Leg 1 transfers. However, there is confidence that reducing the number of champions to this number will not allow GTOC2 #2 to transit to Leg 3 computations either. The best Leg 2 ranking of GTOC2 #2 is consistently achieved for departure windows of 10-years. Expanding the search space to larger departure windows does not profit to the GTOC2 #2 (nor to any other GTOC2 partial sequence) as its classification is worse across all numbers of champion paths. Increasing the search space allows to find more interesting transfers for other sequences, which end up outranking GTOC2 #2.

In terms of computational effort, the results show the expected trends: increasing either the number of champions or the length of the departure windows leads to longer execution times, as the size of the problem grows. However, it appears that the multi-path NNH is quite efficient, as it solves half the of the largest problem (20 champions with 20-years departure windows) in less than one hour. Extrapolating to obtain an estimate of the time needed to solve the complete problem, this would lead to a total execution time of about 2 hours. Note that this is approximately the same value found when using J_2 to evaluate the arc costs (see Table 9.7), which means that there is no added complexity when replacing J_2 with J_1 as arc cost function. However, unlike J_2, this cost function does not allow the forward search to reproduce any of the GTOC2 solutions within the bounds considered.

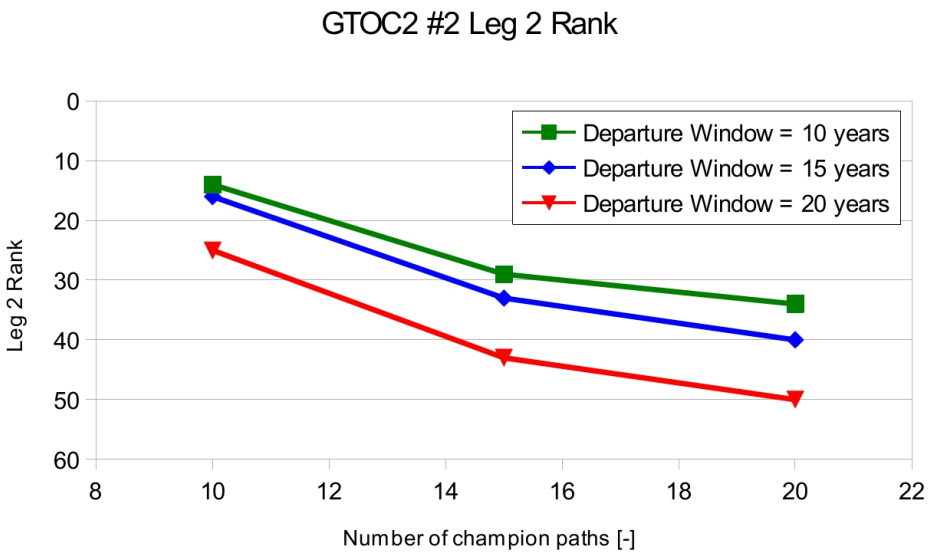

Figure 9.14: Rank of GTOC2 #2 in the J_1 forward search up to Leg 2 as a function of the number of paths, using GS.

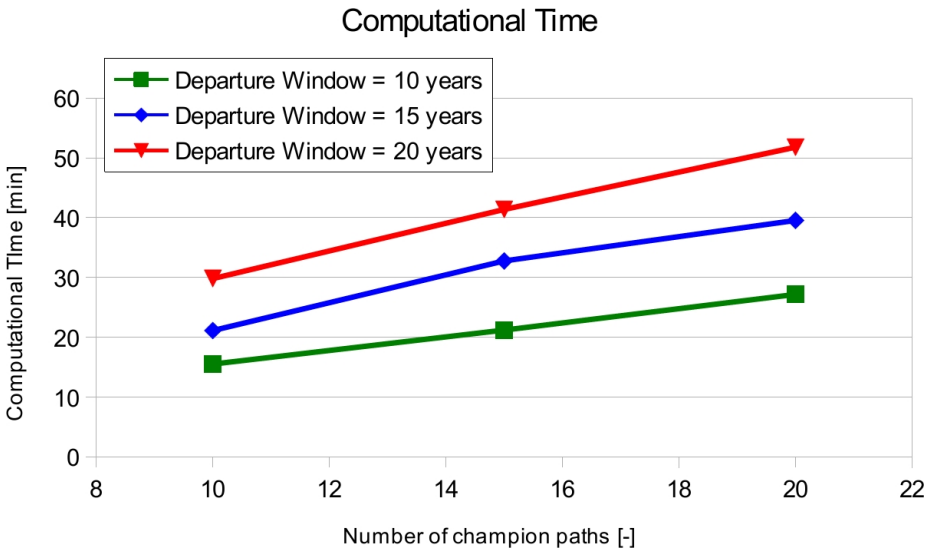

Figure 9.15: Computational time of the J_1 forward search up to Leg 2 as a function of the number of paths, using GS.

Chapter 10

Reduced Asteroid Pool

In light of the results obtained in Chapter 9, it was decided to focus on a reduced set of asteroids in order to facilitate the identification of possible improvements to the model and to the proposed tool to solve the combinatorial aspect of GTOC2. The reduced set of asteroids is the collection of asteroids corresponding to complete GTOC2 solutions corresponding to the pre-established asteroid group order: 4 - 3 - 2 - 1. These are given in Table 10.1.

Table 10.1: The reduced set of asteroids composed of GTOC2 asteroids.

Group 4	Group 3	Group 2	Group 1
3258076	2000060	2000058	2002959
3250293	2000149	2000569	2002483
3170221	2000574	2000209	2011542
3343104	2001990	2000240	2001754
3017309	2000443	2000490	2001345
3288933	2000027	2000110	2001038
	2001707	2000047	2014569
	2001314	2000395	2000659
	2000043	2000074	2009661
	2000169	2000224	
		2000075	

There are in total 36 asteroids: 6 in Group 4, 10 in Group 3, 11 in Group 2, and 9 in Group 1. Given the pre-determined asteroid group sequence, this leads to 5940 different sequences. Given the low dimensionality of the problem, compared with the complete asteroid set, no use is made of champion paths: all possible sequences are evaluated.

Given that, in Chapter 9, the forward search with J_2 was the variant of the multi-path NNH that performed best, we will limit the analysis in the reduced asteroid set to this alternative. Section 10.1 deals with the results obtained with the cached forward search. Thereafter, Section 10.2 leans on the effect of the caching feature as it is removed from the full, forward search. Conclusions with respect to the suitability of DE as optimization technique for the individual

transfers are drawn in Section 10.3, after analysis of the results obtained with that method. Finally, each of the GTOC2 sequences[1] is optimized as a whole in Section 10.4 and a number of observations regarding the suitability of the high-thrust model as preliminary approximation of the GTOC2 model are made.

10.1 Forward Search, Optimal Final Mass, using GS

We start by applying the forward multi-path NNH to the reduced set of asteroids. The departure window for Leg 1 transfers is maintained as the original GTOC2 window: 2015-2035. The departure windows for the following transfers, starting at the time of arrival at the departure asteroid, vary within the same values as in Chapter 9: 10, 15 and 20 years. The time-of-flight is bound within 10 and 600 days.

10.1.1 Final Ranking

The final ranking among the reduced asteroid set of the 12 GTOC2 sequences is given in Table 10.2.

Table 10.2: The final ranking of GTOC2 sequences in the reduced asteroid set, using GS.

GTOC2	Departure Window		
Sequence	10 years	15 years	20 years
#1	801	4211	878
#2	36	432	559
#3	1416	47	59
#4	3704	3268	3923
#5	1370	2047	2520
#6	12	16	16
#7	50	79	2115
#9	2102	2995	3194
#10	122	2608	1493
#11	4019	4888	5172
#12	4576	66	50
#13	1771	2013	2147

The GTOC2 sequences display, in general, large fluctuations in their ranking as a function of the length of the departure windows. Only the GTOC2 #6 sequence is consistently well-ranked, ending in the top-20 sequences independently of the departure windows. On the other hand, the GTOC2 #1, #4, #5, #9, #11, and #13 sequences persistently rank poorly across all departure window values. No GTOC2 sequence outclasses GTOC2 #6 but GTOC2 #2, #3, #7, and #12 are present in the top-50 transfers, depending on the departure window considered.

[1] Each of the GTOC2 sequences who comply with the 4-3-2-1 asteroid group order.

The results found for the reduced set of asteroids with the forward multi-path NNH are rather poor, especially with respect to GTOC2 #1, which does not rank better than 801^{st} independently of the departure window. Except for GTOC2 #2 and #3, which punctually rank within the top-50, none of the top-5 GTOC2 sequences are identified by the multi-path NNH as promising combinations.

A relative ranking of the GTOC2 sequences was established in order to investigate whether the multi-path NNH succesfully identifies a given GTOC2 sequence as being more promising than the ones who ranked below in the GTOC2 competition. GTOC2 solutions with constraint violations, i.e. GTOC2 #12 and #13, were removed from this relative ranking for the sake of clarity. This relative ranking is illustrated in Figure 10.1. The light blue line allows to clearly visualize intersections between the ranking based on the GTOC2 objective function and the multi-path NNH relative classification. Note that, despite the indication in the abcissae axis, GTOC2 #8 was not evaluated since it does not respect the 4 - 3 - 2 - 1 asteroid group order.

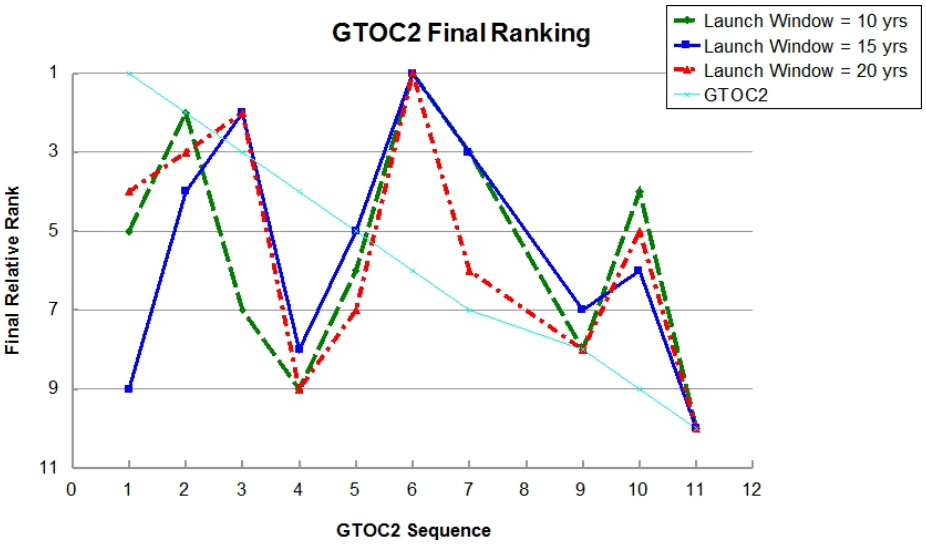

Figure 10.1: Relative ranking of GTOC2 sequences in the reduced asteroid set, using GS.

Looking at the peaks in Figure 10.1, one can state that, besides consistently (and accurately) identifying GTOC2 #11 as the worse sequence, the multi-path NNH does not lead to an accurate relative ranking of GTOC2 sequences. The most prominent examples are perhaps GTOC2 #4 and #6. The latter consistently ends at the top of the relative classification, there outranking 5 sequences which proved to be yield better objective function values during the GTOC2 competition. The former, on the other hand, ranks either 8^{th} or 9^{th} depending on the departure window considered, therefore falling four to five ranks (out of 11) when compared to the GTOC2 ranking. The GTOC2 winner sequence oscillates between the 4^{th} and 9^{th} across the different departure windows. Note also that the relative ranking shows considerable variations as a function of

departure windows length.

10.1.2 Leg Ranking

Let us take advantage of the reduced set of asteroids to investigate the evolution of the ranking of a few interesting GTOC2 sequences, across the leg computations. These are illustrated hereafter, in Figures 10.2 through 10.6, for GTOC2 #1, #2, #3, #4, and #6. Similar graphs for the remaining sequences can be found in Appendix D. Note that the negative trend observable in most graphs can be partly explained by the fact that the total number of unique sequences increases as more legs are added: a given sequence may rank last up to Leg 2 and in the top half of complete sequences and yet have a higher absolute rank in Leg 2 (out of 600 sequences) than in Leg 4 (out of 5940).

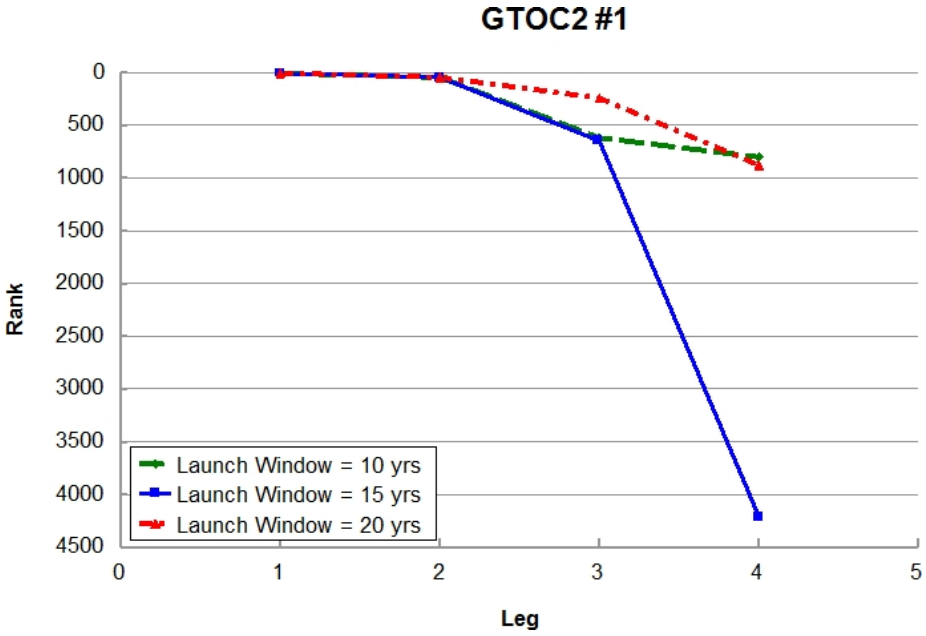

Figure 10.2: Ranking of GTOC2 #1 in the reduced asteroid set, using GS.

The rankings of GTOC2 sequences behave differently for different departure windows. Starting with GTOC2 #1 (Figure 10.2), its Leg 4 ranking is nearly the same for 10- and 20-years departure windows (about 800^{th}) but very different for 15-years departure windows and for the worst: approximately 4000^{th}. It is interesting to notice that up to Leg 3, the ranking for 10- and 15-years departure windows is nearly the same, only to largely differ in the last leg. This seems to indicate that the 15-years departure windows build up to a situation, in Leg 4, where the phasing between the Group 2 and Group 1 does not allow for a "low-cost" transfer. For 20-years departure windows, the phasing of the Group 3 and Group 2 asteroids leads to a more interesting Leg 3 transfer with respect to the other departure windows. However, the phasing in the last leg does not yield the same transfer as with 10-years departure windows and the ranking of GTOC2 plummets to the same level, i.e. about 800^{th}.

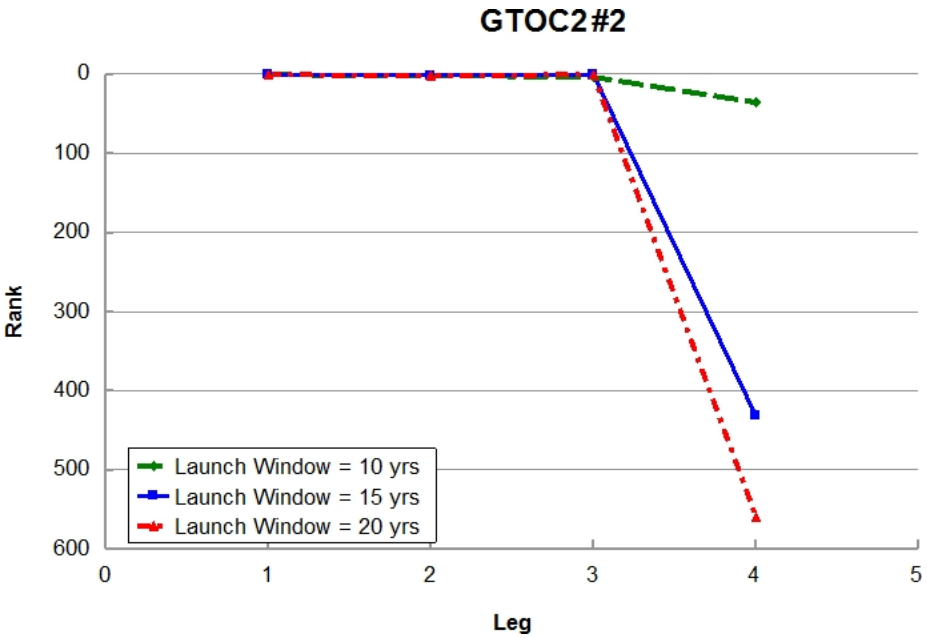

Figure 10.3: Ranking of GTOC2 #2 in the reduced asteroid set, using GS.

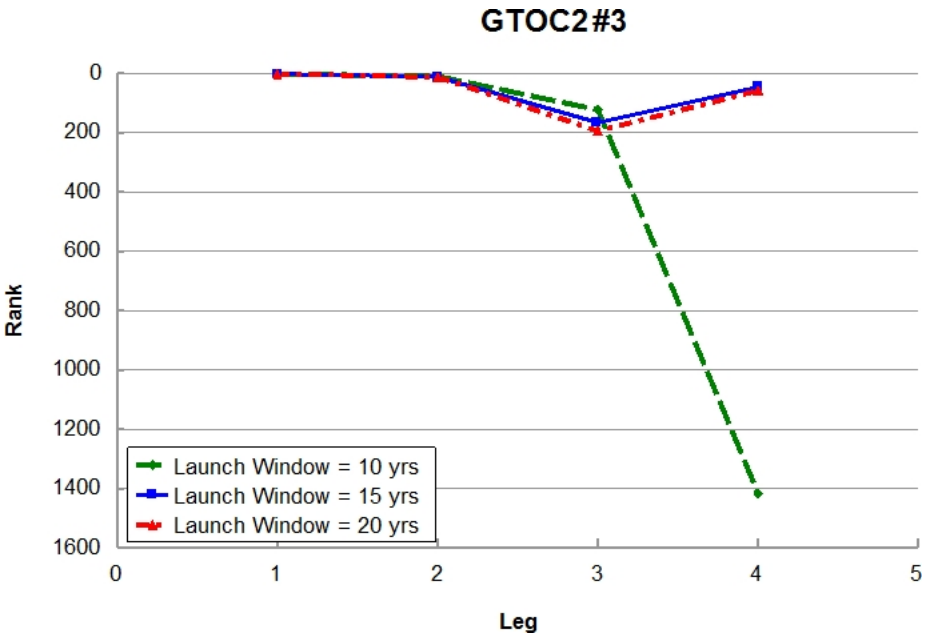

Figure 10.4: Ranking of GTOC2 #3 in the reduced asteroid set, using GS.

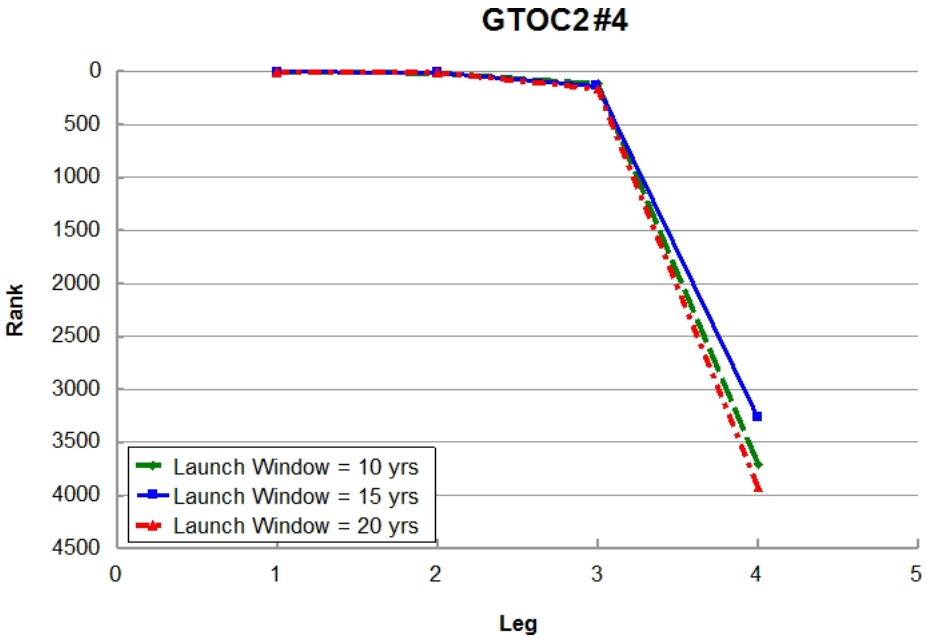

Figure 10.5: Ranking of GTOC2 #4 in the reduced asteroid set, using GS.

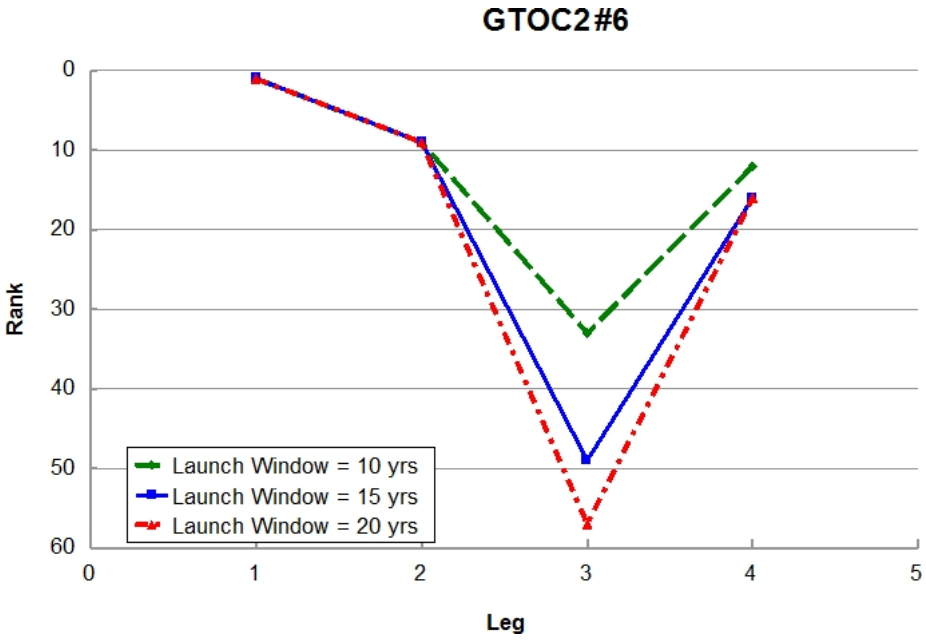

Figure 10.6: Ranking of GTOC2 #6 in the reduced asteroid set, using GS.

For GTOC2 #2 (Figure 10.3), the ranking is approximately constant across the three first legs, independently of the departure window. However, for the last leg, the larger the departure window, the lower the final rank of GTOC2 #2. Again, this is probably due to the heliocentric geometry of the Group 2 and Group 1 asteroids within the time interval considered for Leg 4 computations. The time interval for Leg 4 computations varies in length, but also in starting date, for the different departure windows. Otherwise, the best Leg 4 transfer, which occurs when considering 10-years departure windows, would also be found in the larger time intervals.

Akin GTOC2 #2, the ranking of GTOC2 #3 is roughly the same for the first three legs but not for the last leg, as shown in Figure 10.4. However, unlike the previous sequence, the smallest departure window leads to a considerably worse classification than the 15- and 20-years departure windows. This can also be attributed to phasing, 10-years departure windows leading, in Leg 4, to a range of orbital geometries for the Group 2 and Group 1 asteroids which does not contain the relatively better transfer obtained with 15- and 20-years departure windows. Also, the fact that the ranking of GTOC2 #3 is nearly the same for 15- and 20-years departure windows indicates that the resulting missions are probably the same in terms of departure and arrival dates at the different asteroids. Another interesting observation regarding the evolution of GTOC2 #3 ranking with the larger departure windows is that, unlike GTOC2 #1 and #2, GTOC2 #3 rises in the classification from Leg 3 to Leg 4, from approximately 180^{th} to about 50^{th}. This shows the limitations of the leg-per-leg approach of the multi-path forward NNH used in Chapter 9: sequences may rank rather poorly in an intermediate leg but subsequently reclaim higher classifications. With the implemented multi-path NNH, such sequences would be discarded as soon as they ranked below the specified number of champions. For the GTOC2 #3 sequence, setting the departure windows to 15 years and the number of champions in the multi-path NNH to 50 would lead to it being dicarded, despite ranking above 50^{th} in the final classification (see Table 10.2).

Figure 10.5 illustrates yet another pattern: the ranking of GTOC2 #4 is nearly not affected by departure window. Granted, the variation in Leg 4 ranking (between 3268^{th} and 3923^{rd}, see Table 10.2) is larger than for, e.g., GTOC2 #2, but the impact on the relative ranking is lesser (GTOC2 #4 is systematically in the lower half of the final classification) and therefore less relevant.

Finally, GTOC2 #6 (Figure 10.6) is the GTOC2 sequence that ranks best in Leg 4 in every scenario, as seen in Table 10.2 and Figure 10.1. While up to Leg 2 the ranking of GTOC2 #6 is approximately the same across all departure windows, the situation arises for the remaining two legs where the larger the departure window, the lower the ranking. Note however that the spectrum of the ranking is rather short. Therefore, the degradation of the ranking cannot be linked to the phasing of the GTOC2 #6 asteroids. In fact, the departure and arrival times for the different transfers are probably the same in all three scenarii. However, increasing the departure windows leads to other sequences having better phasing and therefore outranking GTOC2 #6. Note that the evolution of the GTOC2 #6 classification follows the same pattern as some of Figure 10.4 curves, namely with respect to the drop in ranking in Leg 3 with respect to Leg 2 and Leg 4. This would lead, depending on the number of champions specified, to the exclusion of GTOC2 #6 from the final sequences by the multi-path NNH, despite the attractive final ranking in the reduced set of

asteroids.

Based on the reduced set, it was shown that the forward search, multi-path NNH fails to identify GTOC2 sequences as interesting asteroid combinations, with the exception of GTOC2 #6. Moreover, among the GTOC2 sequences themselves, the tool cannot properly rank the combinations according to their GTOC2 fitness. Most combinations display a large fall in ranking when Leg 4 computations are added, adding to the notion that the correlation between high- and low-thrust optimal trajectories is worse for the last leg. The last two legs, and mostly Leg 4, show a visible sensitivity to the length of the departure window, meaning that the phasing is of particular importance for these transfers.

10.2 Full Forward Search, Optimal Final Mass, using GS

In the previous section, the best transfer for each combination of asteroids was saved and reused if the same asteroid combination needed to be evaluated more than once. Take for example the two following sequences:

```
Earth - 3258076 - 2000060 - 2000058 - 2002959
Earth - 3258076 - 2000149 - 2000058 - 2002959
```

These two sequences share the same Leg 4 transfer but have a different Leg 3 transfer. To avoid computing the `2000058 - 2002959` transfer twice, the best transfer found when assessing the first sequence is used in the second sequence. This caching of best transfer values wass intended to reduce the computational complexity of the problem and thus lead to a reduction of the multi-path NNH execution times. This feature may lead to infeasible sequences however, where the departure date from an asteroid occurs before the arrival date of the previous leg. The "recycling" of best transfers values was implemented under the working hypothesis that the departure windows considered were wide enough to allow repeatability of the heliocentric geometry. However, the oscillation of the GTOC2 sequences ranking across the different departure windows, as seen in Table 10.2, leads to the dismissal of this hypothesis. We therefore proceed to a full (as opposed to the recycled variant) forward search with the multi-path NNH.

10.2.1 Final Ranking

The resulting final ranking of the 12 GTOC2 sequences is given in Table 10.3.

Akin Table 10.2, Table 10.3 points to large variations in the final ranking of GTOC2 sequences as a function of the departure windows. This reinforces the importance of phasing when analysing the GTOC2 sequences. The sequence GTOC2 #11 has a very poor classification independently of the departure window, the highest rank achieved being 4746^{th} for 15-years departure windows. Along with this sequence, GTOC2 #1, #5, and #10 rank below 1000^{th} at all times. The only sequence to systematically rank above 100^{th} is GTOC2 #3. The GTOC2 #2, #6, and #7 occasionally rank in the top-20. For 10-years departure windows there are four GTOC2 sequences in the top-50 combinations, while there is only one (GTOC2 #6) in the 50 best combinations with departure

Table 10.3: The final ranking of GTOC2 sequences in the reduced asteroid set without caching, using GS.

GTOC2	Departure Window		
Sequence	10 years	15 years	20 years
#1	1005	4684	1037
#2	12	499	671
#3	33	70	87
#4	112	195	246
#5	5537	2191	2784
#6	697	16	21
#7	7	1907	2353
#9	41	93	114
#10	2634	1308	1735
#11	4792	4746	5292
#12	1419	61	51
#13	74	1841	2405

windows of 15 and 20 years. Note that, according to methods description from Section 3.1.1, these four solutions (GTOC2 #2, #3, #7 and #9) were found via a preliminary screening based on optimal Lambert solutions. The fact that they rank rather high in the asteroid reduced set suggests that these teams may have used similar bounds on the departure windows.

The results for the reduced asteroid set obtained with a full forward multi-path NNH search are not satisfactory. The GTOC2 #1 sequence, which according to its GTOC2 classification should be one of the most promising combinations, ranks below 1000^{th} independent of the length of the departure windows. Only a handful of GTOC2 sequences is occasionally ranked in the top-50 combinations.

We now take a look, in Figure 10.7, to the relative ranking of the GTOC2 sequences. As earlier, GTOC2 solutions with constraint violations were left out of this ranking, and the light blue curve indicates the ranking based on the original GTOC2 objective function. Note that, despite the indication in the abcissae axis, GTOC2 #8 was not evaluated since it does not respect the 4 - 3 - 2 - 1 asteroid group order.

The "sawed" curves in Figure 10.7 contrast with the ideal, nearly linear curve shown in light blue. Meaning that the proposed model and tool do not accurately evaluate the relative fitness of each sequence according to their final GTOC2 cost function value. Depending on the departure window, either GTOC2 #6 or #7 are identified as the most promising sequence. The GTOC2 sequence that ranks last is GTOC2 #11 for the larger departure windows, which is an accurate assessment. For departure windows of 10 years however, that distinction falls upon GTOC2 #5. This sequence is also poorly ranked for the other departure dates. The GTOC2 #2, #3, and #4 sequences systematically reach the top-5 combinations among the GTOC2 features, which is an encouraging feature. The GTOC2 winner sequence, which is expected to rank high, oscillates between the 6^{th} and 9^{th} positions, according to the length of the departure

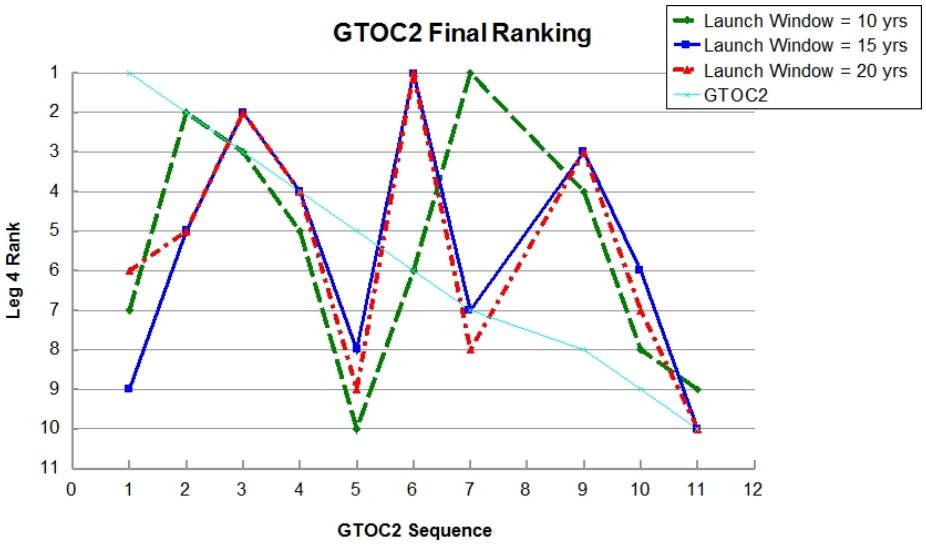

Figure 10.7: Relative ranking of GTOC2 sequences in the reduced asteroid set without caching, using GS.

windows.

10.2.2 Comparison with the cached search

We shall now proceed to comparing the results of the full search with the cached (or recycled) search in order to determine the impact of this feature on the quality of the solutions, as well as on the computational complexity of the problem. Figure 10.8 shows the ranking progression of the GTOC2 sequences when the caching feature is removed. A positive value corresponds to a better ranking in the full search, while a negative value indicates that the ranking of a particular sequence dropped when compared to the recycling of transfers.

The GTOC2 sequences that most profit from the full search are GTOC2 #4 and #9, both progressing about 3000 positions, independently of the departure windows length. Figure 10.8 shows that, besides these two sequences, the caching feature as the least impact for departure windows of 20 years. This can be explained by the fact that the largest the departure window, the largest the probability of heliocentric geometry repeatability. This correlation also explains why the ranking discrepancies are more frequent and in general larger for a departure windows of 10 years. GTOC2 #5 shows the largest degradation in ranking (a drop of about 4000 positions), which occurs precisely for 10 year departure windows. When considering 15-years departure windows, the largest discrepancies in ranking, besides the already mentioned GTOC2 #4 and #9, occur for GTOC2 #7 (an approximate fall of 2000 positions) and GTOC2 #10 (rise of about 1000 positions). These observations are reflected in the difference in relative ranking between both rankings, as shown in Figures 10.9, 10.10, and 10.11.

The improved relative ranking of GTOC2 #4 and #9 is visible in Figures

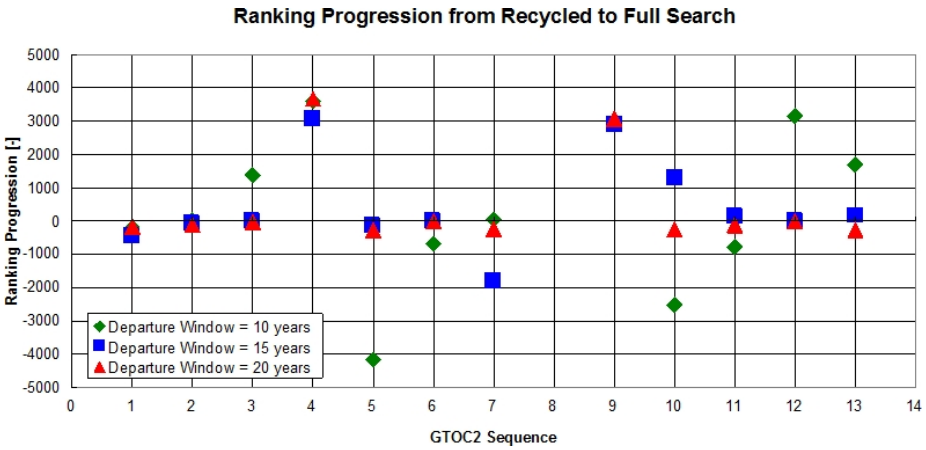

Figure 10.8: Ranking Progression of the GTOC2 sequences in the reduced asteroid set without caching, using GS.

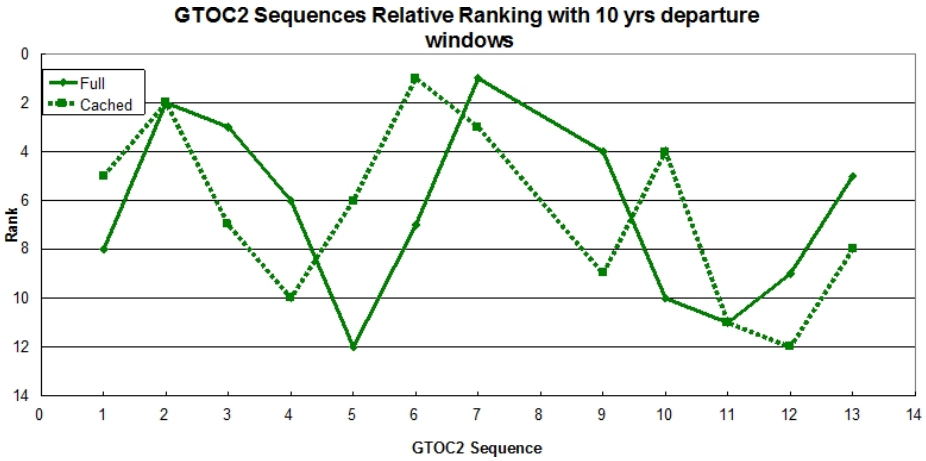

Figure 10.9: Relative ranking of the GTOC2 sequences in the reduced asteroid set for 10-years departure windows, using GS.

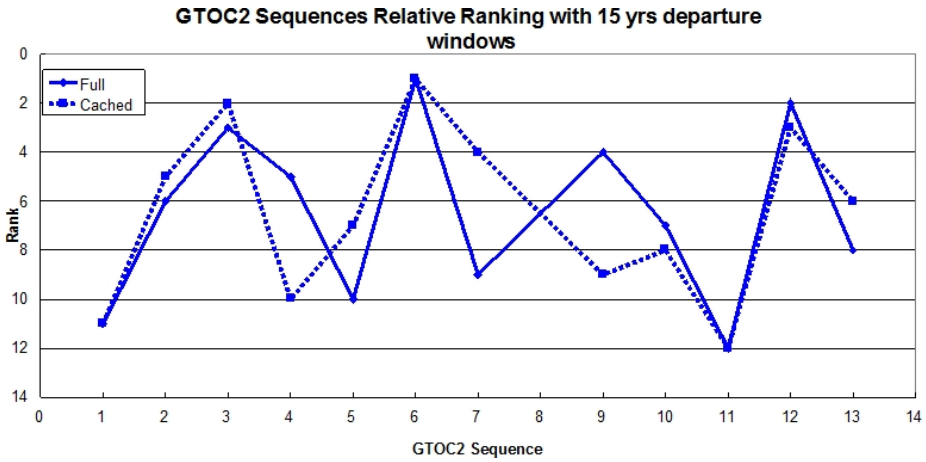

Figure 10.10: Relative ranking of the GTOC2 sequences in the reduced asteroid set for 15-years departure windows, using GS.

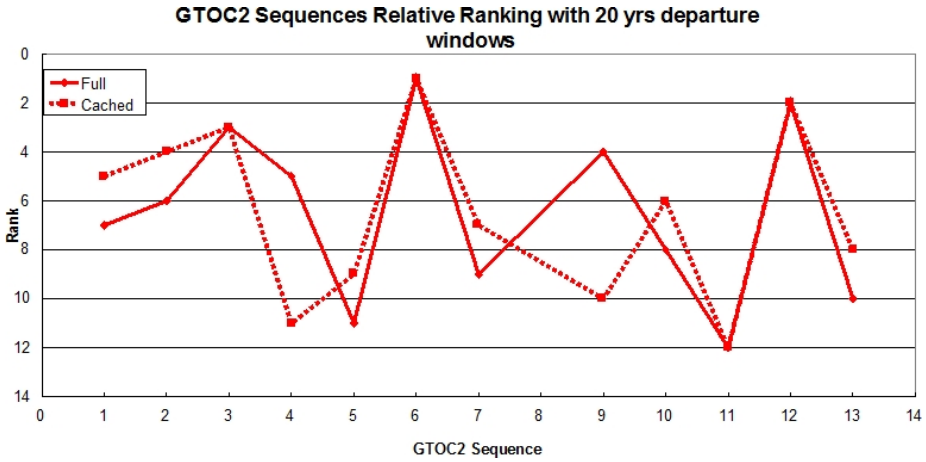

Figure 10.11: Relative ranking of the GTOC2 sequences in the reduced asteroid set for 20-years departure windows, using GS.

10.9 through 10.11. The decreasing impact of caching transfers with the increasing length of the departure windows can be seen in the fact that the larger the latter, the smaller difference between the *Full* and *Cached* curves.

Figures 10.9 through 10.11 only show the impact of the caching feature on the Leg 4 ranking. However, this feature also has an impact on the Leg 3 computations, as shown in Figures 10.12 and 10.13 for GTOC2 #1 with 10-years departure windows and GTOC2 #3 with 20-years departure windows.

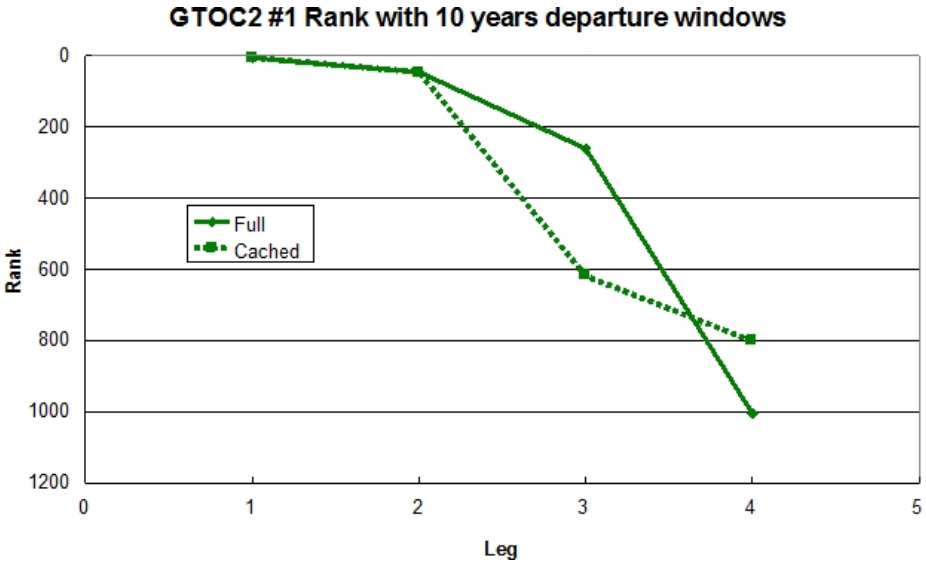

Figure 10.12: Ranking of GTOC2 #1 with 10-years departure windows in the reduced asteroid set, using GS.

In both Figures 10.12 and 10.13, the caching feature degrades the Leg 3 ranking of the GTOC2 sequences but improves their final ranking. While the improvement of the final ranking can be seen as a positive impact, the degradation of the Leg 3 classification may lead, depending on the number of champions, to the sequences being prematurely discarded in a champion-based multi-path NNH. Note that there is no difference in ranking up to Leg 2. This is due to the fact that, up to and including Leg 2, no individual transfer is repeated in different sequences and therefore no use is made of cached transfer values.

In terms of execution times, Figure 10.14 shows the computational time needed to solve the combinatorial problem on the reduced asteroid pool, as a function of both the departure windows length and of whether or not the transfers are cached.

As illustrated in Figure 10.14, removing the caching feature has serious consequences on the computational complexity of the problem. There is a factor of about 20 between the computational times needed by the algorithm to solve the reduced asteroid pool with and without recycling of transfer computations. On the other hand, there is no real improvement in the ranking when not caching transfers between asteroid pairs, except for GTOC2 #4 and #9, as seen in Figure 10.8. The former does not pertain to the top-100 combinations in the reduced asteroid pool while the latter is punctually present in the top-50 (see

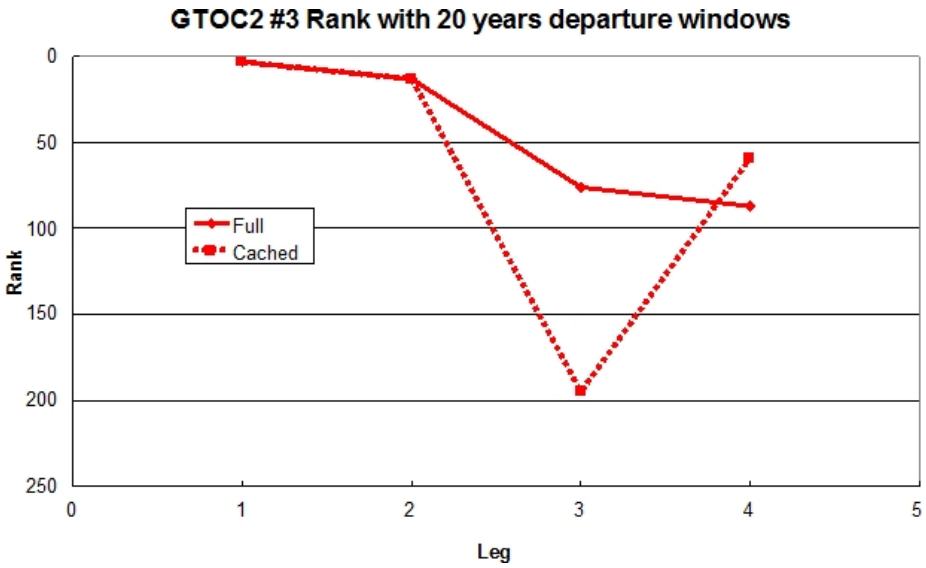

Figure 10.13: Ranking of GTOC2 #3 with 10-years departure windows in the reduced asteroid set, using GS.

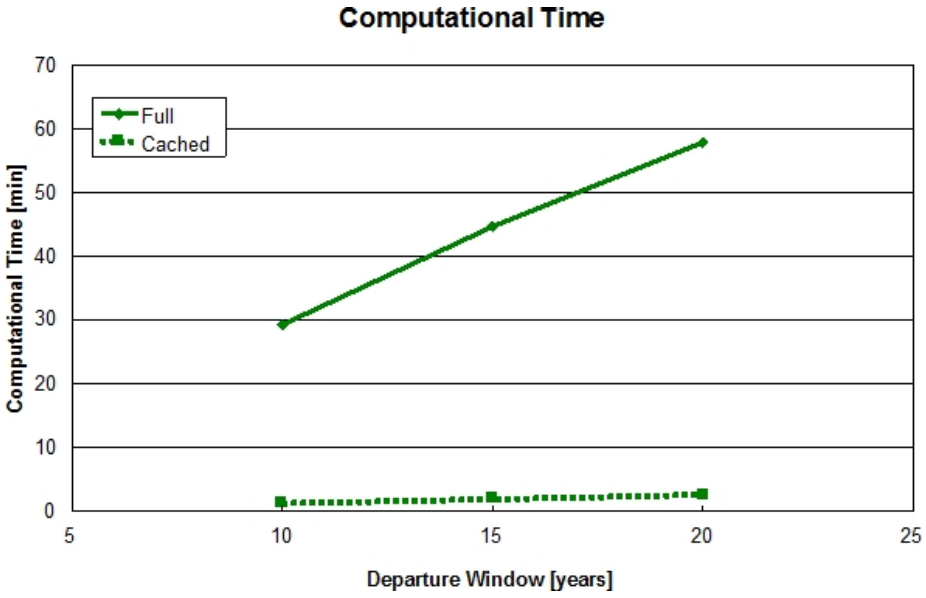

Figure 10.14: Computational time of the forward multi-path NNH search on the reduced asteroid pool as a function of the departure windows length, using GS.

Table 10.3). The effects of caching are more considerable when the departure windows are only 10 years wide. For larger departure windows, the caching feature seems to be interesting given that it effectively reduces the computational complexity without sizeable effects on the overall quality of the solutions.

10.3 Full Forward Search, Optimal Final Mass, using DE

We have seen in Section 6.2.2 that, after tuning of its parameters, DE was up to five times faster than GS. Hence, replacing GS with DE in the full search may lead to considerable computational effort improvements and therefore couteract the absence of the caching feature. We therefore proceed with the full search with DE as optimization technique for the evaluation of the individual arc costs. The parameters of the DE algorithm are as determined in Section 6.2.2: a population of 50 individuals, evolved for 150 generations.

10.3.1 Final Rankings

Starting with the analysis of the final rankings of the 12 GTOC2 sequences, as given in Table 10.4, one can see that there are large changes in ranking for all GTOC2 sequences as the departure windows length increases. The sequence least affected by this is GTOC2 #3, which manages to remain in the top-100 combinations across all departure windows length. GTOC2 #11 consistently ranks very poorly, independent of the length of the departure windows. The sequences GTOC2 #6 and #12 rank much better for the larger departure windows than for 10-years departure windows. The former even ranks in the top-25 combinations for 15- and 20-years departure windows. As when we used GS, the results with DE show that GTOC2 sequences ranking is, overall, better when considering 10-years departure windows: GTOC2 #2, #3, #7 and #9 are part of the top-40 combinations. Bear in mind that these four solutions relied on the optimal Lambert transfers at some point.

With respect to the relative ranking, Figure 10.15 is quite similar to Figure 10.7, which illustrated the relative ranking of GTOC2 sequences in the reduced asteroid set without caching using GS. The "sawed" patterns confirm that the multi-path does not lead to a relative ranking similar to that corresponding to the final GTOC2 objective function value. The GTOC2 winning sequence ranks below 6^{th} at all times, GTOC2 #5 is poorly classified with any departure windows length, and the top combination is bestowed on either GTOC2 #6 or #7. On the other hand, GTOC2 #2, #3 and #4 are systematically in the top-5 combinations, while GTOC2 #11 is always among the worst ranked sequences. Note that the relative rankings corresponding to departure windows of 15 and 20 years are quite similar when compared to the 10-years departure windows relative ranking.

10.3.2 Comparison with the GS results

The resemblance in absolute and relative ranking suggests that replacing GS with DE in the full forward search of the reduced set of asteroids does not lead to significant differences in the final ranking of GTOC2 sequences. This is

Table 10.4: The final ranking of GTOC2 sequences in the reduced asteroid set without caching, using DE.

GTOC2	Departure Window		
Sequence	10 years	15 years	20 years
#1	999	4674	1016
#2	15	511	686
#3	35	76	96
#4	106	193	243
#5	5538	2155	2730
#6	738	18	23
#7	8	1881	2355
#9	39	92	115
#10	2676	1339	1752
#11	4746	4725	5273
#12	1403	55	56
#13	76	1863	2424

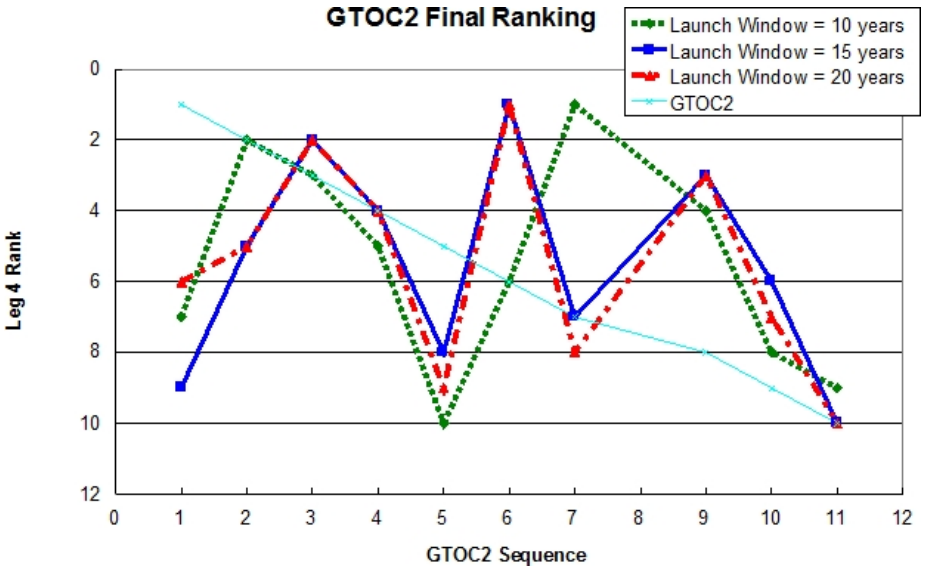

Figure 10.15: Relative ranking of GTOC2 sequences in the reduced asteroid set without caching, using DE.

confirmed when looking at Figure 10.16, where the difference in ranking of the GTOC2 sequences, from arc cost optimization with GS to DE, is plotted.

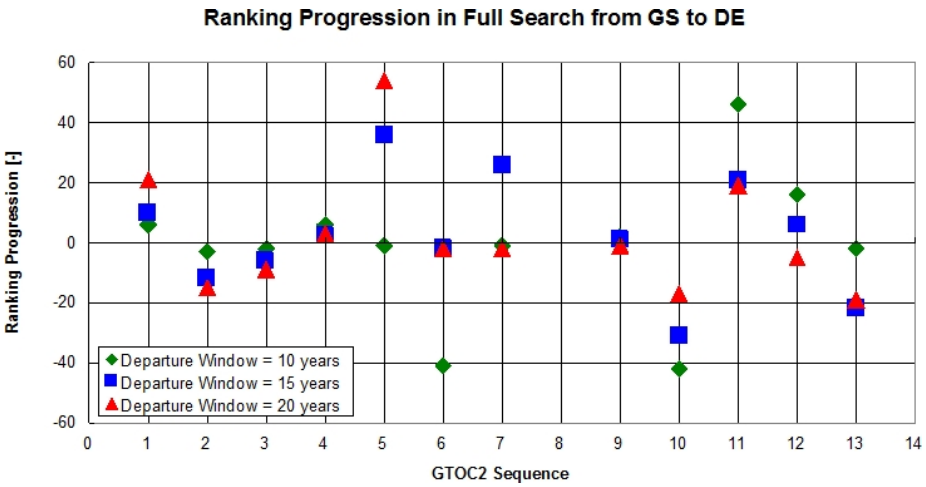

Figure 10.16: Ranking Progression of the GTOC2 sequences in the reduced asteroid set without caching, using DE instead of GS.

We can see that differences in the final ranking of the GTOC2 sequences is approximately ±50 places. While GTOC2 #1 climbs in the classification by up to 20 places, the rankings of GTOC2 #2 and #3 drop a few places. The sequence that most benefits from the change is GTOC2 #5: it climbs 54 places in the ranking for 20-years departure windows (and 36 for 15-years departure windows) but remains below 2000^{th}. GTOC2 #11 rises between 20 and 50 places according to the departure window length. Replacing GS with DE has a negative impact on the ranking of GTOC2 #10, which drops between approximately 40 and 20 places depending on the departure windows length considered. For 10-years departure windows, GTOC2 #6 falls 40 places in the classification. The changes in ranking from GS to DE can be explained by two factors. First, the heuristic nature of DE, and in particular the random initialization of the population, implies that there is no guarantee on the quality of the solution and therefore it may provide, for a particular asteroid pair, an objective function value larger than GS. Second, since DE does not discretize the search space, it has more freedom to provide a better solution than GS. The combination of these two factors, which represent a paradox regarding the solution quality of heuristics when compared to enumerative searches, is responsible for changes in ranking, both positive and negative, of GTOC2 sequences. Note, however, that the changes in ranking are relatively small given the total number (5940) of ranked solutions.

As stated at the beginning of this section, the purpose of replacing GS with DE was to improve the computational performance of the full multi-path NNH search with respect to the search with caching. In order to compare the computational effort needed by each variant to solve the reduced set of asteroids, Figure 10.17 plots the execution times of the algorithms across the range of departure windows considered.

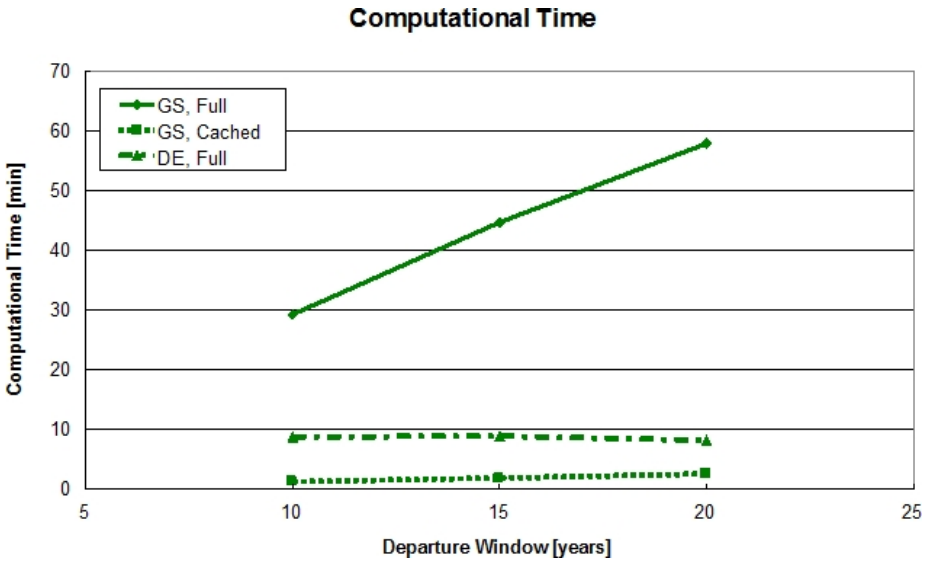

Figure 10.17: Computational time of the forward multi-path NNH search on the reduced asteroid pool as a function of the departure windows length.

Figure 10.17 shows that, despite taking longer than the cached, GS-based search, the full multi-path search with DE takes considerably less time to complete than with GS, as expected. Another interesting feature is that the execution times with DE are constant at about 8 minutes, indepedently of the departure windows length, while using GS (with or without caching) leads to computatonal times that increase with increasing length of the departure windows. Since the step-size in the variables of the GS searches are constant independently of the variables range, increasing the length of the departure windows leads to an increase in the number of grid points and therefore to an increase in the number of transfers evaluated. With DE, on the other hand, the number of evaluations depends on the number of individuals and on the number of generations. Since these are kept constant for the different departure windows, increasing this variable does not increase the computational effort of the algorithm. As a result, the larger the departure windows, the more interesting it becomes to replace GS with DE as arc optimization technique in the full forward multi-path NNH search.

Replacing GS with DE for the full forward search does not considerably impact the ranking of the sequences. It does, on the other, provide significant reductions in the execution times, even more so for large departure windows. Therefore, with DE it becomes interesting to remove the caching feature from the forward multi-path NNH as the resulting sequences will have a more realistic phasing.

10.4 Complete High-thrust Optimization, using DE

In order to determine the impact of adopting a leg-per-leg approach to the optimization of the asteroid sequences, we decide to optimize the complete GTOC2 sequences, more specifically the GTOC2 sequences that comply with the Group 4 - Group 3 - Group 2 - Group 1 asteroid group order. In order to do so, a new optimization problem needs to be formulated. We will focus here on the maximization of the final mass alone, without adding the time-of-flight to the objective function.

10.4.1 Problem Formulation

Given that four consecutive individual transfer are optimized, the number of variables of the problem also increases. Now, based on the problem definition of Section 6.1, the instinctive approach to the definition of the problem variables are four pairs of departure date and time-of-flight, one for each transfer. However, to avoid overlapping departure and arrival dates without placing constraints on the optimzation problems, the departure dates for Legs 2, 3, and 4 are substituted by a new variable, the time-of-wait, t_w, defined as the time spent orbiting the asteroid, between arrival and departure:

$$t_{w,i+1} = t_{d,i+1} - (t_i + t_{d,i}) \ : \ i = 1, ..., 3 \tag{10.4.1}$$

where $t_{d,i}$ and $t_{f,i}$ are the departure date and the time-of-flight, respectively, of Leg i. Given the celestial bodies, v_k, for $k = 0, ..., 4$, the cost function, J, is then defined as:

$$J(t_{d,1}, t_{f,1}, t_{w,2}, t_{f,2}, t_{w,3}, t_{f,3}, t_{w,4}, t_{f,4}) = \frac{1}{m_4(v_0, t_{d,1}, t_{f,1}, v_1, t_{w,2}, t_{f,2}, v_2, t_{w,3}, t_{f,3}, v_3, t_{w,4}, t_{f,4}, v_4)} \tag{10.4.2}$$

Note that maximizing the final mass is, according to Equation (4.4.3), equivalent to minimize the total ΔV which arises from the different Lambert arcs. Given this cost function, the optimization problem can be formulated as:

$$\min J(t_{d,1}, t_{f,1}, t_{w,2}, t_{f,2}, t_{w,3}, t_{f,3}, t_{w,4}, t_{f,4}) \tag{10.4.3}$$

subject to:

$$t_d \in [t_{d,\min}, t_{d,\max}] \tag{10.4.4}$$
$$t_f \in [t_{f,\min}, t_{f,\max}] \tag{10.4.5}$$
$$t_d \in [t_{w,\min}, t_{w,\max}] \tag{10.4.6}$$

The bounds of each parameter are chosen such that the problem is as similar as the combinatorial problem of Section 10.1. The minimum and maximum launch date correspond to the original GTOC2 departure window: January 1^{st}, 2015 and December 31^{st} 2035, respectively. The bounds for the times-of-flight are [10,600]. Finally, the minimum time-of-wait is set to 90 days, while the maximum time-of-wait varies such that it matches the various departure windows defined earlier: 10, 15 and 20 years.

10.4.2 Optimal Sequences

The optimization of the 12 GTOC2 sequences is performed with a self-adaptive DE. Since the number of variables, with respect to the optimization problem (6.1.6), was multiplied by four, the size of the population and the number of generations need to be adapted. The number of individuals is taken to be 200, while the number of generations is raised to 1000. Each sequence is optimized three times, and the best solution found across the three independent runs is taken. The total ΔV of GTOC2 sequences found in this fashion are plotted in Figure 10.18. There is a large gap in terms of ΔV between the ΔV of GTOC2 #11 (about 32 km/s) and the rest of the sequences, which indicates that, in terms of J-value, this sequence is the least attractive of all. For the sake of legibility, GTOC2 #11 is not plotted in Figure 10.18. The numerical values for all GTOC2 sequences optimized can be found in Table D.1.

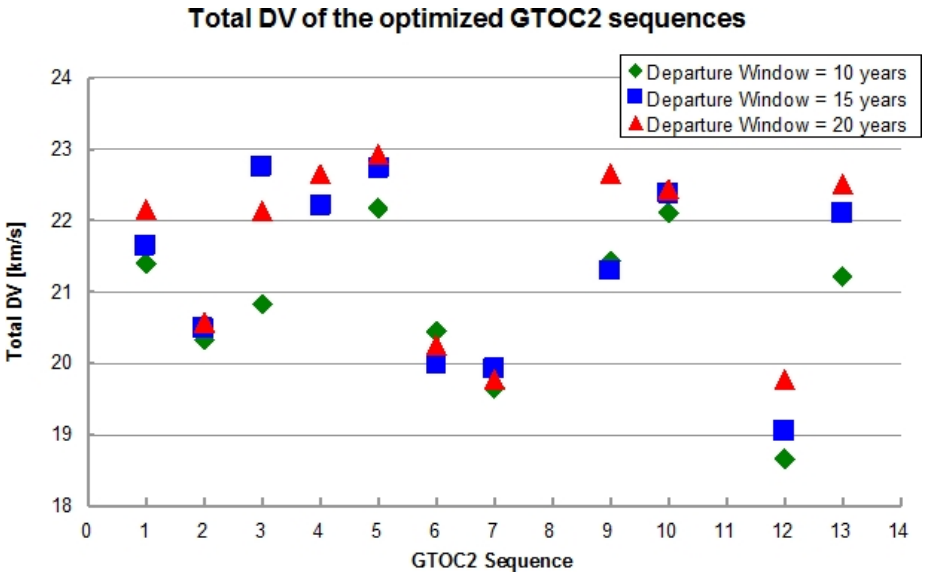

Figure 10.18: Total ΔV of optimized GTOC2 sequences, using DE.

Figure 10.18 shows that, in terms of J-value, GTOC2 #12 is the most interesting sequence, with a total ΔV of about 18.5 km/s for departure windows of 10 years. Other sequences with low total ΔV across all lengths of departure windows are GTOC2 #2, #6 and #7. Overall, the best ΔV budget found tends to correspond to 10-years departure windows. To explain this, one must bear in mind that increasing the maximum time-of-wait leads a "Russian doll" effect: the search space with a lower departure window is contained into the larger search space corresponding to a larger departure window. Therefore, solutions found with 10-years departure windows are also contained in the search space with 15-years departure windows, the solutions of which are in turn contained in the 20-years departure windows search space. With this in mind, increasing the search space can never lead to a degradation of the true optimum of the problem. The reason why the solutions found with 10-years departure windows are generally better than with larger departure windows is mostly due to the fact

that this search space, which contains the optimum, is smaller in this scenario. Since the number of individuals and the number of generations are constant across the different values for the departure windows length, a smaller search space leads to an increased probability of finding the true optimum.

When the quality of the best solution found increases with a larger search space, two reasons can be evoqued. The first reason is simply that the larger search space contains a better solution than the narrower search space. This occurs, e.g., for GTOC2 #6. To illustrate this, let us compare Tables 10.5, 10.6 and 10.7, which list the best solutions found for each of the GTOC2 sequences optimized. Times-of-wait larger than 10 years are highlighted in yellow. We can see that, for GTOC2 #6 with 15- and 20-years departure windows, the solution entails a waiting time for Leg 4 of about 4800 days. This more than 10 years and therefore this solution is not contained in the search space with a maximum time-of-wait of 10 years. Enlarging the search space led to the presence of a better solution, which is reflected in the lower ΔV budget of the GTOC2 #6 sequence for larger departure windows. The second explanation is anchored in the heuristic nature of DE: due to randomness of the initial population, different runs of DE may lead to different solutions. Therefore, despite increasing the search space, a larger departure window may lead to a better solution, which was included in the smaller search space but was not found by DE. This happens for GTOC2 #9: the solution found with 15-years departure windows is better than with 10-years departure windows and yet is contained within the latter's search space.

Table 10.5: Optimized GTOC2 sequences with departure windows of 10 years, using DE.

GTOC2 Sequence	$t_{d,1}$ [JD]	$t_{f,1}$ [days]	$t_{w,2}$ [days]	$t_{f,2}$ [days]	$t_{w,3}$ [days]	$t_{f,3}$ [days]	$t_{w,4}$ [days]	$t_{f,4}$ [days]
#1	2458474.3	318.7	1782.1	426.8	980.0	494.4	585.4	595.3
#2	2462798.2	130.2	349.7	346.0	1460.5	598.0	239.7	600.0
#3	2457025.8	291.7	687.9	392.5	1291.8	308.8	185.6	600.0
#4	2457380.7	298.8	3138.7	367.3	944.6	599.4	90.5	600.0
#5	2458245.5	275.8	980.8	401.2	3626.3	556.2	146.0	600.0
#6	2457817.6	234.8	1114.2	504.6	707.9	421.0	103.2	586.2
#7	2461708.0	315.5	1498.0	453.1	1213.0	394.0	225.8	600.0
#9	2457399.2	282.2	571.7	388.0	1604.2	599.8	217.9	600.0
#10	2457816.4	235.4	462.4	360.8	2354.5	599.6	700.8	594.2
#11	2464590.7	128.9	1322.8	414.1	3432.9	506.7	3179.7	600.0
#12	2457437.0	302.6	1784.7	404.0	772.0	553.8	1808.7	600.0
#13	2457751.3	286.2	413.8	512.6	385.7	553.7	863.7	600.0

Another interesting feature rises from observation of Tables 10.5 through 10.7: the times-of-flight for Leg 4 are nearly systematically close or equal to 600 days, which is the upper limit for that variable. This can also be observed for a few Leg 3 times-of-flight. This seems to indicate that the bounds on the time-of-flight defined in the previous chapters and sections are too stringent and that lower ΔV for Leg 4 transfers can be achieved by increasing the upper limit of the time-of-flight.

In order to discuss the extent to which the high-thrust model used in the

Table 10.6: Optimized GTOC2 sequences with departure windows of 15 years, using DE.

GTOC2 Sequence	$t_{d,1}$ [JD]	$t_{f,1}$ [days]	$t_{w,2}$ [days]	$t_{f,2}$ [days]	$t_{w,3}$ [days]	$t_{f,3}$ [days]	$t_{w,4}$ [days]	$t_{f,4}$ [days]
#1	2458846.26	299.18	1426.82	430.12	969.69	508.32	559.14	599.98
#2	2462813.54	197.66	264.32	335.81	1540.08	597.82	157.94	600.00
#3	2457056.24	241.02	711.84	312.03	1498.85	286.76	98.84	599.98
#4	2457390.45	290.79	3138.49	369.11	925.16	597.88	92.24	600.00
#5	2461069.28	238.32	1202.74	398.25	576.52	511.55	229.78	600.00
#6	2462196.40	282.88	3096.47	503.06	1268.78	460.72	4779.71	600.00
#7	2461351.50	327.40	1843.27	450.77	1254.85	417.73	159.48	600.00
#9	2457028.82	280.70	936.63	445.24	1535.01	599.10	258.22	600.00
#10	2457822.80	232.83	2399.42	440.36	333.66	599.87	684.00	599.98
#11	2463984.61	286.99	1317.74	331.32	4077.05	551.67	3036.17	600.00
#12	2457811.68	224.95	1494.32	399.57	783.76	558.61	1645.55	311.39
#13	2457772.62	268.97	410.65	491.87	587.23	440.66	800.94	600.00

Table 10.7: Optimized GTOC2 sequences with departure windows of 20 years, using DE.

GTOC2 Sequence	$t_{d,1}$ [JD]	$t_{f,1}$ [days]	$t_{w,2}$ [days]	$t_{f,2}$ [days]	$t_{w,3}$ [days]	$t_{f,3}$ [days]	$t_{w,4}$ [days]	$t_{f,4}$ [days]
#1	2458463.78	351.80	354.18	382.47	2416.54	540.03	536.81	600.00
#2	2462227.37	254.56	350.28	443.18	1823.20	590.34	247.31	600.00
#3	2457239.75	262.76	502.05	362.82	1361.81	299.44	219.05	585.04
#4	2457029.70	275.12	3519.97	347.81	465.10	461.49	559.58	600.00
#5	2458296.79	232.62	527.96	273.17	4191.30	557.63	158.71	600.00
#6	2462179.67	270.83	1765.51	515.86	2562.48	472.58	4833.27	600.00
#7	2461704.90	322.26	1492.56	459.55	1152.93	358.81	333.45	600.00
#9	2457064.67	270.95	1357.21	426.97	1162.26	599.89	189.73	600.00
#10	2457797.11	248.74	472.54	347.31	2393.62	577.97	682.92	598.19
#11	2460775.76	172.68	3476.12	256.50	5365.38	565.33	2982.79	600.00
#12	2457998.20	307.79	1226.32	409.00	729.45	578.33	1787.92	600.00
#13	2457246.42	277.46	1431.81	306.54	156.55	495.54	743.70	600.00

optimization of GTOC2 sequences reflects the quality of the combinations based on their original GTOC2 ranking, let us take a look at Figure 10.19. For the sake of clarity, GTOC2 solutions with constraint violations were left out of this ranking, and the light blue curve indicates the ranking based on the original GTOC2 objective function. Bear in mind that GTOC2 #8 was not evaluated since it does not follow the 4 - 3 - 2 - 1 asteroid group order. This could possibly explain the drop in final leg ranking observed for GTOC2 sequences in Chapter 9.

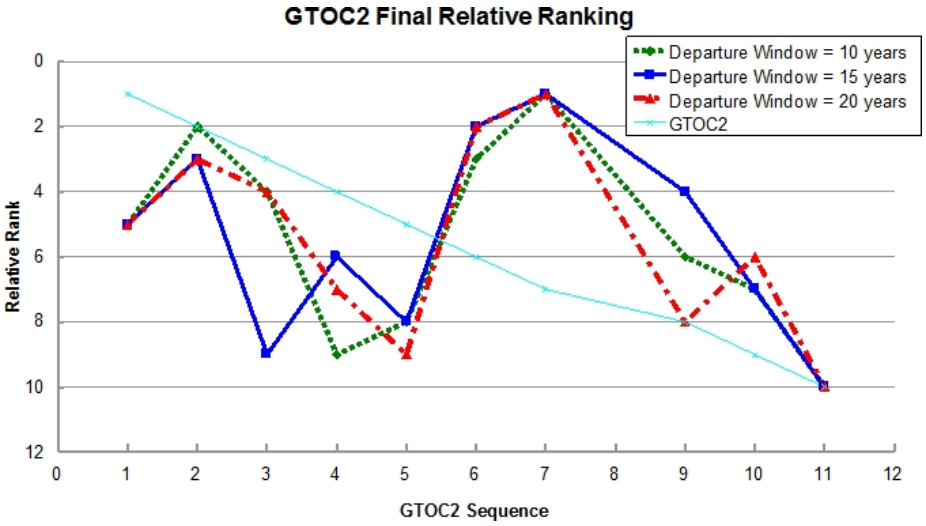

Figure 10.19: Relative ranking of optimized GTOC2 sequences, using DE.

There is no clear decreasing trend in Figure 10.19, as evidenced by the large deviations from the "nominal" light blue curve. While GTOC2 #11 is accurately identified as the worst combination, none of the other sequences is systematically ranked in the vicinity of their original GTOC2 classification. The GTOC2 sequence that ranks the closest to its nominal position is GTOC2 #2, which ranks between 2^{nd} and 3^{rd} depending on the departure window. There is also no departure window that clearly leads to a better ranking than the others. However, out of the sequences that rank in the top-4 for at least one departure window length (GTOC2 #2, #3, #6, #7, and #9), only GTOC2 #6 did not involve the use of high-thrust Lambert arcs in its procurement during the GTOC2 competition.

The evaluation of the GTOC2 sequences via optimization of the final mass under a high-thrust model does not seem to accurately reflect their quality under a low-thrust model with an optimal final mass to time-of-flight ratio objective function. Out of the participating teams that used optimal phase-free, bi-impulsive pruning during the competition, only GTOC2 #4 and #5 do not rank highly in the scenarii studied in this section.

10.4.3 Comparison with Leg-per-Leg Results

Let us now proceed with the comparison, for the GTOC2 sequences, between the complete transfers found with a single, complete optimization and with four consecutive leg optimizations. Figure 10.20 plots the difference in total ΔV, δ, between the sequences found in both fashions, computed according to $\delta = \Delta V_{\text{leg-per-leg}} - \Delta V_{\text{complete}}$. Given this definition, a positive difference translates into the sequence obtained in a leg-per-leg approach having a larger total ΔV than the complete optimized sequence.

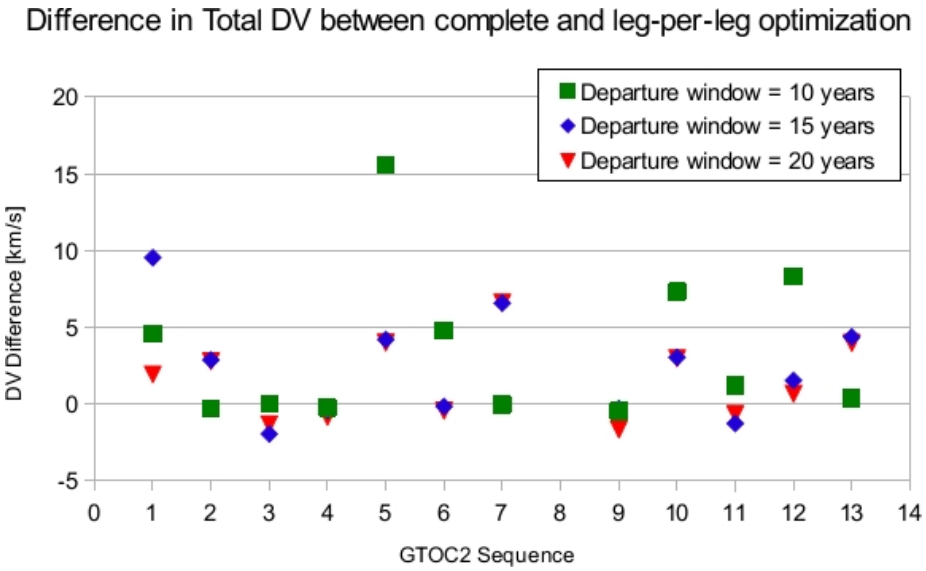

Figure 10.20: Difference in ΔV budget for the GTOC2 sequences between a complete and a leg-per-leg optimization, using DE.

Figure 10.20 shows that sequences optimized in a leg-per-leg fashion have, in general, a larger total ΔV. This reveals the importance of phasing when optimizing full sequences and the limitations of greedy algorithms. By not looking at the "big picture", greedy algorithms are not able to perform trade-offs between the cost of the different transfers to achieve a total mission with lower final cost. Taking a given best leg transfer may lead, due to phasing, to a relatively worse transfer in the following leg. Moreover, this effect can propagate beyond the leg immediately following. This drawback of greedy algorithms was however expected, as discussed in Chapter 5. Note that the difference in total ΔV of sequences obtained via leg-per-leg optimization, with respect to the sequences optimized at once, tends to decrease with increasing departure windows. To understand this phenomenon, we need to take two factors into account. Firstly, increasing the departure windows limits the impact of phasing on the total transfers and, with no limitation on the total time-of-flight, the sequences obtained in both manners will tend to be similar. Second, we have seen in the previous section that increasing the departure windows tends to lead, for the sequences optimized in one go, to worse solutions due to the expansion of the search space. With the quality of leg-per-leg solutions increasing and

that of complete solutions decreasing, the difference between the total ΔV of the sequences tends to decrease with increasing departure windows.

In Figure 10.20, not all sequences are affected in the same manner by the difference in approach to their optimization. While some display large differences (up to 15 km/s for GTOC2 #5), others have differences close to zero, independently of the departure windows length. The latter, such as GTOC2 #4, even exhibit negative differences, meaning that a solution found on a leg-per-leg basis is better than that found on a single optimization run. In this case, the leg-per-leg approach is able to find a very good solution, probably very close to the true optimum of the problem, which does not seem to vary with expanding search spaces. The single-run optimization is not able to improve on that result, probably due to the larger number of variables. Except for GTOC2 #1, the differences for 15- and 20-years departure windows are very similar. This seems to indicate that increasing the departure windows from 15 to 20 years has less impact on the best solution than increasing this variable from 10 to 15 years.

Chapter 11

Complete Asteroid Pool, using DE

We have seen, in the previous chapter, that replacing GS by DE as arc optimization technique leads to considerable savings in terms of computational time when no caching is performed. We therefore proceed to tackling the complete asteroid pool with a self-adaptive DE and without the caching feature. The parameters for DE are consistent with the values defined earlier, namely a population of 50 individuals evolving for 150 generations.

11.1 Full Forward Search, Optimal Final Mass

11.1.1 Comparison with Cached Forward Search, using GS

In order to perform a fair comparison with earlier results, the full forward search, with DE, is performed for departure windows of 10, 15 and 20 years and 5, 10, and 20 champions. As in Section 9.1, a 10-years departure window does not yield any GTOC2 sequence in the final leg computations. For departure windows of 15 and 20 years, the only GTOC2 sequence to systematically reach Leg 4 computations is GTOC2 #2. The difference in its final ranking, plotted in Figure 11.1, is taken to be the final ranking in the cached forward GS search minus the final ranking in the full forward DE search. Therefore, a negative value corresponds to a drop of GTOC2 #2 in the final classification when DE is employed as optimization technique.

Figure 11.1 shows that there is a degradation (up to 28 places) of the GTOC2 #2 final ranking in all scenarii analyzed. This degradation increases with the number of champion paths but decreases with a larger departure window. As discussed in Section 10.2, the differences in ranking are minimized when considering the 20-years departure windows for the effect of caching individual transfer values is lessened. All ranking differences are negative (or zero) which means that GTOC2 #2 ranks worst when moving from a cached GS search to a full DE search. This explanation for this drop is that other sequences profit more from the change, either because their non-cached transfers become more interesting than the cached transfers and/or because there is larger improvement

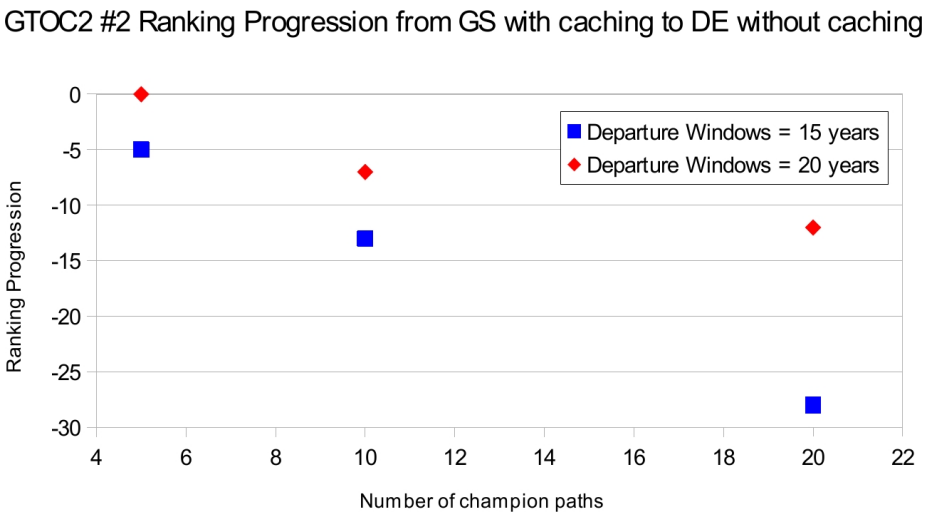

Figure 11.1: Ranking Progression of GTOC2 #2 in the complete asteroid set without caching, using DE.

in their path cost due to the use of DE. Finally, the drop in ranking is more consequent when increasing the number of paths, a feature already observed in Chapter 9. The explanation given at the time is still valid in this scenario: the relative ranking of sequences changes considerably from leg to leg due to the individual contributions to the (partial) path cost of single transfers. Increasing the number of champions allows to consider sequences, which were previously discarded for ranking worse, that end up overtaking other sequences thanks to less costly leg transfers.

With respect to execution times, Table 11.1 lists the computational time needed to solve the different problems by both the cached GS and the full DE searches. It indicates that DE is considerably faster than GS, with the parameters selected for both algorithms. In the worst case scenario (20 champions with 20-years departure windows), the search takes about 16 minutes with DE while it needs almost two hours with GS. This situation is different than the one depicted in Figure 10.17, where the cached GS search on the reduced asteroid set was faster than the full DE search. Bear in mind however, the observation made at the time regarding the fact that the computational effort with DE was less sensitive to the problem dimension: while, on the reduced asteroid pool, the computational effort with GS was lower than with DE, it increased with an expanding search space, namely in terms of departure windows. This can also be observed in Table 11.1, as the execution times of DE are only affected by the number of champions, while for GS the execution times increase with both the number of champions and the departure windows length. Another factor to explain the larger computational effort with the cached GS search with respect to the reduced asteroid pool results is the diversity of transfers: in the reduced set, the number of unique transfers was much lower and therefore the benefits of cached transfers in terms of execution times were greater.

The results from Figure 11.1 and Table 11.1 show that while there is a degra-

Table 11.1: Computational Times for the cached GS search and the full DE search.

Launch Window [years]	Number of Champions	Computational Time [min]	
		Cached GS	Full DE
10	5	14.93	4.47
	10	26.25	7.82
	20	46.28	15.75
15	5	25.66	4.44
	10	45.43	9.14
	20	92.33	16.36
20	5	32.96	4.65
	10	49.78	8.60
	20	113.07	16.37

dation in the final rank of GTOC2 #2, the problem becomes more tractable when GS is replaced with DE as arc optimization technique. To the extent that no benefits in terms of computational time are achieved by caching transfer values. In light of these results, the number of champions is increased, with the expectation that it will allow another, previously discarded GTOC2 sequence to "shine".

11.1.2 Increasing the Number of Champions

Given the considerable computational time savings enabled by the use of a self-adaptive DE, the full forward multi-path search of the complete asteroid pool is performed with larger numbers of champion paths, more specifically with 50 and 100 champions. Given that larger departure windows increase the probability of heliocentric geometry repetition, the analysis is performed with 20-years departure windows.

Table 11.2 gives the rank, leg by leg, of the GTOC2 sequences found by the forward multi-path NNH search with 50 champion paths. The *n.a.* note indicates that a given sequence was not considered in the corresponding leg computations, since it ranked below 50^{th} in the previous leg.

Looking at Table 11.2, one can state that increasing the number of champions did not yield the desired effects. The only two GTOC2 sequences to reach Leg 4 computations are GTOC2 #2 and #12. Both sequences were already present in the final leg computations for 20-years departure windows, the latter for numbers of champions larger than 10. There is therefore no novelty regarding which of the GTOC2 sequences transit to the last leg computations. Moreover, the decreasing trend in ranking for GTOC2 #2 and #12, seen in Tables 9.5 and 9.6, respectively, continues: GTOC2 #2 is now 499^{th} and GTOC2 #12 122^{nd}, below their classification with 20 champions.

The results from Table 11.2 for Leg 2 show that while the partial GTOC2 #1 and #11 sequences rank well beyond the specified number of champions, other sequences, such as GTOC2 #4, have a classification relatively close to the threshold. Given that the execution time of the search with 50 champions lasted less than 40 minutes, there is some leeway for increasing the number of

Table 11.2: Leg rank of the GTOC2 (partial) sequences as found by the forward multi-path NNH with 50 champion paths, using DE.

Leg	1	2	3	4
GTOC2 #1	37	2786	n.a.	n.a.
GTOC2 #2	2	5	23	499
GTOC2 #3	12	117	n.a.	n.a.
GTOC2 #4	12	82	n.a.	n.a.
GTOC2 #5	3	453	n.a.	n.a.
GTOC2 #6	2	94	n.a.	n.a.
GTOC2 #7	18	387	n.a.	n.a.
GTOC2 #9	12	17	2393	n.a.
GTOC2 #10	2	5	1069	n.a.
GTOC2 #11	39	7885	n.a.	n.a.
GTOC2 #12	2	16	24	122
GTOC2 #13	12	81	n.a.	n.a.

champions even further. Based on Table 11.2, the number of champion paths is increased to 100, in an attempt to include GTOC2 #4, #6 and #13 in the Leg 3 computations. The resulting leg ranks are given in Table 11.3.

Table 11.3: Leg rank of the GTOC2 (partial) sequences as found by the forward multi-path NNH with 100 champion paths, using DE.

Leg	1	2	3	4
GTOC2 #1	37	2849	n.a.	n.a.
GTOC2 #2	2	4	25	929
GTOC2 #3	12	125	n.a.	n.a.
GTOC2 #4	12	89	1183	n.a.
GTOC2 #5	3	456	n.a.	n.a.
GTOC2 #6	2	95	1148	n.a.
GTOC2 #7	18	396	n.a.	n.a.
GTOC2 #9	12	18	4423	n.a.
GTOC2 #10	2	4	2736	n.a.
GTOC2 #11	39	8501	n.a.	n.a.
GTOC2 #12	2	15	27	197
GTOC2 #13	12	87	511	n.a.

Unfortunately, raising the number of champions to 100 does not enable any GTOC2 sequence other than #2 and #12 to feature in the final leg computations. It did allow GTOC2 #4, #6 and #13 to be evaluated in Leg 3 but their corresponding classification ranked well below the champion threshold and these sequences were therefore discarded by the multi-path NNH. The degradation of GTOC2 #2 and #12 continues. The drop of GTOC2 #2 in the classification is rather steep, as it now ranks below 900^{th}. GTOC2 #12 on the other hand is

still in the top-200 sequences.

In order to include GTOC2 #4 in the final leg computations, the number of champions would need to be increased to more than 1000. The computational time needed to obtain the results in Table 11.3 is equal to 80 minutes. Extrapolating from the computational times corresponding to 50 and 100 champions, solving the full asteroid set with a 1000 champions would require about 15 hours of computations. Given the tendency of lower-ranked sequences to overtake higher-ranked sequence from leg to leg, increasing the number of champions to such levels would lead to a very large computational effort with no guarantee on the outcome. In the light of this considerable impact on the efficiency of the combinatorial tool, it is decided not to further increase the number of champions.

11.1.3 Increasing the Maximum Time-of-Flight

We have seen in Section 10.4 that the complete optimal solutions found led to times-of-flight for Leg 4 and, to a minor extent, Leg 3 transfers close to their upper bound of 600 days. In this section, the full forward search of the complete asteroid set is performed with an upper bound on the time-of-flight of 1200 days instead. Increasing the upper bound for the time-of-flight is expected to only lead to significant changes in the Leg 3 and Leg 4 computations. Two scenarii are analyzed, with two different lengths for the departure windows: 15 and 20 years. The 10-years departure windows scenario is not considered, since it did not produce satisfying results in the previous sections. Finally, due to time constraints, only two values for the number of champion paths were considered: 15 and 20.

20-Years Departure Windows

We have seen in Section 9.1 that, for numbers of champions larger than 13, two GTOC2 sequences reached the final leg computations: GTOC2 #2 and #12. This is again the case when increasing the bounds on the time-of-flight: only these two sequences reach the final leg computations. In the earlier scenario, GTOC2 #12 ranked the highest of the two GTOC2 sequences. This relative ranking is inaltered when increasing the maximum time-of-flight. Moreover, not only is GTOC2 #12 the best GTOC2 sequence, it is the overall best sequence found. This occurs for both 15 and 20 champion paths, as seen in Table 11.4.

Table 11.4: Final rank of GTOC2 sequences and computational time corresponding to a 20-years departure window and a maximum time-of-flight of 1200 days, using DE

Number of Champion Paths	15	20
GTOC2 #2	45	58
GTOC2 #12	1	1
Computational Time [min]	13.41	18.03

With respect to GTOC2 #2, increasing the number of champions leads to a degradation of its final ranking, a feature already observed in other scenarii.

Setting the number of champion paths to 15 allows GTOC2 #2 to be placed within the top-50 sequences found. With 20 champions, its ranking drops to 58^{th}.

Focusing on the execution times, the algorithm is extremely fast, completing the search in less than 20 minutes with either 15 or 20 champions. This effectively means that the initial billions of possible combinations are boiled down to less than 2000 sequences in a very short time. Among the top 60 combinations, the highest ranked led to a GTOC2 solution that, albeit exhibiting position and velocity constraints, yielded a higher objective function value than the solution handed in at the time by the TU Delft-led team[1]. Down the ranking, one finds a sequence that corresponds to the GTOC2 runner-up.

15-Years Departure Windows

In Section 9.1, when restricting the departure windows length to 15 years, the GTOC2 #12 solution was discarded from the final leg computations. Increasing the maximum time-of-flight does not alter this fact: only GTOC2 #2 is present when evaluating the final leg, as indicated by the *n.a.* note accompanying the GTOC2 #12 entries in Table 11.4. Note that GTOC2 #12 ranked first in Table 11.4, which once more highlights the sensitivity of the sequence analysis to the phasing of the bodies.

Table 11.5: Final rank of GTOC2 sequences and computational time corresponding to a 15-years departure window and a maximum time-of-flight of 1200 days, using DE

Number of Champions	15	20
GTOC2 #2	35	50
GTOC2 #12	n.a.	n.a.
Computational Time [min]	13.12	17.32

Akin what was observed in Section 9.1, reducing the departure windows length leads to a better ranking of GTOC2 #2. Also, the execution times are not affected by the length of the departure windows, a fact observed and explained earlier.

GTOC2 #2 ranks in the top-50 sequences for both numbers of champions. Therefore, setting the maximum time-of-flight to 1200 days and the departure windows length to 15 years drives the forward multi-path NNH to drastically reduce the combinatorial search space: a subset of 50 combinations, containing the sequence corresponding to the GTOC2 winner-up in less than 20 minutes.

A Note on GTOC2 #2 Rank for 15-Years Departure Windows

It should be disclaimed that the GTOC2 #2 rank, as found by the forward multi-path NNH with 15 champion paths, given in Table 11.5 does not correspond to the solution found in the first run of the algorithm. The first time the

[1]Bear in mind that this TU Delft solution contained a maximum mission time-of-flight violation besides position and velocity constraint violations.

forward multi-path NNH search was performed, it ranked GTOC2 #2 as the best sequence found. Subsequent re-runs of the search were unable to reproduce this result but consistently placed GTOC2 #2 in 35^{th}, the rank indicated in Table 11.5. The nominal (35^{th}) and non-nominal (1^{st}) sequences can be seen in Appendix E.

This illustrates the heuristic nature of DE: the quality of the results is subject to a certain degree of randomness, most notably in terms of population initialization, and there is no guarantee on the optimality of the solution returned. Granted, this dependence can be limited by increasing the DE parameters, i.e. the number of individuals and the number of generations. Increasing these parameters did not allow to reproduce the solution that allowed GTOC2 #2 to hoist itself up to the top of the final ranking but confirmed the solution that places it in the 35^{th} position.

It is therefore hypothesized that this non-nominal solution is the fruit of a sub-optimal Leg 3 transfer that led to a better phasing in Leg 4, resulting in the overall lower path cost. Note that this solution is also not present in the 20-years departure windows analysis. This drives to the conclusion that 20-years departure windows do not allow for repeatability of the relative geometry of asteroids 2000569 and 2002483.

Comparison with Original Bounds

In order to highlight the differences in the ranking of the two GTOC2 sequences when the cached GS search with a maximum time-of-flight of 600 days is replaced by a full DE search with a maximum time-of-flight of 1200 days, Table 11.6 lists the classification side by side. Additionally the execution times with both variants are given in that Table.

Table 11.6: Final rank of GTOC2 sequences and computational time for different maximum times-of-flight.

	Technique			
	DE, Full		GS, Cached	
	$t_{f,max}$ [days]			
	1200		600	
Departure windows = 20 years				
Number of Champions	15	20	15	20
GTOC2 #2	45	58	96	186
GTOC2 #12	1	1	38	45
Computational Time [min]	13.41	18.03	69.60	112.80
Departure windows = 15 years				
Number of Champions	15	20	15	20
GTOC2 #2	35	50	115	148
GTOC2 #12	n.a.	n.a.	n.a.	n.a.
Computational Time [min]	13.12	17.32	62.40	92.4

Table 11.6 does not allow to make explicit observations regarding the added

computational time related by the increase in maximum time-of-flight. However it is expected that, in the same way that the execution times with DE are not affected by increasing the departure date search space, expanding the range of possible time-of-flight values won't affect the computational performance of the DE search. Note however that the DE search is up to 8 times faster than GS search.

In terms of GTOC2 sequence ranking, the same trends are observable in both scenarios. With 15-years departure windows, GTOC2 #12 does not feature in the last leg computations. With 20-years departure windows on the other hand, this sequence outranks GTOC2 #2. Increasing the departure windows length and/or the number of champions leads to a decline in the final ranking of GTOC2 #2.

Finally, there is a large difference in ranking when considering the searches with 600 days and the 1200 days as maximum value for the time-of-flight. Increasing the upper bound leads to significant improvements on the final ranking of GTOC2 #2 and #12, the latter outranking all other sequences for departure windows of 20 years, independently of the champion thresholds considered. To sum up, raising the upper bound on the time-of-flight yields very positive results as GTOC2 #2 is always present in the top-60 combinations and GTOC2 #12 is the overall best when included. Bear in mind that other parameters were changed from the GS to the DE search, namely the removal of the caching feature and the improvement of the optimization technique. However, we have shown, in Sections 10.2 and 10.3, that such modifications had little impact on the final rankings of the sequences, especially for the departure windows considered. We can therefore state that the ranking improvements shown in Table 11.6 are mostly due to the raising of the time-of-flight upper bound.

11.1.4 Increasing the Departure Window Length

The results from Section 11.1.3 suggested that departure windows of 20 years did not allow to conduct a phase-free analysis of the high-thrust GTOC2 model. We therefore decide to increase these departure windows beyond 20 years: departure windows of 25, 30, 40 and 50 years were considered. In order to cope, in terms of accuracy, with this widening of the bounds of one of the optimization variables, the number of individuals and generations of the self-adaptive DE are also increased, to 100 and 200 respectively. The maximum time-of-flight is taken to be the same as in Section 11.1.3, i.e. 1200 days. Finally, we vary the number of champions of the forward multi-path NNH. The final rank of GTOC2 #2 is plotted, in the form of a heat map, in Figure 11.2, for the different number of champion paths and departure windows lengths. Note that, in that figure, dark red (which corresponds to a rank of 40^{th}) indicates that GTOC2 #2 is discarded from Leg 4 computations.

At first glance, it would appear that increasing the departure windows to 25 years is sufficient to obtain a phase-free evaluation of GTOC2 #2, as illustrated in Figure 11.2. For a number of champion paths larger or equal to 10, with the exception of $N = 40$, GTOC2 #2 is always present in Leg 4 evaluations. Moreover, it consistently ranks above 25^{th} and, for any number of champions, above the threshold set by that parameter[2]. The exception that occurs for

[2]I.e. GTOC2 #2 is in the top-10 for N=10, in the top-15 for N=15, and so on.

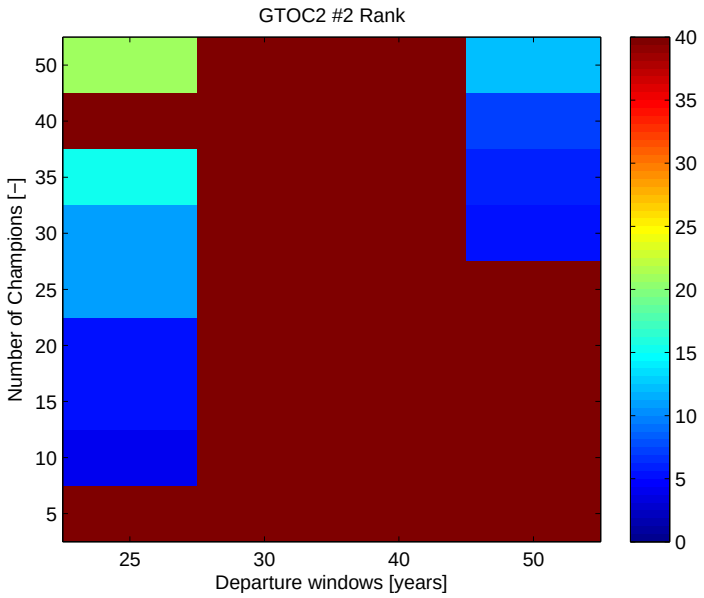

Figure 11.2: Final ranking of GTOC2 #2 for large departure windows, using DE.

$N = 40$ can be explained: enlarging the number of champions from 35 to 40 leads to the inclusion, in Leg 3 evaluations, of transfers which lead to sequences with a total path cost lower than GTOC2 #2 and which therefore push this sequence just below 40^{th} in Leg 3. Increasing the number of champions further to 50 does not translate into significant changes in Leg 3 ranking but does allow GTOC2 #2 to be included in Leg 4 computations. Note that for $N = 50$, despite being below 40^{th} in Leg 3, GTOC2 #2 rises up to 21^{st} in Leg 4.

Further widening the departure windows shows that 25 years does not allow to obtain a phase-free analysis of the high-thrust GTOC2 model. Indeed, with 30- and 40-years departure windows, GTOC2 #2 is discarded from the final leg computations (see Figure 11.2). This can be attributed to other sequences having lower path costs than GTOC2 #2 for these departure windows, indicating that their evaluation was not independent of phasing for shorter departure windows[3].

For 50-years departure windows, GTOC2 #2 is once more (for certain numbers of champion paths) among the Leg 4 computations, disproving the hypothesis that 25 years is a large enough window to allow geometry repeatability for all of its individual legs. In order for GTOC2 #2 to reach the final leg computations, the number of champions has to be set to 30 or higher (see Figure 11.2). While this might seem like a large number, notice that this leads to a final GTOC2 #2 ranking above 15^{th} for $30 \leq N \leq 50$, which is considerably better than the rankings found for the initial, shorter departure windows (see

[3]Remember that widening the departure windows allows for a larger range of orbital configurations, which should ideally repeat themselves with the specified interval therefore leading to the overall optimal transfer being included in the corresponding search space.

e.g. Table 11.6).

The reason for which GTOC2 #2 is only present in the final leg computations for large numbers of champions can be attributed to a relatively poor Leg 3 transfer. This is illustrated in Figure 11.3, where the ranking of GTOC2 #2 in each leg for 50-years departure windows is given. Notice the dip in ranking for Leg 3. While GTOC2 #2 ranking is constant for any number of champions up to Leg 2, Leg 3 computations leads to a drop in ranking, more pronounced as the number of champions increases. This drop leads to the exclusion of GTOC2 #2 from Leg 4 computations for a number of champions up to 25. For larger N values, GTOC2 #2 stays afloat above the champion threshold and is included in Leg 4 transfers, hoisting itself up to the top-15 sequences. Note that for N=5, GTOC2 #2 is discarded after Leg 2 evaluations.

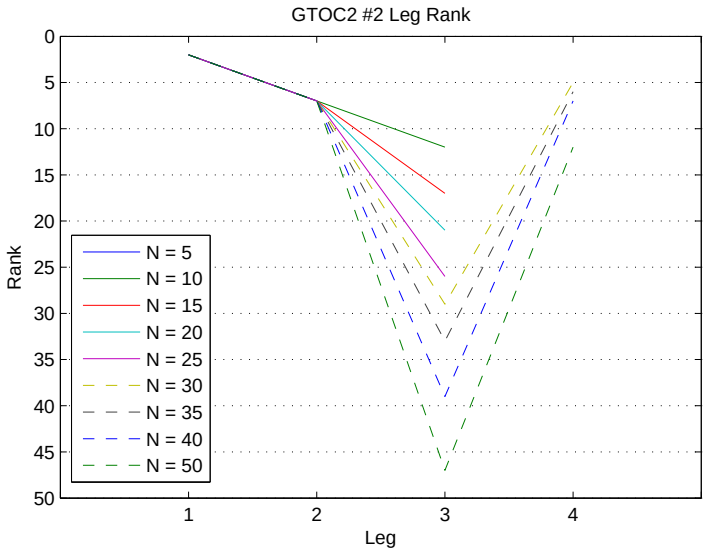

Figure 11.3: Ranking of GTOC2 #2, leg per leg, for 50-years departure windows, using DE.

In Figure 11.4, the final rank of GTOC2 #12 for large departure windows is plotted. For numbers of champions larger than 15, GTOC2 #12 ranks in the top-10 sequences for 25- and 30-years departure windows. This seems to indicate a similar pattern as that illustrated in Figure 11.3 for GTOC2 #2, namely in terms of the Leg 4 ranking being better than its Leg 3 equivalent. We have established by now that these departure windows do not allow for geometry repeatability of all transfers. Increasing their length to 40 and 50 years leads to an improvement of the path costs of other sequences which relegate GTOC2 #12 below the champion thresholds and therefore exclude it from Leg 4 computations. There is one exception to this statement, for 40-years departure windows and N = 50. However, the fact that it is not reproducible for 50-years yields the conclusion that increasing the departure windows leads to an overall improvement of the sequences path costs, a scenario in which GTOC2 #12 does not perform well.

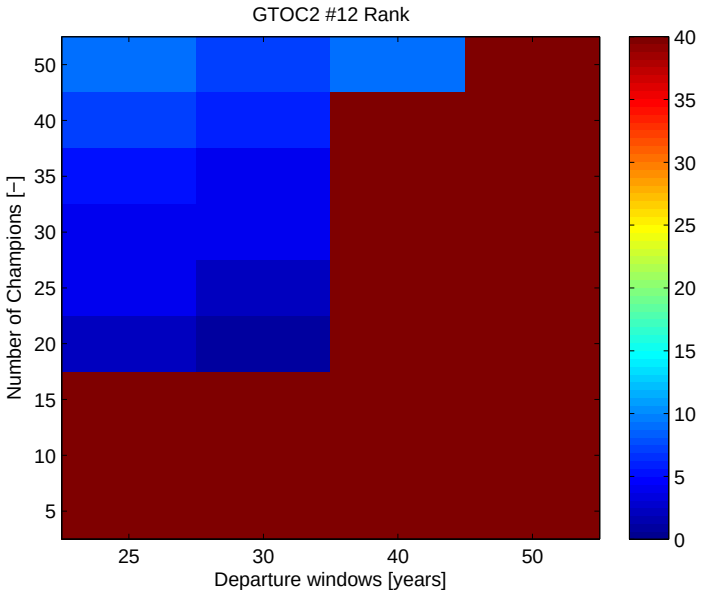

Figure 11.4: Final ranking of GTOC2 #12 for large departure windows, using DE.

To conclude the analysis of the GTOC2 high-thrust model with larger departure windows, let us take a look at Figure 11.5, which indicates the computational time, in minutes, needed for the forward NNH search to complete. Despite having increased the number of individuals and generations with respect to the previous section, the forward multi-path NNH takes between 10 and 120 minutes, depending on the number of champions, meaning that the tool retains its attractive computational effort characteristics. Focusing on the search parameters that yielded the best GTOC2 #2 rankings, i.e. 50-years departure windows and a large number of champions, the algorithm runs in 70 to 120 minutes, which is rather positive given the size of the original combinatorial problem.

Figure 11.5 also shows that, as we discussed earlier, increasing the number of champion paths leads to an increase in computational effort, as more transfers need to be evaluated. A increase in computational effort from 25/30- to 40/50-years departure windows is also visible, especially for larger numbers of transfers. This is probably due to the convergence criteria of the self-adaptive DE routine employed. A smaller search space means that the optimum can be more easily found and therefore the optimization heuristic convergences faster. The final impact on computational times is amplified by the number champions. Note however that this does not single-handedly account for the large differences in computational times between Tables 11.5 and 11.4 and Figure 11.5, for 15 and 20 champions. Those are mostly the effect of the larger DE parameters in used in this analysis.

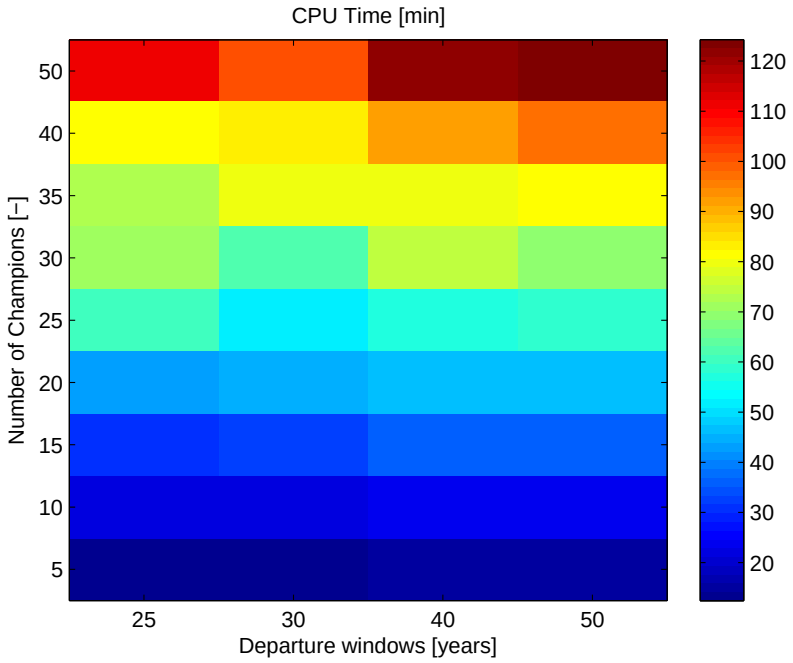

Figure 11.5: Computational time for large departure windows, using DE.

Part IV

Conclusions and Recommendations

Chapter 12

Conclusions

This report documents the author's attempt at designing an efficient combinatorial tool for preliminary space mission analysis, applied to a theoretical problem, defined in the 2^{nd} Global Trajectory Optimization Competition (GTOC2). With a focus on efficiency, a number of design choices and hypotheses were formulated, implemented and verified. We started by decoupling the continuous part from the integer facet of the problem. The combinatorial problem was modelled as a static Shortest Path Problem (SPP), the complexity of dynamic variant being expected to yield large computational times which would hinder the efficiency of the combinatorial tool. The SPP model is solved on a leg-per-leg basis with a greedy algorithm. While the basis of this algorithm is the Nearest Neighbour Heuristic (NNH), a multi-path variant was adopted in an attempt to improve the fidelity of this heuristic. At each consecutive leg, the individual transfers, formulated as continuous, box-constrained problems, were optimized using two different optimization techniques: Grid Search (GS) and Differential Evolution (DE). Two different cost functions were implemented, both based on the ΔV of high-thrust, bi-impulsive Lambert transfers. An efficient Lambert routine was implemented and tested for this purpose. The resulting combinatorial tools were called into play on both the complete asteroid pool and a subset of said pool. A diversity of scenarii, namely with respect to the search spaces of the continuous problems, as well as to the parameters of the multi-path NNH, were analyzed.

This chapter lists the main conclusions that can be drawn from the work documented in this report. These are grouped into categories that reflect their scope of application. As a result, we first take a look at the continuous optimization methods employed, and then move towards considerations regarding the problem definition of the continuous problems, more specifically with regard to the bounds of the different variables. The subsequent section concerns itself with the high-thrust approximation and corresponding cost functions. Thereafter, observations regarding the greedy algorithm are made. Finally, the last two sections are in respect to the characteristics of the GTOC2 problem and the evaluation of the resulting sequences.

12.1 Continuous Optimization Technique

For similar accuracy levels, the self-adaptive DE is considerably faster than GS. It was shown throughout the report that, after tuning, the self-adaptive DE leads to considerable time savings with respect to GS, while providing results with a similar, if not increased, fidelity (see e.g. Sections 6.2.2 and 10.3). Even adding complexity to the problem, namely through the removal of the caching feature, the complete asteroid pool is dealt with by the forward multi-path NNH up to 7 times faster when replacing GS with DE as arc optimization method (Section 11.1.1).

GS is more sensitive, in terms of execution times, to a varying search space size. The parameters of GS and DE were kept constant irrespective of variations in the size of the problem. The number of champions affects the number of individual transfers and therefore impacts the computational times of the search with both GS and DE. The departure windows length, on the other hand, affects the size of the individual, continuous optimization problems. By defining a constant step-size for the mesh of GS, the algorithm evaluates a different number of grid points and this is reflected in the computational effort (see e.g. Section 9.1). The number of evaluations in DE is mostly defined by the number of individuals and generations, which are constant. Increasing the departure windows length has thus no impact on the computational effort of DE (see Section 10.3).

The heuristic nature of DE leads to less consistency in the results. This was observed when optimizing the full GTOC2 sequences (Section 10.4), as well as when solving the GTOC2 combinatorial problem with a raised upper bound on the time-of-flight (Section 11.1.3): different DE optimization runs yields different results due to the random components of the algorithm. This randomness can be limited by setting more stringent parameters, i.e. by increasing the number of individuals and/or of generations, at the cost, however, of added computational complexity.

In DE, higher accuracy is more easily achieved by raising the number of generations than by increasing the number of individuals. When tuning the parameters of the self-adaptive DE (Section 6.2.2) it was shown that a larger number of generations is more effective to improve the quality of the results than a larger number of individuals. By increasing the number of generations, solutions have more opportunities to evolve into a better set of variables, diminishing the impact of the random population initialization.

An expanding search space should be accompanied by an increase in the DE parameters. When optimizing the full GTOC2 sequences (Section 10.4), it was observed that increasing the parameters of the search, namely in terms of the upper bound on the times-of-wait, led to a degradation of the total ΔV. Given the "Russian doll" property of the increasing search spaces, i.e. the fact that the smaller search spaces are contained within the larger ones, this degradation can only be explained by an underperforming optimization

step. This can be compensated by increasing the number of individual and/or generations.

12.2 Optimization Problem Definition

The largest departure windows length considered initially, 20 years, is not broad enough to perform a fully phase-free analysis. This conclusion is recurrent to different parts of the MSc Thesis work and was brought into the spotlight as the result of the "odd" simulation that ranked GTOC2 #2 as the best sequence found when using DE in the forward search with a maximum time-of-flight of 1200 years, a departure windows length of 15 years, and 15 champion paths (Section 11.1.3). Gradually increasing the departure windows length up to 50 years in Section 11.1.4 lead to ever-changing results. The departure windows were not increased beyond 50 years due to time constraints. There is however confidence that this value considerably reduces the impact of phasing on the results.

An upper bound of 600 days for the time-of-flight is too restrictive. Analysis of the GTOC2 sequences optimized as a whole (Section 10.4) suggested that the upper bound of 600 days placed on the maximum time-of-flight was too limiting, especially for Leg 3 and Leg 4 transfers. This was confirmed when raising that bound to 1200 days. The GTOC2 sequences to reach the final leg computations remained unchanged, for most sequences are discarded prior to Leg 3 computations. The GTOC2 sequences that were preserved until the Leg 4 computations saw their ranking improve considerably with respect to the results yielded by a maximum time-of-flight of 600 days, with special emphasis on GTOC2 #12, which for 20 years departure windows becomes the overall best sequence (Section 11.1.3).

12.3 High-Thrust Cost Function

The high-thrust approximation allows to quickly sort through the possible combinations. Given the low complexity of the analytically described Lambert arcs, adopting such a high-thrust model for the preliminary analysis of the GTOC2 combinatorial problem allows to rapidly assess a considerable number of sequences. For the forward multi-path NNH search, the execution times range from less than 20 minutes with DE to 2 hours with GS for the 20-years departure windows with 20 champion paths (Section 11.1.1). Increasing the DE parameters to cope with larger departure windows led to execution times up to 2 hours for 50 champion paths.

The high-thrust cost function investigated does not fully correlate with the low-thrust GTOC2 cost function. The complete high-thrust optimization of GTOC2 sequences (Section 10.4) revealed that the optimal, high-thrust final mass cost function (Equation (6.1.5)) does not lead to an evaluation coherent with their GTOC2 ranking. This is most evident when looking at the relative rankings of GTOC2 sequences in the reduced asteroid set as well as in the complete optimization.

Not including the time-of-flight in the high-thrust cost function yields better results. Comparison of the results of the forward multi-path NNH search with J_1 (Equation (6.1.4)) and J_2 shows that the latter produces more satisfying solutions (Sections 9.1 and 9.3). Therefore, introducing the time-of-flight in the high-thrust cost function decreases the correlation with the low-thrust GTOC2 objective function. This is mostly due to the large differences in time-of-flight between optimal low- and high-thrust transfers.

When compared to the routine developed by [Lancaster and Blanchard, 1969], the Izzo Lambert targeter achieves similar accuracy while being faster and more robust. The high-thrust cost of each individual transfer is evaluated based on the corresponding Lambert solution. After extensive testing and tuning of the implemented routine, it was shown that the implemented Lambert routine is twice as fast as the targeter initially designed by [Lancaster and Blanchard, 1969] and further improved by [Gooding, 1990], on similar accuracy levels (Section 7.4.4). Moreover, not suffering from a near-π singularity, the algorithm turns out to be more robust over a larger spectrum of problems.

12.4 Greedy Algorithm

The leg-per-leg approach typically decreases the quality of the sequences. Comparing the ΔV budget of the sequences optimized in a single run (Section 10.4) and in a leg-per-leg approach reveals that, overall, dividing the sequences into separately optimized legs leads to a larger final ΔV. This indicates that the departure windows considered do not allow to completely remove the effect of phasing from the results. As a result the separately optimized legs affect the phasing of subsequent legs and therefore the minimum ΔV achieved. Optimizing the complete sequences in a single run enables to perform trade-offs between the individual transfer cost to achieve a lower final ΔV budget. Note however that this would not be the case if the leg-per-leg approach was accompanied by settings leading to a fully phase-free manner.

Results display a heightened sensitivity to the decision metric. The number of champions plays an important role on the quality of the results. Throughout the scenarii investigated, the final ranking of GTOC2 sequences is highly dependent on this paramater (see e.g. Section 9.1). If it is too stringent, sequences are discarded from the following leg computations. If it is too large, the GTOC2 sequences are bumped down the ranking by previously discarded sequences. This sensitivity can be seen as a property of the search space.

The backward search does not perform as well as the forward search. In retrospect, the underperformance of the backward multi-path NNH search may be related to limiting parameters bounds, as it was shown that increasing the upper bound of the time-of-flight and of the departure windows leads to better results for the forward search (Sections 11.1.3 and 11.1.4, respectively). Nonetheless, within the bounds considered, the relatively good results for Leg 4 (GTOC2 #1 is highest ranking GTOC2 sequence for that leg) are overshadowed by the deceiving Leg 3 computations which see GTOC2 #1 fall below 50^{th} in

the best case scenario (Section 9.2). Lowering the champion threshold to that level would lead to a large computational effort with GS, although using DE may lessen the computational drawbacks. In any case, analyzing both legs leads to labelling Leg 3 as a good candidate to be the patch leg in a bi-directional multi-path NNH search.

Computational times achieved with NNH are very attractive. Within the parameters considered, the computational times inherent to the greedy algorithm are very attractive. Even more so when the individual arc cost are optimized with DE. For the forward search, the algorithm takes at most 2 hours to solve the combinatorial problem, with GS. Using DE instead, the largest execution times encountered, for the same departure windows length, are about 20 minutes (Section 11.1.1). Increasing the DE parameters and the number of champions in Section resultsdemaxwind led to maximum running times of 2 hours. This very low computational effort allows room for added complexity in the parameters of the NNH, in the parameters of the continuous optimzer or in the continuous search space, in the definition of the model.

12.5 GTOC2 Sequence Evaluation

The GTOC2 winning sequence could not be reproduced. The GTOC2 winning sequence, referred to throughout the report as GTOC2 #1, was systematically discarded at early stages, independently of the approach or the cost function. Moreover, the complete high-thrust optimization (Section 10.4) did not identify this sequence as the most promising combination. In the description of the methods used to find this sequence, high-thrust Lambert arcs are not explicitely referenced. The author believes that the reason why, in the forward search, GTOC2 #1 is rapidly discarded lies with the fact that, using the "free" excess velocity allowed in the problem definition, the goal of the first transfer was to get as far as possible from Earth (and therefore as close as possible from Group 3 asteroids). This design option would translate into a higher ΔV cost which explains why it is discarded. With respect to the backward search (Section 9.2), GTOC2 #1 is the best-ranked GTOC2 sequence in Leg 4, and the second in Leg 3 despite its overall ranking being below 80^{th} at all times in that leg. The poor overall ranking may be caused by the stringent upper bound on the time-of-flight, discussed earlier.

GTOC2 #11 is accurately and consistently ruled out as promising sequence. The GTOC2 #11 sequence, which led to the solution ranked last in the competition among the non-disqualified solutions, is systematically ruled out as promising sequence, early in the process. This occurs in every scenario considered, independently of the number of champions, of the departure windows, or of the direction of the search (see e.g. Chapter 9). One can therefore conclude that the tool accurately discards sequences which will lead to a rather poor GTOC2 objective function value.

The GTOC2 runner-up is the only non-disqualified sequence identified as promising. For departure windows length of 15 to 25 years, GTOC2 #2 reaches the last leg computations in the forward search for optimal final

mass, both with GS (Section 9.1) and DE (Section 11.1). Under favourable conditions, namely in terms of number of champions and departure windows, it ends up in the top-15 sequences identified. The fact that it is identified as a promising sequence is in tune with results from the complete optimization (Section 10.4), where it always landed on the podium. The best rank achieved (4^{th}) occured for the forward search using DE for 25 years departure windows, 10 champion paths and a larger interval for time-of-flight values. Increasing the length of the departure windows to 50 years and the number of champions to values larger than 25 lead to a slight drop in final ranking but such departure windows allow for a more robust analysis, as the effect of phasing is further diminished. For the largest number of champions considered (50), the GTOC2 runner-up is included in the top-15 subset, obtained in 2 hours (Section 11.1.4). Interestingly, this is the only sequence, along with GTOC2 #9, that was obtained exclusively via optimal Lambert solutions (Section 3.1.1).

For departure windows of 20 years, GTOC2 #12 ranks as the absolute best sequence. In the last chapter, the forward search was performed using DE to optimize individual transfers where the maximum time-of-flight was increased with respect to previous scenarii (Section 11.1.3). In this framework, for 20 years departure windows, GTOC2 #12 is tagged as the absolute most promising sequence among the ones considered in the final leg computations. This occurs for both 15 and 20 champion paths. This result is also obtained in less than 20 minutes. While the corresponding GTOC2 solution violated a number of constraints, it represents an improvement with respect to the solution submitted at the time by the team under TU Delft leadership. However, increasing the departure windows length to 50 years revealed that GTOC2 #12 would most probably be discarded from a phase-free analysis.

Other GTOC2 sequences obtained with optimal Lambert transfers could not be reproduced. A large number of GTOC2 participating teams used optimal, bi-impulsive transfers at some point when obtaining their asteroid combination (Section 3.1.1). Out of these, only GTOC2 #2 and #12 could be reproduced. The absence of the remaining sequences from the batch of final, complete sequences returned by the multi-path NNH could be explained by a preliminary pruning (GTOC2 #3, #4, and #5) or a mixed low-thrust/high-thrust model (GTOC2 #3 and #7). The absence of sequence GTOC2 #9 from the results obtained is more difficult to interpret based on the method description from Section 3.1.1. It may be related to differences in the definition of the continuous optimization problem, such as different departure windows.

12.6 GTOC2 Problem

It is difficult to reproduce all GTOC2 sequences with a single approach. Although the continuous low-thrust optimization aspect is straightforward, the GTOC2 integer problem is very large and does not contain an obvious sequence which would likely contain the optimal complete trajectory. Pruning the integer search space based on another metric than the GTOC2 cost function will likely cause promising sequences to be discarded from the resulting subset. The preliminary pruning applied by most of the GTOC2 participating

teams likely resulted in a reduced set of asteroids that excluded combinations considered promising according to the model established in this MSc Thesis.

Using a comparative evaluation of sequences is limiting. As mentioned in the previous paragraph, the GTOC2 is quite vast, both in its integer (asteroid sequences) and continuous (trajectory optimization) aspects. As a result, one can never know with certainty the true optimum of the problem. The only knowledge that we have regarding the problem is the best known trajectory, resulting from the GTOC2 winning sequence. Limiting the evaluation of the sequences found to the comparison with a very small subset of solutions, obtained in a variety of ways, does not provide an accurate estimate of their quality. Indeed, some of the sequences yielded by the proposed tool may correspond to combinations which were considered, at some point, by the GTOC2 participating teams. One can also not exclude the possibility that one of the resulting sequences contains the true optimum of the problem, as it is not known.

The GTOC2 combinatorial problem is very sensitive under the high-thrust approximation. Modifying any parameter of the search leads to different sequence rankings. This was observed when the number of champions was altered, when the size of the departure windows was changed and when the maximum time-of-flight was extended. This shows that the (partial) path costs of the different sequences are very close to one another, small modifications immediately leading to alterations in the ranking of the (partial) sequences.

Leg 4 is very sensitive to phasing. Phasing plays an important role in the final legs, and in particular in Leg 4. It explains the large variations in the final rank of sequences when the departure windows are altered. The relatively larger importance of phasing in later legs was perhaps most visible when looking at the evolution of GTOC2 sequences, leg per leg, in the reduced asteroid set (Section 10.1). Therefore, care must be taken to ensure that the last leg computations are as phase free, i.e. consider the largest array of departure and arrival dates, possible. The relatively larger effect of phasing for the final leg is linked to the orbital properties of the Group 1 asteroids, which are typically further away from the Group 2 asteroids and are, in general, more elliptical. These two factors reduce the frequency of the repeatability of the relative geometry between Group 2 and Group 1 asteroids. Increasing the departure windows in Section 11.1.4 revealed that also Leg 3 results are dependent on phasing. For this leg, the reason is the exact opposite: the proximity between Group 3 and Group 2 asteroids leads to a relative geometry that changes very slowly over time and that therefore takes a longer time to repeat.

Phasing plays a major role. While the use of a high-thrust phasing to obtain an initial guess for low-thrust trajectories has been discouraged, phasing has a major impact on the path costs of the (partial) sequences. This is true for the last legs but also for the early legs, as revealed when analyzing the effect of the caching feature (Section 10.2). To eliminate this effect, the departure windows should be large enough to allow repeatability of the relative heliocentric geometry between departure and arrival bodies. We have shown that 20 years

is not large enough to ensure said repeatability for all asteroid pairs, 50 years being a more sensible value.

Chapter 13

Recommendations

There was an effort from the author to conduct a thorough and sensible analysis of the performance of the different components and variants of the designed combinatorial tool, to improve the models and techniques based on the feedback provided by the results achieved. However, this iterative process can easily become neverending and the investigation needed to be interrupted due to timeliness considerations. As a result, this chapter lists a number of recommendations for possible future work. These recommendations are based on the conclusions discussed in the previous chapter and therefore follow the same structure.

13.1 Continuous Optimization Technique

Use an implicit enumerative method for global continuous optimization instead of a pure enumerative method. One of the first recommendations which arises from the work carried out in this MSc Thesis is the use of implicit enumerative methods, such as DE, to perform global, continuous optimization. When compared to pure enumerative techniques, e.g. GS, they achieve similar, if not better, solutions with much less computational effort. This allows to remove computational load from the optimization procedure and redistribute it to other areas of the problem, e.g. the implementation of a more accurate model, with the improvement of the results as a goal.

For the purposes of space trajectory optimization, stricter DE parameters should be used. This is particularly true for problems with larger search spaces than the ones consider in this report. Increasing the number of individuals in the solution population and/or the number of generations for which the population is evolved confers an added consistency to the results. Besides the consistency, it increases the probability of finding the true optimum and therefore the confidence in the output from the optimization procedure. Given the low computational times achieved in the MSc Thesis work, there is definitely room for a larger number of individuals and generations.

Investigate the quality of the results found with different optimization techniques. Now that the interface between PaGMO and Tudat has been successfully established and is well documented, replacing the self-adaptive DE

with any other optimization algorithm available in the external toolbox comes at (nearly) no cost. An interesting avenue may be the use of an integer optimizer such as Ant Colony Optimization (ACO).

13.2 Optimization Problem Definition

Increase the departure windows length. We have seen the impact of phasing on the solutions returned by the multi-path NNH. In order to eliminate, or at least minimize, its effect, the maximum departure windows considered could be extended beyond the 50 years considered in this report. If doing so leads to significant computational drawbacks, the departure window can be determined for each specific asteroid pair, as not all asteroid couples will require the same departure window length to conduct a phase-free analysis. An estimate of the necessary time span could be bbased on the circular, co-planar synodic period, as given in Equation (6.2.9), although its suitability for this purpose should first be investigated.

Increase the maximum time-of-flight. When the maximum time-of-flight was increased from 600 to 1200 days, the ranking of the GTOC2 sequences improved considerably. Moreover, the dependence on the number of champions was mitigated. It is conceivable that pushing the upper bound for the time-of-flight of individual transfers further will have the same effect and lead to even better results.

13.3 High-Thrust Cost Function

Introduce a low-thrust model. The primary reason for selecting a high-thrust cost function for the preliminary analysis of the (low-thrust) GTOC2 problem was the fact that high-thrust trajectories can be computed considerably faster than low-thrust transfers. Given the very short computational times observed, there is space for introducing a low-thrust cost function. This cost function could be based on e.g. shape-based models (see e.g. [Petropoulos and Longuski, 2004], [Wall and Conway, 2009]) which have proven to provide a more reliable initial guess for low-thrust trajectories than high-thrust models. In order to avoid compromising the computational tractability of the resulting problem, this low-thrust cost could be applied only to one leg. This approach was adopted to obtain GTOC2 #3 and #7: the Leg 2 transfer was taken to be a low-thrust arc, as it corresponds to the most costly leg under high-thrust assumption (see [Evertsz, 2008]).

13.4 Greedy Algorithm

Build up the sequences by adding each new leg to the partial sequence optimization. We have already discussed the important effects of phasing on the quality of the results. Earlier, it was suggested to increase the departure windows length to mitigate the impact of phasing. However, should this approach be unpractical due to e.g. severe execution times drawbacks, the phasing issue can be tackled from a different angle. Instead of trying to get rid

of the phasing via a phase-free analysis, one can accept it into the optimization procedure. At each leg, instead of optimizing the particular transfer in an isolated fashion, the trajectory, up to that leg, can be optimized as a whole. In doing so, the optimizer performs the necessary trade-offs to yield the (partial) trajectory with the lowest cost. Doing so would also limit the adverse effects of a leg-per-leg approach.

Use a more flexible decision metric. The sensitivity of the results with respect to the number of champion paths has been thouroughly discussed. Possible ways to mitigate this sensitivity rely on switching from an absolute and constant number of champions to a decision metric which determines the number of champion paths based on the characteristics of the solution space. A relative number of champions, as described in Section 5.2.1, would selected the champion paths based on the size of the solution space. Another approach is to take into consideration the fitness of the population of solutions and adapt the champions accordingly. Such a decision metric could rely on fitting a distribution to the solutions, as suggested by [Alemany and Braun, 2007].

Use a more accurate combinatorial algorithm. A conscious choice was made, during the design process of the combinatorial tool, to opt for a greedy algorithm and to attempt to improve its accuracy while preserving its low computational load. Given that positive results were obtained with NNH in very short execution times, a more complex combinatorial optimization algorithm, such as B&B, can be considered. Note that this type of algorithms has already been investigated by [Gorter, 2010] with a simplified arc cost evaluation, based on the orbital elements of the asteroids.

13.5 GTOC2 Sequence Evaluation

Perform an absolute evaluation of the asteroid combinations. It is recommended to have an accurate low-thrust model before further investigations on the combinatorial aspect of GTOC2 are undertaken. With such a model, coupled to a suitable optimizer, an absolute (and fairer) evaluation of the quality of the sequences can be achieved. Such a model is currently being developed by MSc students at the A&S department and therefore should be available in the near future.

Bibliography

Aardal, K. (2011). Private Communication.

Aarts, E. and Lenstra, J. K. (1997). *Local Search in Combinatorial Optimization*. A Wiley-Interscience Publication.

ACT (Last accessed: March 2011). ACT Informatics - GTOP Database: Global Optimisation Trajectory Problems and Solutions. `http://www.esa.int/gsp/ACT/inf/op/globopt.htm`.

Ahuja, R. K., Orlin, J. B., Pallottino, S., and Scutellá, M. G. (2003). Dynamic Shortest Path Minimizing Travel Times and Costs. *Networks*, 14:197–205.

Alemany, K. and Braun, R. D. (2007). Design Space Pruning Techniques for Low-Thrust, Multiple Asteroid Rendezvous Trajectory Design. In *17th AAS/AIAA Space Flight Mechanics Meeting*.

Battin, R. H. (1999). *An Introduction to the Mathematics and Methods of Astrodynamics*. AIAA Education Series.

Bellman, R. (1958). On a Routing Problem. *Quarterly of Applied Mathematics*, 16(1):87–90.

Biscani, F., Izzo, D., and Yam, C. H. (2010). A Global Optimisation Toolbox for Massively Parallel Engineering Optimisation. In *4th International Conference on Astrodynamics Tools and Techniques (ICATT 2010)*, ESA/ESAC, Madrid, Spain.

Brest, J., Greiner, S., Boskovic, B., Mernik, M., and Zumer, V. (2006). Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems. *IEEE Transactions on Evolutionary Computation*, 10(6):646–657.

Casalino, L., Colasurdo, G., Sentinella, M. R., and Cacciatore, F. (2007). 2nd Global Trajectory Optimization Competition Workshop - Team 4. In *17th AAS/AIAA Space Flight Mechanics Meeting*.

Chabini, I. (1998). Discrete Dynamic Shortest Path Problems In Transportation Applications: Complexity And Algorithms With Optimal Run Time. *Transportation Research Records*, 1645:170–175.

Coello, C. A. C., Lamont, G. B., and Veldhuizen, D. A. V. (2006). *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Dial, R. B. (1969). Algorithm 360: Shortest-Path Forest with Topological Ordering [H]. *Commun. ACM*, 12:632–633.

Dijkstra, E. W. (1959). A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik*, 1:269–271.

Evertsz, C. (2008). GTOC2: Multiple Asteroid Rendezvous. Master's thesis, TUDelft.

Gooding, R. H. (1990). A Procedure for the Solution of Lambert's Orbital Boundary-Value Problem. *Celestial Mechanics and Dynamical Astronomy*, 48(2):145–165.

Gorter, H. (2009). Analysis of Models and Methods for Solving Multiple Leg Low-Thrust Trajectory Problems: Literature Survey. Master's thesis, TUDelft.

Gorter, H. (2010). Models and Methods for GTOC2: Analysis of a Multiple Asteroid Rendezvous Mission. Master's thesis, TUDelft.

Hartmann, A. K. and Weigt, M. (2005). *Phase Transitions in Combinatorial Optimization Problems - Basics, Algorithms and Statistical Mechanics*. Wiley-VCH.

Izzo, D. (2011). Private Communication.

Kemble, S. (2010). Interplanetary Mission Analysis. In *4th International Conference on Astrodynamics Tools and Techniques*, ESA/ESAC, Madrid, Spain.

Kennedy, J. and Eberhart, R. (1995). Particle Swarm Optimization. In *IEEE International Conference on Neural Networks IV*, University of Western Australia, Perth, Australia.

Kumar, K. (2011). Design of a Modular and Robust Astrodynamics Toolbox. In *New Trends in Astrodynamics and Applications IV*, Courant Institute of Mathematical Sciences, New York University, New York, NY, USA.

Kumar, K., Abdulkadir, Y., van Barneveld, P., Belien, F., Billemont, S., Brandon, E., Dijkstra, M., Dirkx, D., Engelen, F., Gondelach, D., van der Ham, L., Heeren, E., Iorfida, E., Leloux, J., Melman, J., Mooij, E., Musegaas, P., Noomena, R., Persson, S., Römgens, B., Ronse, A., Secretin, T. L. P., Minh, B. T., and Vandamme, J. (2012). Tudat: A Modular and Robust Astrodynamics Toolbox. In *5th International Conference on Astrodynamics Tools and Techniques (ICATT 2012)*, ESA/ESTEC, Noordwijk, the Netherlands.

Lancaster, E. R. and Blanchard, R. C. (1969). A Unified Form of Lambert's Theorem. Technical Report NASA TN D5368, NASA.

Lewis, J., Matthews, M., and Guerrieri, M. (1993). *Resources of near-Earth space*. University of Arizona Space Science Series. University of Arizona Press.

Melman, J. (2011). The N Commandments. `http://tudat.tudelft.nl/projects/tudat/repository/changes/documents/Developers/Protocols/The_N_Commandments.pdf`.

Munkres, J. (1957). Algorithms for the Assignment and Transportation Problems. *Siam Journal on Control and Optimization*, 5(1):32–38.

Myatt, D., Becerra, V., Nasuto, S., and Bishop, J. (2004). Advanced Global Optimisation Tools for Mission Analysis and Design. Technical Report 03-4101a, European Space Agency, the Advanced Concepts Team.

Nannicini, G. and Liberti, L. (2008). Shortest Path on Dynamic Graphs. *International Transactions in Operational Research*, 15:1–13.

Oldenhuis, R. (2011). Private Communication.

Petropoulos, A. E. (2006). *Problem Description for the 2nd Global Trajectory Optimisation Competition*. Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA.

Petropoulos, A. E. (2007). *Final Rankings and Brief Descriptions of the Returned Solutions and Methods Used for the 2nd Global Trajectory Optimisation Competition*. Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA.

Petropoulos, A. E. and Longuski, J. M. (2004). Shape-Based Algorithm for the Automated Design of Low-Thrust, Gravity Assist Trajectories. *Journal of Spacecraft and Rockets*, 41:787–796.

Price, K., Storn, R. M., and Lampinen, J. A. (2005). *Differential Evolution: A Practical Approach to Global Optimization (Natural Computing Series)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Schrijver, A. (2010). *A Course on Combinatorial Optimization*. Department of Mathematics, University of Amsterdam, Amsterdam, The Netherlands.

Secretin, T. (2011). Design of a Combinatorial Tool for Preliminary Space Mission Analysis, applied to the GTOC2 problem: Literature Survey. Master's thesis, TUDelft.

Secretin, T. and Noomen, R. (2012). Design of a Combinatorial Tool for Preliminary Space Mission Analysis. In *5th International Conference on Astrodynamics Tools and Techniques (ICATT 2012)*, ESA/ESTEC, Noordwijk, the Netherlands.

Storn, R. and Price, K. (1995). Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces. Technical Report TR-95-012, International Computer Science Institute.

TheMathPage (Last accessed: March 2012). Trigonometry. `http://www.themathpage.com/atrig/trigonometric-identities.htm`.

Tudat (2011). Tudat Developers' Manual. `http://tudat.tudelft.nl/projects/tudat/wiki`.

Tudat (Last accessed: August 2012). Tudat Organogram. `http://tudat.tudelft.nl/projects/tudat/wiki`.

Wakker, K. (2007a). *AE4-874: Astrodynamics - I*. Lecture Notes, TUDelft.

Wakker, K. (2007b). *AE4-874: Astrodynamics - II.* Lecture Notes, TUDelft.

Wall, B. J. and Conway, B. A. (2009). Shape-Based Approach to Low-Thrust Rendezvous Trajectory Design. *Journal of Guidance, Control, and Dynamics*, 32(1).

Wikipedia (Last accessed: April 2010a). Asteroid belt. `http://en.wikipedia.org/wiki/Main_belt`.

Wikipedia (Last accessed: April 2010b). Asteroid spectral types. `http://en.wikipedia.org/wiki/Asteroid_spectral_types`.

Wikipedia (Last accessed: September 2011a). Accretion (astrophysics). `http://en.wikipedia.org/wiki/Accretion_(astrophysics)`.

Wikipedia (Last accessed: September 2011b). Dawn (spacecraft). `http://en.wikipedia.org/wiki/Dawn_(spacecraft)`.

Wikipedia (Last accessed: September 2011c). Hayabusa. `http://en.wikipedia.org/wiki/Hayabusa`.

Wikipedia (Last accessed: September 2011d). Rosetta (spacecraft). `http://en.wikipedia.org/wiki/Rosetta_(spacecraft)`.

Yeomans, D. K. (Last accessed: September 2011). Why Study Asteroids? `http://ssd.jpl.nasa.gov/?why_asteroids`.

Ziliaskopoulos, A. and Wardell, W. (2000). An Intermodal Optimum Path Algorithm for Multimodal Networks with Dynamic Arc Travel Times and Switching Delays. *European Journal of Operations Research*, 125:486–502.

Appendix A

Trigonometric Identities

This appendix contains a number of trigonometric identities and is based on [TheMathPage, 2012]. These were mostly used for the derivation of the equations from the Lagrange-Gauss equations in Chapter 7.

A.1 Tangent and Cotangent Identities

$$\tan a = \frac{\sin a}{\cos a} \tag{A.1.1}$$

$$\cot a = \frac{\cos a}{\sin a} \tag{A.1.2}$$

A.2 Pythagorean Identities

$$\sin^2 a + \cos^2 a = 1 \tag{A.2.1}$$

$$1 + \tan^2 a = \frac{1}{\cos^2 a} \tag{A.2.2}$$

$$1 + \frac{1}{\tan^2 a} = \frac{1}{\sin^2 a} \tag{A.2.3}$$

A.3 Sum and Difference Formulae

$$\sin(a + b) = \sin a \cos b + \cos a \sin b \tag{A.3.1}$$

$$\sin(a - b) = \sin a \cos b - \cos a \sin b \tag{A.3.2}$$

$$\cos(a + b) = \cos a \cos b - \sin a \sin b \tag{A.3.3}$$

$$\cos(a - b) = \cos a \cos b + \sin a \sin b \tag{A.3.4}$$

A.4 Half-angle Formulae

$$\cos^2 \frac{1}{2}a = \frac{1}{2}(1 + \cos a) \tag{A.4.1}$$

$$\sin^2 \frac{1}{2}a = \frac{1}{2}(1 - \cos a) \tag{A.4.2}$$

A.5 Products as Sums

$$\sin a \cos b = \frac{1}{2}\left[\sin(a + b) + \sin(a - b)\right] \tag{A.5.1}$$

$$\cos a \sin b = \frac{1}{2}\left[\sin(a + b) - \sin(a - b)\right] \tag{A.5.2}$$

$$\cos a \cos b = \frac{1}{2}\left[\cos(a + b) + \cos(a - b)\right] \tag{A.5.3}$$

$$\sin a \sin b = -\frac{1}{2}\left[\cos(a + b) - \cos(a - b)\right] \tag{A.5.4}$$

A.6 Sums as Products

$$\sin a + \sin b = 2 \sin \frac{1}{2}(a + b) \cos \frac{1}{2}(a - b) \tag{A.6.1}$$

$$\sin a - \sin b = 2 \sin \frac{1}{2}(a - b) \cos \frac{1}{2}(a + b) \tag{A.6.2}$$

$$\cos a + \cos b = 2 \cos \frac{1}{2}(a + b) \cos \frac{1}{2}(a - b) \tag{A.6.3}$$

$$\cos a - \cos b = -2 \sin \frac{1}{2}(a + b) \sin \frac{1}{2}(a - b) \tag{A.6.4}$$

Appendix B

Algorithms Tuning and Verification

This appendix contains additional data relative to the tuning and verification of the external self-adaptive DE routine and of the implemented Lambert solver.

B.1 Self-Adaptive Differential Evolution Tuning

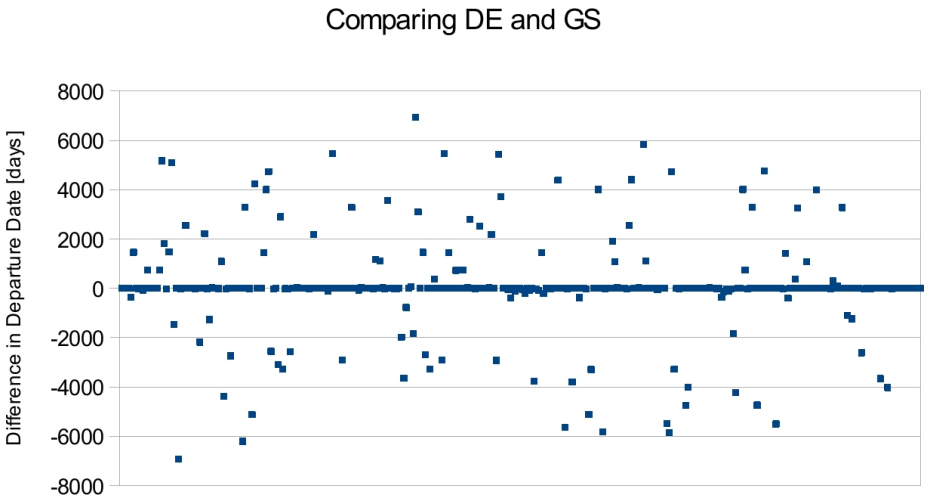

Figure B.1: Difference in departure date between the best Leg 1 transfers found by DE (with 100 individuals and 50 generations) and GS.

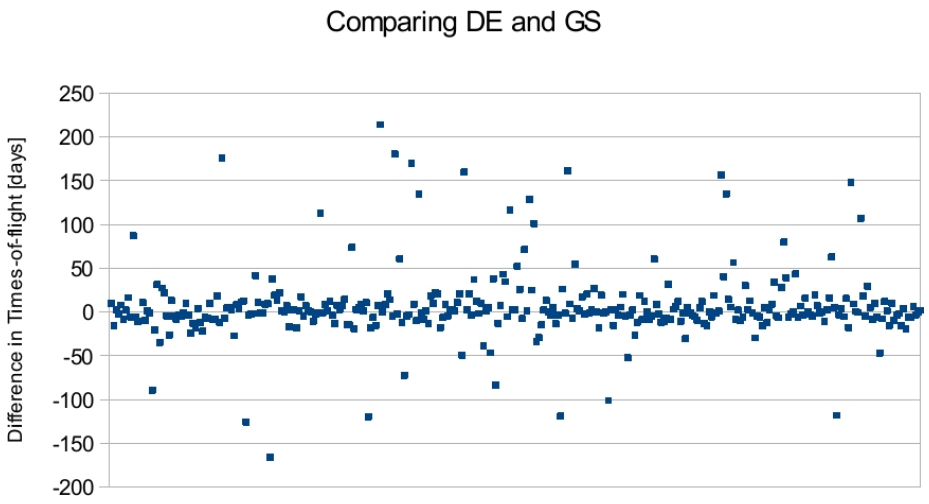

Figure B.2: Difference in time-of-flight between the best Leg 1 transfers found by DE (with 100 individuals and 50 generations) and GS.

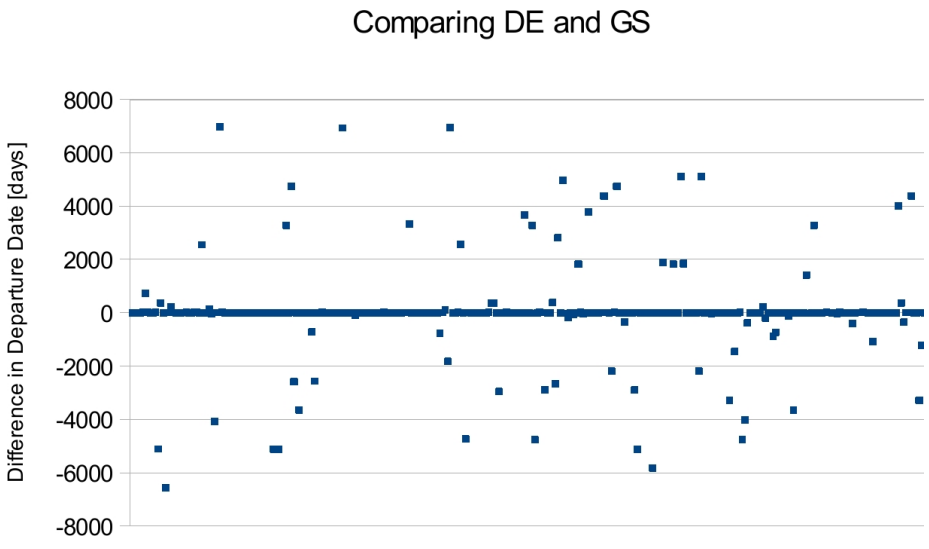

Figure B.3: Difference in departure date between the best Leg 1 transfers found by DE (with 50 individuals and 150 generations) and GS.

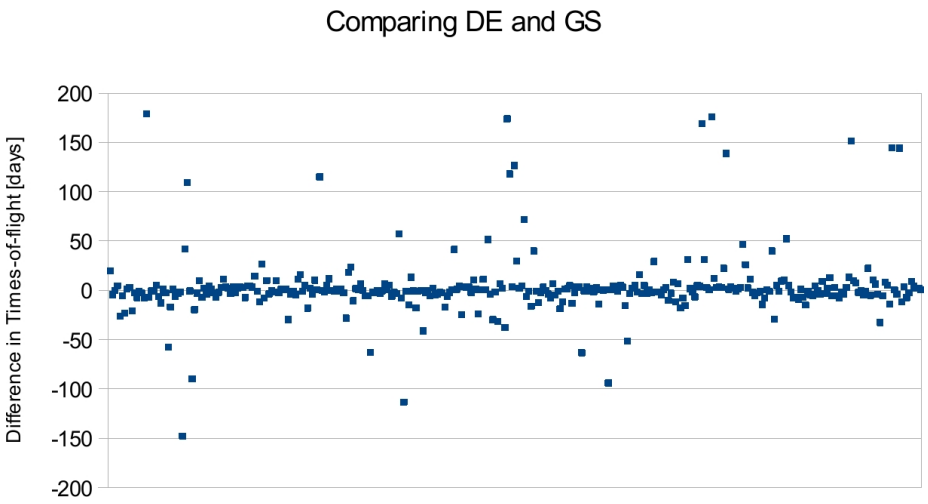

Figure B.4: Difference in time-of-flight between the best Leg 1 transfers found by DE (with 50 individuals and 150 generations) and GS.

B.2 Lambert Solver Verification

B.2.1 Test Case 1

Table B.1: Maximum and mean absolute errors for Test Case 1, for each run.

Quantity		Run #1	Run #2	Run #3
V_{x1}	Maximum	$4.760 \cdot 10^{-7}$	$2.386 \cdot 10^{-7}$	$2.052 \cdot 10^{-7}$
	Mean	$2.288 \cdot 10^{-8}$	$1.605 \cdot 10^{-8}$	$1.414 \cdot 10^{-8}$
V_{y1}	Maximum	$1.256 \cdot 10^{-6}$	$6.601 \cdot 10^{-7}$	$3.919 \cdot 10^{-7}$
	Mean	$3.449 \cdot 10^{-8}$	$2.034 \cdot 10^{-8}$	$1.729 \cdot 10^{-8}$
V_{z1}	Maximum	$1.104 \cdot 10^{-6}$	$1.723 \cdot 10^{-7}$	$3.287 \cdot 10^{-7}$
	Mean	$3.025 \cdot 10^{-8}$	$1.512 \cdot 10^{-8}$	$1.558 \cdot 10^{-8}$
V_1	Maximum	$1.739 \cdot 10^{-6}$	$7.133 \cdot 10^{-7}$	$5.512 \cdot 10^{-7}$
	Mean	$5.573 \cdot 10^{-8}$	$3.447 \cdot 10^{-8}$	$2.996 \cdot 10^{-8}$
V_{x2}	Maximum	$1.240 \cdot 10^{-6}$	$4.177 \cdot 10^{-7}$	$4.648 \cdot 10^{-7}$
	Mean	$3.579 \cdot 10^{-8}$	$2.183 \cdot 10^{-8}$	$1.648 \cdot 10^{-8}$
V_{y2}	Maximum	$1.020 \cdot 10^{-6}$	$1.545 \cdot 10^{-7}$	$2.818 \cdot 10^{-7}$
	Mean	$3.105 \cdot 10^{-8}$	$1.339 \cdot 10^{-8}$	$1.631 \cdot 10^{-8}$
V_{z2}	Maximum	$6.563 \cdot 10^{-7}$	$5.580 \cdot 10^{-7}$	$1.210 \cdot 10^{-7}$
	Mean	$2.166 \cdot 10^{-8}$	$1.732 \cdot 10^{-8}$	$1.483 \cdot 10^{-8}$
V_2	Maximum	$1.735 \cdot 10^{-6}$	$7.140 \cdot 10^{-7}$	$5.489 \cdot 10^{-7}$
	Mean	$5.548 \cdot 10^{-8}$	$3.461 \cdot 10^{-8}$	$3.089 \cdot 10^{-8}$

B.2.2 Test Case 2

Table B.2: Parameters for the Test Case 2. Varying parameters are represented in the $i : s : f$ format, with i, s and f the initial, step-size and final values, respectively.

Time-of-flight, t_f [days]	200:50:1000	
Gravitational parameter, μ [km^3/s^2]	$1.32712440018 \times 10^{11}$	
Orbital parameters	Planet	
	Earth	Mars
Semi-major axis, a [AU]	1	1.5
Eccentricity, e [-]	0	0
Inclination, i [deg]	0	0
RAAN, Ω [deg]	0	0
Argument of periapsis, ω [deg]	0	0
Mean anomaly, M [deg]	0	1:1:360

B.2.3 Sensitivity Analysis

Table B.3: Maximum values for the absolute erros in the velocity components and norms and for the number of iterations as a function of the convergence tolerance.

ϵ	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-9}	10^{-12}
Component	Maximum Absolute Error [m/s]					
$V_{1,x}$	$6.36 \cdot 10^{-1}$	$5.50 \cdot 10^{-2}$	$5.21 \cdot 10^{-3}$	$1.18 \cdot 10^{-5}$	$1.18 \cdot 10^{-5}$	$1.18 \cdot 10^{-5}$
$V_{1,y}$	1.06	$4.12 \cdot 10^{-2}$	$1.36 \cdot 10^{-3}$	$1.33 \cdot 10^{-5}$	$1.33 \cdot 10^{-5}$	$1.33 \cdot 10^{-5}$
$\|V_1\|$	1.12	$6.87 \cdot 10^{-2}$	$5.38 \cdot 10^{-3}$	$1.51 \cdot 10^{-5}$	$1.51 \cdot 10^{-5}$	$1.51 \cdot 10^{-5}$
$V_{2,x}$	1.20	$4.20 \cdot 10^{-2}$	$4.51 \cdot 10^{-3}$	$1.25 \cdot 10^{-5}$	$1.25 \cdot 10^{-5}$	$1.25 \cdot 10^{-5}$
$V_{2,y}$	1.22	$7.15 \cdot 10^{-2}$	$3.09 \cdot 10^{-3}$	$1.94 \cdot 10^{-5}$	$1.94 \cdot 10^{-5}$	$1.94 \cdot 10^{-5}$
$\|V_2\|$	1.71	$7.49 \cdot 10^{-2}$	$5.47 \cdot 10^{-3}$	$2.26 \cdot 10^{-5}$	$2.26 \cdot 10^{-5}$	$2.26 \cdot 10^{-5}$
Maximum Number of Iterations [-]						
# Iterations	4	5	5	5	6	7

Table B.4: RMS values for the absolute erros in the velocity components and norms and for the number of iterations as a function of the convergence tolerance.

ϵ	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-9}	10^{-12}
Component	RMS Absolute Error [m/s]					
$V_{1,x}$	$5.21 \cdot 10^{-2}$	$1.66 \cdot 10^{-3}$	$7.02 \cdot 10^{-5}$	$1.06 \cdot 10^{-6}$	$1.06 \cdot 10^{-6}$	$1.06 \cdot 10^{-6}$
$V_{1,y}$	$3.15 \cdot 10^{-2}$	$7.45 \cdot 10^{-4}$	$2.02 \cdot 10^{-5}$	$9.24 \cdot 10^{-7}$	$9.25 \cdot 10^{-7}$	$9.25 \cdot 10^{-7}$
$\|V_1\|$	$6.08 \cdot 10^{-2}$	$1.82 \cdot 10^{-3}$	$7.30 \cdot 10^{-5}$	$1.41 \cdot 10^{-6}$	$1.40 \cdot 10^{-6}$	$1.40 \cdot 10^{-6}$
$V_{2,x}$	$4.87 \cdot 10^{-2}$	$1.15 \cdot 10^{-3}$	$5.79 \cdot 10^{-5}$	$9.97 \cdot 10^{-7}$	$9.98 \cdot 10^{-7}$	$9.98 \cdot 10^{-7}$
$V_{2,y}$	$4.99 \cdot 10^{-2}$	$1.53 \cdot 10^{-3}$	$4.82 \cdot 10^{-5}$	$1.51 \cdot 10^{-6}$	$1.51 \cdot 10^{-6}$	$1.51 \cdot 10^{-6}$
$\|V_2\|$	$6.97 \cdot 10^{-2}$	$1.91 \cdot 10^{-3}$	$7.53 \cdot 10^{-5}$	$1.81 \cdot 10^{-6}$	$1.81 \cdot 10^{-6}$	$1.81 \cdot 10^{-6}$
RMS Number of Iterations [-]						
# Iterations	3.31	3.88	4.09	4.27	5.09	5.63

Appendix C

Best Sequences in the Complete Asteroid Pool, using GS

This appendix contains the 10 best sequences, for the forward multi-path NNH search with J_2 as a cost function, for 10-years departure windows, according to the number of champion paths. This specific departure windows length did not lead to any GTOC2 sequence being present in the final leg computations (see Section 9.1). Full results for all search directions, cost functions, departure windows and number of champions investigated in Chapter 9, found in the complete asteroid pool using GS, can be obtained by contacting the author.

Table C.1: Top-10 combinations, according to J_2, found for 10-years departure windows by a forward multi-path NNH with N=3, using GS

Rank	Group 4	Group 3	Group 2	Group 1	Total ΔV [m/s]
1	3250293	2001990	2000058	2001345	18851.9
2	3250293	2001990	2000442	2001754	19529.4
3	3250293	2001990	2000058	2001754	19945.3
4	3250293	2001990	2000821	2032511	19995.1
5	3250293	2001990	2000821	2003134	20139.2
6	3250293	2001990	2000442	2002483	20671.0
7	3250293	2001990	2000821	2002959	20706.0
8	3250293	2001990	2000442	2001345	21281.3
9	3250293	2001990	2000442	2015278	21486.3
10	3250293	2001990	2000442	2011542	21789.9

Table C.2: Top-10 combinations, according to J_2, found for 10-years departure windows by a forward multi-path NNH with N=4, using GS

Rank	Group 4	Group 3	Group 2	Group 1	Total ΔV [m/s]
1	3250293	2001990	2000058	2001345	18851.9
2	3250293	2001990	2000442	2001754	19529.4
3	3250293	2001990	2000058	2001754	19945.3
4	3250293	2001990	2000821	2032511	19995.1
5	3250293	2001990	2000821	2003134	20139.2
6	3250293	2001990	2000442	2002483	20671.0
7	3250293	2001990	2000821	2002959	20706.0
8	3042555	2001133	2000021	2001038	20991.8
9	3250293	2001990	2000442	2001345	21281.3
10	3250293	2001990	2000442	2015278	21486.3

Table C.3: Top-10 combinations, according to J_2, found for 10-years departure windows by a forward multi-path NNH with N=5, using GS

Rank	Group 4	Group 3	Group 2	Group 1	Total ΔV [m/s]
1	3250293	2001990	2000058	2001345	18851.9
2	3250293	2001990	2000442	2001754	19529.4
3	3250293	2001990	2000585	2001754	19621.3
4	3250293	2001990	2000058	2001754	19945.3
5	3250293	2001990	2000821	2032511	19995.1
6	3250293	2001990	2000821	2003134	20139.2
7	3250293	2001990	2000442	2002483	20671.0
8	3250293	2001990	2000821	2002959	20706.0
9	3042555	2001133	2000021	2001038	20991.8
10	3250293	2001990	2000442	2001345	21281.3

Table C.4: Top-10 combinations, according to J_2, found for 10-years departure windows by a forward multi-path NNH with N=10, using GS

Rank	Group 4	Group 3	Group 2	Group 1	Total ΔV [m/s]
1	3250293	2000270	2000395	2003134	18442.6
2	3250293	2000937	2000395	2003134	18523.3
3	3250293	2001990	2000058	2001345	18851.9
4	3250293	2001990	2000442	2001754	19529.4
5	3250293	2000851	2000206	2001754	19769.8
6	3250293	2000901	2000135	2014569	19889.4
7	3250293	2001990	2000058	2001754	19945.3
8	3250293	2001990	2000821	2032511	19995.1
9	3250293	2001990	2000821	2003134	20139.2
10	3250293	2000851	2000206	2001345	20639.8

Table C.5: Top-10 combinations, according to J_2, found for 10-years departure windows by a forward multi-path NNH with N=13, using GS

Rank	Group 4	Group 3	Group 2	Group 1	Total ΔV [m/s]
1	3250293	2000270	2000395	2003134	18442.6
2	3250293	2000937	2000395	2003134	18523.3
3	3170221	2000951	2000395	2003134	18526.5
4	3250293	2001990	2000058	2001345	18851.9
5	3250293	2001990	2000442	2001754	19529.4
6	3250293	2001990	2000585	2001754	19621.3
7	3250293	2000851	2000206	2001754	19769.8
8	3170221	2000951	2000418	2002959	19870.7
9	3250293	2000901	2000135	2014569	19889.4
10	3250293	2001990	2000058	2001754	19945.3

Table C.6: Top-10 combinations, according to J_2, found for 10-years departure windows by a forward multi-path NNH with N=15, using GS

Rank	Group 4	Group 3	Group 2	Group 1	Total ΔV [m/s]
1	3250293	2000270	2000395	2003134	18442.6
2	3250293	2000937	2000395	2003134	18523.3
3	3170221	2000951	2000395	2003134	18526.5
4	3250293	2001990	2000058	2001345	18851.9
5	3250293	2001990	2000442	2001754	19529.4
6	3250293	2001990	2000585	2001754	19621.3
7	3250293	2000851	2000206	2001754	19769.8
8	3170221	2000951	2000418	2002959	19870.7
9	3250293	2000901	2000135	2014569	19889.4
10	3250293	2001990	2000207	2001038	19918.6

Table C.7: Top-10 combinations, according to J_2, found for 10-years departure windows by a forward multi-path NNH with N=20, using GS

Rank	Group 4	Group 3	Group 2	Group 1	Total ΔV [m/s]
1	3250293	2000270	2000395	2003134	18442.6
2	3250293	2000937	2000395	2003134	18523.3
3	3170221	2000951	2000395	2003134	18526.5
4	3042555	2000296	2000240	2001038	18533.2
5	3250293	2001990	2000058	2001345	18851.9
6	3042555	2001133	2000144	2011542	19224
7	3042555	2000296	2000240	2009661	19279
8	3250293	2001990	2000442	2001754	19529.4
9	3250293	2001990	2000585	2001754	19621.3
10	3250293	2000851	2000206	2001754	19769.8

Appendix D

Additional Data from the Reduced Set of Asteroids

This appendix contains the leg-per-leg evolution of the rank of GTOC2 #5, #7 and GTOC2 #9 through #13, according to the forward multi-path search performed with GS and without caching in Section 10.1. Moreover, the total ΔV of the GTOC2 sequences optimized in a complete, single DE run (Section 10.4) is given. Additional data concerning the investigation in the reduced asteroid pool can be obtained by contacting the author.

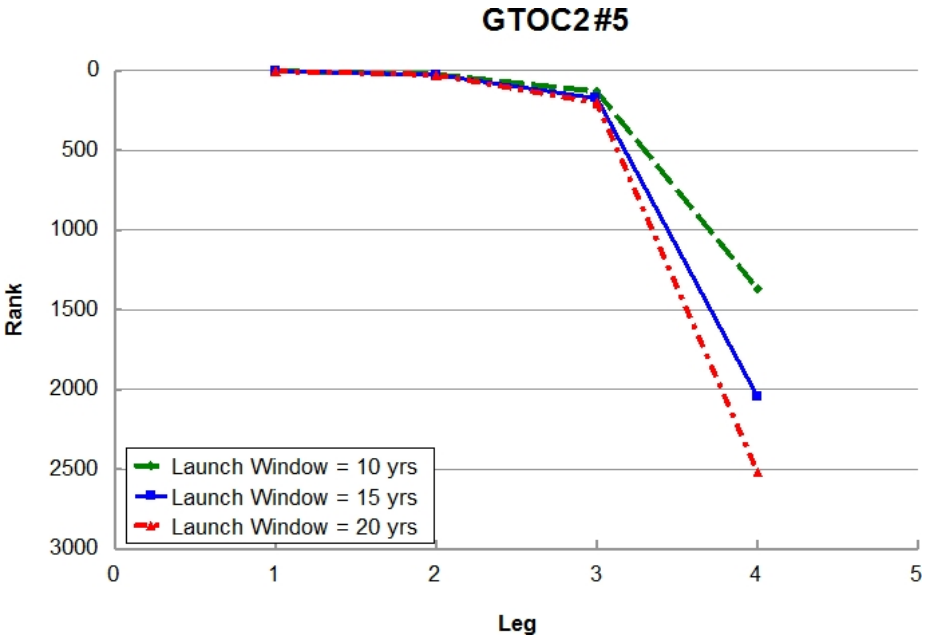

Figure D.1: Ranking of GTOC2 #5 in the reduced asteroid set, using GS.

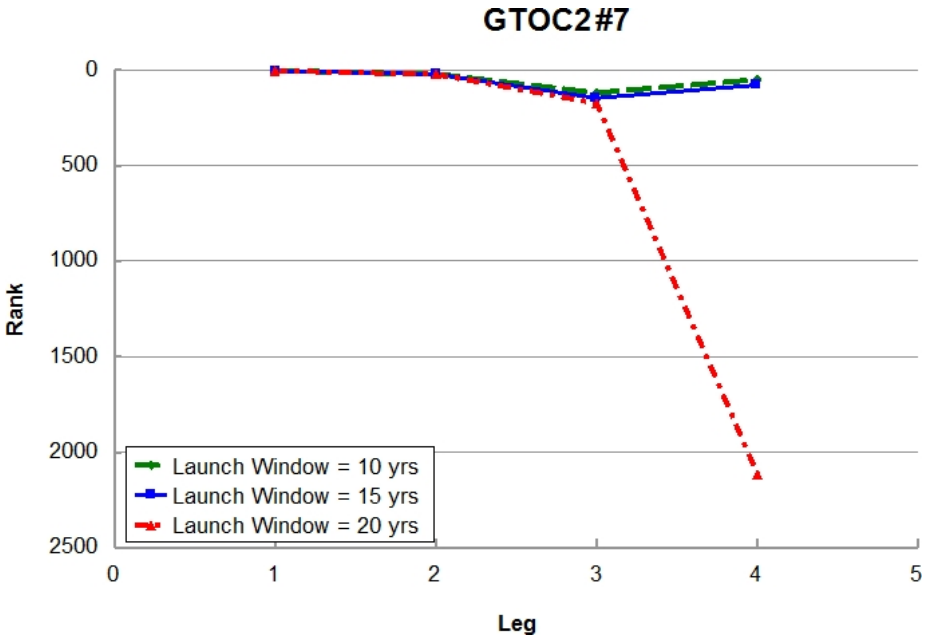

Figure D.2: Ranking of GTOC2 #7 in the reduced asteroid set, using GS.

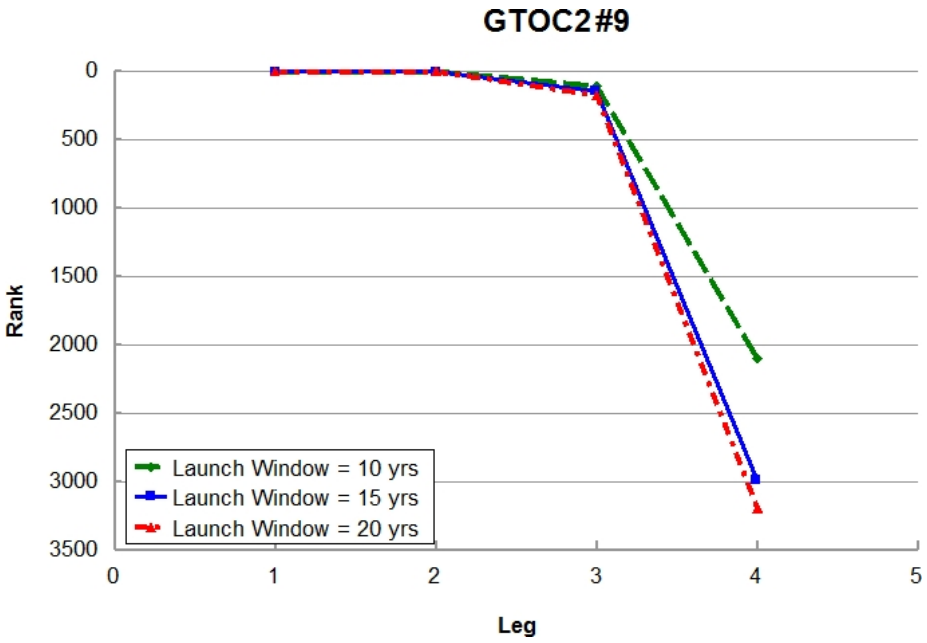

Figure D.3: Ranking of GTOC2 #9 in the reduced asteroid set, using GS.

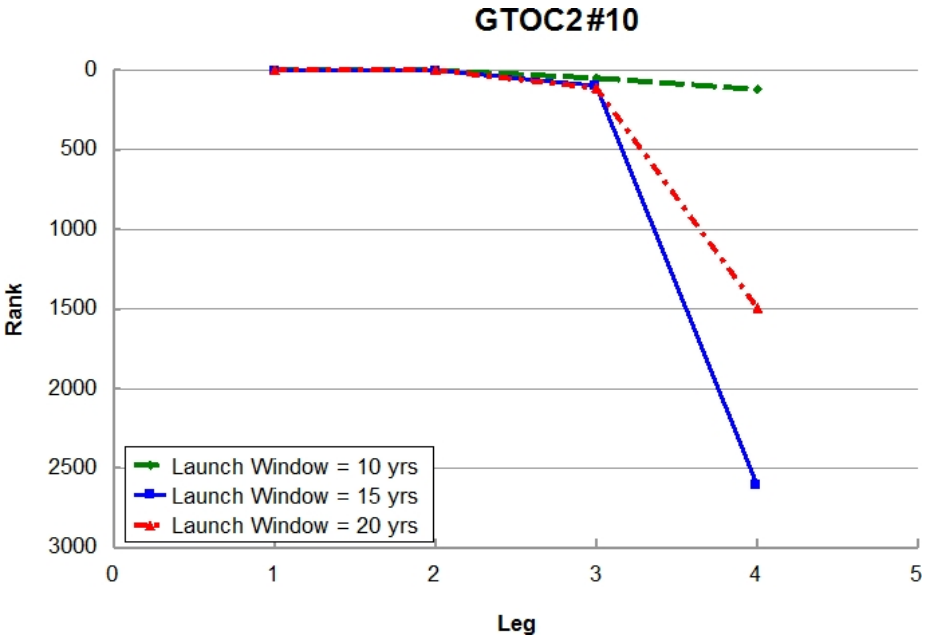

Figure D.4: Ranking of GTOC2 #10 in the reduced asteroid set, using GS.

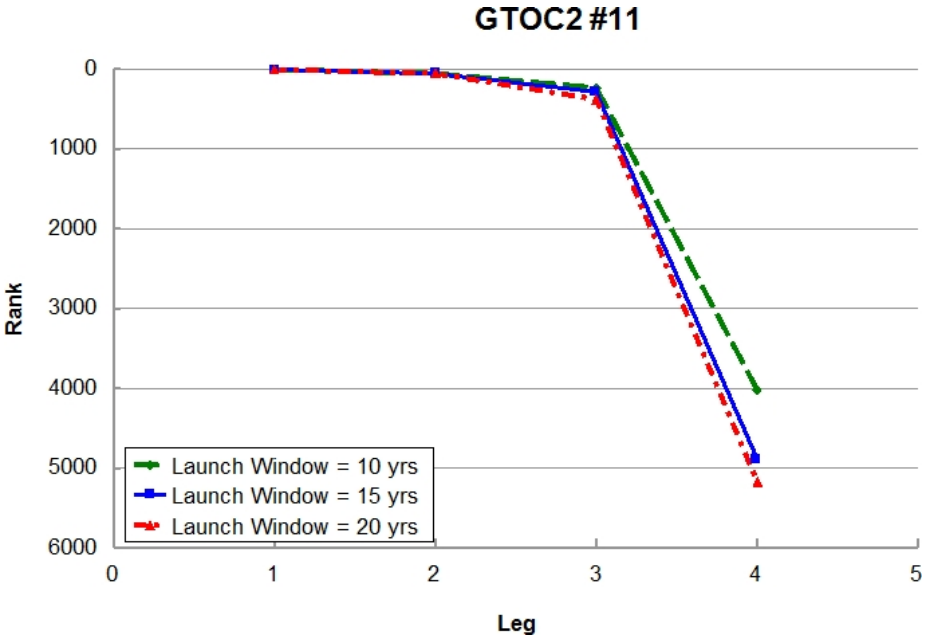

Figure D.5: Ranking of GTOC2 #11 in the reduced asteroid set, using GS.

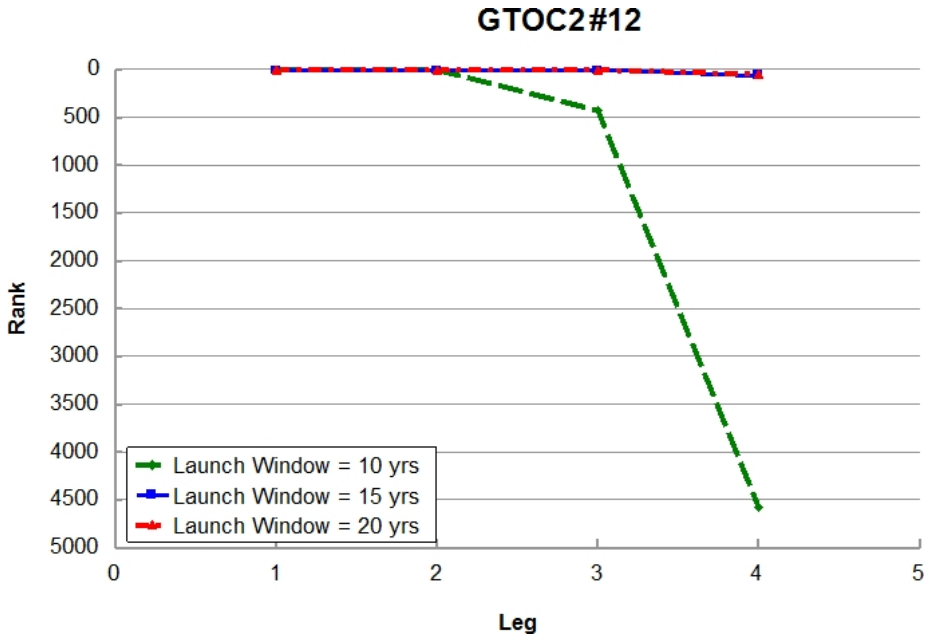

Figure D.6: Ranking of GTOC2 #12 in the reduced asteroid set, using GS.

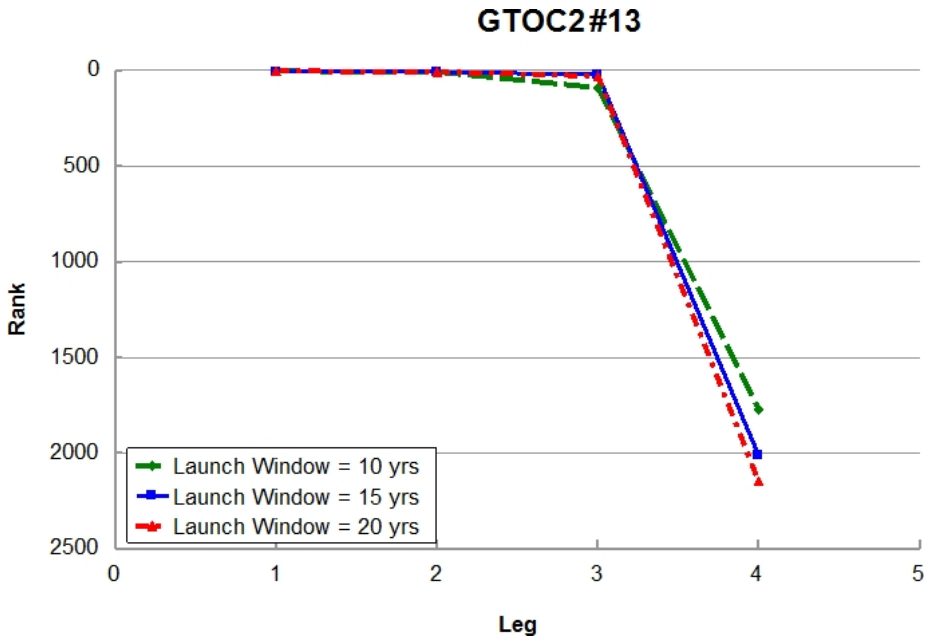

Figure D.7: Ranking of GTOC2 #13 in the reduced asteroid set, using GS.

Table D.1: ΔV of the optimized GTOC2 sequences, using DE

Sequence	Departure Windows [years]		
	10	15	20
	Total DV [km/s]		
GTOC2 #1	21.40	21.65	22.15
GTOC2 #2	20.33	20.48	20.58
GTOC2 #3	20.83	22.76	22.13
GTOC2 #4	22.21	22.21	22.65
GTOC2 #5	22.17	22.73	22.92
GTOC2 #6	20.45	19.98	20.25
GTOC2 #7	19.66	19.93	19.77
GTOC2 #9	21.43	21.29	22.65
GTOC2 #10	22.11	22.37	22.44
GTOC2 #11	32.61	32.59	31.98
GTOC2 #12	18.67	19.05	19.78
GTOC2 #13	21.22	22.12	22.51

Appendix E

Best Sequences in the Complete Asteroid Pool, using DE

This appendix contains the 10 best sequences, according to the departure windows and number of champions, investigated in Section 11.1.3 for a maximum time-of-flight of 1200 days. It also gives the non-nominal result obtained for 15-years departure windows and 15 champions. When available, the ranking of the GTOC2 sequences is given. The GTOC2 sequences can be identified via cross-referencing with Table 2.2. The complete results, according to departure windows, number of champions, and variable bounds investigated in Chapter 11, acquired in the complete asteroid pool using DE, can be obtained by contacting the author.

E.1 15-Years Departure Windows

Table E.1: Top 10 combination, according to J_2, found for 15-years departure windows by a forward multi-path NNH with N=15, using DE without caching. The maximum time-of-flight was set to 1200 days. GTOC2 #2 rank is also given.

Rank	Group 4	Group 3	Group 2	Group 1	Total ΔV [m/s]
1	3250293	2000149	2000240	2011542	17952.14
2	3250293	2000149	2000240	2002483	18000.94
3	3250293	2000851	2000449	2011542	18041.48
4	3250293	2000149	2000240	2002959	18184.58
5	3250293	2002411	2000442	2002959	18328.18
6	3250293	2000851	2000206	2001345	18441.91
7	3250293	2000851	2000449	2001345	18494.88
8	3250293	2000149	2000240	2003134	18550.55
9	3017309	2000228	2000135	2014569	18662.16
10	3250293	2001990	2000585	2001754	18691.88
...					
35	3250293	2000149	2000569	2002483	19208.03

Table E.2: Top 10 combination, according to J_2, found for 15-years departure windows by a forward multi-path NNH with N=20, using DE without caching. The maximum time-of-flight was set to 1200 days. GTOC2 #2 rank is also given.

Rank	Group 4	Group 3	Group 2	Group 1	Total ΔV [m/s]
1	3170221	2000951	2000418	2003134	17893.78
2	3250293	2000149	2000240	2011542	17952.14
3	3250293	2000149	2000240	2002483	18000.94
4	3250293	2000851	2000449	2011542	18041.48
5	3250293	2001415	2000111	2003134	18107.95
6	3250293	2000149	2000240	2002959	18184.58
7	3250293	2002411	2000442	2002959	18328.18
8	3170221	2000883	2000407	2003134	18342.16
9	3250293	2000851	2000206	2001345	18441.91
10	3250293	2001990	2000034	2001754	18457.35
...					
50	3250293	2000149	2000569	2002483	19208.03

E.1.1 Non-Nominal Result

Table E.3: Top 10 combination, according to J_2, found for 15-years departure windows by a forward multi-path NNH with N=15, using DE without caching. The maximum time-of-flight was set to 1200 days. Note that GTOC2 #2 ranks first.

Rank	Group 4	Group 3	Group 2	Group 1	Total ΔV [m/s]
1	3250293	2000149	2000569	2002483	17377.18
2	3250293	2000149	2000240	2002959	17935.44
3	3250293	2000851	2000449	2011542	18041.48
4	3250293	2000149	2000569	2003134	18060.67
5	3250293	2002411	2000442	2002959	18328.18
6	3250293	2000851	2000206	2001345	18441.91
7	3250293	2001990	2000034	2001754	18457.35
8	3250293	2000851	2000449	2001345	18494.88
9	3250293	2000149	2000569	2011542	18567.47
10	3250293	2000149	2000240	2011542	18586.93

E.2 20-Years Departure Windows

Table E.4: Top 10 combination, according to J_2, found for 20-years departure windows by a forward multi-path NNH with N=15, using DE without caching. The maximum time-of-flight was set to 1200 days. GTOC2 #2 rank is also given.

Rank	Group 4	Group 3	Group 2	Group 1	Total ΔV [m/s]
1	3250293	2000443	2000058	2002959	17724.46
2	3250293	2000149	2000240	2011542	17939.47
3	3250293	2000149	2000240	2002483	17988.27
4	3042555	2001133	2000021	2002959	17993.69
5	3250293	2000851	2000449	2011542	18041.48
6	3250293	2000443	2000442	2002959	18163.29
7	3250293	2000149	2000240	2002959	18171.91
8	3250293	2000443	2000058	2001345	18349.06
9	3017309	2000228	2000135	2001038	18458.85
10	3250293	2000851	2000449	2001345	18494.88
...					
45	3250293	2000149	2000569	2002483	19195.36

Table E.5: Top 10 combination, according to J_2, found for 20-years departure windows by a forward multi-path NNH with N=20, using DE without caching. The maximum time-of-flight was set to 1200 days. GTOC2 #2 rank is also given.

Rank	Group 4	Group 3	Group 2	Group 1	Total ΔV [m/s]
1	3250293	2000443	2000058	2002959	17724.46
2	3170221	2000951	2000418	2003134	17893.78
3	3250293	2000149	2000240	2011542	17939.47
4	3250293	2000149	2000240	2002483	17988.27
5	3042555	2001133	2000021	2002959	17996.75
6	3250293	2000851	2000449	2011542	18041.48
7	3170221	2000951	2000418	2002483	18045.87
8	3250293	2000443	2000442	2002959	18163.29
9	3250293	2000149	2000240	2002959	18171.91
10	3250293	2000749	2000442	2002959	18309.07
...					
58	3250293	2000149	2000569	2002483	19195.36