Acta Futura 11 (2018) 79-90

DOI: 10.5281/zenodo.1139258

Acta Futura

GTOC9: Results from University of Trento (team ELFMAN)

ENRICO BERTOLAZZI *, FRANCESCO BIRAL, MATTEO RAGNI

DEPARTMENT OF INDUSTRIAL ENGINEERING, UNIVERSITY OF TRENTO

Abstract. The GTOC9 competition requires the design of a sequence of missions to remove debris from the LEO orbit. A mission is a sequence of transfer of the spacecraft from one debris to another. Both missions and transfer must fulfill a set of constraints. The work presents the procedures to develop a solution for the GTOC9 problem (i.e the mission sequence) that does not violates constraints.

The solution is obtained through an evolutionary algorithm that combines pre-computed basic missions stored in a database. The main objective of the algorithm is to minimize the overall cost of the solution, in order to maximize the competition score.

The database of pre-computed missions is derived by connecting trasfers stored in a database of transfers, through a combinatorial approach that considers the problem constraints.

The database of transfer is formulated through the solution of a constrained minimization problem upon the control action (the magnitude of the overall impulsive velocity changes ΔV). Only a subset of all possible transfers (selected on the basis of *acceptable* ΔV), enters in the database.

1 Introduction

The 9^{th} Global Optimization Competition (GTOC9) requires the design of a sequence of missions—i.e. the *solution*—in order to cumulative clean up the Sun-synchronous Low-Earth-Orbit from 123 debris that may trigger the Kessler effect, while minimizing an overall cost.

A single mission is characterized by a sequence of rendezvous spacecraft trajectories between debris. The spacecraft is controlled with impulsive changes in velocity. For each debris, the spacecraft activates a de-orbit package that removes the debris from the LEO orbit.

Each mission starts from a debris—i.e. it is not necessary to design the launch from the Earth—and continues for an arbitrary number of transfers, limited only by the fuel consumption. Each mission has to comply with some rules:

- the spacecraft has to wait 5 days to activate the de-orbit package;
- the time between two rendezvous must not exceed 30 days;
- a maximum of 5 velocity impulses are allowed;
- the spacecraft may never reach an orbital periapsis lower than $6600\,\mathrm{km}$.

At least 30 days must be accounted between two subsequent missions. The performance index is:

$$J = \sum_{i=1}^{N} \left(c_i + \alpha (m_{0,i} - m_{\mathrm{dry}})^2 \right) \qquad (1)$$

*Corresponding author. E-mail: enrico.bertolazzi@unitn.it

where c_i is a submission cost that is proportional to submission time (favoring earlier submission), $m_{0,i}$ is the initial mass of the spacecraft at mission i, and m_{dry} is its dry mass (favoring lighter mission). α is a constant scaling factor.

The fuel consumption of a single change in velocity is calculated by the Tsiolkovsky equation:

$$\Delta m = \left(1 - \exp\Big(-\frac{\Delta V}{I_{\mathrm{sp}} g_0}\Big)\right) m_i \tag{2}$$

During the transfer between two successive debris, the spacecraft trajectory is approximated with a Keplerian motion perturbed by the effect of an oblate Earth—i.e. J_2 factor—and it is modeled by the ODE:

$$\begin{aligned} \dot{\mathbf{r}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= -\frac{\mu}{r^3}\mathbf{r} + \mathbf{J}(\mathbf{r}), \\ \mathbf{J}(\mathbf{r}) &= \frac{3}{2} J_2 \frac{r_{\mathrm{eq}}^2 \mu}{r^5} \begin{pmatrix} x - 5\,x\,(z/r)^2 \\ y - 5\,y\,(z/r)^2 \\ 3z - 5\,z\,(z/r)^2 \end{pmatrix} \end{aligned} \tag{3}$$

where $\mathbf{r} = \begin{pmatrix} x & y & z \end{pmatrix}^T$ is the position vector, $r = ||\mathbf{r}||$ and $\mathbf{v}$ is the velocity vector. For more details, constants definitions and constraints, refer to problem description [1].

Sec. 2 explores the complex procedures to obtain a solution that fulfill problem statement constraints. The Section is divided in three major parts. Sec. 2.1 explores the formulation of the minimization problem to formulate a transfer between debris, leveraging the dynamical system of the spacecraft. The final result of the section is the *database of transfers*. Sec. 2.2 focuses on the exploration of a database of transfers in order to build chains of transfer that constitute a single mission, through combinatorial based approach, that creates the *database of missions*. Sec. 2.3 concatenates the missions in order to develop a final solution that becomes the actual submission to the authorities.

2 Implemented solution

At the core of the problem there is the transfer of the spacecraft from one debris to another. The transfer implies a loss in mass, due to the control action of the spacecraft ΔV, thus the evaluation of the control action is fundamental for the solution of the problem. The key of the proposed solution lays in the separation between mass loss evaluation and visited debris sequence in a single mission. The mass loss is treated as an independent problem with respect to the sequence.

The mission sequences result from a searching problem in a very large graph. The graph is explored by means of queries to a database of trajectories and *mean mass losses* for transfer from one debris to another.

The solving algorithm is divided in three logical steps:

- the first part identifies all maneuvers between two orbiting debris at a defined departure epoch, with an *acceptable* loss due to ΔV: only a subset of transfer is considered since the whole set of trajectory would be too large to handle, as detailed in Section 2.1. The trajectories are saved in a database for the following solution step.

- the second step connects maneuvers saved in the previous database to obtain a subset of all missions: also in this case there is a pruning policy to reduce the total size of the database that will contain all *acceptable* missions. This step is described in Section 2.2.

- as a third step, the missions are combined together, in such a way the total cost is minimized, as described in Section 2.3

2.1 Database of Transfers (DBT)

A database is filled with the transfer maneuvers that approximate the minimum fuel consumption. Virtually, the database contains an infinite number of transfers, thus, some heuristic criteria are used to group maneuvers into equivalence classes and store only one rendezvous maneuver for each class.

Let's consider d_i as the current debris, while d_j, with $i \neq j$, is the arrival debris. The starting and arrival epochs of each transfer belong to a prescribed domain.

The maneuvers may differ in terms of total transfer time and starting epoch. Heuristically, the longer the traveling time, the lower the fuel required to perform the transfer. To generate trajectories with both short and long traveling time, an optimization problem is performed on several time windows—e.g. time intervals of $[0, 1]$ days, $[0, 5]$ days and $[0, 15]$ days are used.

The resolution for starting epoch is one day. For maneuvers with starting and ending time that differs less than 1 day, only the most proficient ones are stored in the database.

 DOI: 10.5281/zenodo.1139258

The evaluation of the transfer that minimizes the mass loss is a result of a complex optimization problem that is formulated in order to reduce the computational resources required. The large number of trajectories to be generated, for the whole set of debris combination and starting epoch, makes the approach costly so that a number of speed-up strategy are used.

For a limited amount of time, the debris orbit can be well approximated as a Keplerian orbit. The theoretical minimum ΔV to transfer from one orbit to another can be computed semi-analytically (see [2]) with a very fast procedure based on the computation of the roots of an 8^{th} order real polynomial. Only the maneuvers with a theoretical ΔV below a reasonable threshold are considered for further computations. The details of the approach and its implementations are described in Section 2.1. The minimum ΔV is attained considering only a departing and an arriving point on the Keplerian orbits, disregarding the actual positions of the debris. The spacecraft transfer is approximated with a Keplerian trajectory obtained by a Lambert maneuver.

It is easy to find the starting epoch when the debris d_i is in the right departing position for the Lambert maneuver. The real challenge is the synchronization with the arrival debris d_j. An integer optimization problem is solved in order to minimize the distance between the arrival debris d_j and the spacecraft, which is approaching the target orbit—cfr. Section 2.1. The result of the integer optimization is the number of complete revolution that must be performed by the rocket on the Keplerian transfer orbit, the initial, and final time necessary to reach the debris d_j.

This transfer still approaches only approximatively the target debris d_j, thus, a new Lambert maneuver using initial and final time with initial and final position is computed. This second maneuver, in general, applies ΔV that is not too far from the theoretical minimum ΔV.

The rocket dynamics is not Keplerian, consequently the Lambert transfer is only an approximation of the required trajectory, that is the solution of a *two-burns* optimal variational problem (2B-OVP). The 2B-OVP is described in detail in Section 2.1 where the Lambert transfer is adopted as initial guess.

An high quality solution is evaluated only when a submission to the authorities is required—cfr. Section 2.3.

The trajectories are inserted in a database, where each record contains:

- starting and final epoch
- starting and arrival debris
- intermediate burn event time (if present)
- ΔV initial, final, intermediate (if present)

Those information are enough to reconstruct the transfer trajectory.

Minimum ΔV Transfer

The work of Zhang, Zou and Mortari [2] is the cornerstone for the minimum ΔV estimation. In this work, the authors derive a semi-analytical procedure that computes the travel time t_w which minimize the $\Delta V(t_w)$ required to transfer from one Keplerian orbit to another with initial and final position fixed.

The minimum is computed searching the points where derivatives of $\Delta V(t_w)$ is zeroed. This $\Delta V(t_w)$ is the sum of two norms $||\delta \mathbf{v}_0(t_w)||$ and $||\delta \mathbf{v}_1(t_w)||$ that are not differentiable near zero so that the derivative is computed formally as

$$\begin{aligned} 0 = \frac{\mathrm{d}}{\mathrm{d}t_w}\left(\Delta V(t_w)\right) &= \frac{\mathrm{d}}{\mathrm{d}t_w}\sum_{i=0,1}||\delta \mathbf{v}_i(t_w)|| \\ &= \sum_{i=0,1}\frac{\mathrm{d}\left(||\delta \mathbf{v}_i(t_w)||^2\right)/\mathrm{d}t_w}{||\delta \mathbf{v}_i(t_w)||} \end{aligned} \tag{4}$$

so that the relation

$$\frac{\mathrm{d}\left(||\delta \mathbf{v}_0(t_w)||^2\right)/\mathrm{d}t_w}{||\delta \mathbf{v}_0(t_w)||} = -\frac{\mathrm{d}\left(||\delta \mathbf{v}_1(t_w)||^2\right)/\mathrm{d}t_w}{||\delta \mathbf{v}_1(t_w)||} \tag{5}$$

is squared on both sides and an 8^{th} degree polynomial in t_w is obtained. The positive real roots are the candidates for t_w and the minima are discriminated by a simple procedure (see [2]). The roots are calculated through the fast Jenkins–Traub algorithm [3].

This procedure is extremely fast and is the core of the computation of the optimal debris transfer, and works as follows:

- The two orbits are sampled with e.g. nearly equally spaced points and all the combinations of pairs of starting and arrival points are evaluated for searching minimum ΔV. The minima are refined applying a re-sampling with points near the best candidates. This procedure is repeated a couple of time. At the end of this procedure the initial and final point $\mathbf{r}_0$ and $\mathbf{r}_1$ with the transfer time t_w are set.

DOI: 10.5281/zenodo.1139258

- Let t_0 the time at which the initial debris reaches point $\mathbf{r}_0$—i.e. the initial point of Lambert trajectory—and t_1 the time at which the arrival debris intercepts point $\mathbf{r}_1$. Let T the period of the Lambert trajectory and T_1 the period of the target debris Keplerian orbit. A rendezvous satisfies the following equation

$$t_0 + \underbrace{t_w + n\,T}_{\text{travel time}} = \underbrace{t_1 + m\,T_1}_{\text{intercept time}} \tag{6}$$

  where n is the number of revolutions the Lambert trajectory should complete, such that the target debris reaches the arrival point, at the same time, after m revolutions. Since there is no perfect match in practice, the previous equation is transformed in:

$$\arg\min_{m,\,n} \left|(t_0 + t_w + n\,T) - (t_1 + m\,T_1)\right| \tag{7}$$

  where m and n belongs to a limited range, such that

$$0 \le \left\{ \begin{array}{c} t_w + n\,T \\ t_1 - t_0 + m\,T_1 \end{array} \right\} \le \Delta t_{\max} \tag{8}$$

  where $\Delta t_{\max}$ is the maximum travel time considered.

- Once n is evaluated, the arrival time $t_f = t_0 + t_w + n\,T$ is used to compute the true position of the arrival debris. With this data—i.e. initial time and position, final time and position, and number of revolutions—a new Lambert problem is solved and used as initial guess for the 2B-OVP of Section 2.1.

The Lambert solver is a C++ routine based upon a MATLAB script written by Dario Izzo which implements an efficient and fast algorithm [4, 5, 6]

Equinoctial Coordinates and Integration

The rocket model proposed in (3) is an approximation of a LEO orbit, which contains a perturbation term due to oblate Earth, which makes numerical integration using Cartesian coordinates impractical. In fact, an high order numerical integration scheme is required to keep the prescribed tolerance with a reasonable time step.

To avoid the usage of an highly accurate numerical scheme, the ODE (3) is reformulated in terms of equinoctial coordinates, where the oblate perturbation is modeled as a low thrust action. This permits to integrate through low order numerical methods [7] that maintains the required accuracy. Equinoctial coordinates and the disturbances vector, due to oblate Earth, are:

$$\begin{aligned} \mathbf{y} &= \begin{pmatrix} p & f & g & h & k & L \end{pmatrix}^T \\ \mathbf{\Gamma} &= \begin{pmatrix} \Gamma_r & \Gamma_t & \Gamma_n \end{pmatrix}^T \end{aligned}$$

and equations of motion for the spacecraft can be stated as:

$$\dot{\mathbf{y}} = \mathbf{A}(\mathbf{y})\mathbf{\Gamma}(\mathbf{y}) + \mathbf{b}(\mathbf{y}).$$

The equinoctial dynamic is well known, but for completeness is hereby reported:

$$\begin{aligned} \mathbf{A}(\mathbf{y}) &= \frac{1}{q}\sqrt{\frac{p}{\mu}} \begin{pmatrix} 0 & 2p & 0 \\ q\,s_L & a_{2,2} & -g\,a_{6,3} \\ -q\,c_L & a_{3,2} & f\,a_{6,3} \\ 0 & 0 & \frac{1}{2}s^2 c_L \\ 0 & 0 & \frac{1}{2}s^2 s_L \\ 0 & 0 & a_{6,3} \end{pmatrix} \\ \mathbf{b}(\mathbf{y}) &= q^2 p^{-3/2}\sqrt{\mu}\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}^T \end{aligned}$$

with:

$$\begin{aligned} c_L &= \cos(L) & a_{2,2} &= (q+1)c_L + f \\ s_L &= \sin(L) & a_{3,2} &= (q+1)s_L + g \\ C_f &= c_L + f & a_{6,3} &= h s_L - k c_L \\ S_g &= s_L + g & q &= 1 + f\,c_L + g\,s_L \\ \alpha^2 &= h^2 - k^2 & s^2 &= 1 + h^2 + k^2 \end{aligned}$$

The equinoctial coordinates $\mathbf{y}$ are related to the state $(\mathbf{r}, \mathbf{v})$ expressed in Cartesian coordinates according to the following identities:

$$\begin{aligned} \mathbf{r}(\mathbf{y}) &= \frac{p}{q s^2}\begin{pmatrix} c_L\left(1+\alpha^2\right) + 2hk\,s_L \\ s_L\left(1-\alpha^2\right) + 2hk\,c_L \\ 2\,a_{6,3} \end{pmatrix} \\ \mathbf{v}(\mathbf{y}) &= \frac{1}{s^2}\begin{pmatrix} 2hk\,C_f - \left(1+\alpha^2\right)S_g \\ \left(1-\alpha^2\right)C_f - 2hk\,S_g \\ 2(\mu/p)^{1/2}\left(h\,C_f + k\,S_g\right) \end{pmatrix} \end{aligned} \tag{9}$$

The vector $\mathbf{\Gamma}(\mathbf{y})$, in equinoctial reference frame, is the matrix vector product $\mathbf{Q}(\mathbf{y})^T\mathbf{J}(\mathbf{r})$, where $\mathbf{Q}(\mathbf{y})$ is the orthogonal matrix:

$$\mathbf{Q}(\mathbf{y}) = \begin{pmatrix} \frac{\mathbf{r}}{\|\mathbf{r}\|} & \frac{(\mathbf{r}\times\mathbf{v})\times\mathbf{r}}{\|\mathbf{r}\times\mathbf{v}\|\|\mathbf{r}\|} & \frac{\mathbf{r}\times\mathbf{v}}{\|\mathbf{r}\times\mathbf{v}\|} \end{pmatrix} \tag{10}$$

DOI: 10.5281/zenodo.1139258

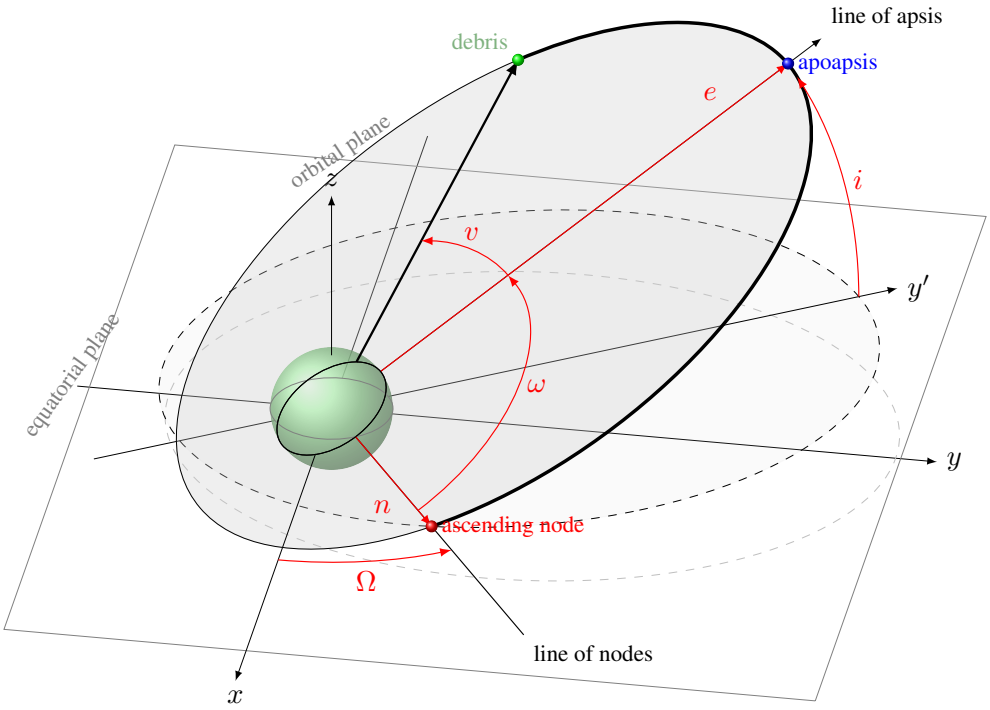

FIGURE 1. *Equinoctial coordinates for Sun-synchronous LEO orbits*

with $\mathbf{r}$ and $\mathbf{v}$ as defined in (9). The use of equinoctial coordinates permits to integrate a trajectory through a second order numerical method with a fixed time step of 6 s, maintaining the required accuracy for a integration domain of 30 day.

The forward integration is the classical second order Runge-Kutta (Heun scheme), while the discretization for the OVP is based upon Crank-Nicolson scheme [8, 9].

Optimal Two-Burn Variational Problem

Lets formulate the problem with cartesian coordinates for the sake of clarity.

Minimize:

$$\Delta V = ||\mathbf{v}(\mathbf{y}(t_0)) - \mathbf{v}_0|| + ||\mathbf{v}(\mathbf{y}(t_f)) - \mathbf{v}_1|| \quad (11)$$

subject to

$$\dot{\mathbf{y}} = \mathbf{A}(\mathbf{y})\mathbf{\Gamma} + \mathbf{b}(\mathbf{y}), \quad (12)$$

$$\mathbf{r}(\mathbf{y}(t_0)) = \mathbf{r}_0, \qquad \mathbf{r}(\mathbf{y}(t_f)) = \mathbf{r}_1, \quad (13)$$

$$||\mathbf{r}(\mathbf{y}(t))|| \geq r_{p_m}, \qquad t \in [t_0, t_f] \quad (14)$$

notice that in cartesian coordinates $\Delta V = ||\delta \mathbf{v}_0|| + ||\delta \mathbf{v}_1||$ and $\mathbf{r}_0$ and $\mathbf{r}_1$ are the initial and final point of the trajectory transfer defined in Section 2.1. Moreover, $\mathbf{r}(\mathbf{y})$ and $\mathbf{v}(\mathbf{y})$ are given in (9) and the value of r_{p_m} is given in [1].

The problem is solved by the custom made PINS solver used in other contexts [10, 11, 12, 13], an indirect problem solver for optimal control problem. PINS is able to solve OCP in the form:

Minimize:

$$\Phi(\mathbf{x}(a), \mathbf{x}(b)) + \int_a^b J(\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}, t)\mathrm{d}t \quad (15)$$

subject to

$$\mathbf{M}(\mathbf{x}(t), \mathbf{p}, t)\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}, t), \quad (16)$$

$$\mathbf{b}(\mathbf{x}(a), \mathbf{x}(b), \mathbf{p}) = \mathbf{0}, \quad (17)$$

where $\mathbf{u}$ is the control action, $\mathbf{p}$ is a parameter vector, and $\mathbf{M}(\mathbf{x}, \mathbf{p}, t)$ is a nonsingular mass matrix. Problem (11)–(14) fits PINS formulation with $\mathbf{M}(\mathbf{x}, \mathbf{p}, t) =$

$\mathbf{I}$ and empty $\mathbf{u}$ and $\mathbf{p}$. Bound (14) is well approximated in $J(\mathbf{x}, \mathbf{u}, \mathbf{p}, t) \equiv J(\mathbf{x}, t)$ using a barrier function.

PINS transforms the variational problem (15)–(17) in a two point boundary value problem that is solved as a non-linear system with a Newton-like iterative method. For the solution of the OVP problem (11)–(14) the initial guess required for the Newton-like method is based upon the Lambert trajectory, built in Section 2.1. The problem has a quite coarse time discretization (600 s), that brings to a very low precision solutions for the trajectory. With this mesh the computational time is very short, and the trajectory is roughly approximated, but still the estimate of the ΔV is good. The computational mean time for problem (11)–(14) on a MacBookPro with 2.9GHz Intel Core i7 is less than one second.

If a specific trajectory is selected as candidate for the final submission, is then re-calculated upon a mesh with a finer time discretization (6 s) that fulfils the tolerance requirements. This new problem uses as initial guess the solution found using the coarser mesh and the mean time on the same hardware is less than ten seconds.

Considering the solution of the integer problem, if the number of estimated revolution is too high—i.e. more than 100 revolutions—computing the solution of 2B-OVP is too costly. To reduce the computational effort, a first part of the trajectory, after the first burn—i.e. the one estimated through the Lambert problem—is integrated forward (see Section 2.1), and only the very few last revolutions of the trajectory—i.e. almost 10—are evaluated through the 2B-OVP, making the whole maneuver a three burns transfer (3B-OVP).

2.2 Database of Missions (DBM)

The second database contains sequences of maneuvres that form a mission. The exploration of database of transfers (DBT), to build the database of missions (DBM), considers the limitations proposed in the problem statement. The sequences are also limited implicitly by the available fuel mass, that is used to generate the pulses. Each time a mission is completed, the sequence is inserted into the mission database.

Even in this case, the number of mission that may be generated is huge, thus some pruning policy must be adopted.

The average cost

In the selection of the better candidate missions the average removal cost is taken into account. This cost is defined as the cost of the mission divided by the number of debris removed:

$$c_{\text{ave}} = \frac{c_b + \alpha \Delta m^2}{n_d} \tag{18}$$

where c_b is the time dependent base cost of the mission and Δm is the fuel consumed in the mission. For simplicity c_b is set to the maximum [1].

The average cost is a projection of the overall cost of the debris removal, and allows to forecast the performances of the solution proposed.

The mean cost has a trend similar to the one depicted in Fig. 2. When the number of debris removed is increasing, the mean cost tends to decrease. This trend is inverted after a certain number of debris removed, when the additional mass Δm required for the mission inverts the cost trend.

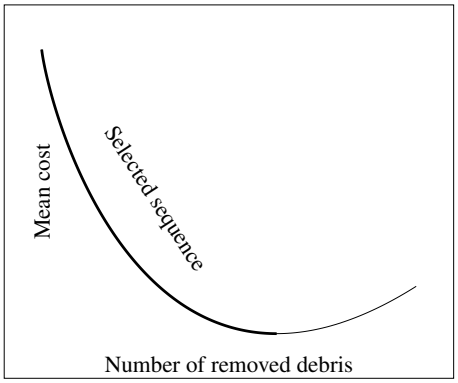

FIGURE 2. *Given a sequence of debris removal the average cost (18) decreases initially, reach a minimum, and then increases. The minimum is an indicator of the optimal number of debris to be removed and of the final cost of the solution The mean cost also permits to compare the solution with the performance of the other competitors.*

The mean cost depends on the length of the mission and it is re-evaluated every time a new transfer is added to the sequence.

Database creation policy

To generate the DBM, the total possible time for performing the debris removal is take into account and

 DOI: 10.5281/zenodo.1139258

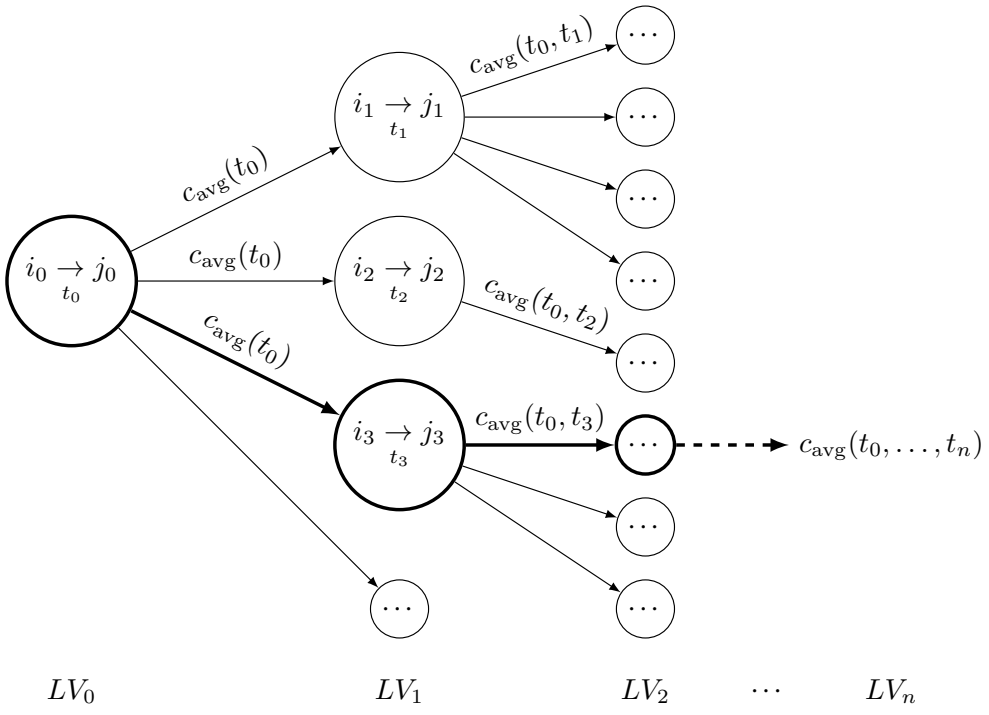

FIGURE 3. *A representation of the graph for the transfer combination*

sliced in time windows $[t_b, t_e]$ where:

$$\begin{aligned} t_b &\in \{0, 100, \ldots, 2900\} \\ t_e - t_b &\in \{100, 150, \ldots, 400\} \end{aligned} \tag{19}$$

Time windows may overlap. For each time window the N best missions, in term of average cost, are evaluated and stored in DBM.

The construction of missions to be inserted in database is performed in two steps:

- seeds initialization: a seed is the first transfer between two debris in a mission, that determines the starting debris;
- sequences exploration: starting from the seeds, continues the sequences of removal until the maximum fuel consumption limit is satisfied.

The seeds are initialized as follow:

1. from the manouver database (DBM), all the transfers with starting time in a limited initial time frame of the window are selected;
2. for each transfer from debris i to debris j of the previous selection, only the one with minimal average cost is kept;
3. the remaining selection is sorted with respect to the average cost, and only the first N are used as starting seed for the sequence search.

Starting at level 0—i.e. the seed—the exploration continues and at each new level the average cost of the mission is computed. Then, for each level pruning is applied:

- topologically equivalent missions—i.e. with the same set of debris removed and the same arrival debris—are pruned, and only the one with lowest average cost is kept;
- the remaining missions are furthemore pruned and only the N best mission are kept.

Then the exploration continues with the next level. At each level the best N missions are stored in the database.

DOI: 10.5281/zenodo.1139258

Each record of the DBM is stored as a sequence of pointers, where each pointer points at a manouveur that is stored in the DBT.

2.3 Missions combination

A series of mathematical objects are defined to formalize the GTOC problem as a minimization problem.

Definition 1 *The set $\mathcal{T}$ is the collection of all possible time intervals for the GTOC problem:*

$$\mathcal{T} := \{[t_0, t_1] \subset \mathbb{R} : 0 \le t_0 < t_1 \le t_{\max}\} \cup \{\emptyset\} \quad (20)$$

Definition 2 *Define with $\mathcal{M}$ the following set:*

$$\mathcal{M} := \left\{[\mathbf{w}, \tau, c] \in \{0,1\}^{123} \times \mathcal{T} \times \mathbb{R}\right\} \quad (21)$$

it represents a minimal encoding of all possible missions. The vector $\mathbf{w}$ marks all visited debris during the mission, the interval τ represents the initial and final time of the mission, which also considers the dead time for the deorbit package release, and c represents the overall cost of the mission.

The *null mission*, i.e. the mission that does not remove any debris is encoded with $m_\emptyset = [\{0\}^{123}, \emptyset, 0]$, while $\widetilde{\mathcal{M}}$ denotes a subset of $\mathcal{M}$ that contains all the possible queries of the *computed* DBM. The set $\overline{\mathcal{M}}$ denotes a subset of $\mathcal{M}$ of all the missions that fulfill the GTOC problem requirements.

Definition 3 *The operators W, T and C*

$$\begin{aligned} W &: \mathcal{M} \to \{0,1\}^{123} \\ T &: \mathcal{M} \to \mathcal{T} \\ C &: \mathcal{M} \to \mathbb{R} \end{aligned} \quad (22)$$

are defined for $m = [\boldsymbol{\omega}, \eta, \xi] \in \mathcal{M}$ and returns

$$W(m) = \boldsymbol{\omega}, \quad T(m) = \eta, \quad C(m) = \xi. \quad (23)$$

The set $\mathcal{M}^{123}$ contains all the possible sequence of missions for the GTOC problem. For example a sequence of only 5 removal missions can be fit in $\mathcal{M}^{123}$ by appending 118 *null mission* $m_\emptyset$.

Definition 4 *Let $s = [m_1, \ldots, m_{123}]$ a sequence of mission in $\mathcal{M}^{123}$ the operator*

$$Z : \mathcal{M}^{123} \to \mathbb{N} \quad (24)$$

is the number of debris not removed, i.e.

$$Z(s) = 123 - \mathbf{u} \cdot \mathbf{u}, \quad \mathbf{u} = \bigvee_{i=1}^{123} W(m_i)$$

*where the $\vee$ operator is the **element-wise or**.*

Definition 5 *The overall cost of a (possibly unfeasible) set of missions is:*

$$C : \mathcal{M}^{123} \to \mathbb{R} \quad (25)$$

which is the sum of the cost of each mission plus the cost of the n_r debris not removed:

$$C(s) = \sum_{i=1}^{123} C(m_i) + Z(s) c_b, \quad (26)$$

for $s = [m_1, \ldots, m_{123}]$ and c_b the maximum submission cost (cfr. equation (18)*).*

The set $\mathcal{M}^{123}$ permits to define the set of admissible solutions:

Definition 6 *The set $\mathcal{S}(\mathcal{N})$ is:*

$$\mathcal{S}(\mathcal{N}) := \left\{ \begin{array}{l} [m_1, \ldots, m_{123}] \in \mathcal{N}^{123} : \\ \text{for } i \neq j \\ W(m_i) \wedge W(m_j) = \{0\}^{123} \\ T(m_i) \cap T(m_j) = \emptyset \end{array} \right\} \quad (27)$$

*where the $\wedge$ operator is the **element-wise and**, and the set $\mathcal{N} \subset \mathcal{M}$.*

Remark 1 *Notice that the set $\mathcal{S}(\mathcal{N})$ can be extracted from $\mathcal{N}$ easily so that the set is never explicitly constructed.*

With the previously defined mathematical objects, it is easy to state the minimization problem at the core of GTOC.

$$\text{Find } s \in \mathcal{S}(\overline{\mathcal{M}}) \text{ which minimize } C(s). \quad (28)$$

Remark 2 *The problem* (28) *is a mixed integer minimization problem. Thus, gradient based minimization cannot be used. Moreover, the set $\mathcal{S}(\overline{\mathcal{M}})$ is uncountable. Thus, combinatorial based minimization cannot be used. In this form, the problem* (28) *is not numerically computable.*

 DOI: 10.5281/zenodo.1139258

The problem (28) becomes numerically tractable when the set $\mathcal{S}(\overline{\mathcal{M}})$ is reduced to the subset $\mathcal{S}(\widetilde{\mathcal{M}})$ which has finite cardinality:

$$\boxed{\text{Find } s \in \mathcal{S}(\widetilde{\mathcal{M}}) \text{ which minimize } C(s).} \tag{29}$$

Remark 3 *The problem* (29) *is a integer minimization problem defined on a finite cardinality discrete set* $\mathcal{S}(\widetilde{\mathcal{M}})$ *that rules out the use of algorithms based upon gradient or other analytical tools. However, a combinatorial approach is impractical due to the large cardinality, thus evolutionary approach must be preferred.*

The bigger the set $\widetilde{\mathcal{M}}$, the higher the chances that $\mathcal{S}(\widetilde{\mathcal{M}})$ contains a good minimum. Nevertheless, since evolutionary methods have extremely slow convergence rates, a bigger cardinality of such set reduce the chances to find a good approximation of this minimum in the time frame of the GTOC competition.

The pruning policy of section 2.2 is essential to reduce the density of set $\widetilde{\mathcal{M}}$ eliminating solutions in $\mathcal{S}(\widetilde{\mathcal{M}})$ that expose nearly the same structure, retaining only the best one.

2.4 Evolutionary minimization

The mimization of (29) is done through an evolutionary algorithm [14, 15, 16] that is a search technique based on the principles of evolution and natural selection. Schematically, an evolutionary algorithm requires a population of agents—i.e. a solution candidate—with a genomics that encodes a possible solution. An agent is an element of the set $\mathcal{P}$

$$\mathcal{P} := \left\{ \begin{array}{l} [m_1, \ldots, m_{123}] \in \widetilde{\mathcal{M}}^{123} \ : \\ T(m_i) \cap T(m_j) = \emptyset \ \text{ for } i \neq j \end{array} \right\} \tag{30}$$

The set $\mathcal{P}$ is larger of the set $\mathcal{S}(\widetilde{\mathcal{M}})$ and an element $s \in \mathcal{P}$ is also an element of $\mathcal{S}(\widetilde{\mathcal{M}})$ it is a solution of the GTOC problem. A population of agents:

$$\boldsymbol{P} = (s_1, \ldots, s_q), \qquad s_i \in \mathcal{P} \tag{31}$$

is evolved to become elements of $\mathcal{S}(\widetilde{\mathcal{M}})$. The *fitness function* of one agent is the the overall cost (26). It is also useful to introduce a function $G : \mathcal{P} \to \mathbb{N}$ that measures the *gap* of an element in $\mathcal{P}$ from the set $\mathcal{S}(\widetilde{\mathcal{M}})$. The *gap* is the number of multiple removed debris:

$$G(s) = \#\{k \ : \ v_k > 1\} \tag{32}$$

where v_k is the number of times the kth debris is removed and is computed as $\mathbf{v} = \sum_{m \in s} W(m)$. The evolutionary algorithm guides the genomic mutation of the population to minimize the fitness and to nullify the *gap* by cyclic on:

1. augment the population by adding random agents;
2. augment the population by cloning with mutation;
3. remove the worst agents with respect to gap;
4. select the next generation with respect to fitness.

In step 1 the population is increased by 10% by selecting randomly from $\mathcal{S}(\widetilde{\mathcal{M}})$. In step 2 the population is increased by 50% through cloning of randomly chosen agents. The cloned agents are then muted accordingly to the rules described in section 2.4. In step 3 the population has more than q agents and it is clustered and sorted in classes with increasing gap. The next population is obtained by the union of the first k classes such that the number of agents in the union is greater or equal to q and minimized—cfr. Figure 4. In this way the evolution rewards a population with smaller gap.

In step 4 the population is sorted by fitness and the first q agents are selected for the next generation.

Mutation rules

Given the agent s the mutation rules change an element m_k of s where k is chosen randomly. The mutated agent is denoted with s' and the mutated mission with m'_k. Let be g the minimum value of $G(s')$ for all the possible mutations $m'_k \in \widetilde{\mathcal{M}}$, then, the set $\mathcal{G}$ contains all $m'_k \in \widetilde{\mathcal{M}}$ such that the gap $G(s') = g$. If the set $\mathcal{G}$ is too small it is enlarged considering elements with such that the gap $G(s') = g + 1$ or more.

M1 Within the set $\mathcal{G}$ select the element that minimizes the cost $C(s')$. It is extremely unlikely to find more than one minima, but in case of duplicates the first found is chosen.

M2 Within the set $\mathcal{G}$ select m'_k randomly with uniform probability.

M3 Within the set $\mathcal{G}$ select m'_k randomly from a conditional probability with respect to the binary vector of the mission. In particular, an higher probability is associated with missions that cross debris that are rarely removed. The rarity of a debris is easily deduced from the database $\widetilde{\mathcal{M}}$.

DOI: 10.5281/zenodo.1139258

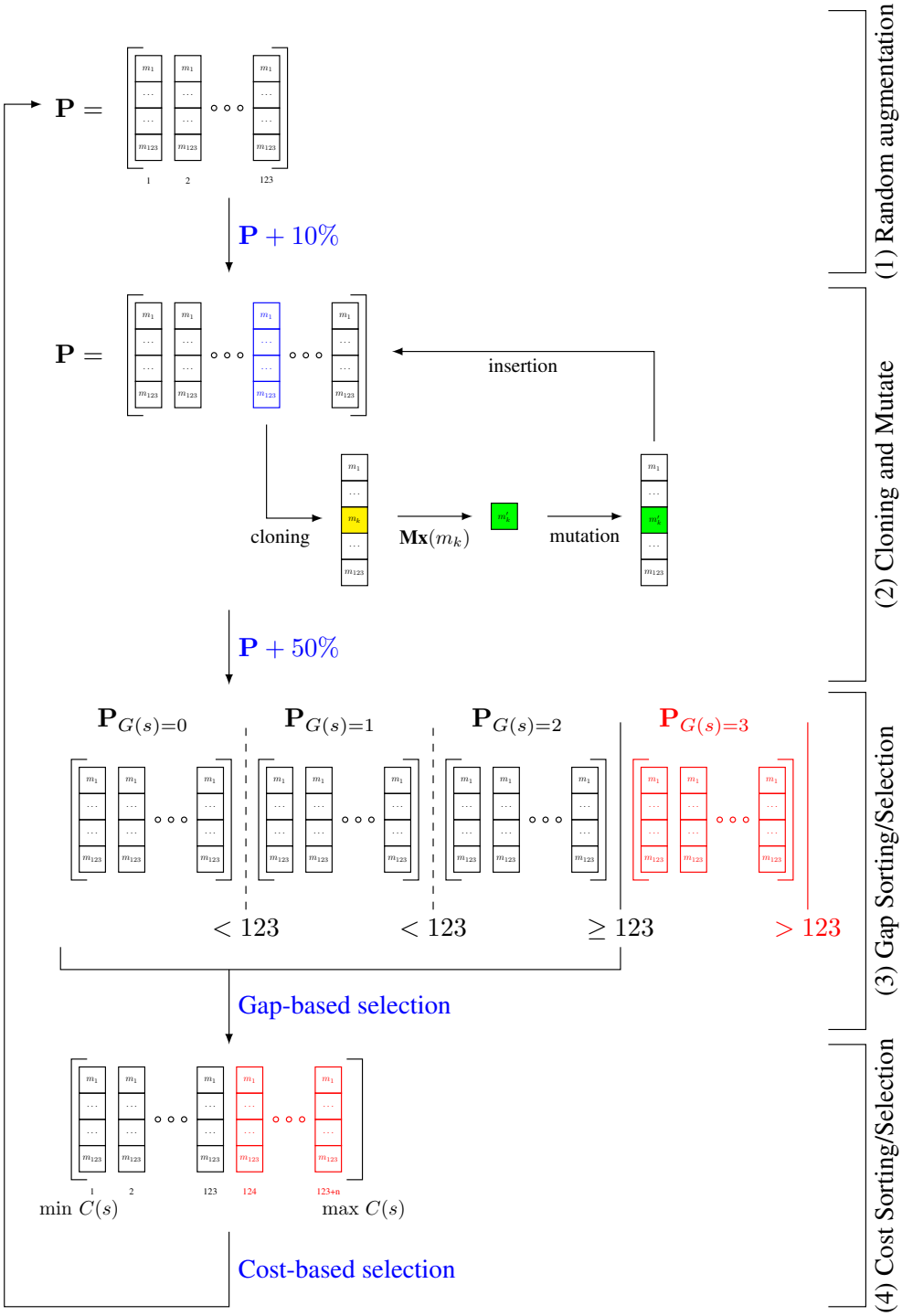

FIGURE 4. *Algorithm main loop*

DOI: 10.5281/zenodo.1139258

Only one of the mutations **M1**–**M2**–**M3** is applied, and it is randomly selected.

3 Conclusions

The work presented a strategy to solve the GTOC9 problem, that guarantees a complete compliance with all the constrained provided in the problem statement, and it is a cleaner formulation of the actual strategy used during the competition by the authors. In particular, evolutionary minimization was introduced when it became clear that a simple combinatorial approach was unpractical.

The first step was the construction of a database of transfers, from one debris to another, without considering the removal, upon the whole window allowed for missions in the problem statement. This database is based upon a ΔV minimization. The trajectories in the database where used to build a second database, the database of missions, through combinatorial algorithms. The final step was to chain the different missions in a single solution to be submitted for evaluation. This final solution was identified by a evolutionary approach.

Due to the limited time span for the competition, the very last component of the puzzle, the evolutionary minimization, was not completely developed. Thus an incomplete version of the algorithm was used to explore the solution of (28). Moreover, very few transfers between debris were added to the solution provided by the genetic minimization, using other forms of heuristic approach.

Whit this strategies the ELFMAN team ranked 13th removing 119 debris with a final score of 1107.69367526485.

References

[1] D. Izzo and M. Märtens. The Kessler Run: On the Design of the GTOC9 Challenge. *Acta Futura*, 11:11–24, 2018.

[2] Gang Zhang, Di Zhou, and Daniele Mortari. Optimal two-impulse rendezvous using constrained multiple-revolution Lambert solutions. *Celestial Mechanics and Dynamical Astronomy*, 110(4):305, 2011.

[3] M. A. Jenkins. Algorithm 493: Zeros of a Real Polynomial [C2]. *ACM Trans. Math. Softw.*, 1(2):178–189, June 1975.

[4] E.R. Lancaster, R.C. Blanchard, United States. National Aeronautics, Space Administration, and Goddard Space Flight Center. *A unified form of Lambert's theorem*. Technical note. National Aeronautics and Space Administration, 1969.

[5] R. H. Gooding. A procedure for the solution of Lambert's orbital boundary-value problem. *Celestial Mechanics and Dynamical Astronomy*, 48:145–165, June 1990.

[6] Dario Izzo. Revisiting Lambert's problem. *Celestial Mechanics and Dynamical Astronomy*, 121(1):1–15, 2015.

[7] Francesco Casella and Marco Lovera. High-accuracy simulation of orbital dynamics: An object-oriented approach. *Simulation Modelling Practice and Theory*, 16(8):1040–1054, 2008. EUROSIM 2007.

[8] J. C. Butcher. *Runge–Kutta Methods*. John Wiley & Sons, Ltd, 2008.

[9] J. Crank and P. Nicolson. A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. *Advances in Computational Mathematics*, 6(1):207–226, Dec 1996.

[10] F. Biral, E. Bertolazzi, and P. Bosetti. Notes on numerical methods for solving optimal control problems. *IEEJ Journal of Industry Applications*, 5(2):154–166, 2016.

[11] F. Biral, E. Bertolazzi, and M. Da Lio. *Modelling, Simulation and Control of Two-Wheeled Vehicles*, chapter The Optimal Manoeuvre, pages 119–154. Wiley Blackwell, 2014.

[12] P. Bosetti and E. Bertolazzi. Feed-rate and trajectory optimization for CNC machine tools. *Robotics and Computer-Integrated Manufacturing*, 30(6):667–677, 2014.

[13] P. Bosetti and M. Ragni. Milling part program preprocessing for jerk-limited, minimum-time tool paths based on optimal control theory. *IEEJ Journal of Industry Applications*, 5(2):53–60, 2016.

[14] Marco Dorigo and Thomas Stützle. *Ant Colony Optimization*. Bradford Company, Scituate, MA, USA, 2004.

[15] Randy L. Haupt and Sue Ellen Haupt. *Practical genetic algorithms*. Wiley-Interscience [John Wiley & Sons], Hoboken, NJ, second edition, 2004.

[16] John H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1992.

 DOI: 10.5281/zenodo.1139258