GTOC9: Methods and Results from Strathclyde University. On the generation and evolution of multiple debris removal missions

Carlos Ortega Absil[†], Lorenzo A. Ricciardi[†], Marilena Di Carlo[†], Cristian Greco[‡], Romain Serra[†], Mateusz Polnik[†], Aram Vroom[‡], Annalisa Riccardi[†,*], Edmondo Minisci[†], Massimiliano Vasile[†]

[†] Department of Mechanical and Aerospace Engineering, University of Strathclyde, 75 Montrose Street, G1 1XJ Glasgow (United Kingdom)

[‡] Faculty of Aerospace Engineering, Delft University of Technology, Kluyverweg 1, 2629 HS Delft (The Netherlands)

May 12, 2017

Abstract

The design and planning of space trajectories is a challenging problem in mission analysis. In the last years global optimisation techniques have proven to be a valuable tool for automating the design process that otherwise would mostly rely on engineers experience. The paper presents the optimisation approach and problem formulation proposed by the team Strathclyde++ to address the problem of the 9^{th} edition of the Global Trajectory Optimisation Competition. While the solution approach is introduced for the design of a set of multiple debris removal missions, the solution idea can be generalised to a wider set of trajectory design problems that have a similar structure.

*Corresponding author. E-mail: annalisa.riccardi@strath.ac.uk

1 Introduction

The Global Trajectory Optimisation Competition (GTOC) [4] is a yearly worldwide challenge that was initiated by the European Space Agency in 2005 with the aim of advancing the field of research on global optimisation techniques for space mission design. During the years the challenge has been the breeding ground for the testing and development of new computational intelligence techniques for the design of a variety of trajectory design problems. This year challenge, *The Kessler run* [5], has been to design a set of missions to deorbit 123 debris on Low Earth Orbit (LEO). The only manoeuvres allowed to control the spacecraft trajectory are instanta-

Team	N_L	N_d	score
JPL	10	123	731.2756
NUDT Team	12	123	786.2145
XSCC-ADL	12	123	821.3796
Tsinghua-LAD	12	123	829.5798
NPU	13	123	878.9982
Strathclyde++	14	123	918.9808

Table 1: Final rank GTOC9

neous changes of the spacecraft velocity and the problem objective function J is the sum of a linear term, the launch cost, that increases linearly with submission time, and a quadratic cost that is the sum of propellant mass and deorbiting kits. Moreover constraints on propellant mass, minimum pericentre, time between rendezvous and between launches have to be considered in the problem formulation.

The paper presents the optimisation techniques and the solution approach adopted by the team Strathclyde++ that ranked 6^{th} over the 69 teams that registered to the competition (see Table 1). The second Section is dedicated to present an overview on the overall problem solving methodology designed for the problem, Section 3 presents the different fidelity dynamical models used for the combinatorial search (Section 4) and final solution optimisation/local refinement (Section 6). Section 5 presents an evolutionary approach adopted to recombine and improve solutions and Section 7 and 8 presents results and conclusions.

2 Solution approach

The solution to the problem was found by using a three-step process that included both low fidelity and high fidelity models, as well as global and local optimisation solvers to converge to an optimal and feasible solution. As a first step, a Beam Search algorithm and a sequence patching method have been used to generate initial guesses for multi-launch debris removal campaigns, considering the debris sequences and the initial guesses for the departure time from each debris. The combinatorial algorithms used a low fidelity model to calculate the required ΔV for each transfer and estimate the final mission cost. A set of these solutions have been used as initial population for an evolutionary optimisation approach that, by optimising the times of transfers, was able to modify the order of the debris in the sequences themselves as well as to improve the distribution of initial mass among the launches. After the generation of these initial campaigns, a second step was used to process the output of the combinatorial search. In this step, for each debris to debris transfer, the time of application of each impulse as well as the magnitude and direction of each of them has been optimised by means of a global and local optimisation algorithms using low and high fidelity models. The bounds on departure time from each debris and ΔV components have been set from the values returned by the combinatorial search. By imposing mission's time, position and mass constraints, these new sequences formed the overall campaign.

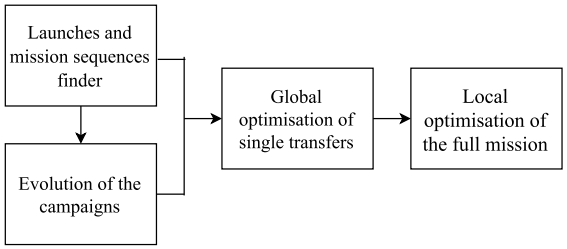

Figure 1: Flowchart of the solution approach.

As a third step, the entire launch sequence has been optimised locally with the high fidelity dynamical model to exploit the correla-

tion between subsequent transfers and reduce the propellant consumption further while ensuring that the constraint tolerances were met.

A flowchart of the solution approach is shown in Figure 1. In the following sections, the various components of this approach are described in detail.

3 Impulsive models

Low Fidelity estimation

In order to have a good but fast approximation of the cost of a transfer between pairs of debris, a low fidelity model, neglecting the J_2 perturbation, was used. For the sake of simplicity, the transfer was divided into two parts: an in-plane part, modifying only the shape of the orbit and an out-of-plane part, changing only the direction of the angular momentum. The phasing was not included as it comes with no extra cost under Keplerian dynamics assuming there is no time constraint on the rendezvous. Given the low eccentricity of all the debris, the departure and target orbits were approximated to be circular. Thus the cost of the in-plane part can be obtained from a classic Hohmann transfer. As for the change of plane, this was computed as a single manoeuvre modifying both the inclination and the right ascension of the ascending node at the same time [8]. Since its cost is proportional to the orbital velocity, this manoeuvre comes first if the departure orbit is at higher altitude than the target, second otherwise.

High Fidelity computation

A high fidelity estimation of the cost of the transfer between pairs of debris was obtained by solving a constrained global optimisation problem using the evolutionary algorithm MP-AIDEA [2]. In order to reduce the number of variables and, therefore, facilitate convergence to the global optimum, the maximum number of allowed manoeuvres was set to $n_{\Delta V} = 5$. The vector $\mathbf{y}$ of optimisation variables includes the time of applications of each impulsive manoeuvre and the three components of the $\mathbf{\Delta V}$ vector, for a total of $n = 4n_{\Delta V} = 20$ variables. The constrained optimisation problem has been formulated as:

$$\begin{aligned} &\min_{\mathbf{L}\leq\mathbf{y}\leq\mathbf{U}} F(\mathbf{y}) = \sum_{i=1}^{n_{\Delta V}} \Delta V_i(\mathbf{y}) \\ &\text{s.t.} \quad \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \\ &\qquad a_i(1-e_i) \geq 6600 \text{ km} \qquad i = 1, \ldots, n_{\Delta V} \\ &\qquad \mathbf{x}(t_f) = \mathbf{x}_D(t_f) \end{aligned} \tag{1}$$

where $\mathbf{y}$ is the vector encoding the 20 optimisation variables, $\mathbf{x}$ is the state vector of the spacecraft, $\mathbf{x}_D$ is the state vector of the targeted debris and t_f is the time at the end of the transfer. MP-AIDEA is run for a total of 1e6 function evaluations for each transfer. For the first 7e5 function evaluations a non-expensive dynamical model is used, in which it is assumed that the spacecraft's mean orbital elements a, e and i remain constant between two impulses, Ω and ω change according to their secular variations due to J_2 [8], while M changes according to $M = M_0 + \overline{n}\,(t - t_0)$ where $\overline{n}$ is the mean motion perturbed by J_2 [10]:

$$\overline{n} = n\left[1 + \tfrac{3}{2}J_2\left(\tfrac{R_\oplus}{p}\right)^2\sqrt{1-e^2}\left(1 - \tfrac{3}{2}\sin^2 i\right)\right]$$

$R_\oplus$ is the Earth's radius and $p = a(1-e^2)$. The best solutions obtained at the end of this stage are then used to initialise the population for the next phase of the optimisation process, where the complete high fidelity dynamics, including osculating J_2 effects, was considered. In

this phase dynamic equations were integrated with an 8-th order Adam-Bashforth-Moulton algorithm with a fixed step-size. At the end of the global optimisation, a local search was run from the best solution obtained: the Matlab solver *fmincon* with active-set algorithm was applied to problem 1.

4 Combinatorial search

Full campaign

At this stage, a solution is considered to be a list of couples $\{(D_j, t_j)\}$ defining the itinerary in terms of debris to visit and time of transfer, and with a predicted cost J. By considering a new launch as a particular case of transfer, a full first-guess launch campaign can be built incrementally in a tree-like fashion. The base tree exploration heuristic of choice was the Beam Search, this is a Breadth-First Tree Search including two stages of pruning: at the parent level up to a branching factor Br –branching phase–, and at the generation level up to a beaming factor Be– beaming phase. This method has been successfully applied in other GTOCs [3] and was selected primarily due to the fact that upper bounds on its time and space complexity are easily obtained.

The algorithm operated on a memory-loaded time-discretised precomputation of the ΔV cost of all debris-to-debris transfers as predicted by the Low-Fidelity Model in Section 3. With this modelling, analysis pointed towards the underestimation in time of flight as an important source of error due to the J_2 drift. For seamless interaction with the subsequent of the solution pipeline, a zero-order approximation of the time of flight was used as correction in the precomputation of ΔV, together with margins on the prediction and over-constrained transfer windows.

Upon this baseline, a family of problem-specific techniques was conceived. These can be classified in two conceptual variations:

– *Cyclic Beam Search*: takes one or more roots S_0 consisting in an initial debris-time couple (D_0, t_0) and a set of available debris and available mission time intervals. At iteration i, a move of depth 1 is appended to node S_i to generate next-level nodes S_{i+1}. This consists in expanding the launch campaign with either a transfer or a new launch to the couple (D_1, t_1), with D_1 non-visited and t_1 satisfying the respective transfer or launch constraints imposed by the sequence associated to S_i. Note the latter do not exclude a potential launch at $t_1 < t_0$ provided that t_1 belongs to the available mission time intervals. Additional restrictions are imposed on the launch heuristics to render the execution practical.

– *Concurrent Beam Search*: a meta-algorithm on the method above, consists in a scheduler that manages the branching phase of N Cyclic Beam Searches sharing the non-visited debris and available mission times in a competitive fashion, and allowing at each meta-iteration $N_t \leq N$ of them to advance of an iteration as defined above.

Note these methods can be applied to either the computation of single mission itineraries or complete campaigns, as well as to the expansion of sub-campaign itineraries. Over 95% of the solutions eventually refined and/or evolved were generated with the Cyclic Beam Search.

Additional heuristics. In the Kessler run problem, a competent solution needs to visit all the debris available. This fact poses an issue for incremental approaches such as the ones described hereby; different launch missions will be in competition for a fraction of the reachable targets, hence greedy ap-

proaches risk to exhaust the search space in early iterations, leading to unexpensive missions that cannot be aggregated into competent campaigns. This effect was mitigated using heuristics that enforce variety amongst the subcampaigns represented by the parent nodes of an iteration of the search, namely:

– *Pruning of twin transfers:* a limited number of transfers n_t to the same debris is appended to node S_i during its branching. A minimum time separation Δt_t is enforced between each of them.

– *Pruning of twin campaigns:* a maximum number of itineraries n_s visiting the same subset of debris is allowed during the beaming phase. This parameter is self-adapted if the generation is underpopulated.

– *Modified cost functions:* considering several definitions of a per-debris rarity bonus or rewarding homogeneous campaigns in terms of ΔV budget per launch. Campaigns thus obtained contained features that proved to be of special interest for the seeding of the approach presented in Section 5.

Initialisation. For the Cyclic Beam Search, the properties of the search space and algorithm allowed for a brute-force initialisation approach; light searches in terms of computational resources were initialized from each of the initial debris at a monthly discretization of the available mission timespan. As data was gathered, progressively heavier searches were conducted, giving priority to the most promising t_0 values in the database while letting the algorithm itself select amongst all initial debris.

For the concurrent beam search, even for moderate values of N, naïve initialization of all sub-searches from all debris results impractical. To overcome this limitation, a multi-variate time-series clustering was conducted in the features $x_j = \cos(\Omega_j(t))$, $y_j = \sin(\Omega_j(t))$, where $\Omega_j(t)$ is the RAAN of debris j at time t, and pre-pruning conducted in terms of size of the cluster. The clustering algorithm of choice was Partition Around Medioids, using segments of 75 days, Euclidean and Penrose distances and number of clusters selected by means of Silhouette Width in each segment. Yet other options tested for the Concurrent Beam Search are its initialisation by means of N non-intersecting itineraries corresponding to independent launches, or solution of a single-objective optimization problem for the designation of N launch sites in terms of RAAN and time of launch, maximising the total number of reachable debris.

Sequence patching

The goal of a sequence patching is to assemble a launch campaign out of feasible sequences built from cheap debris to debris transfers. Figure 2 suggestes that such transfers are common within certain time windows. All possible transfers with ΔV below a predefined threshold were precomputed and used to generate sequences. A sequence is described by time windows within its transfer occur and debris intended for removal. The order in which debris are visited is not relevant for the patching algorithm and can be established later using a cost optimizer.

Building a launch campaign is equivalent to finding a clique in an unidirectional graph $G(V, E)$ where V is the set of sequences and E the set of edges. Two sequences are connected by an edge if they target distinct subsets of debris and do not overlap in time. Finding a maximum clique is a well known NP-hard problem [1] with efficient solvers available open source [9]. With this ap-

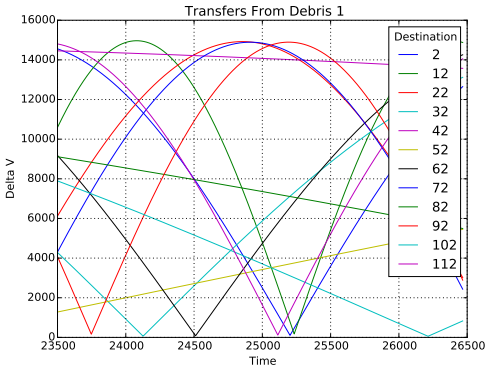

Figure 2: Velocity required to reach a destination debris at a point in time.

proach partial campaigns for data sets containing 8.5×10^5 sequences have been found. However this approach became impractical for larger datasets where a depth-first search was used instead. To accelerate the patching algorithm sequences were sorted according to an index function that took into account a relative cost of a debris' removal and its frequency among all sequences. Furthermore, the depth-first search was started from sequences that remove the rarest debris first to significantly reduce the number of sequences that later can be added to a campaign. The results obtained from sequence patching algorithm heavily depend on the quality of the initial data set. Final campaigns obtained from patching a data set containing 2.2×10^6 elements covered up to 116 debris without launches dedicated for one debris removal.

5 Evolution of solutions

As an attempt to further improve the solutions obtained by the combinatorial searches, the problem was reformulated as

$$\begin{aligned} &\min_{\mathbf{L_t} \leq \mathbf{t} \leq \mathbf{U_t}} \mathbf{J}^*(\mathbf{t}), \qquad \mathbf{t} = (t_1, ..., t_j, ..., t_{123}) \\ &\text{s.t.} \\ &5 \leq t_{s_j+1} - t_{s_j} \leq 30 \qquad \forall s_j \in M_i, \forall M_i \\ &t_{s_1, M_{i+1}} - t_{s_{\text{end}}, M_i} \geq 43 \quad \forall M_i \end{aligned} \tag{2}$$

where J^* is the bi-objective function that has as first objective the original objective function and as second objective the mass violation; M_i is the $i-$th mission and t_{s_j} is the time of transfer from debris s_j. With this encoding, to each debris was associated a time between the minimum and maximum epoch allowed for the mission ($\mathbf{L_t}$ and $\mathbf{U_t}$). This vector of times was then sorted in ascending order, and the resulting sort index vector was saved to allow reverting to the original order

$$\mathbf{t_{sort}} = (t_{s_1}, ..., t_{s_j}, ..., t_{s_{123}}), \; t_{s_j} \leq t_{s_j+1}$$

Once the times were sorted, missions M_i emerged from the differences between consecutive times: sequences of debris separated each by less than 30 days defined a mission, and the union of missions defined a full campaign.

$$M_i = \{t_{s_k} | t_{s_{k+1}} - t_{s_k} \leq 30 \text{ for } s_k \in \mathbf{t_{sort}}\}$$

At this point, the order of debris visited couldn't change, and the time values could be interpreted as the departure time for each debris, or as the launch time if a mission was composed by only one debris. Time differences were then imposed to be between 5 and 30 days between each subsequent debris of a mission, and of at least 43 days between the final time of a mission and the start time of

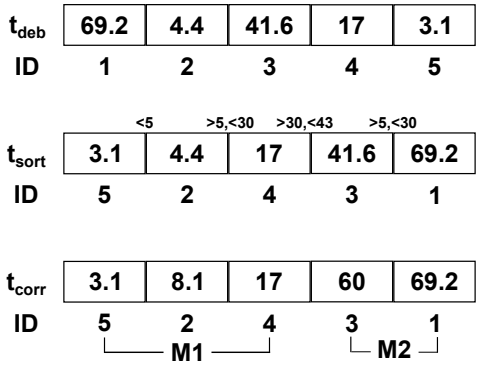

Figure 3: Scheme of the time processing in the evolutionary scheme

the next, which includes 3 days of safety margin as the transfers were considered instantaneous at this level but not with the full dynamics employed in the refinement stage. Once these time constraints were enforced, the resulting ΔV of each debris to debris transfer were computed with the low fidelity estimation, resulting in the propellant and launch mass of each mission and ultimately of the whole campaign. Problem 2 was tackled with the MACS algorithm [7] with a bi-level approach: on the upper level, the evolutionary heuristics of MACS generated possible solutions $\mathbf{t}$, which were sorted and made feasible by a lower level simply enforcing the constraints and returning to the outer level feasible solutions with the original ordering. MACS was seeded with the best 14 solutions coming from the combinatorial search, and was run for 1e7 function evaluations and standard parameters, with a gradient based refinement performed only on the lower level every 100 iterations.

6 Solution refinement

As last step, the solutions of the single transfers between debris computed by the high fidelity model presented in Section 3 are refined by a local optimiser handling an entire mission of multiple transfers in order to meet the constraint tolerances and further reduce the propellant consumption. Two steps are employed for this process. In the first one the mission is optimised using a single-shooting method. For each transfer, the optimisation variables are the same ones defined in Section 3, but the total number of variables is now $n = 4\ N\ n_{\Delta V}$ where N is the number of transfers in the mission. The problem is solved using Matlab *fmincon* with the active-set algorithm. In the second step, the solution obtained by the single-shooting is used as first-guess for a direct multiple-shooting algorithm, using WORHP as sparse nonlinear programming (NLP) solver [11], employed to reduce the numerical integration error and improve the convergence performance.

The constrained optimisation problem has been formulated as already outlined in Equation 1, but the number of variables to optimise for a single transfer is increased to $n = 4n_{\Delta V}+6(n_{\Delta V}-1)(m-1)+3(n_{\Delta V}-2)$, where m is the number of discretization intervals between two impulses. In order to enhance the computational efficiency, the sparsity patterns of the associated *Jacobian* and *Hessian* matrices, resulting from the multiple-shooting transcription scheme, have been derived and exploited in the NLP step. Furthermore, the full-J_2 dynamical model has been augmented with the associated variational dynamics, and the system of equations numerically propagated using a Runge-Kutta 4 integrator, to compute the gradient information. This approach resulted in a decreased computational load and a more accurate derivative computation with respect to the finite-difference approach [6].

7 Results

Table 2 details the number of launches and cost of several of the submissions ordered by submission date, together with the associated relevant improvements on the solution process. For fairness in the comparison, the cost is computed as if they had all been submitted at the time of the last submission.

Figure 4 details the mission timeline for the solutions in Table 2 as well as the initial mass of the spacecraft in each of the launches. It can be observed how an increase in the quality of the solution is associated to an increase in homogeneity in the mass of each of the independent launches. This relates to the fact that, since all the debris need to be visited for a solution to be competent, any two launches can be in competition for a subset of the targets. Hence greedy approaches in terms of ΔV of each transfer will be ill-suited for the resolution of this problem, as they will often lead to unexpensive missions that cannot be aggregated into competent campaigns. Following this line of reasoning, note also how the bottleneck derived from the utilisation of an incremental combinatorial approach is here manifest, as can be induced from the gaps in the timelines of solutions 1 to 4. These are caused by the search exhausting the available debris before exhausting the available mission time, thus failing to explore a region of the search space. Whereas solutions were generated using some of the heuristics detailed in Section 4 that mitigated this effect, this was always at the expense of the final objective function value. A similar phenomenon was encountered regarding the number of launches – solutions of as few as 13 launches yet suboptimal to solution 4 were generated before solution 5. This tendency ends with the introduction of the evolution of solutions in the pipeline, in solutions 5 to 7. Besides a reduced number of launches and the lack of the aforementioned gaps in the mission timeline, these campaigns also show an increased homogeneity in terms of initial mass of each of the launches. It is hence interesting to remark the outstanding synergy obtained between the first and second stages of the solution process described in Section 1. Figure 5 shows the difference in time of arrival at debris between the 14 individuals used by MACS as initial population, and the final solution submitted, solution 7. Note that these times will alone define the sequence of a campaign, hence these differences can be taken as an indicator of the similarity of the chromosomes of different campaigns. It can be observed that mainly two similar first-guess itineraries serve as baseline for the evolution of the final submission. The algorithm manages nevertheless to take features from other solutions in order to enhance the quality of the population; whereas an approximate exchange with another solution can be detected in some cases, the mechanics of the evolution is more concealed in others as for example debris 97. Figure 6 is a representation of the evolution in time of the RAAN of the final submitted campaign. It can be observed that the solution generally follows the natural J_2 drift as expected.

8 Conclusions

The paper presents the approach developed by team Strathclyde++ for the solution of the 9^{th} GTOC problem. The proposed methodology separates the combinatorial component of finding the optimal sequence of debris removals within the mission timeline, from the continuous problem of finding the best set of manoeuvres for each transfer. In particu-

Solution ID	N. Launches	$\hat{J}$	Relevant improvements in solution process
1	26	1713.07	Beam Search in the first 100 mission days.
			No thorough trajectory refinement.
2	18	1133.94	Cyclic Beam Search.
			Improved high-fidelity model.
			Added single and multiple-shooting refinement.
3	16	1059.54	Improved Cyclic Beam Search heuristics.
			Improved low-fidelity model.
			Improved global optimisation on high-fidelity model.
4	16	1028.72	Further relaxation of search overconstraints.
5	14	967.49	Added evolution algorithm to solution process,
			Small population of best submitted solutions.
6	14	945.15	Multi-objective formulation of evolution
7	14	918.98	Larger population including diverse features.

Table 2: Evolution of the solution process and quality of some of the submissions. Column $\hat{J}$ computed with $C_0 = 54.945$ as if submitted at the time of submission of solution 7 (best submitted).

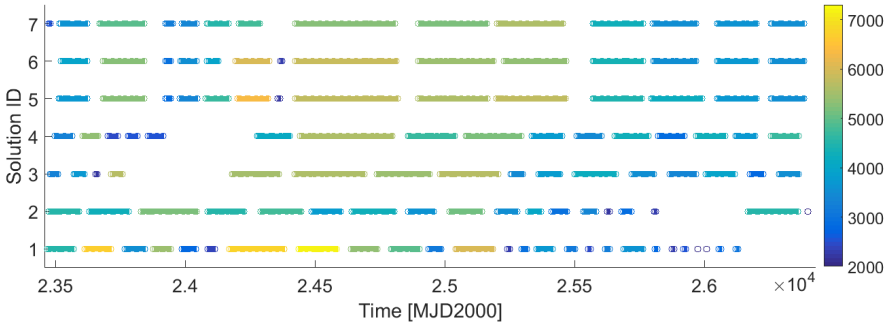

Figure 4: Launches and debris removal epochs of the solutions in Table 2. Colour relates to initial mass of each of the launches.

lar, it introduces a continuous formulation of the combinatorial problem that solved by a population-based global algorithm was able to evolve a set of first-guess solutions and generate new ones by means of recombination and hybridisation of the initial individuals, thus overcoming the manifest limitations of incremental combinatorial approaches. The fundamentals of the proposed methodology can be generalised to a family of multiple rendezvous problems, and are of special interest for the design of missions in which the set of available targets needs to be exhausted.

9 Acknowledgment

The team would like to thank Archie-WEST, the supercomputing centre of West of Scotland for the support and computing resources, and the colleagues at the University of Strathclyde, namely Gianluca Filippi, Francesco Torre and Victor Rodriguez for the always insightful discussions about the prob-

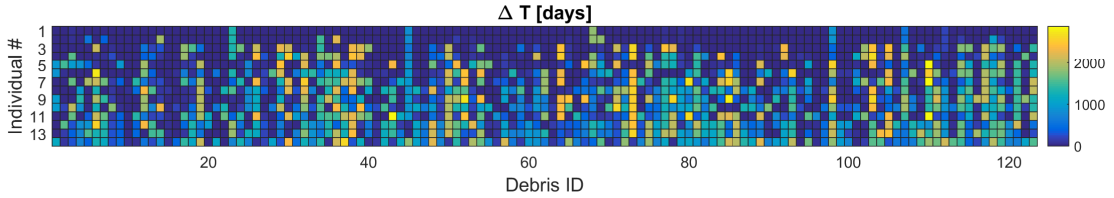

Figure 5: Difference in time at debris between individuals in the initial population and final solution obtained after evolution. Most similar individuals on top.

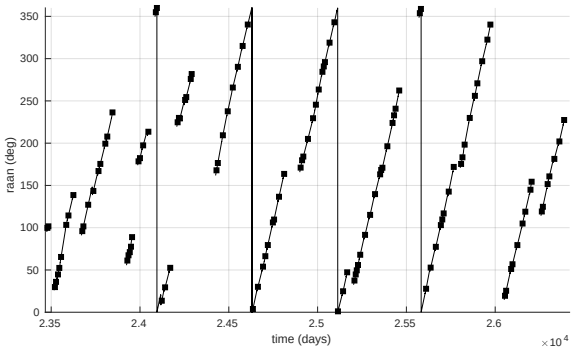

Figure 6: Evolution in time of the RAAN of the final submitted solution.

lem.

References

[1] Immanuel M Bomze, Marco Budinich, Panos M Pardalos, and Marcello Pelillo. The maximum clique problem. In *Handbook of combinatorial optimization*, pages 1–74. Springer, 1999.

[2] Marilena Di Carlo, Massimiliano Vasile, and Edmondo Minisci. Multi-population inflationary differential evolution algorithm with Adaptive Local Restart. In *Evolutionary Computation (CEC), 2015 IEEE Congress on*, pages 632–639. IEEE, 2015.

[3] D. Izzo, D. Hennes, L.F. Simões, and M. Märtens. Designing complex interplanetary trajectories for the global trajectory optimization competitions. pages 151–176, 2016.

[4] Dario Izzo. *GTOC portal*.

[5] Dario Izzo. *Problem description for the 9th Global Trajectory Optimisation Competition*, May 2017.

[6] Stephen Kemble. *Interplanetary mission analysis and design*. Springer, 1st edition, 2006.

[7] Lorenzo A. Ricciardi and Massimiliano Vasile. Improved archiving and search strategies for Multi Agent Collaborative Search. In *EUROGEN, 2015*.

[8] David A. Vallado. *Fundamentals of astrodynamics and applications*, volume 12. Springer Science & Business Media, 2001.

[9] Nikolaj van Omme, Laurent Perron, and Vincent Furnon. or-tools user's manual. Technical report, Google, 2014.

[10] Karel F. Wakker. *Fundamentals of Astrodynamics*. Institutional Repository, Delft University of Technology, 2015.

[11] Dennis Wassel, Florian Wolff, Jan Vogelsang, and Christof Buskens. The ESA NLP-Solver WORHP - Recent Developments and Applications. In Giorgio Fasano and Janos D. Pinter, editors, *Modeling and Optimization in Space Engineering*, chapter 4, pages 85–110. Springer, New York, 2013.