GTOC9: Methods and Results from the Jet Propulsion Laboratory Team

Anastassios Petropoulos*, Daniel Grebow, Drew Jones, Gregory Lantoine, Austin Nicholas, Javier Roa, Juan Senent, Jeffrey Stuart, Nitin Arora, Thomas Pavlak, Try Lam, Timothy McElrath, Ralph Roncoli, David Garza, Nicholas Bradley, Damon Landau, Zahi Tarzi, Frank Laipert, Eugene Bonfiglio, Mark Wallace, Jon Sims

Jet Propulsion Laboratory, California Institute of Technology,
4800 Oak Grove Dr., Pasadena CA 91109, USA

Abstract

The removal of 123 pieces of debris from Sun-synchronous orbit is accomplished by a 10-spacecraft campaign wherein the spacecraft, flying in succession over an 8-yr period, rendezvous with a series of the debris objects, delivering a de-orbit package at each one before moving on to the next object by means of impulsive manoeuvres. This was the GTOC9 problem, as posed the European Space Agency. The methods used by the Jet Propulsion Laboratory team are described, along with the winning solution that was found by the team. Methods include branch-and-bound searches that exploit the natural nodal drift to compute long chains of rendezvouses with debris objects, beam searches for synthesising campaigns, ant colony optimisation, and a genetic algorithm. Databases of transfers between all bodies on a fine time grid are made, containing an easy-to-compute yet accurate estimate of the transfer ΔV. Lastly, a final non-linear programming optmisation is performed to ensure the trajectories meet all the constraints and are locally optimal in initial mass.

1 Introduction

The 9^{th} Global Trajectory Optimisation Competition (GTOC9) considered the problem of debris removal from Sun-synchronous orbits around the Earth [1]. In this paper we describe the methods developed at the Jet Propulsion Laboratory to tackle the problem in the short, one-month competition timeframe. We also present the results obtained, both the submitted winning solution which removed all 123 debris objects in a campaign of ten missions, and results obtained shortly after the close of the competition.

An initial rough sizing of the problem was performed to understand the dynamics, estimate the ΔV sensitivities, and characterise the effect of number of missions on the cost function. The range of inclinations of the debris object orbits (about 96° – 101°) and semimajor axes (about 600 – 900 km larger than the Earth's radius) were sufficient to result in considerable variation in the drift rate of the ascending nodes (0.75°/day – 1.3°/day); the eccentricity, ranging from about 0.02 down to almost zero provided only a second order effect on the drift rate. The drift rate must of course be exploited in the transfers, because the ascending nodes are spread over the full circle and the cost of large plane changes is prohibitive (about 1.3 km/s per ten degrees). Drift-rate changes of about 0.1°/day can be achieved for about 100 m/s ΔV; for larger drift-rate changes, above about 0.2°/day, changing inclination is increasingly more effective (per unit ΔV) than raising apoapsis in affecting the node rate, and also permits an increase in the node rate, not just a decrease. The ΔV cost of matching the phasing and argument of periapsis of the debris was deemed of secondary importance.

Initial estimates of the number of missions that would be required were difficult to make, a fact reflected in the initial range of estimates that were made — from about 5 to 20. The base cost of a mission was comparable to the mass-dependent cost of a fully fueled spacecraft.

*Corresponding author. E-mail: Anastassios.E.Petropoulos@jpl.nasa.gov

Thus the trade-off between few missions with large ΔV requirements versus many missions with low ΔV requirements had to remain in play. This tradeoff was captured graphically a few days into the competition as shown in the Results section.

It was also realised early on that using the debris dynamics to propagate the spacecraft trajectory, rather than the full, unaveraged, J_2 dynamics, would be sufficiently accurate for good preliminary solutions. A variety of methods were then used to develop databases of body-to-body transfers, chains of bodies comprising a single mission, and complete campaigns of missions. The remaining sections of the paper describe these main facets of our solution methods, followed by a section on our results.

2 Propagation

A variety of spacecraft propagation methods were used, depending on the specifics of the broad-search method and the accuracy required. In the initial broad search, the debris dynamics, which correspond to the averaged J_2 dynamics, were used as an approximation of the spacecraft dynamics. In later searches, a correction to the mean motion, n, was incorporated [2]:

$$\dot{M} = n\left[1 + \frac{3}{2}J_2\left(\frac{r_{eq}}{a}\right)^2 \frac{1 - \frac{3}{2}\sin^2 i}{(1-e^2)^{\frac{3}{2}}}\right]$$

When phase is relevant, it is important both to use the correction term to the mean motion and to propagate from initial values for the elements that are obtained from computing the mean orbital elements rather than from the osculating element values at a particular epoch. Doing so results in relatively small errors of about 5 to 50 km in position after ten days. An asymptotic solution to the main problem was also implemented but not extensively used. For full accuracy, the spacecraft equations of motion were directly integrated using the Gragg-Bulirsch-Stoer extrapolation, either in their given Cartesian form or in equinoctial elements. Since some broad searches used the accurate integrations, some effort was spent to optimise the integration speed.

For final optimisation (see Final Optmisation), a simple, 8-th order Runge-Kutta integrator was used for propagating the dynamics (via Cartesian states). Other dynamical models such as equinoctial elements or the averaged equations were considered but deemed not as convenient, accurate, or robust.

3 Body-to-Body Transfers

A number of body-to-body transfer techniques and databases were developed to approximate chains so that they could be more easily computed, or so that bodies could be easily added to existing chains using simple database lookups. These databases grew in accuracy and size as different methods were developed throughout the competition.

Transfer techniques

One method for estimating the body-to-body ΔV was coded in a subroutine called AF2. The goal of AF2 was to find debris-to-debris candidate transfer opportunities below a user defined ΔV threshold, for a range of departure and transfer times. The algorithm estimated total transfer ΔV by selectively adding or taking the root-sum-square of individual ΔVs required for matching node, inclination, periapsis, apoapsis, argument of periapsis and approximate phase change. Actual propagations in full J_2 dynamics were done (for node matching) to capture the effect of varying transfer time on ΔV. After refinements, the ΔV estimates from AF2 were found to be within 5% of the actual optimised transfer cost for most of the cases.

Another technique was based on the debris dynamics and provided an analytic estimate of the ΔV needed to match the semimajor axis, node angle, and inclination of the target debris in a specified transfer time. The ΔV was split between an initial ΔV to change the drift rate and a final ΔV to match the semi-major axis, node and inclination. The ΔV was split optimally between the two manoeuvres such that a linear combination of ΔVs needed to change each of those three elements individually was optimised.

A final technique involved making a quadratic fit of the plane-change ΔV as a function of time of the manoeuvre, which means that this simplified solution space needs little information to describe it. The actual departure and arrival time can be optimised later with relatively small adjustments to fix phase.

Databases

Throughout the competition, the team created many databases for quick lookups for estimating the ΔV for transfers between bodies. One of the first databases provided estimates of the transfer ΔV between all body pairs at one day intervals assuming debris dynamics and

transfer durations of approximately one day. This estimate did not exploit changing the node rate, but intended to capture transfers between bodies whose orbit planes naturally drifted close to each other. The estimation included primarily a plane change at the relative node and one or two further impulses to change the eccentricity vector and semimajor axis.

Another database was also developed early in the competition that provided ΔV-optimal, full-phase transfers based on two-impulses using the full J_2-dynamics. At discrete times and for a set of discrete flight times, multi-revolution Lambert arcs were used as initial guesses to seed optimisation with Matlab's *fmincon*. (A similar database of optimal single-impulse transfers was also computed based on JPL optimisation software.)

The final database, named the GIGABASE, was perhaps the most reliable database created by the team during the competition. The GIGABASE was created roughly halfway through the competition and further developed in the final weeks. Transfers in the GIGABASE exploited the possibility of changing the node rate using ΔV and also matched phase, overall providing a good estimate of the optimal ΔV.

To quickly estimate the ΔV and flight time needed to transfer between debris objects, the GIGABASE assumed the spacecraft followed the dynamics of the debris. First derivatives (with respect to inclination and semimajor axis) of the nodal drift rate and the argument of latitude ($= \omega + \theta$) rate were used to approximate the required changes in inclination (δi) and semimajor axis (δa) for transfers with a given transfer time, a given integer number of revolutions, and with a given propulsive change in the right ascension of the ascending node, Ω. These simple equations require only solving a two-dimensional linear system of equations.

These initial δi and δa values are then differentially corrected to satisfy the full dynamics. The ΔVs to effect the differentially corrected initial δi and δa as well as the given change in Ω are computed assuming two-impulses. Subsequently, after the given transfer duration, another pair of impulses matches a, i and Ω of the debris (also θ has at this point drifted into alignment). Within the next orbit, the final two impulses match eccentricity and argument of periapsis of the debris.

Since the method is simple and fast, it is possible to loop over all possible number of integral revolutions and choose the lowest ΔV for each transfer. Furthermore, the simplicity of the computations allows looping over transfer times discretised in a fine grid, target bodies, departure times and departure bodies. Thus, a large database (the GIGABASE) was made with the following spacings: Debris, from every object to every object (123^2 = 15,129 options); departure epoch, 2-day spacing (1477 options); transfer time, 1 to 25 days, steps of 2 days (13 options). The resulting database contained about 290 million rows of data.

In order to facilitate the conversion of broad-search to detailed solution, a script called the decoderRing was created to reproduce more accurately the actual manoeuvre times and ΔV vectors approximated in the GIGABASE as a (good) initial guess into the final optimisation process. The inputs to this function were the bodies, departure epoch, transfer time, intermediate a, i and Ω values computed using the approximate method and stored in the GIGABASE. These were sufficient to allow analytic computation of all six manoeuvres, and a final one-dimensional corrections scheme added the missing secular term to the spacecraft J_2 dynamics. The decoderRing was only used when transitioning from campaign-level search to local optimisation and therefore did not need to be particularly fast.

4 Chain Building

Branch-and-bound

To construct low-ΔV chains (and sets of compatible chains), early in the competition a heuristic was developed that only considered plane change manoeuvres. Starting from an initial debris object at time t_0 a set of unvisited near-planar (to a tolerance) debris was identified. The minimum node difference was computed numerically, and a transfer of 1 – 2 days was initiated at the minimum. The following three cases were considered for when the minimum occurs: (i) within the feasible time interval, ΔV done to match inclination; (ii) before the feasible time interval (moving apart), immediate inclination change to alter the drift rate to match node in two days, and a second manoeuvre to match inclination; and (iii) after the feasible time interval (moving together), long stay at the initial debris object apply drift rate change manoeuvre and manoeuvre to match inclination two days later.

A branch-and-bound algorithm was used to build good chains of varying length with the heuristic ΔV. A grid of starting epochs and the set of unvisited debris initialized chain construction. A chain was deemed complete when there were no further targets, or when

it ran up against the maximum time constraint. Filtering (bounding) was applied to chain length (minimum), total fuel, and average ΔV. The branch-and-bound algorithm was parallelized to quickly yield a database of low-ΔV chains over all epochs, with chain lengths from 3 – 22. The longest were used as backbones to seed building of campaigns.

Ant Colony Optimisation

A previous investigation applied Ant Colony Optimisation (ACO) to the removal of debris objects [3]; modifications were made to this to accommodate J_2 drift and the constraint that missions cannot overlap in epoch. Briefly, the ACO algorithm seeds "agents" at starting debris objects, where these agents then build chains of encounters by interspersing random steps and the directed following of "pheromone" trails laid by previous generations. When an agent reaches the defined propulsive limits for an individual mission, it resets by "launching" to a new debris object and begins again, repeating this behavior until a complete mission set is generated. At the end of each generation, the mission sets are evaluated using the GTOC9 cost function, with the best performing agents chosen to lay pheromones along the routes they followed. In order to differentiate among the multiple possible transfers between any debris object pair, the GIGABASE solution giving

$$\min \Delta t^{\xi} \Delta V$$

was selected, where ξ is a tuning parameter and Δt is the time interval between arrivals at the respective debris objects (prior to the GIGABASE, an earlier database was utilized in a similar manner). Time-varying aspects of the problem were addressed by seeding objects and initial chain epochs based on debris clustering information that created groups of common orbital elements at differing epochs across the available mission window. By the end of the competition, two main variants of ACO were employed: i) a "subset" routine that searched among a limited set of remaining debris objects and available launch windows in order to complete an existing set of missions, and ii) a "full" search that began at the end of the mission window and worked forward to build mission sets eliminating all 123 debris objects. Throughout the competition, the "subset" approach reliably discovered the minimal set of additional launches to remove all remaining objects, with the complete sets then fed into the genetic algorithm and other refinement methods; by the end of the competition, the "full" ACO was reliably generating complete mission sets removing all 123 objects for roughly 950 MEUR using 10-13 launches.

ACM

Built upon the ΔV estimation capability of AF2, ACM's primary task was to rapidly build chains using an algorithm which preserves diversity while preventing exponential increase in the number of solutions. This was achieved by using a hash-function based chain identification algorithm which kept equal proportions of best-in-time and best-in-ΔV solutions during the chain building process. Randomized additions were also done with a 1% chance of being accepted to the next length level. After tuning, ACM was able to generate hundreds of thousands of chains of length greater than 10 and up to 23 in matter of few seconds.

5 Campaign Building

Campaign Beam Search

Starting with a backbone (or two), typically from the branch-and-bound method, partial campaigns were built using a Beam Search variant [4] with probabilistic mixing. Every generation (or depth) adds a new mission (chain), and the starting epochs for chain building are selected randomly from the currently valid time interval sets. After each highly-parallel generation evaluation, a subset of solutions are maintained for the next generation based on minimizing the heuristic campaign cost:

$$h = J + h_{\alpha} \frac{J}{D} (123 - D)$$

where J is the running GTOC cost function, h_{α} is a constant weight (scaling cost to go), and D is the number of de-orbited debris. Various knobs are available to ensure diversity and to prevent an overly greedy algorithm.

The chains and partial campaigns were advantageous in seeding combinatorial algorithms (e.g. ACO and GA), since they represent quality populations with an inherent reduction in the degrees-of-freedom (relative to the initial problem).

Chain Recombination

Given a database body-to-body transfers or a database of chains, a directed graph was created where the nodes were defined by the debris object ID and time, and the edges indicated a transfer to a new object. Edges contained information like transfer time and ΔV. Chains were created by traversing the graph following different criteria: minimum ΔV, maximum chain length, mission time interval, whether or not the mission should contain a particular debris object, etc.

Given a database of chains, an undirected graph was created where the nodes contained a mission (*i.e.*, chain) and the edge indicated that two missions were compatible. A standard algorithm was used to search for cliques (a subset of nodes that are totally connected, that is, a subset of chains that are compatible). Only the cliques with the maximum number of debris were reported.

Chains visiting the same debris object at similar times would be connected in the directed graph, allowing "mixing-up". In this way, the graph can be traversed following an existing chain and at the common node it can transfer to another existing chain, creating in the process a new chain. "Re-jiggering" was often used to find better alternatives to a given mission: A directed graph was created with only the given chain and then populated with compatible transfers from the GIGABASE. The graph becomes very dense, allowing several permutations of debris objects and transfer times. Once the directed graph is created, we can traverse it with the above criteria.

Manual Completion of Campaigns

Two dedicated tools were implemented to insert missing bodies automatically using a ΔV database (normally the GIGABASE or the single-impulse database), or AF2. The tool looks for long sitting times in a chain (> 7 d) and time gaps between missions (> 36 d) and computes the ΔV-optimal insertion point for each debris.

Anchor Bodies

Late in the competition the idea of anchor bodies was developed. These bodies are ones that will likely not appear in the same chain with each other due to having generally unfavourable relative geometry. For example, bodies 74, 102, 109 have not only very high ΔV to transfer between any pair of them, but also comparatively high ΔV to transfer to all of the other bodies. This is clearly visible in Fig. 1 which shows the minimum ΔV, over the entire 8.1-yr launch period, needed to transfer from a body to any other body in 25 days or less, computed by scanning the GIGABASE. Each column corresponds to a different departure body, IDs 0 through 122 left-to-right, while each row corresponds to an arrival body, IDs 0 through 122 bottom-to-top. The plot is nearly symmetric. The three bright stipes in each direction correspond to bodies 74, 102, 109, whose inclinations and node rates of 100.98°,101.07°, 96.24°, and 1.28°/day, 1.30°day, 0.75°/day, make them outliers by at least 0.4° and 0.06°/day (body 74), 0.01°/day (body 109).

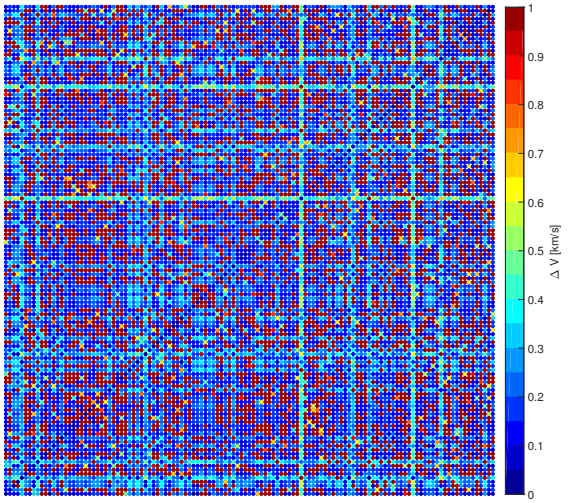

Figure 1: Minimum body-to-body transfer ΔV.

Converse to the idea of anchor bodies, but relevant in that chains would have to be built around them, are the following two observations, facilitated by the GIGABASE. First, the minimum ΔV over all possible transfers from a departure body over the whole launch period is less than 90 m/s for all departure bodies, and about 40 m/s when averaged over all departure bodies. Second, the "mode" of the distribution of the minimum body-to-body ΔVs shown in the matrix plot of Fig. 1 is about 125 – 225 m/s. This mode is seen in Fig. 2 which provides a histogram of the ΔVs, binned in 25 m/s bins.

In an attempt to build chains and complete campaigns around these anchor bodies at a suitable epoch, a tool called AMM was created based on the tools and algorithms developed in ACM and AF2. The main mo-

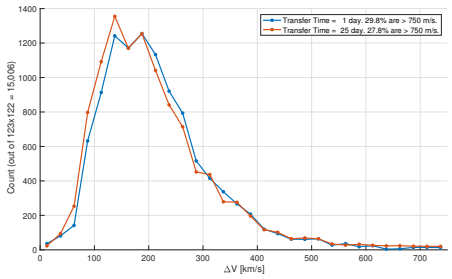

Figure 2: Histogram of minimum transfer ΔVs.

tivation behind AMM was to force the chain building process to maintain long, campaign-compatible chains with near equal lengths. The tool was developed late in the competition and therefore did not have sufficient time to mature to produce results that could be used. The utility of the anchor bodies would thus have been to reduce the dimension of the search space, allowing a fuller search to be conducted in the same time.

Rare bodies

A variation on the Anchor Body concept was also considered late in the competition but not fully explored due to time constraints. Using the quadratic-fit ΔV estimates, the minimum ΔV for given sequence of debris objects is readily found as a quadratic programming problem (almost always convex with inequality constraints on time between manoeuvres). An attempt at building a campaign (i.e., a complete set of missions) begins by tallying how many times each debris occurs in the entire pool of debris sequences. We pull out the subset of missions that include the "rarest" debris and permute that set with the set of missions containing the object with the second fewest occurrences. This way the most difficult objects are built into the mission sets early in the design process. After each permutation, the set is filtered by cost per unique object and number of "difficult" objects deorbited.

6 Campaign Re-Adjustment

Genetic Algorithm

Early in the contest, campaigns were assembled by piecing together locally-optimised missions spanning different bodies and different epochs. About halfway through, a need for campaign-level, global optimisation was identified as critical to improving our score. One approach, and the one which was ultimately used for the remainder of the contest, was to pose the problem as a single-traveling-salesman formulation of the campaign. A customised genetic algorithm (GA), named GIGA, was written in Matlab, based on the generic GA of Kirk [5]. The problem was represented as a list of nodes, visited sequentially and separated by a manoeuvre time. The key breakthrough in formulating the problem was the inclusion of both the debris objects and launch offsets (time from end of prior mission to start of next mission) as nodes, as opposed to keeping separate lists of debris and absolute epochs for each launch. This problem is similar to the Time-Dependent Travelling Salesman Problem (edge costs depend on order), but with an added dimension of an associated choice variable (manoeuvre time) at each edge.

The problem was encoded into a genome which took discrete values. Each genome was an ordered list of debris and launch offsets, each of which had an associated time code which mapped into a time-between-nodes that considered different values for debris and launch offsets (Fig. 3). In constructing the problem this way, only complete and time-feasible campaigns were modelled and produced. This eliminated the need to have any constraints enforced numerically (the 5000 kg propellant limit manifested strongly enough in the cost function that an explicit constraint was not required), other than the total campaign time constraint which was enforced with a barrier function of a 10% cost increase per day above the maximum time.

The fitness of each genome was a near-instantaneous, full evaluation of the campaign cost function using ΔVs from the GIGABASE. Being able to directly and rapidly optimise the campaign cost function was essential to this approach. The selection of which genomes advanced to the next generation was accomplished using a deterministic tournament of size 4 and tournament players were selected randomly. Other sizes were explored, but 4 was found to be a good balance between flexibility and selectiveness. The winner of the deterministic tournament advanced without modification to the next generation and was selected for 3 replications.

Because the node list had to be unique (non-repeating), a simple crossover was not implemented due to the possibility of generating infeasible genomes. Thus mutation is the only mechanism for improving a genome. Each mutation is constructed by first selecting two genes, say I and K, from each genome. A few factors were used for selecting the genes: Random; propellant mass fraction of that node (average of trans-

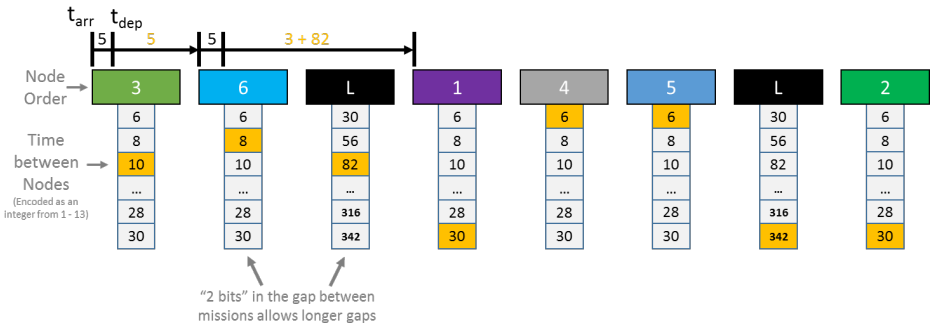

Figure 3: Encoding of the Campaign for the Genetic Algorithm

fers to and from); propellant mass fraction compared to full mission (launch fraction of the whole mission); Distance between I and K. By the end of the contest, there were 13 different mutations of roughly six types: (i) Swap, switches the list position and/or time code of node I and node K, (ii) Slide, puts node K next to node I and slides all the nodes and/or times between I and K over by 1, (iii) Flip, reverses the order of all nodes and/or times between I and K, (iv) Time Mutate, randomizes the time code of nodes I and K, (v) Net-Zero Time Mutate, randomize the time code of node I, then changes K by the inverse amount, and (vi) Insertion (Net-zero Time Slide), puts node K between node I and $I+1$ and adjusts the times of I, $I+1$, K, and $K-1$ so as to minimize the perturbations of the epochs of the rest of the campaign. The net-zero operators were invented because, although a certain slide or mutation would help locally, it would perturb the epochs of many other bodies, ultimately causing the overall cost to increase. Near the end of the contest, 15 heuristic combinations of weighting values for gene selection and mutation types were implemented, and one was selected randomly in each run. For instance, one combination focused on inserting high-mass-fraction nodes into low-mass-fraction missions while others focused on optimising the time only (without reordering nodes).

A completely random initialisation of GIGA was never successful in converging to a competitive-scoring campaign, though it did converge to a feasible campaign. Instead, initialisation was accomplished by encoding a previous solution, then mutating it 10–30 times to introduce sufficient randomness in the initial population without totally destroying the good seed solution. One problem with this implementation was its propensity to lose diversity. To partially correct this, if no improved solutions were found within a certain number of iterations, an additional 10 mutation steps were introduced without selection.

A single-threaded GA could run through tens of thousands of iterations with a population of a few hundred genomes in under an hour on a typical PC. In the last days of the contest, GIGA was parallelized and running on 12 nodes of a cluster (total of 144 threads). Each call of GIGA would search stored solutions for the globally best solution for initialisation and attempt to improve it using a random heuristic setting. By the end of the contest, approximately 6,400 full GIGA runs had been completed, based on the initial inputs of about 30 seeds, which means approximately 10^{11} campaigns were evaluated, or approximately 10^{13} body-to-body transfers.

Human-Guided Adjustments

If or when GIGA gets stuck in a local optimum, it can be advantageous to provide new, slightly varied initial solutions as a "kick" in hopes that subsequent runs may find new, lower-cost solutions. To construct new initial guesses, the cost-contour plot (Fig. 6) was used extensively to identify ΔV-infeasible missions and the worst-performing missions in terms of mission cost. Debris involved in high-ΔV transfers were typically removed from their respective chains and added to shorter, low-scoring missions, either by prepending, appending, or inserting. The GA could often improve considerably on the new initial guesses even if one or two of the new transfers had high ΔV.

These same techniques were also leveraged to reduce the number of missions in a given campaign. For example, the initial 10-mission and 9-mission solutions provided to the GA were created by starting with 11- and 10-mission campaigns, respectively, disbanding the shortest mission entirely, and distributing those bodies across the remaining missions.

7 Final Optimisation

The local optimiser used for individual missions was based on the OPTIFOR framework [6]. The complete trajectory is decomposed into different legs, which facilitates the modeling of rendezvous constraints and makes the process less sensitive to the control variables (multiple shooting formulation). A forward-backward strategy is implemented to reduce sensitivity with respect to the initial guess. Additionally, each leg is discretized into multiple segments, where a segment corresponds to an impulsive ΔV followed by a propagation of the full J_2-spacecraft dynamics. If no manoeuvre is needed at the beginning of a segment, the optimiser drives the corresponding ΔV to zero. The optimisation of the number of impulses as well as their respective locations is therefore automatically resolved. A total of 10 manoeuvres per revolution were considered to allow sufficient variety in the manoeuvre locations. Initial mass was minimized, while the final mass was constrained to be equal to the dry mass. Initial guesses were ballistic during the first part of the competition, then initial guesses from the decoderRing (see Databases) were used when available. The resulting discrete problem is solved using SNOPT [7]. Smooth convergence after 2,000 iterations was observed for most missions (the number of iterations can probably be decreased by changing some step size parameters and better scaling of variables and constraints). The HDDP solver [8] was also tested on long debris-to-debris transfers, but it was found to be generally slower to converge.

8 Putting it all together

As described in this paper, the team had a variety of methods for estimating body-body ΔV and creating databases of body-body transfers and chains. Thus the human-in-the-loop was an essential part of the workflow and eventual finding of the winning solution (see Fig. 4). The most important contributions to attaining competitive solutions was the use of the branch-and-bound search, the associated beam search and campaign-completion strategies (manual and ACO) to feed GIGA, which in turn fed promising solutions to the final optimisation step. After the creation of GIGA was completed in the final week of the competition, GIGA became the primary mechanism for global optimisation at the campaign level. GIGA was very good at making large numbers of significant changes to existing solutions, but had difficulty in finding truly new global optima due to the extremely strong local optima in this problem, and sometimes missed "obvious" single changes. So, human analysts focused on: creating qualitatively different input seeds for GIGA (including reducing the number of launches), and could even include launches which grossly violated the propellant mass limit (by 10,000+ kg); manually modifying GIGA outputs to make an obvious swap or insertion, or to inject some desirable features, such as trying to even out the number of debris in each launch or "smash" smaller chains together. Results coming out of GIGA were either directly optimised and submitted to Kelvins, or were further refined by human adjustment and then optmised and submitted, as showing in Fig. 4.

9 Results

Figure 5 provides a summary of all the complete missions sets submitted by JPL during the competition. In general cost decreases as the number of missions decreases, and flattens near 10 missions. Also depicted on this graph are 10-mission solutions found by JPL completed shortly after the competition ended, costing 720 MEUR and 711 MEUR. The best cost the team found for a 9-mission solution set was 750 MEUR.

Figure 6 shows cost contours for various ΔVs per transfer and rendezvous per mission. Overlayed on the contour plot is our initial 20-mission solution, the solution we submitted on April 27 when JPL occupied the top of the leaderboard for the first time (labeled 'Intermediate Set'), and our final submitted solution (numbered in white by mission number). As our solutions improved they moved from the top-left to the bottom-right side of the contour plot. For the final submitted solution, the smallest chain has nine rendezvous and the largest has 21.

A complete summary of the final submitted solution is provided in Tables 1 and 2. Every mission ends with $m_{dry} = 2000$ kg.

10 Conclusions

The debris rendezvous problem posed for this edition of the GTOC series was a challenging problem of interdependent and time-dependent combinatorics. The insights into the problem dynamics, which readily yielded a plethora of low-ΔV chains of transfers be-

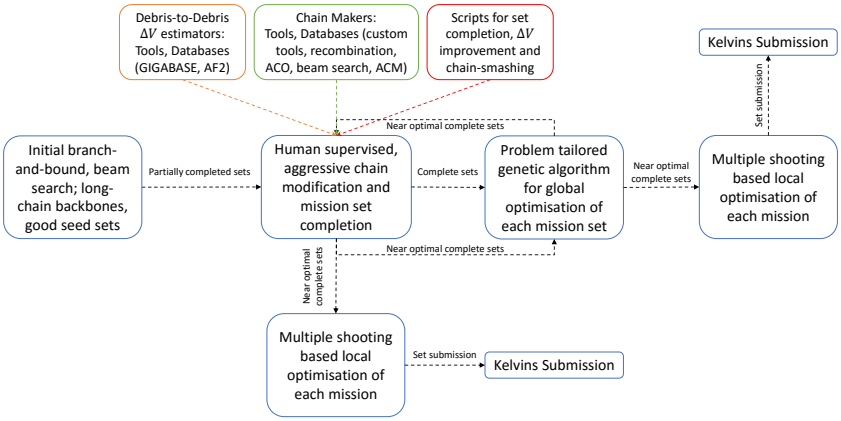

Figure 4: Workflow process.

Table 1: Campaign Overview, UTC submission times on 01 May 2017 indicated.

Mission	Start MJD2000	End MJD2000	Launch Mass, kg	Debris	UTC
1	23557.18	23821.03	5665.38	23,55,79,113,25,20,27,117,121,50,95,102,38,97	20:17
2	23851.08	24024.53	4666.15	19,115,41,26,45,82,47,85,7,2,11,77	20:17
3	24057.47	24561.49	6589.58	72,107,61,10,28,3,64,66,31,90,73,87,57,35,69,65,8,43,71,4,29	21:42
4	24637.26	24916.44	5679.10	108,24,104,119,22,75,63,112,37,32,114	20:18
5	24946.47	25232.94	4906.59	84,59,98,1,40,51,36,67,62,99,54,122,76,15	20:18
6	25262.95	25455.15	5062.74	101,48,53,5,12,39,58,13,60,74	20:18
7	25485.20	25682.33	4082.33	49,9,70,93,105,46,88,118,18,91	20:18
8	25712.38	25915.53	3725.73	86,34,100,30,92,6,110,96,81	20:19
9	25946.06	26237.29	4897.35	33,68,116,106,14,52,120,80,16,94,83,89	20:19
10	26267.80	26416.00	3438.62	44,111,56,78,0,17,109,103,42,21	20:19

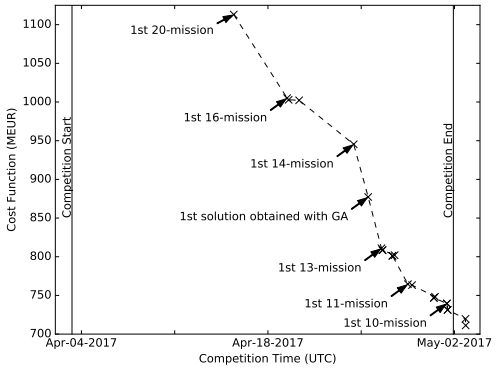

Figure 5: Evolution of score during competition.

tween debris objects, coupled with a tuned beam search to build near-complete, multi-spacecraft campaigns fed into grid-searched and ant-colony-optimisation-based design phases to complete the campaigns. As indicated by the drop in cost annotated in Fig. 5 by the introduction of the genetic algorithm, complete campaigns benefited greatly from the the genetic algorithm. However it must be stressed that the genetic algorithm required initially very good seed sets to further modify and improve, and even then, human-guided searches either feeding into or out from the GA where key to yielding the winning solution (recall "Human supervised, aggressive chain modification" at the center of flow chart in Fig. 4).

The JPL team thanks the Advanced Concepts Team of the European Space Agency, in particular the team lead Dario Izzo, for posing this fascinating and relevant problem, for making the logistics of problem dissemination and solution verification almost trivial, and for introducing the excitement of real-time solution ranking.

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

Table 2: Mission Characteristics.

Mission	*Rendezvous Time, days*
1	5.00,5.00,5.04,5.01,5.01,5.03,5.00,5.00,5.00,5.03,5.03,5.04,5.04,5.00
2	5.00,5.02,5.02,5.00,5.04,5.00,5.05,5.02,5.07,5.03,5.02,5.00
3	5.00,5.06,5.01,5.02,5.07,5.02,5.04,5.02,5.01,5.02,5.01,5.07,5.06,5.02,5.01,5.01,5.06,5.01,5.02,5.04,5.00
4	5.00,6.01,6.01,6.03,6.05,6.05,6.04,6.01,6.06,6.04,5.00
5	5.00,5.02,5.07,5.04,5.01,5.01,5.02,5.06,5.06,5.02,5.06,5.01,5.07,5.00
6	5.00,5.02,5.01,5.04,5.07,5.02,5.01,5.02,5.02,5.00
7	5.00,5.00,5.06,5.06,5.04,5.06,5.04,5.06,5.03,5.00
8	5.00,5.01,5.03,5.00,5.01,5.04,5.07,5.02,5.00
9	5.00,5.51,5.53,5.53,5.53,5.55,5.54,5.53,5.54,5.55,5.52,5.00
10	5.00,5.54,5.50,5.50,5.52,5.52,5.54,5.53,5.52,5.00

Mission	*Transfer Time, days*
1	24.86,24.98,22.42,24.99,0.29,10.63,25.00,2.70,1.51,1.41,24.67,24.31,5.86
2	24.93,0.28,0.73,0.39,17.07,1.61,22.42,2.39,15.88,24.97,2.49
3	14.16,24.94,2.87,8.10,9.00,23.13,23.09,23.09,22.83,24.98,24.98,24.93,24.94,9.10,13.44,24.99,24.94,24.99,24.98,24.96
4	23.96,6.48,16.72,23.97,23.95,23.95,23.96,23.99,23.94,23.96
5	0.45,3.17,24.93,10.34,12.53,7.11,13.44,24.94,24.94,24.98,22.19,24.99,22.01
6	24.91,0.30,18.39,3.08,20.24,24.96,24.85,24.97,0.28
7	15.69,0.50,9.83,24.94,24.90,24.48,20.87,24.91,0.66
8	10.03,24.00,2.83,24.99,24.99,24.96,21.19,24.98
9	22.69,4.24,24.47,24.46,24.47,24.44,24.46,24.46,24.46,18.54,9.22
10	0.81,11.59,7.66,1.11,17.46,6.47,20.47,24.47,3.99

Mission	ΔV, *m/s*
1	161.8,139.2,65.8,208.2,115.2,300.1,564.9,78.3,105.0,233.3,453.5,340.4,300.8
2	659.0,301.1,252.1,143.8,146.8,68.6,40.6,84.2,105.3,448.5,148.0
3	219.1,80.8,105.2,55.2,140.2,85.5,95.0,237.6,205.9,149.9,245.2,71.6,197.3,160.4,132.2,240.0,161.2,364.3,230.4,232.5
4	86.1,103.1,62.6,222.9,709.1,553.9,219.9,233.9,739.0,232.6
5	129.6,45.2,172.9,52.6,160.7,280.8,221.1,163.5,98.2,115.7,164.8,674.8,291.1
6	156.0,198.0,305.8,71.2,194.4,920.5,314.1,353.0,272.8
7	400.6,173.6,211.3,374.4,109.6,171.2,145.1,194.3,233.0
8	287.9,111.9,112.2,144.5,540.0,260.1,198.8,82.7
9	83.3,148.1,495.9,464.9,405.2,285.9,254.8,62.3,156.6,36.5,174.9
10	189.4,112.9,110.0,121.3,117.9,280.1,300.4,120.6,70.2

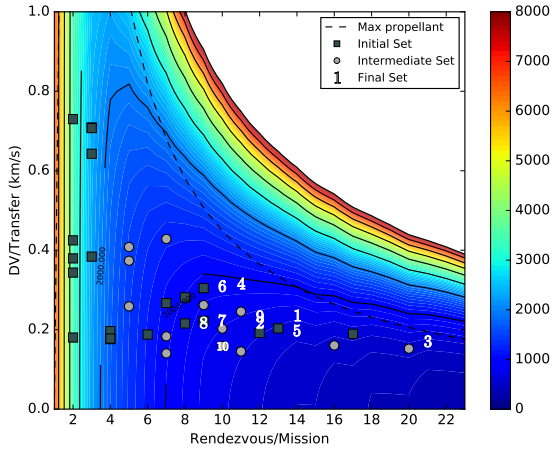

Figure 6: Cost-contour plot with final solution, numbered by mission; base cost: 55 MEUR.

References

[1] D. Izzo. *Problem description for the 9th Global Trajectory Optimisation Competition*, doi: 10.5281/zenodo.570193, May 2017.

[2] P. Gurfil. Analysis of j2-perturbed motion using mean non-osculating orbital elements. *Celest. Mech. Dyn. Astron.*, 90((3–4)):289–306, 2004.

[3] J. R. Stuart, K. C. Howell, and R. S Wilson. Application of multi-agent coordination methods to the design of space debris mitigation tours. *Advances in Space Research*, 57(8):1680–1697, 2016.

[4] L. Simoes, D. Izzo, E. Haasdijk, and A. Eiben. Multi-rendezvous spacecraft trajectory optimization with beam p-aco. *https://arxiv.org/pdf/1704.00702.pdf*, 2017.

[5] J. Kirk. *Travelling Salesman Problem GA, Matlab code*, https://www.mathworks.com/matlabcentral/fileexchange/13680-traveling-salesman-problem-genetic-algorithm, April 2017.

[6] G. Lantoine. *A Methodology for Robust Optimization of Low-Thrust Trajectories in Multi-Body Environments*. PhD thesis, Georgia Institute of Technology, 2010.

[7] P. E. Gill, W. Murray, and M. A. Saunders. Snopt: An sqp algorithm for large-scale constrained optimization. *SIAM Journal on Optimization*, 12:979–1006, 2002.

[8] G. Lantoine and R. P. Russell. A hybrid differential dynamic programming algorithm for constrained optimal control problems, part 1. *Theory, Journal of Optimization Theory and Applications*, 154(2):382–417, 2012.