A biobjective branch and bound procedure for planning spatial missions

Dalal Madakat, Jérôme Morio, D. Vanderpooten

A biobjective Branch and Bound procedure for planning spatial missions

Dalal Madakat[1,2]*, Jérôme Morio*[2]*, Daniel Vanderpooten*[1,*]*.*
dalal.madakat@onera.fr, jerome.morio@onera.fr, vdp@lamsade.dauphine.fr

[1] *Université Paris Dauphine, LAMSADE.*
[2] *Onera - The French Aerospace Lab, F-91761 PALAISEAU,*
[*] *This research has been partially supported by the project*
ANR-09-BLAN-0361
' GUaranted Efficiency for PAReto optimal solutions Determination'(GUEPARD).

November 22, 2017

Abstract

More than 90% of the space objects orbiting around the earth are space debris. Since the orbits of these debris often overlap the trajectories of spacecraft, they create a potential collision risk. The problem of removing the most dangerous space debris can be modeled as a biobjective time dependent traveling salesman problem (BiTDTSP).
In this paper, we study an approach based on a branch and bound procedure to determine the Pareto frontier of the BiTDTSP.

Keywords: Space debris; Biobjective time dependent traveling salesman problem; Pareto Frontier; Fathoming rules

1 Introduction

Since the launch of the first satellite in 1957, an increasing number of objects has been put into orbit around the earth. The number of space objects with diameter above 10 cm is estimated, nowadays, at about 15,000. Only 10% of these objects are operational, the other objects constitute space debris. Even if future launches are suspended, the number of space debris will continue to increase, e.g. owing to collisions between debris, making the collision with an operational satellite more probable [10]. Since consequences of collisions with debris may prove dramatic, avoidance maneuvers or missions to remove debris are necessary [5]. As removing all space debris would be quite expensive, the idea is to determine the debris which are more likely to cause collisions and remove them at least cost. Removal is performed by

achieving a space rendezvous between a moving space vehicle and each debris followed by a soft capture using a robotic arm. The shuttle has to meet each debris on its orbit until all debris have been dealt with and then return to its initial orbit. This tour has to be achieved in the less expensive and the fastest possible way. Most collisions are not debris/satellite but rather debris/debris and result in an increasing number of space debris [9, 1]. So the earlier the removal is finished, the less new debris are generated.

In this article, we study the problem of removing a list of space debris and extend the results obtained in [12]. We propose an exact algorithm based on a branch and bound procedure to compute the set of non dominated (cost, duration) vectors and give for each of these vectors a feasible solution. Two specificities that make our model more realistic are taken into account: between each pair of targets, several transfer possibilities, with different costs and times, are considered. These costs and times depend on the start time from the initial target in the pair. Therefore, the problem is modeled as a biobjective time dependent traveling salesman problem defined on a multigraph. To the best of our knowledge, BiTDTSP has not been studied before, although several articles study the time dependent traveling salesman problem (TDTSP) and the biobjective traveling salesman problem (BiTSP). TDTSP is a variant of TSP where distances depend on the arrival time to each vertex. Malandraki and Daskin [13] described dynamic programming (DP) algorithms for TDTSP extended by Bellman [3] and finally Held and Karp [7]. Schneider [18] also proposed a simulated annealing heuristic to deal with TDTSP. BiTSP has been addressed by several authors. Gendreau et al. [4] used the ϵ-constraint method to efficiently generate the Pareto front of the traveling salesman problem with profits. Schmitz and Niemann [17] were interested in a BiTSP problem motivated by various applications in the context of service delivery in which the second objective relates to priorities among locations to be visited. Paquette and Stutzle [14] analyzed algorithmic components of Stochastic local search (SLS) algorithms for the multiobjective traveling salesman problem. Based on the insights gained, they engineered SLS algorithms for this problem. Lust and Jaszkiewicz [11] proposed a heuristic resolution based on the two-phase local search method with speed-up techniques for BiTSP.

The paper is organized as follows. In Section 2, we model the problem of removing dangerous space debris. The proposed approach is presented in Section 3. Section 4 is devoted to implementation issues. Computational experiments and results are reported in Section 5 and conclusions are provided in a final section.

2 Problem description

2.1 Context and notations

Given n debris to be removed, the space shuttle has to move from its own orbit and visit the n debris in order to collect them and then return to its first orbit. The total quantity of fuel burned during each transfer from a debris i to a debris j, denoted by c_{ij}, represents the transfer cost. The duration of the transfer is denoted by d_{ij}. The quantity of fuel burned during the mission should not exceed the shuttle capacity, thus the cost cannot exceed a fixed cost c_{max}. Moreover, the duration of the mission should not exceed a fixed duration d_{max}. We assume here that the mission is not carried out by an unmanned space shuttle. To remove each debris, the shuttle has to perform a rendezvous with each debris on its orbit. Thus in the following we associate each debris with its orbit. Costs and durations depend on the way the rendezvous is achieved.

A space rendezvous between a debris and a space shuttle is an orbital maneuver where both arrive at the same orbit and approach to a very close distance. There are several ways to achieve a space rendezvous as shown in [6, 20, 19]. In our case, we have chosen to perform a rendezvous following the Lambert method where the shuttle moves between the two orbits undergoing exactly two pulses. Figure 1 shows how the transfer is performed in the Lambert elliptical case. The cost of this transfer is the quantity of fuel burned in order to perform the first and second pulses.

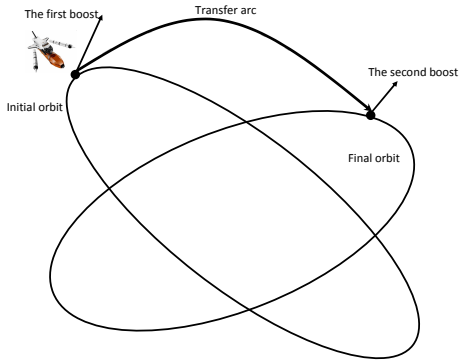


Figure 1: The transfer orbit of the space shuttle using the elliptical maneuver

For each debris i, t_i denotes the time at which the shuttle reaches the

orbit of debris i. Once the shuttle has reached orbit i, it can immediately start the next transfer to reach another orbit j or wait before beginning the transfer. Indeed, waiting on an orbit may be cheaper and/or quicker to reach the next orbit. The duration of the transfer is the sum of the waiting time in the departure orbit and the travel time to the arrival orbit. In the following we assume that each elementary transfer requires a minimum duration δt_{trans} and a minimum cost δc_{min}. We assume as well that the service time on an orbit i takes a duration of δt_{serv_i}.

2.2 Formulation of the problem

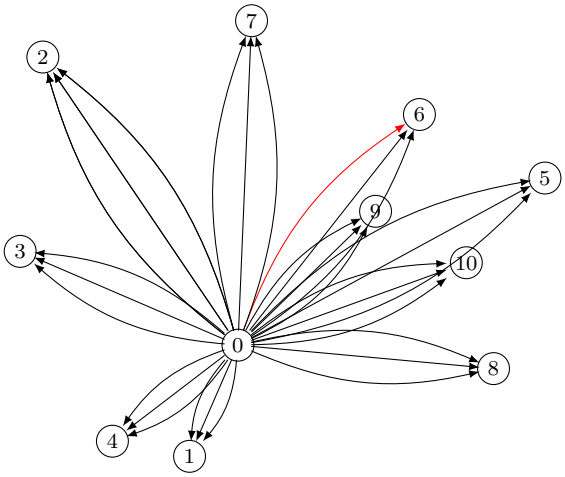
The studied space objects and the possible transfers between their orbits are modeled by a complete valued digraph $G = (V, A)$ where V represents the set of object's orbits numbered from 0 to n. Vertex 0 represents the shuttle and vertices from 1 to n represent the n debris. The set A corresponds to the set of feasible transfers between orbits. Several arcs may link each pair of orbits depending on the moment on when the shuttle reaches the departure orbit. To each 3-tuple (i, j, t_i), corresponds a set of feasible transfers $A_{ij}(t_i)$. Each element of $A_{ij}(t_i)$ induces a bivalued arc linking i to j whose value is a pair (c_{ij}, d_{ij}) corresponding to the cost and duration of the travel. As the arcs representing possible transfers depend on the time at which the shuttle reaches the departure orbit, G is a dynamic multigraph. In the figure 2, we have shown the way the digraph is constituted as the shuttle performs its mission.

After it reaches an orbit i, the shuttle must serve it before going on. Thus, the shuttle can leave the orbit i at least at $t_{i_d} = t_i + \delta t_{serv_i}$. Due to mission duration constraints, the shuttle should leave the orbit i before an instant limit t_{i_l}. When the shuttle leaves i at t_{i_l}, it has barely time to achieve the mission before $t_0 + d_{max}$. Thus, if the shuttle reaches l after visiting $V' \subset V$ debris, one has

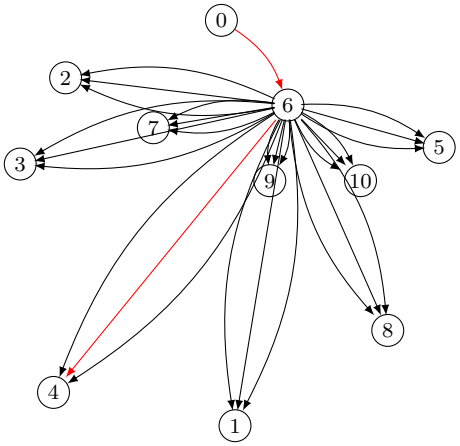
$$t_{i_l} = t_0 + d_{max} - (n + 1 - |V'|) \times \delta t_{trans} - \sum_{i \in V \setminus V'} \delta t_{serv_i} - \delta t_{trans}$$

In the equation above, $(n + 1 - |V'|) \times \delta t_{trans}$ is the least time needed to travel to each unvisited debris, $\sum_{i \in V \setminus V'} \delta t_{serv_i}$ is the time needed to serve debris and δt_{trans} is required to go back to the first orbit. The duration of time corresponding to possible departure times is discretized using a time step δt. Hence, the transfer can start at any time $t_{i_d} + w \times \delta t$, where $w \in I(t_i)$ and $I(t_i)$ determines the set of possible departure times if the shuttle reaches orbit i at t_i that is $I(t_i) = \{0, 1, ..., \lfloor \frac{t_{i_l} - t_{i_d}}{\delta t} \rfloor\})$.

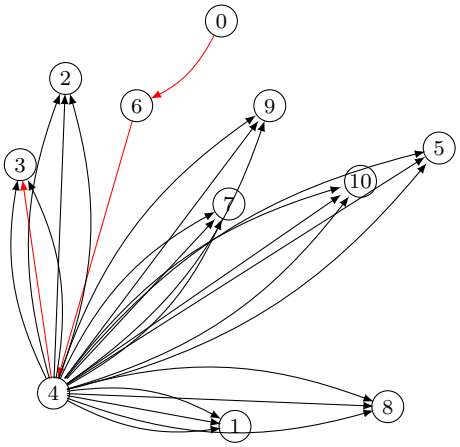
For each departure time several transfer durations are possible, the set of possible durations is denoted as D. A departure time t_i, $w \in I(t_i)$ and transfer duration $p \in D$ define a new arc allowing the shuttle to reach j at $t_j = t_{i_d} + d_{ij}(t_i, w, p)$. The valuation of this arc is:



(a) Situation in the initial orbit



(b) Situation after the first choice



(c) Situation after the second choice

Figure 2: Progression of the graph of the studied objects

$(c_{ij}(t_i, w, p), d_{ij}(t_i, w, p))$.
The transfer possibilities can be seen in Figure 3.

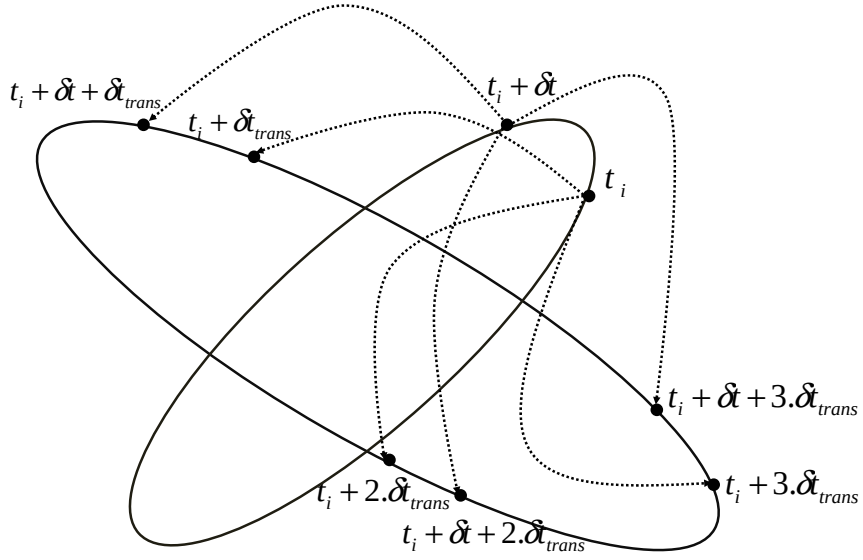


Figure 3: Transfer possibilities when the shuttle reaches orbit i at t_i

3 Solution approach

3.1 Preprocessing

The cardinality of $A_{ij}(t_i)$ depends on the number of possible departure times from i and on the number of possible durations to perform the transfer from i to j. Therefore $A_{ij}(t_i)$ may contain a very large number of possible transfers. The computation of the cost corresponding to each transfer possibility using Lambert algorithm is very time-demanding (see Section 5.2.2). The algorithm spends much more time computing transfer costs than optimizing the shuttle trajectory. Therefore, optimizing the algorithm performances requires limiting the number of calls to the cost computation function. This is achieved by reducing the cardinality of $A_{ij}(t_i)$ as will be seen in Section 5.2.1.

3.2 The branch and bound procedure

In order to enumerate the non dominated vectors for BiTDTSP, we propose a branch and bound enumeration scheme. In our case, each node at level k in the search tree corresponds to a partial solution visiting k debris starting from the shuttle orbit. Its successors are partial solutions visiting $k+1$ debris. A partial solution s is characterized by:

- V_s the set of debris visited in that partial solution.
- l the last visited element.
- $T(s)$ the time spent to visit the set of debris V_s.
- $C(s)$ the visit cost incurred to visit the set of debris V_s.

3.2.1 Branching

The branching procedure consists in choosing the next node that will be separated and in generating its successors. This requires generating $A_{lj}(t_l)$ for each debris $j \notin V_s$. A strategy of selection dictates which of the active nodes in the search tree will be separated next. Different search strategies are adopted in our case. Once a node is chosen, we generate its successors by computing $A_{lj}(t_l)$ for each $j \notin V_s$.

3.2.2 Bounding

A bounding procedure allows deleting some partial solutions that may not lead to efficient complete solutions. We present some fathoming rules. Fathoming rules can lead to substantial improvements in algorithmic performance [2]. Given a partial solution s visiting k debris, an *extension* of s is any feasible solution in which the k first visited debris are identical to s.
In the following, the set of possible extensions of a partial solution s is denoted by $Ext(s)$. In our context we define the two following fathoming rules.

- First fathoming rule D_1

  A pool of good candidate solutions is initially generated (see Section 3.3). If the performance of the most optimistic extension of a partial solution is dominated by one of the solutions pool, this partial solution can be discarded. Consider a partial solution $s = (V_s, l, T(s), C(s))$, the performance of any extension of s is at least
  $T^*(s) = T(s) + (|V| - |V_s| + 1) \times \delta t_{trans} + \sum_{i \in V \setminus V_s} \delta t_{serv_i} + \delta t_{serv_l}$ and
  $C^*(s) = C(s) + (|V| - |V_s| + 1) \times \delta c_{min}$.

Thus, if x is a solution from the pool corresponding to a criterion vector $(C(x), T(x))$ such that:

$$T(x) \leq T^*(s)$$
$$C(x) \leq C^*(s)$$

then each extension of s is in the best case as good as x and can thus be discarded.

- Second fathoming rule D_2

  Consider two partial solutions $s = (V_s, l, T(s), C(s))$ and $s' = (V_{s'}, l', T(s'), C(s'))$ such that

$$V_s \supseteq V_{s'}, \quad l = l', \quad T(s) \leq T(s') \quad \textit{and} \quad C(s) \leq C(s')$$

The last rule is valid thanks to two assumptions, whose validity will be discussed in Section 5.2.1.

Assumption 1 *We assume that, for any orbits i and j, if t_i and t'_i are two possible arrival times to i such that the time difference between these dates is greater than the period of orbit i then each transfer possibility corresponding to the arrival time t'_i is dominated by at least one transfer possibility corresponding to the arrival time t_i.*

More precisely:

$\forall w' \in I(t'_i),\ \forall p' \in D$
$t'_i \geq t_i + P_i \ \Rightarrow\ \exists w \in I(t_i)\ \exists p \in D$ such that

$$\begin{cases} c_{ij}(t_i, w, p) \leq c_{ij}(t'_i, w', p') \\ d_{ij}(t_i, w, p) \leq d_{ij}(t'_i, w', p') \end{cases}$$

where P_i refers to the period of orbit i. According to this assumption, waiting more than one period on any orbit i before traveling to orbit j does not lead to efficient transfers.

Assumption 2 *The second assumption concerns the triangle inequality of travel costs and durations. Given three debris i, j and l, all transfers that allow reaching l from i through j when the shuttle reaches i at t'_i are dominated by at least one transfer that corresponds to an earlier possible arrival time t_i to orbit i and that reaches l directly from i.*

This assumption can be formalized as follows:

$\forall w'_1 \in I(t'_i)\ \forall (p'_1, p'_2) \in D^2\ \forall w'_2 \in I(t'_i + \delta t_{serv_i} + d_{ij}(t'_i, w'_1, p'_1))$
$t_i \leq t'_i \ \Rightarrow\ \exists w \in I(t_i) \exists p \in T$

$$\begin{cases} t_i + \delta t_{serv_i} + d_{ik}(t_i, w, p) \leq t'_i + \delta t_{serv_i} + d_{ij}(t'_i, w'_1, p'_1) + \delta t_{serv_j} + d_{jk}(t'_j, w'_2, p'_2) \\ c_{ik}(t_i, w, p) \leq c_{ij}(t'_i, w'_1, p'_1) + c_{jk}(t'_j, w'_2, p'_2) \end{cases}$$

Proposition 1 *Assumptions 1 and 2 ensure the validity of the fathoming rule D_2.*

Proof 1 *Let $s = (V_s, l, T(s), C(s))$ and $s' = (V_{s'}, l', T(s'), C(s'))$ be two partial solutions such that:*
$V_s \supseteq V_{s'}$, $l = l'$, $T(s) \leq T(s')$ et $C(s) \leq C(s')$. Assume that the travel is performed by a shuttle $N1$ in the first solution while in the second solution it is achieved by another shuttle $N2$.

Case where: *$V_s = V_{s'}$*

Let e' be an extension of s'.

If, in the solution e', $N2$ leaves l at t such that $T(s) + \delta t_{serv_l} \leq t \leq T(s) + \delta t_{serv_l} + P_l$ hence $N1$ can wait on the orbit l during $t - (T(s) + \delta t_{serv_l})$ which is possible since this duration is smaller than the period of orbit l.

Knowing that $N1$ and $N2$ have achieved a rendezvous with debris l and that they are moving at the same speed, at t they will be in the same place.

Hence, it is possible for $N1$ to perform the same sequence of transfers that $N2$ has achieved and to finish the mission in the same time than $N2$ at a lower cost. Thus e' is a possible extension for s too.

If $N2$ leaves the orbit l at $t \geq T(s) + \delta t_{serv_l} + P_l$ then, according to Assumption 1, there exists a transfer that leaves l at $t' \in [T(s) + \delta t_{serv_l}, T(s) + \delta t_{serv_l} + P_l]$ and allows $N1$ to visit the next debris visited in e' earlier and at lower cost. This transfer brings us back to the same initial situation with the only difference that another orbit has been visited. Repeating the same procedure, we construct an extension of s that dominates e'.

Case where: *$V_{s'} \subsetneq V_s$*

Let $A = V_s - V_{s'}$ be the set of debris visited in the solution s and not visited in the solution s'. Let us also define B the set of debris that have to be visited in extensions of both s and s'. Hence, an extension of s' has to visit the set of debris $A \cup B$.

Let e' be a completion of s' that visits the set $A \cup B$ in a given order. Thus, the completion has the form $\{l, A_1, B_1, A_2, B_2, .., A_h, B_h\}$ where $A = \bigcup_{i=1}^{h} A_i$ and $B = \bigcup_{i=1}^{h} B_i$. The idea is to construct an extension of s from e' that costs less and which allows to achieve the mission more quickly than e'.

The shuttle $N1$ reaches l at $t_l = t_0 + T(s)$ while $N2$ reaches it at $t'_l = t_0 + T(s')$. $t_l \leq t'_l$ so according to the triangle inequality, there is a sequence of transfers that allow the shuttle $N1$ to reach the first element visited in B_1 faster than $N2$. Elements of B_1 have to be visited in e and e' as well, with a similar reasoning in the case 1, we can extends e in a way that $N1$ reaches the last element in B_1 faster than $N2$. This brings us back to the initial situation with the difference that we have visited $A_1 \cup B_1$ in addition. By repeating this process, we construct an extension e of s that allows $N1$ to reach the last element to visit in B_h faster than $N2$.

This validates the second fathoming rule.

3.3 The Branch and Bound algorithm

A solution pool is initially generated so as to use the fathoming rule D_1 as early as possible. For the procedure to be efficient, this pool must be generated quickly while containing reasonably good solutions. For this purpose, we compute costs and durations corresponding to solutions visiting debris in their orbit inclination order which will form the set initializing the solution pool. This order was chosen because it corresponds to some solutions that promote least cost transfers. In fact, transfers that require large changes in inclination cost more than transfers performed between near inclination orbits [8]. Besides, numerical results show that numbering debris according to their inclination order decreases computation time (see Section 5.2.2). The set of these solutions is stored in an ordered list ND which is updated each time a complete solution is generated.

A working list L contains current partial solutions. Initially L contains one element which is the initial partial solution $(\emptyset, 0, 0, 0)$. As long as there still exist elements in L, the algorithm computes successors of the first element in the list. The separated node is then removed and its successors are inserted in the working list L. Different orders were tested for list L. Once a solution is complete it is inserted into ND and then we remove from ND all dominated solutions. At the end of the algorithm, all elements stored in ND are non dominated. For each partial solution s, Γ_s denotes the set of successors of s.

Algorithm 1: Branch and bound algorithm

Output: Set of non dominated vectors ND
begin
  $L \leftarrow \{(\emptyset, 0, 0, 0)\}$
  Initialize ND a set of candidate non dominated vectors
  while $L \neq \emptyset$ **do**
    /*assume $ND = \{p_1, p_2, ..., p_r\}$ where solutions are ordered by non increasing costs*/
    Select $s \in L$
    $L \leftarrow L \backslash \{s\}$
    if *s is complete* **then**
      $Dominated \leftarrow false$
      $j \leftarrow 1$
      while $j \leq r$ *and* $C(p_j) \leq C(s)$ *and* $Dominated = false$ **do**
        if $T(p_j) \leq T(s)$ **then**
          $Dominated \leftarrow true$
        if $C(s) = C(p_j)$ *and* $T(s) \leq T(p_j)$ **then**
          $ND \leftarrow ND \backslash \{p_j\}$
        $j \leftarrow j + 1$
      if $Dominated = false$ **then**
        $ND \leftarrow ND \cup \{s\}$
        while $j \leq r$ *and* $T(s) \leq T(p_j)$ **do**
          $ND \leftarrow ND \backslash \{p_j\}$
          $j \leftarrow j + 1$
    else
      Generate Γ_s
      Update(L, Γ_s, ND)
  return ND

Algorithm 2: Procedure Update (L, L', ND)

Input: Two lists of partial solutions L and L' and list of pool solutions ND
Output: Working list L updated
begin
  /*assume $L' = \{s'_1, s'_2, ..., s'_q\}$ and $ND = \{p_1, p_2, ..., p_s\}$*/
  for $i \leftarrow 1$ **to** q **do**
    $Dominated \leftarrow false$
    $j \leftarrow 1$
    while $Dominated = false$ *and* $j \leq s$ **do**
      if p_j D_1 s'_i **then**
        $Dominated \leftarrow true$
      $j \leftarrow j + 1$
    if $Dominated = false$ **then**
      Insert s'_i in L unless it is dominated with respect to D_2
      Remove from L partial solutions that are dominated by s'_i

Algorithm 1 returns exactly the set of non dominated vectors and provides for each non dominated vector one efficient solution. In procedure 'Update', we first use D_1 to test dominance as it is easier to verify and may cut many branches of the search tree. Notice that the way partial solutions are stored influences the computation time since it can limit the number of dominance tests which will be checked. Three different orders were tested in our case as will be seen in next section.

4 Implementation issues

4.1 Debris order

The order in which debris are numbered is a crucial implementation issue in branch and bound procedures. We would like to number debris in an order allowing least cost solutions to be generated first. Two elements influence the cost of the transfer between two orbits: the difference between the two orbits inclination and the difference between the two orbits altitudes. In the following, we test two ways of numbering and compare them with a random numbering:

- The orbits inclination numbering.
- The orbits altitude numbering.

4.2 Data storage

In the solution pool, complete solutions are inserted in the lexicographic order of the criteria vector: (cost, duration). $\preceq_1$ is defined as follows: Two solutions $s_1 = (V_1, l_1, C(s_1), T(s_1))$ and $s_2 = (V_2, l_2, C(s_2), T(s_2))$ are such that $s_1 \preceq_1 s_2$ if and only if:
$C(s_1) < C(s_2)$ or $(C(s_1) = C(s_2)$ and $T(s_1) \leq T(s_2))$
The interest of this order is that for each pair of solutions (s_1, s_2) such that $s_1 \preceq_1 s_2$, one has s_2 cannot dominate s_1. The way a new solution is inserted in the solution pool is shown in Figure 4.

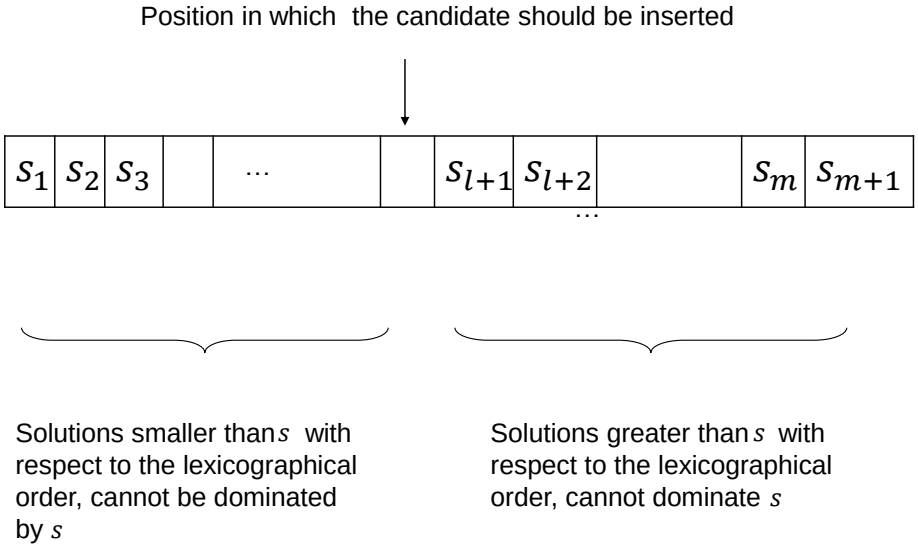


Figure 4: The insertion of a new solution in the solution pool

For the partial solutions, we have tested the three orders $\preceq_1$, $\preceq_2$ and $\preceq_3$. The order $\preceq_2$ is defined for each pair (s_1, s_2) as follows: $s_1 \preceq_2 s_2$ if and only if:
$|V_1| > |V_2|$ or $(|V_1| = |V_2|$ and $C(s_1) < C(s_2))$ or $(|V_1| = |V_2|$ and $C(s_1) = C(s_2)$ and $T(s_1) \leq T(s_2))$
The interest of this order is that it allows separating least cost partial solutions from those that visit more debris first and decreases the number of dominance tests. The insertion of a new element is done as explained in Figure 5.

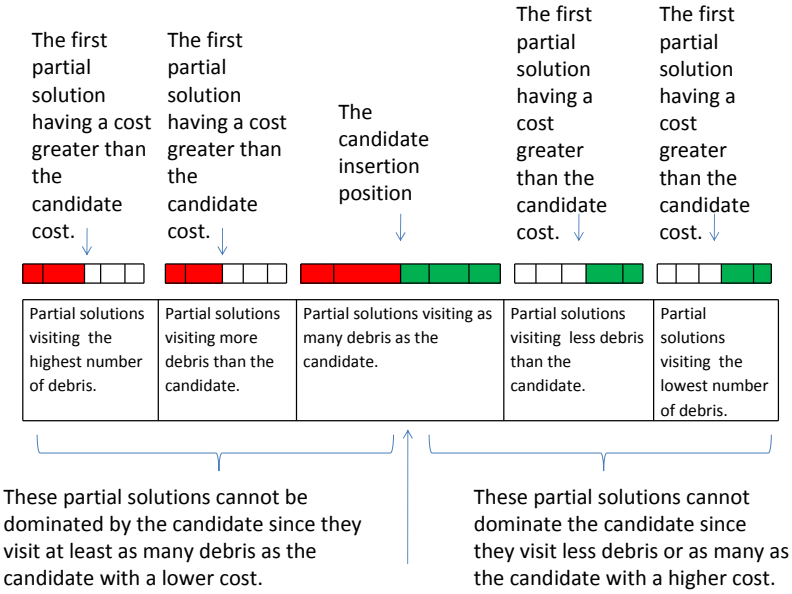


Figure 5: The insertion of a new partial solution with respect to $\preceq_2$ order

The order $\preceq_3$ is defined for each pair (s_1, s_2) as follows:
$s_1 \preceq_3 s_2$ if and only if: $C(s_1) + T(s_1) \leq C(s_2) + T(s_2)$
The last order has the advantage to allow less dominance tests to execute without promoting any of the two criteria.

5 Computational experiments and results

5.1 Experimental design

To get the description of space object orbits, we use the most complete source of orbital element information available to the public today which is the two-line elements (TLEs) set given by NORth American aerospace Defense command (NORAD). NORAD distributes orbital information on a very large number of space objects in the form of two-line elements sets (TLEs). Each TLE consists of two lines of 69 characters, containing an object identifier, the epoch and a set of orbital elements that have been estimated by NORAD. The NORAD element sets are "mean" values obtained by removing periodic variations in a particular way [15]. In order to obtain good predictions, these periodic variations must be reconstructed in exactly the same way they were removed by NORAD . Hence, The NORAD element

Instances	Trajectories starting at t_i	Trajectories starting at $t_i + 0.25 \times P_i$	Trajectories starting at $t_i + 0.5 \times P_i$	Trajectories starting at $t_i + 0.75 \times P_i$	Trajectories starting at $t_i + P_i$
1^{st} instance	100	100	100	100	100
2^{nd} instance	100	100	100	100	100
3^{rd} instance	100	100	100	100	100
4^{th} instance	100	100	100	100	100
5^{th} instance	100	100	100	100	100

Table 1: The percentage of trajectories verifying the triangle inequality

sets must be used with the propagation algorithm: SGP4. In the present article, we use this algorithm to predict debris positions.

5.2 Results

The goals of the experiments are:

- to determine the best order to sort partial solutions.
- to determine the best order for numbering debris in our approach.
- to analyse the impact of using fathoming rules.
- to compare Pareto fronts obtained with different time steps.
- to view some Pareto fronts and compare them with inital solutions pool.

5.2.1 Preprocessing

Verification of the validity of the triangle inequality

We consider five triplets of debris orbits. For each of these triplets (i, j, k), we compute costs and durations of efficient trajectories allowing the shuttle to go from i to k through j given a date t_i at which the shuttle reaches orbit i. We compute the same set of trajectories when the shuttle reaches i at $t_i + 0.25 \times Pi$, $t_i + 0.5 \times Pi$, $t_i + 0.75 \times Pi$ and $t_i + Pi$. The idea is to show that these sets of trajectories are dominated by at least one transfer of $A_{il}(t_i)$. For this purpose, we compute the percentage of each of these sets dominated by at least one transfer among $A_{il}(t_i)$. Results are reported in Table 1. As shown in the table, most of these transfers are dominated which legitimizes the triangle inequality hypothesis.

Determination of transfer parameters

The computation time is directly related to the cardinality of the set of possible transfers. Reducing the computation time requires decreasing this set cardinality. The cardinality of $A_{ij}(t_i)$ depends on:

- The maximum waiting duration allowed on each orbit.

Debris pairs	The % of ND transfers obtained after waiting one period	The % of ND transfers obtained after waiting two periods	The % of ND transfers obtained after waiting three periods
1^{st} pair	100	100	100
2^{nd} pair	85.7	100	100
3^{rd} pair	87.5	87.5	100
4^{th} pair	92.8	100	100
5^{th} pair	100	100	100
6^{th} pair	83.33	100	100
7^{th} pair	75	100	100
8^{th} pair	100	100	100

Table 2: The percentage of efficient transfers progression

- The number of possible transfer durations.

- Choice of the maximum transfer duration

  One may allow the shuttle to take a long time in order to achieve a transfer, but the loss on time must be compensated by a significant gain on cost. We consider in Figure 6, five pairs of LEO debris and study the evolution of the transfer cost depending on the transfer duration. As shown in the figures, most transfers that take more than 10 days to be performed are dominated. In the following, we assume that an interorbital transfer takes at most 10 days to be achieved.

- Choice of the maximum waiting duration

  The departure possibilities number must be limited. For this purpose, we test all the possible waiting durations using an increment of 30 minutes and less than ten times the period of the departure orbit. We then determine the costs of the transfers that depart from each of these positions and take complete days to be achieved without exceeding one month. We retain only non dominated transfers. The percentage of the non dominated transfers already obtained after a number of periods is then computed. We show in Table 2 the evolution of the efficient transfers duration obtained with eight different pairs of orbits. As shown in the table, the majority of non dominated transfers is already obtained after testing departure positions allowed by waiting one period. Therefore, in the following, we assume that the waiting time in any orbit cannot exceed the period of this orbit.

5.2.2 The impact of partial solutions sorting order and of debris numbering

We compare here the performance the algorithm when partial solutions are inserted in the three orders and when we number debris in their orbit in-

Instance size	$\preceq_1$				$\preceq_2$				$\preceq_3$			
	Altitude order		inclination order		Altitude order		inclination order		Altitude order		inclination order	
	B & B time	global time	B & B time	global time	B & B time	global time	B & B time	global time	B & B time	global time	B & B time	global time
3	2	34	2	29	2	30	2	26	2	37	2	39
4	3	167	3	145	3	143	2	132	3	174	4	163
5	7	768	6	679	5	645	5	630	8	701	7	689
6	25	1568	23	1481	21	1206	18	1250	28	1591	27	1390
7	67	5423	51	5360	53	4493	52	4199	63	4839	61	5951
8	154	8456	116	7855	135	7813	105	7600	140	8810	154	8721
9	456	9874	449	9324	330	9804	371	9341	510	10549	511	10034
10	680	11569	632	11429	653	10345	658	10321	743	11600	734	11003

Table 3: The obtained computation time using different orders

Instance size	With D_1		With D_2		With $D = D_1 \cup D_2$	
	B & B time	global time	B & B time	global time	B & B time	global time
3	3	32	3	35	2	26
4	3	145	3	159	2	132
5	5	756	6	680	5	630
6	23	1373	22	1327	18	1250
7	67	5643	62	5472	52	4199
8	156	8340	134	7983	105	7600
9	463	10376	423	10562	371	9341
10	714	11540	702	11582	658	10321

Table 4: The impact of fathoming rules use on computation time

clination order or in their orbit altitude order. The time step is set to 15 minutes.
As shown in Table 3, the best order to sort partial solutions is $\preceq_2$ which is the lexicographic order of $(|V|, cost, duration)$. Furthermore the best way of numbering the orbits is the inclination order. The lexicographic order promotes partial solutions visiting more debris which accelerates the construction of complete solutions which improves the solutions pool and strengthen the first fathoming rule. Furthermore numbering debris in their inclination order promotes least cost solutions to appear first.

5.2.3 Impact of fathoming rules

We use here the $\preceq_2$ to sort partial solutions, and we number debris in their inclination order. To view the impact of fathoming rules, we compare the computation time with and without each fathoming rule. As shown in Table 4 the use of fathoming rules decrease notably the computation time in the two cases. The best computation time is obtained when both of the fathoming rules are used.

5.2.4 Impact of time step

In order to view the impact of the time step, we insert partial solutions in the order $\preceq_2$, use both of dominance relations and number debris in their orbit inclination order. For several instance sizes, we generate the set of non dominated solutions obtained with each instance and compute the number of non dominated vectors, and the corresponding computation time.

Size of the instance	15'		25'			35'		
	$\|ND\|$	Computation time	$\|ND\|$	Computation time	ϵ_{moy}	$\|ND\|$	Computation time	ϵ_{moy}
3	54	26	38	23	0.06	33	17	0.25
4	63	132	43	103	0.06	39	90	0.4
5	84	630	63	549	0.05	49	465	0.44
6	135	1250	127	1278	0.07	63	1130	0.35
7	169	4199	157	3745	0.07	112	3087	0.27
8	250	7600	237	7620	0.04	183	5145	0.42
9	343	9341	365	9273	0.06	312	7652	0.42
10	483	10321	389	10492	0.08	369	8622	0.34

Table 5: The impact of time step on computation time

We increase the time step and compute the set of non dominated vectors obtained using greater time steps.

The idea is to compare the quality of these sets of non dominated debris. We use the ϵ-approximation relation introduced in [21]: for a given accuracy $\epsilon > 0$, an $(1 + \epsilon)$-approximation is a subset of solutions which contains, for each efficient solution, a solution that is at most at a factor $(1 + \epsilon)$ on both objective values. Thus, to evaluate the quality of the Pareto front obtained with greater time steps, we compute the least value of ϵ such that the second Pareto front is an ϵ-approximation of the first one. As shown in Table 5, the use of greater time steps decreases the computation time but deteriorates notably the quality of non dominated set.

5.2.5 Studying some Pareto fronts

In this paragraph, we propose to view some Pareto fronts obtained with several instance sizes. In each case we compare the quality of the non dominated vectors set and the solutions pool used in the beginning of the algorithm. To compare these two sets quality a worst case criterion is used. Hence we compute the least value of ϵ such that the solutions pool is an ϵ-approximation of the final set of non dominated vectors.

In Figure 7, we present the set of non dominated vectors obtained with a six debris instance (solutions marked with '+') and performances of the pool of solutions initially generated for the same instance (solutions marked with points). As can be seen in the figures, solutions that visit debris according to their inclination order present a good approximation of the set of non dominated vectors. Hence, the least value of the ϵ for which the solutions pool is an ϵ-approximation of the final set of non dominated vectors is 0.49 for the 6 debris instance, 0.33 for the seven's 0.22 for the eight's and 0.13 for the nine debris instance. In addition, the order of the visit in the set of non dominated vectors obtained in each case is not very different from the inclination order. Thus, most of the non dominated vectors differ from the set of solutions pool by less than three swaps.

6 Conclusions

The purpose of this work was to propose a new modeling of the problem of removing potentially dangerous space debris. The problem was presented as a time dependent biobjective traveling salesman problem. The BiTDTSP has not been studied in the literature before.

To efficiently solve the BiTDTSP, we proposed a branch and bound approach. The use of dominance relations avoids exploring some partial solutions that would not lead to non dominated vectors, and the limitation of the 'on orbit waiting durations' gives a good approximation of the set of non dominated vectors in less time. As a perspective of this work, we will study more deeply the interesting area in each non dominated solution set by using smaller time steps. As removing all space debris is too expensive another perspective of this work is to model and efficiently solve the problem of determining the most troublesome debris.

References

[1] L. Anselmo, A. Rossi, and C. Pardini. Updated results on the long-term evolution of the space debris environment. *Advances in Space Research*, 23:201–211, 1999.

[2] C. Bazgan, H. Hugot, and D. Vanderpooten. Solving efficiently the 0-1 multi-objective knapsack problem. *Computers & Operations Research*, 36(1):260–279, 2009.

[3] R. Bellman. Dynamic programming treatment of the travelling salesman problem. *Journal of the ACM (JACM)*, pages 61–63, 1962.

[4] J.F. Bérubé, M. Gendreau, and J.Y. Potvin. An exact-constraint method for bi-objective combinatorial optimization problems: Application to the traveling salesman problem with profits. *European Journal of Operational Research*, pages 39–50, 2009.

[5] R. Crowther. Orbital debris: a growing threat to space operations. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 361(1802):157–168, 2003.

[6] A. E. Petropoulos and R. P. Russell. Low-thrust transfers using primer vector theory and a second-order penalty method. In *AIAA/AAS Astrodynamics Specialist Conference*, pages 18–21, 2003.

[7] M. Held and R.M. Karp. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied Mathematics*, 10(1):196–210, 1962.

[8] B. J. Wadsley and R. G. Melton. Optimal visitation order for spacecraft servicing missions. *Advances in the astronautical sciences*, 129(3):2705–2720, 2008.

[9] D.J. Kessler. Collisional cascading: The limits of population growth in low earth orbit. *Advances in Space Research*, 11(12):63–66, 1991.

[10] J.-C. Liou. Collision activities in the future orbital debris environment. *Advances in Space Research*, 38(9):2102–2106, 2006.

[11] T. Lust and A. Jaszkiewicz. Speed-up techniques for solving large-scale biobjective tsp. *Computers & Operations Research*, 37(3):521–533, 2010.

[12] D. Madakat, J. Morio, and D. Vanderpooten. Biobjective planning of an active debris removal mission. *Acta Astronautica*, 84:182 – 188, 2013.

[13] C. Malandraki and R.B. Dial. A restricted dynamic programming heuristic algorithm for the time dependent traveling salesman problem. *European Journal of Operational Research*, 90(1):45–55, 1996.

[14] L. Paquete and T. Stützle. Design and analysis of stochastic local search for the multiobjective traveling salesman problem. *Computers & Operations Research*, 36(9):2619–2631, 2009.

[15] F. R. Hoots and R. L. Roehrich. Models for propagation of NORAD element sets., 1980. Space track report no. 3.

[16] T. R. Stodgell and D. B. Spencer. Satellite rendezvous tours using multiobjective evolutionary optimization. *Advances in the astronautical sciences*, 129(3):2069–2094, 2008.

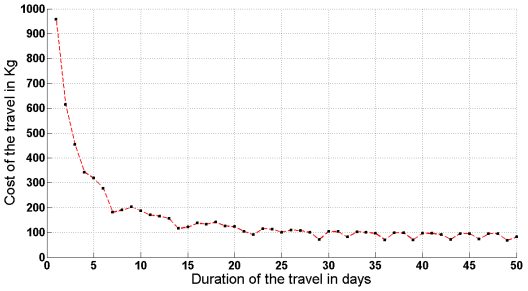
[17] H. Schmitz and S. Niemann. A bicriteria traveling salesman problem with sequence priorities. *Metaheuristics in the Service Industry*, pages 1–14, 2009.

[18] J. Schneider. The time-dependent traveling salesman problem. *Physica A: Statistical Mechanics and its Applications*, 314(1-4):151–155, 2002.

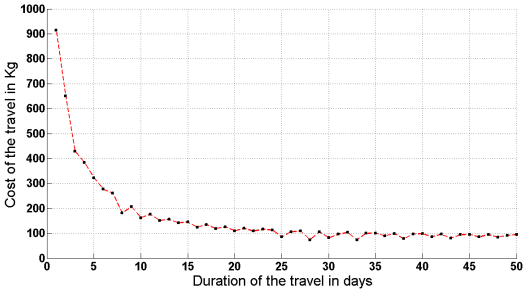
[19] A. V. Arutyunova, D. Yu. Karamzinb, and F. Pereira. Pontryagin's maximum principle for constrained impulsive control problems. *Nonlinear Analisys*, 75:1045–1057, 2011.

[20] Y. Z. Luo, J. Zhang, H. Li, and G. J. Tang. Interactive optimization approach for optimal impulsive rendezvous using primer vector and evolutionary algorithms. *Acta Astronautica*, 67(3-4):396–405, 2010.

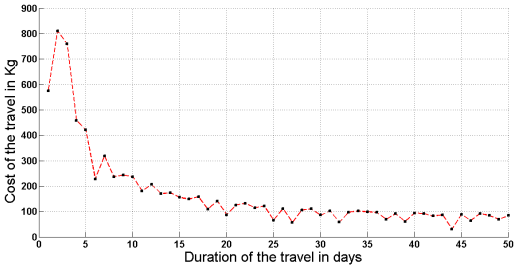
[21] E. Zitzler, L. Thiele, M. Laumanns, C. M Fonseca, and V.G. Da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *Evolutionary Computation, IEEE Transactions on*, 7(2):117–132, 2003.



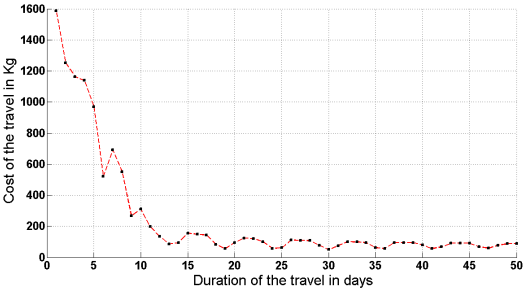
(a) Instance 1



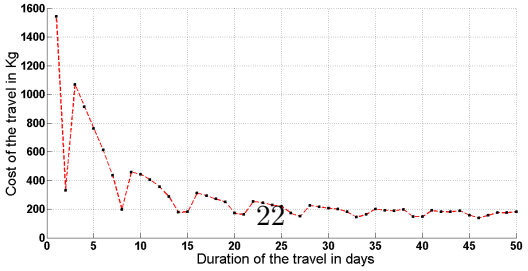
(b) Instance 2



(c) Instance 3

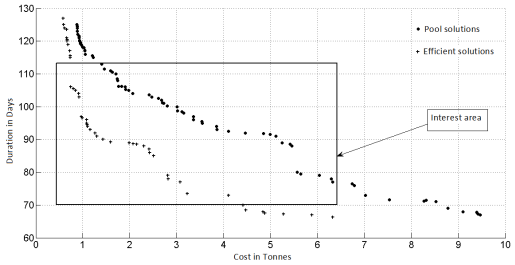


(d) Instance 4

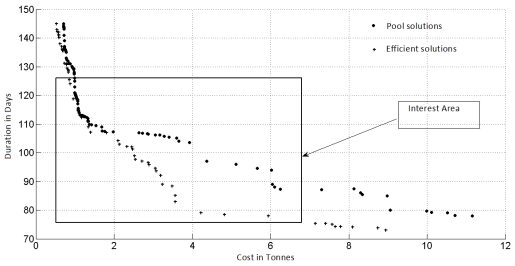


(e) Instance 5

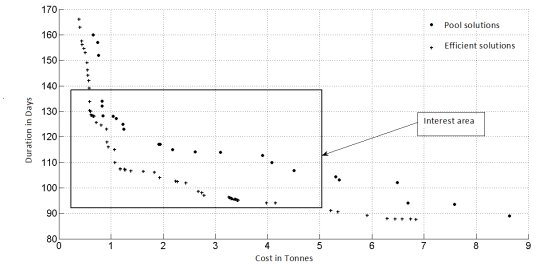
Figure 6: Changes in transfer cost depending on travel duration



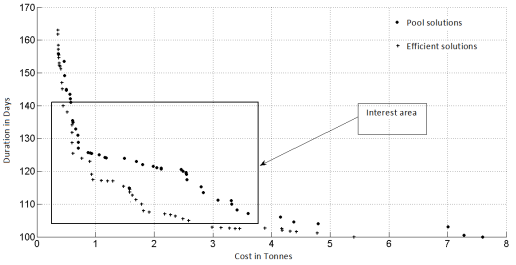
(a) The set of non dominated vectors and pool solutions obtained with an instance of 6 debris



(b) The set of non dominated vectors and pool solutions obtained with an instance of 7 debris



(c) The set of non dominated vectors and pool solutions obtained with an instance of 8 debris



(d) The set of non dominated vectors and pool solutions obtained with an instance of 9 debris

Figure 7: Non dominated vectors and pool solutions for different instances of debris