2024 S.T. Yau High School Science Award (Asia)

Research Report

The Team

Registration number: Comp-136

Name of team member: Isaac HUNG

School: Sha Tin College

Name of supervising teacher: Bing XU

Job title: Assistant Professor

School/Institution: Department of Aeronautical and Aviation Engineering, The Hong Kong Polytechnic University

Title of Research Report

Multi-objective optimization for effective active debris removal mission planning with memetic algorithms

Date

August 14, 2024

Multi-objective optimization for effective active debris removal mission planning with memetic algorithms

Isaac HUNG

Abstract

The amount of space debris in Low Earth Orbit (LEO) has been steadily increasing since the beginning of space exploration. This debris severely threatens the safety of operational satellites and missions due to potential damage caused by collisions at orbital velocities. Furthermore, collisions between debris will lead to greater fragmentation, causing a cascading effect known as the Kessler syndrome, rendering LEO unusable for space exploration.

Active debris removal (ADR), which involves using a satellite to capture and deorbit debris, is a promising approach to stabilizing the growth rate of space debris in LEO, especially as the amount of debris would still increase even in the absence of further launches due to fragmentation caused by collisions.

Several challenges still stand in the way of effectively using ADR missions to mitigate the Kessler syndrome. Constant fragmentation and change in the dynamic LEO environment, combined with limitations in current debris tracking and monitoring technology, makes it difficult for researchers and engineers to plan safe and scalable ADR missions. Furthermore, to account for the rapid growth of the space industry, ADR missions must be done in a time-efficient and cost-effective manner, enabling progress in space research without compromising on the long-term sustainability of the LEO environment.

This paper presents an ADR mission planning problem formulation based on multi-objective optimization and the Time-Dependent Orienteering Problem (TDOP), and develops a novel method to solve the problem based on memetic algorithms. We show that our algorithm produces solutions of high quality and provides a high degree of flexibility in planning ADR missions to achieve various mission-specific objectives and meet certain constraints. We also present the use of long-term LEO environment simulations such as ESA's `cascade` library as a novel part of our methodology for evaluating solutions.

Keywords: Active Debris Removal, Combinatorial Optimization, Multi-Objective Optimization, Evolutionary Computation, Time-Dependent Orienteering Problem, Memetic Algorithms

Acknowledgement

Isaac Hung:

I would like to thank my research supervisor, Dr. Bing Xu, for mentoring and supporting me to work on this project and providing expertise on orbital mechanics. Without his expertise, guidance and feedback, this research project would not have been possible.

I would also like to thank Dr. Tianqi Wang for his guidance and support in the field of mathematical optimization and linear programming, as well as Dr. Sean Swei for his insights into the current state of active debris removal (ADR) and the requirements for ADR mission planning to create viable solutions for engineers in practice.

I would also like to thank the developers of and contributors to the software packages and libraries I have used in this project, including `cascade`, `jMetalPy`, `matplotlib`, `NumPy`, `pandas`, `pykep`, `SciPy` and `sgp4`.

Finally, I would like to extend my utmost gratitude to my family members who have been very supportive of me during my research project.

Commitments on Academic Honesty and Integrity

We hereby declare that we

1. are fully committed to the principle of honesty, integrity and fair play throughout the competition.
2. actually perform the research work ourselves and thus truly understand the content of the work.
3. observe the common standard of academic integrity adopted by most journals and degree theses.
4. have declared all the assistance and contribution we have received from any personnel, agency, institution, etc. for the research work.
5. undertake to avoid getting in touch with assessment panel members in a way that may lead to direct or indirect conflict of interest.
6. undertake to avoid any interaction with assessment panel members that would undermine the neutrality of the panel member and fairness of the assessment process.
7. observe the safety regulations of the laboratory(ies) where we conduct the experiment(s), if applicable.
8. observe all rules and regulations of the competition.
9. agree that the decision of YHSA(Asia) is final in all matters related to the competition.

We understand and agree that failure to honour the above commitments may lead to disqualification from the competition and/or removal of reward, if applicable; that any unethical deeds, if found, will be disclosed to the school principal of team member(s) and relevant parties if deemed necessary; and that the decision of YHSA(Asia) is final and no appeal will be accepted.

(Signatures of full team below)

X [illegible]

Name of team member: Isaac HUNG

X [illegible]

Name of supervising teacher: Bing XU

Noted and endorsed by

(signature) [illegible]

SHA TIN COLLEGE * HONG KONG *

Name of school principal: Carol LARKIN, Sha Tin College

Contents

1 Introduction

Low Earth Orbit (LEO) is typically defined as an orbital region within 2000 km above Earth's surface [1]. LEO orbits require the least energy for satellite placement, and the close proximity of LEO satellites to the surface make them easier to maintain and suitable for applications such as communication, where bandwidth and latency play key roles in performance. Because of these characteristics, LEO contains most of the artificial space objects in orbit and is a typical choice for human spaceflight.

As such, the recent rapid increase of space debris in LEO is becoming a major concern for space researchers and engineers due to the increased risk of debris colliding with operational satellites. Furthermore, with an increasing number of fragmentation events caused by collisions between existing debris, researchers expect the density of space debris in LEO to increase to the extent that collisions cause a cascading effect, increasing the likelihood of further collisions and rendering LEO unusable for space exploration, a scenario known as the Kessler syndrome [2].

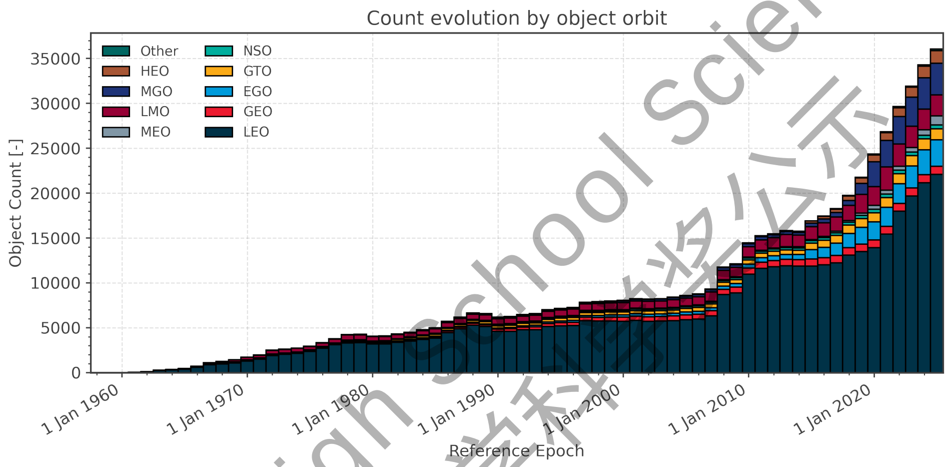


Figure 1: Evolution of number of objects by orbital region [3]

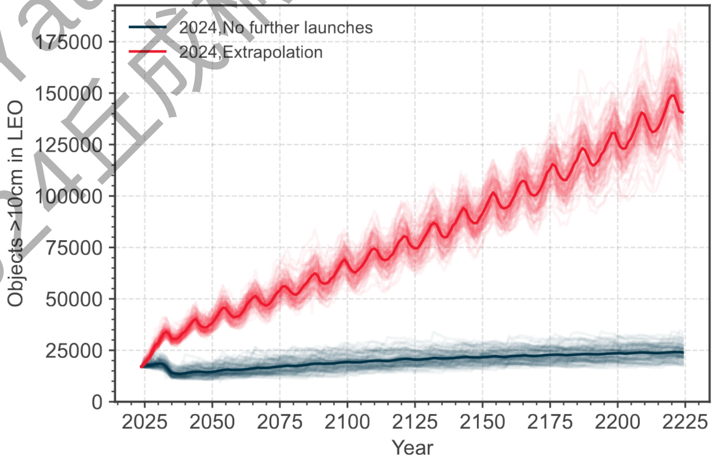


Figure 2: Extrapolation of future LEO space debris population [3]

Space debris is defined as non-functional manmade objects, including fragments or elements of man-made objects, that are in orbit around Earth or re-entering the atmosphere [1]. Types of space debris include defunct payloads, rocket bodies, and fragmentation debris [3]. Figure 1 shows the upwards trend of the number of objects in LEO, including functional satellites and debris, and Figure 2 shows an extrapolation by the European Space Agency (ESA) predicting the increase in large (greater than 10 cm) objects in LEO in the future [3].

Active debris removal (ADR) refers to the use of spacecraft missions that interact with debris to remove it or reduce its lifetime. Studies have shown that even when following the Inter-Agency Space Debris Coordination Committee (IADC) Space Debris Mitigation Guidelines [1] and in the absence of any further launches, the amount of space debris in LEO will still continue to grow due to collision fragmentation [4]. As such, it has been recommended that at least five large objects must be removed from dense LEO regions each year to stabilize the LEO environment [5].

An ADR mission is implemented through the use of an ADR satellite, the "chaser", which performs a sequence of orbital manuevers to rendezvous with each piece of space debris, performing a capture or removal operation to either collect or deorbit it, such as attaching a deorbiting kit to it [6]. Each mission will have a specific set of objectives, for example removing a greater amount of or more threatening debris, as well as a set of constraints, such as ensuring that the mission is completed within a time limit.

Mission planning for ADR is challenging. This is due to the continuous nature of the orbital mechanics of space debris, creating an infinite amount of possibilities for mission trajectories and making it difficult to routinely apply techniques for solving combinatorial optimization problems.

In this report, we will introduce background topics relevant to this investigation (Section 2), discuss typical methods employed to address the problem of ADR mission planning (Section 3), present a problem statement that simplifies the task by using a discrete representation of time (Section 4), and propose a novel approach to solving the problem with memetic algorithms (Section 5). We then evaluate our methodology and discuss the viability of our approach for ADR mission planning (Sections 6, 7, and 8).

We summarize the main contributions of this report. Firstly, a novel memetic algorithm, the Active Debris Removal Memetic Algorithm (ADR-MA), is introduced to solve the problem of ADR mission planning. Secondly, a novel local search algorithm that exploits ADR-specific characteristics of the problem to produce better results is presented. Thirdly, we develop a novel simulation-based methodology, the Simulation-based Debris Threat Index (SDTI), for evaluating the threat posed by LEO space debris in causing fragmentation contributing to the Kessler syndrome.

2 Background

This section will introduce the background topics in the field of optimization which are relevant to ADR mission planning. For a summary of orbital mechanics relevant to ADR missions, refer to Appendix A.

2.1 Combinatorial optimization

The objective of ADR is to protect operational satellites by removing threatening debris and mitigate the Kessler syndrome by avoiding further debris fragmentation [4]. Due to the high time and monetary costs of space missions, ADR mission planning should seek to achieve the maximum impact on protecting the LEO environment while keeping costs at a reasonable level. ADR missions consist of a set of debris to remove and a time-dependent sequence of maneuvers to rendezvous with debris, creating a combinatorial explosion of possible missions, out of which an optimal or near-optimal mission sequence is desired. Hence, ADR mission planning is formulated as a combinatorial optimization problem.

Optimization is a field of mathematics and computer science that refers to finding the "best" or optimal solutions to problems within the set of all feasible solutions [7]. This typically involves finding a solution to maximize or minimize an objective function that indicates the quality and desirability of a solution, while satisfying a set of constraints. The basic structure of an optimization problem is as follows:

$$\min_{x \in \mathbf{\Omega}} f(x) \tag{1}$$

where x is a solution, f is the objective function and $\mathbf{\Omega}$ represents the solution space, which encapsulates the constraints of the problem.

Combinatorial optimization is a subfield of optimization where the optimal solution is contained within a finite, discrete solution space, emphasizing the combinatorial nature of the optimization problem formulation and solutions [8]. It finds many applications in the field of operations research, where common problems include routing problems such as the Travelling Salesman Problem (TSP) [9] and the Vehicle Routing Problem (VRP), scheduling problems like the Job-shop Problem (JSP), decision-making problems such as the Knapsack Problem (KP), and various others.

Due to the nature of combinatorics, the size of the solution space for a combinatorial optimization problem often increases rapidly as the problem grows in scale, such as when more decision variables are added. This makes it impossible to enumerate and search the entire feasible set for most problems, especially at real-world scale. Because of this, metaheuristic approaches were developed, which are algorithms that find sufficiently good solutions to optimization problems that would otherwise be unsolvable through brute-force techniques. Although they provide no guarantee that a globally optimal solution will be found, a well-designed metaheuristic can provide a solution close to the optimum within a reasonable execution time. Popular metaheuristic approaches include evolutionary algorithms (EAs), ant colony optimization (ACO), particle swarm optimization (PSO), simulated annealing (SA), and local search (LS) techniques.

However, there are still more challenges in solving combinatorial optimization problems. Many such problems exhibit a rugged solution space, increasing the risk of converging upon a local maxima — a solution which appears to be superior to all other solutions near or similar to it, but is not the globally optimal solution. To decrease the likelihood of premature convergence, researchers consider the trade-off between "exploration and exploitation" [10], where exploration refers to exploring the solution space to avoid getting stuck in local maxima, and exploitation refers to improving solutions to obtain a maximum once the algorithm believes that the solution space has been sufficiently explored.

2.2 Multi-objective optimization

Many real-life optimization scenarios feature more than one objective to consider and optimize for, often with conflicts between objectives. In the context of ADR mission planning, objectives may include maximization of benefit towards the LEO environment, minimization of propellant cost or minimization of mission time.

Economist Vilfredo Pareto introduced the concept of Pareto optimality [11], referring to an outcome in which no objective can be improved upon without leaving another worse off, and hence the lack of an outcome that is better in every objective. The concept has been applied to multi-objective optimization problems (MOPs) [12], where solving a MOP involves finding a set of Pareto optimal solutions known as the Pareto set, either exactly or approximately.

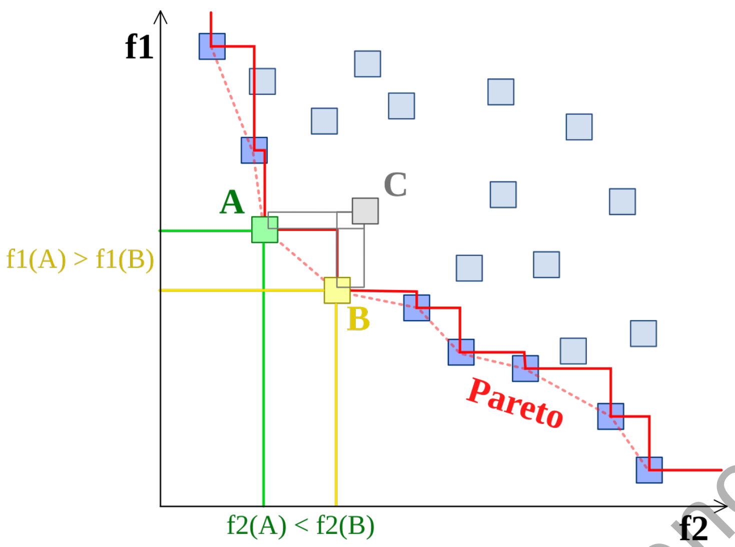


Figure 3: Example of a Pareto front [13]

Figure 3 shows an example of a Pareto front for an MOP where both objectives f_1 and f_2 are to be minimized. The red line is the Pareto front, which includes points A and B, because there are no solutions that have smaller values for both f_1 and f_2; in contrast, point C is not on the front because it is dominated — at least one solution exists that has more optimal values for both objectives.

Definition 1 *A solution x^* **dominates** another solution x, i.e. $x^* \succ x$ iff $\forall f \in F : f(x^*) \leqslant f(x) \wedge \exists f \in F : f(x^*) < f(x)$ where F is the set of objectives in a minimization MOP and $x, x^* \in \mathbf{\Omega}$.*

Definition 2 *A solution x^* is **Pareto optimal** if it is non-dominated, i.e. $\nexists x, x^* \in \mathbf{\Omega} : x \succ x^*$.*

Definition 3 *The **Pareto set** is the set of all Pareto optimal solutions.*

Definition 4 *The **Pareto front** is the image of the Pareto set in the objective space.*

Once a Pareto set has been found, a Decision Maker (DM) [14] can select a solution that lies on the Pareto front based on their knowledge of the problem and preference of one objective over another. The advantage of solving MOPs is that DMs are granted more flexibility in comparison to single-objective optimization problems, where the set of optimal solutions is typically much more limited.

2.3 Genetic algorithms

Genetic algorithms (GAs) are a specific kind of evolutionary algorithm that take inspiration from the biological process of natural selection [16]. GAs are categorized as naturally-inspired population-based metaheuristic optimization techniques [17], referring to its use of a population of individuals and the fact that they do not find exact solutions to problems. The flowchart of the standard GA is shown in Figure 4 [15].

Algorithm 1 shows the basic procedure for a GA, which maintains a population of candidate solutions, known as individuals, which are iteratively evaluated against the objective function, selected and modified to improve the fitness of solutions [16]. The time complexity of GAs can be calculated from their main components to be $O(n_t n_{\text{pop}} m)$, where m is the length of an individual chromosome.

GAs are known to be extremely flexible algorithms, with most of their phases and parameters being customizable, including the parameters, genetic encoding, genetic operators and termination conditions [18]. This allows algorithm designers to strategically employ domain-specific or problem-specific

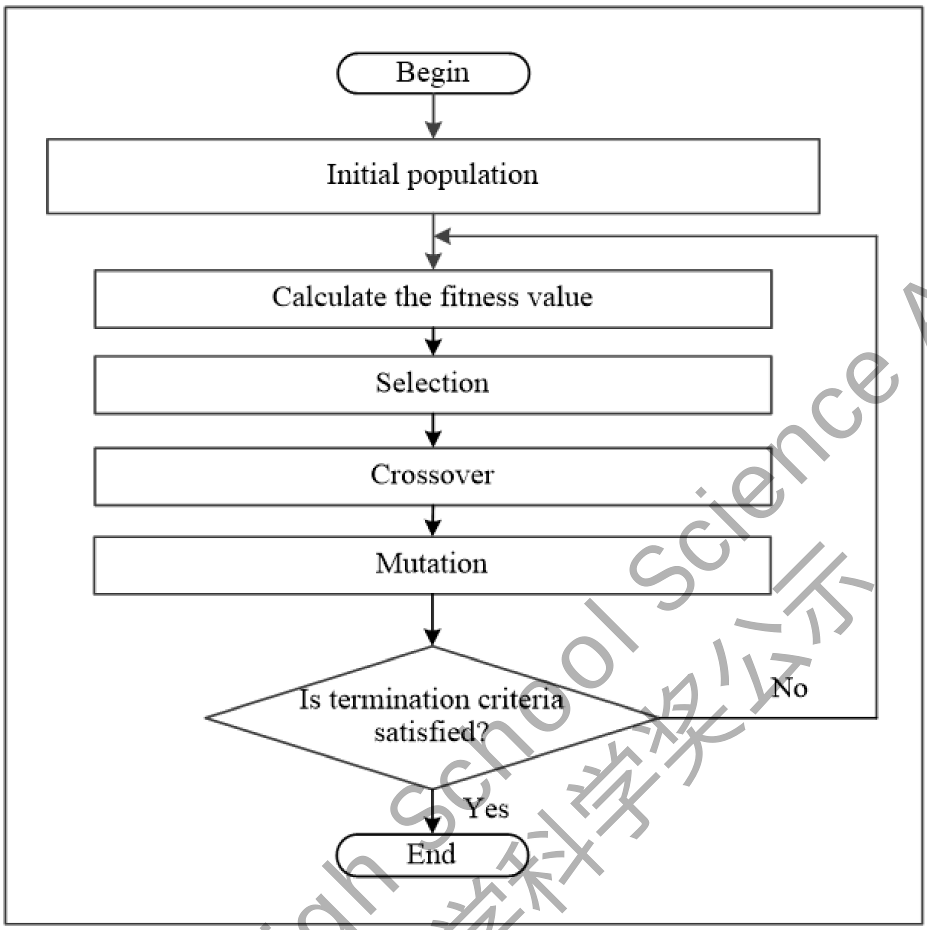

Begin
Initial population
Calculate the fitness value
Selection
Crossover
Mutation
Is termination criteria satisfied?
No
Yes
End

Figure 4: Flowchart of the genetic algorithm procedure [15]

Algorithm 1 Genetic algorithm

```
$P_0 \leftarrow$ INITIALIZE($n_{\text{pop}}$)        ▷ Randomly initialize a population $P$ of $n_{\text{pop}}$ individuals
for $t \leftarrow 1$ to $n_t$ do        ▷ Run for $n_t$ generations
    for all $x \in P_t$ do
        $f_x \leftarrow$ FITNESS($x$)        ▷ Evaluate the fitness of each individual
    end for
    $P_{t+1} \leftarrow \varnothing$
    while $|P_{t+1}| < n$ do
        $a, b \leftarrow$ SELECTION($P_t, f$)        ▷ Parents are selected based on fitness
        if RANDOM-PROBABILITY() $\leqslant P(crossover)$ then
            $a, b \leftarrow$ CROSSOVER($a, b$)        ▷ Crossover operator mixes genetic information
        end if
        if RANDOM-PROBABILITY() $\leqslant P(mutation)$ then
            $a, b \leftarrow$ MUTATION($a$), MUTATION($b$)        ▷ Mutation operator explores the solution space
        end if
        $P_{t+1} \leftarrow P_{t+1} \cup \{a, b\}$
    end while
end for
```

operators and configurations to achieve the best performance both in terms of solution quality and algorithmic performance, enabling GAs to be applied to various discrete and continuous optimization problems. However, GAs also have disadvantages, namely the risk of premature convergence on local maxima, sensitivity to parameter configurations, and computational efficiency [19]. Maintaining of genetic diversity and ensuring a balance between exploration and exploitation remain major challenges in the area of GAs.

3 Related work

3.1 ADR mission planning approaches

Various articles discussing the task of ADR mission planning have been published in the literature. Several problem formulations have been proposed, most prominently variants of the TSP with consideration of the time-dependent nature of such missions, such as the Time-Dependent Travelling Salesman Problem (TDTSP) and the Time-Dependent Orienteering Problem (TDOP).

In addition, many approaches have been proposed for solving the NP-hard combinatorial optimization problems. Federici et al. proposes a formulation of the TDOP as a standard search problem as well as the use of the A* graph search algorithm to solve it, employing a novel Longest Short Path (LSP) heuristic that can be solved efficiently with dynamic programming [20]. Zona et al. propose a similar formulation with multiple chasers, using genetic algorithms and the TSP 2-opt local search technique to solve the problem [21]. Other techniques, such as ant colony optimization (ACO) [22, 23], simulated annealing (SA) [24] and machine learning (ML) [25] have been proposed. Yang et al. propose a methodology using reinforcement learning (RL) [26], proposing a novel RL problem formulation and methodology combining RL with upper confidence trees (UCTs), finding that the RL-based methodology is both robust and resistant to premature convergence.

Consideration of multiple objectives for ADR missions has been studied by researchers [27, 28], possibly motivated by the multiple requirements of the thriving space industry. Common objectives presented in MOPs include maximization of the benefit derived from removing debris, minimization of orbital transfer cost ΔV, minimization of mission time or minimization of the number of chasers required to remove a cluster of debris.

Choi et al. propose a bi-objective formulation that aims to maximize the score of the removed debris and minimize the total ΔV of the mission [29]. The novel Hazard Criticality Index (HCI) is developed in their investigation. They use multi-objective evolutionary algorithms (MOEAs) [14] combined with a Random-Key (RK) genetic encoding scheme, finding that NSGA-II [30] achieves the best performance in terms of Pareto optimality and solution diversity.

Missel and Mortari focus on a specific mission, *Space Sweeper with Sling-Sat (4S)*, which features a chaser capable of using impulsive momentum exchanges from deorbiting debris, exploiting the fast orbital velocities of debris to reduce the total ΔV required for the mission [31]. They show that this consideration of orbital mechanics allows them to drive 73% of a certain debris removal sequence at no cost, and present a GA-based methodology for solving their problem formulation.

3.2 Strategies for effective ADR

While the design of ADR mission planning algorithms is important, it is also necessary to understand that effective ADR is more than application of combinatorial optimization techniques. Several studies have been published on the topic of the evolution of the LEO environment, using simulations such as NASA's LEGEND model [32] and ESA's MASTER model [33] to monitor the growth of the LEO debris population.

Liou et al. use the LEGEND model to find that “the LEO environment can be stabilized in the next 200 years with an ADR removal rate of five objects per year” [4]. This metric has since been used as a benchmark for subsequent studies on ADR mission planning [23, 34]. The study also introduces the concept of a “selection criterion based on the mass and collision probability of each object”, emphasizing the importance of prioritizing removal of debris that is more threatening to the LEO environment.

White and Lewis further investigate the optimal ADR removal rate with the DAMAGE and CASCADE models, finding that the optimal ADR removal fluctuates due to both in-orbit debris fragmentation and external factors such as further launches or uncertainties in future advancements in technology [35]. They model the change in size of the LEO population and present an adaptive strategy for ADR missions.

Rossi et al. further develop the selection criteria of debris to remove [36]. They present the Criticality of Spacecraft Index (CSI), which considers the physical characteristics, orbit, and environment of the object, and give a formula for the CSI in terms of these parameters.

4 Problem statement

The objectives and constraints on active debris removal will vary between missions, depending on the specific requirements of those who are implementing it. As such, there is no single problem statement that can address the problem entirely, but an overall framework and set of techniques for ADR mission planning can be developed. In particular, due to the nature of the space industry, most missions will have various, potentially conflicting objectives, so we choose to formulate it as a MOP.

For the rest of this report, we assume that the goal of ADR mission planning is to find a solution that optimizes the set of removed debris to maximize the impact of the mission, while simultaneously optimizing the maneuver sequence to minimize the transfer cost, with mission duration being considered as a constraint. This specific MOP has been previously investigated [29] and provides a realistic example of conflicting objectives in ADR mission planning.

Consider a cluster of $N \in \mathbb{N}$ debris numbered from $1, 2, \ldots, N$. Each debris is assigned a score, with $s_i \in \mathbb{R}^+$ used to denote the score of the i^{th} debris, quantifying the threat level posed by the debris remaining in orbit. This process is further explained in Section 5.4.

The task of the ADR mission planning algorithm is to produce a sequence of maneuvers between a selected subset of the debris targeted for removal, maximizing the total score of the removed debris, while minimizing the orbital transfer cost along the trajectory, without exceeding the constraints of the mission.

4.1 Orienteering Problem (OP)

The problem of ADR mission planning can be modeled with a variant of the Orienteering Problem (OP) [37]. Inspired by the sport of orienteering, the OP involves a “player” that must traverse a course with many checkpoints, each with a different score. The course is represented by a weighted complete graph, with vertices representing checkpoints and edges representing paths between them, with edge weights representing the distance or time required to travel between two checkpoints. The player must finish the course within a certain time limit, making it impossible to visit all checkpoints, so they must strategically select a subset of checkpoints and navigate between them to maximize their score. In the context of ADR, the player is the chaser, and the checkpoints are the debris.

The OP is a combination of the Knapsack Problem (KP) [38], which corresponds to debris selection, and the Travelling Salesman Problem (TSP) [9], which corresponds to trajectory optimization. As both subproblems are NP-hard, the OP is therefore also regarded as an NP-hard problem. The OP has other applications in problems such as the mobile crowdsourcing problem or the tourist trip design problem [39].

4.2 Discretized representation of time

As time is a continuous variable, the number of possible missions is infinite. While there are only a finite number of possible subsets of debris to select, there are an infinite number of orbital transfer possibilities as they can start and end at any time. While metaheuristic approaches such as GAs and PSO are able to tackle continuous optimization problems, calculating the cost of orbital transfers is non-trivial, and doing so within the mission planning algorithm would result in a performance bottleneck.

Hence, we use the strategy proposed by Zona et al. [21] to precompute all of the possible transfers within the time frame of the mission before trajectory optimization, using a discrete representation of time. Time is broken down into a set of N_t time epochs $\boldsymbol{\tau}$ as follows:

$$\boldsymbol{\tau} = \{\tau_1, \tau_2, \ldots, \tau_n\} \tag{2}$$

$$\tau_k = k\Delta t \tag{3}$$

$$\Delta t = \frac{t_{\max}}{N_t} \tag{4}$$

Δt represents the length of one time epoch, and $t_{\max}$ represents the maximum length of an ADR mission.

4.3 Selected orbital transfer strategy

We use the transfer strategy proposed by Cerf to perform orbital transfers between space debris [40], exploiting the J_2 perturbation to correct the RAAN of the chaser Ω_c during a drift orbit, represented by d. The steps to perform such an orbital transfer are as follows:

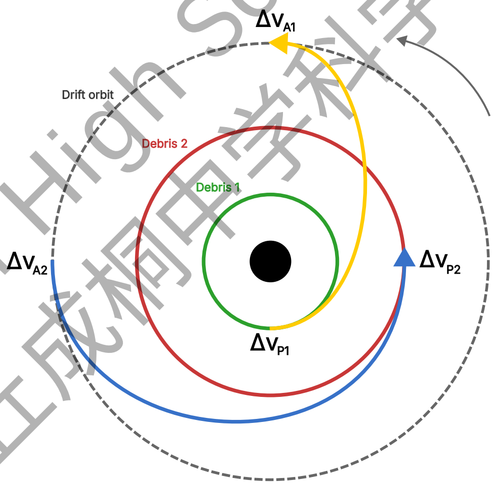


Figure 5: The proposed orbital transfer strategy

1. Perform a Hohmann transfer from the orbit i to the drift orbit d
2. Wait in the drift orbit for the J_2 perturbation to correct Ω_c
3. Perform a Hohmann transfer from the drift orbit d to the orbit j

Figure 5 depicts the 1$^{\text{st}}$ and 3$^{\text{rd}}$ stages of the orbital transfer strategy.

The cost to perform an orbital transfer between the i^{th} debris and j^{th} debris, departing from i at time epoch τ_k and arriving at j at epoch τ_{k+m} is referred to as ΔV_{ijkm}, where $1 \leqslant i, j \leqslant N$ and $\tau_k, \tau_{k+m} \in \boldsymbol{\tau}$ and $1 \leqslant m \leqslant m_{\max}$, where $m_{\max}$ limits the maximum length of the transfer.

To perform the transfer efficiently, it is necessary to find the optimal drift orbit with a semi-major axis a_d and inclination of i_d. This can be done through solving a nonlinear optimization problem with the drift orbit parameters as the decision variables and the resulting ΔV_{ijkm} as the cost function [20]:

$$\min_{a_d, i_d} \Delta V_{ijkm}(a_d, i_d) \tag{5}$$

such that

$$\Omega_c(\tau_{k+m}) = \Omega_j(\tau_{k+m}) \tag{6}$$

where

$$\Omega_c(\tau_{k+m}) = \Omega_c(\tau_0) + \dot{\Omega}_i(\tau_k - \tau_0) + \dot{\Omega}_d(\tau_{k+m} - \tau_k) \tag{7}$$

$$\Omega_j(\tau_{k+m}) = \Omega(\tau_0) + \dot{\Omega}_j(\tau_{k+m} - \tau_0) \tag{8}$$

and $\dot{\Omega}_h$ in $\mathrm{rad\,s^{-1}}$ for any $h \in \{i, j, d\}$ is determined by Equation 49. ΔV_{ijkm} can be represented as a function of a_d and i_d [41]:

$$\Delta V_{ijkm}(a_d, i_d) = \Delta V_{P1} + \Delta V_{A1} + \Delta V_{P2} + \Delta V_{A2} \tag{9}$$

This function is a summation of four separate maneuvers, which represent the impulsive maneuvers performed at the perigee P and apogee A of an orbit during two Hohmann transfers and are calculated with the law of cosines [41].

$$\Delta V_{P1} = \sqrt{v_i^2 + v_{P1}^2 - 2v_i v_{P1} \cos(\Delta i_i s_i)} \tag{10}$$

$$\Delta V_{A1} = \sqrt{v_d^2 + v_{A1}^2 - 2v_d v_{A1} \cos(\Delta i_i (1 - s_i))} \tag{11}$$

$$\Delta V_{P2} = \sqrt{v_j^2 + v_{P2}^2 - 2v_j v_{P2} \cos(\Delta i_j s_i)} \tag{12}$$

$$\Delta V_{A2} = \sqrt{v_d^2 + v_{A2}^2 - 2v_d v_{A2} \cos(\Delta i_j (1 - s_i))} \tag{13}$$

where

$$v_h = \sqrt{\frac{GM_\oplus}{a_h}} \tag{14}$$

$$\Delta i_i = |i_i - i_d| \tag{15}$$

$$\Delta i_j = |i_j - i_d| \tag{16}$$

$$s_i = \frac{1}{\Delta i_i} \arctan\left(\frac{\sin \Delta i_i}{\sqrt{a_d^3/a_i^3} + \cos \Delta i_i}\right) \tag{17}$$

$$s_j = \frac{1}{\Delta i_j} \arctan\left(\frac{\sin \Delta i_j}{\sqrt{a_d^3/a_j^3} + \cos \Delta i_j}\right) \tag{18}$$

Nonlinear programming (NLP) can be used to solve the optimization problem for each value of ΔV_{ijkm} prior to trajectory optimization, which can be stored in a tensor of shape $N \times N \times N_t \times m_{\max}$.

4.4 Time-Dependent Orienteering Problem (TDOP)

Due to the orbital mechanics of space debris, the OP in its standard form cannot be applied directly to solve the problem of ADR mission planning, as the transfer cost between debris is constantly changing. Hence, the problem must be formulated with a time-dependent extension of the OP, known as the Time-

Dependent Orienteering Problem (TDOP) [37].

After the transfer cost ΔV_{ijkm} has been precomputed for all possible orbital transfers during the mission, the problem reduces to the TDOP. The algorithm must output a mission plan $\boldsymbol{M}(\boldsymbol{d}, \boldsymbol{t})$, composed of an ordered subset of $n \leqslant N$ debris $\boldsymbol{d} = \{d_1, d_2, \ldots, d_n\}$ to remove, at the corresponding strictly increasing rendezvous times $\boldsymbol{t} = \{t_1, t_2, \ldots t_n\}$, where for each i, the ${d_i}^{\text{th}}$ debris is removed at time epoch t_i. The mission imposes certain constraints on the algorithm, requiring that the debris removal sequence be completed within a time limit $t_{\max}$ and that the total ΔV does not exceed $\Delta V_{\max}$.

The problem can be classified as a NP-hard bi-objective constrained combinatorial optimization problem. Formally:

$$\max_{\boldsymbol{M}} \sum_{i=1}^{n} s_{\boldsymbol{d}_i} \tag{19}$$

$$\min_{\boldsymbol{M}} \sum_{i=1}^{n-1} \Delta V_{\boldsymbol{d}_i \boldsymbol{d}_{i+1} \frac{\boldsymbol{t}_i}{\Delta t} \frac{\boldsymbol{t}_{i+1}-\boldsymbol{t}_i}{\Delta t}} \tag{20}$$

Equations 19 and 20 describe the two objectives of the optimization problem.

$$\sum_{i=1}^{n-1} \Delta V_{\boldsymbol{d}_i \boldsymbol{d}_{i+1} \frac{\boldsymbol{t}_i}{\Delta t} \frac{\boldsymbol{t}_{i+1}-\boldsymbol{t}_i}{\Delta t}} \leqslant \Delta V_{\max} \tag{21}$$

$$\boldsymbol{t}_n \leqslant t_{\max} \tag{22}$$

Equations 21 and 22 represent the constraints of the optimization problem.

$$\forall 1 \leqslant i \leqslant n: \ \boldsymbol{d}_i \in \mathbb{N}^+, 1 \leqslant \boldsymbol{d}_i \leqslant N \tag{23}$$

$$\forall 1 \leqslant i, j \leqslant n, \ i \neq j: \ \boldsymbol{d}_i \neq \boldsymbol{d}_j \tag{24}$$

$$\forall 1 \leqslant i \leqslant n: \ \boldsymbol{t}_i \in \boldsymbol{\tau} \tag{25}$$

$$\forall 1 \leqslant i \leqslant n-1: \ \boldsymbol{t}_i < \boldsymbol{t}_{i+1} \tag{26}$$

Equations 23 to 26 ensure that the solution $\boldsymbol{M}$ is well-formed. This formulation of the ADR mission planning problem with the TDOP is based on that proposed by Federici et al. [20], extended to be a MOP.

5 Approach

We propose a novel approach based on memetic algorithms (MAs) for solving the problem of ADR mission planning. MAs refer to hybrid metaheuristic algorithms that combine a global search process with a local search technique [42]. While the global search process can be any metaheuristic, research on MAs are often focused on combining EAs with local search [43], and indeed MAs have been described within the analogy of EAs, where the global search process is analogous to natural selection and the local search process is analogous to individuals learning from experience over their lifetime [44].

5.1 Genetic algorithm

Our MA-based approach uses a multi-objective GA as its basis. We employ the Non-dominated Sorting Genetic Algorithm (II) or NSGA-II [30], an elitist multi-objective GA characterized for its computational efficiency. We will first briefly introduce NSGA-II and justify why we use it, and then describe and justify our choices of GA parameters and operators.

5.1.1 NSGA-II

NSGA-II is an improvement on the original NSGA multi-objective GA [30]. Instead of keeping one population of size n, NSGA-II keeps two populations P and Q. On every iteration, both populations are combined and sorted into k non-dominated fronts $F_0, F_1, \dots, F_{k-1}$. To perform the non-dominated sorting process, the initially Pareto optimal or non-dominated solutions are assigned to F_0, and then F_0 is removed from consideration. The solutions that are now non-dominated when F_0 is not considered are added to F_1 and the process repeats until no solutions remain unsorted.

Once non-dominated sorting is complete, each front is individually sorted by crowding distance, from highest to lowest. Crowding distance is a metric that indicates the size of the hypervolume in the objective space between a solution, the previous solution and the next solution when sorted in terms of performance on one objective. Crowding distance for a front F on a M-objective MOP can be calculated with the equation [30]:

$$c(F_i) = \begin{cases} \infty & i \in \{0, |F| - 1\} \\ \sum_{k=1}^{M} \frac{f_k(F_{i+1}) - f_k(F_{i-1})}{\max(F, f_k) - \min(F, f_k)} & \text{otherwise} \end{cases} \tag{27}$$

assuming that F has been sorted in terms of each objective before calculation. Since the goal of crowding distance is to preserve solution diversity, the lowest and highest fitness solutions in terms of each objective are assigned a crowding distance of ∞ to ensure they are kept. NSGA-II then uses binary tournament selection [45] to select the next generation, using typical crossover and mutation operators in other GAs.

NSGA-II is a popular multi-objective GA due to its efficiency, with a time complexity of $O(MN^2)$ [30], where M is the number of objectives and N is the population size. We also favor it over other MOEAs due to its flexibility in allowing us to easily implement the local search procedure to create a memetic algorithm. Algorithms based on NSGA-II have demonstrated favorable results in solving the ADR mission planning problem [29].

5.1.2 Genetic representation

Due to the combinatorial and time-dependent nature of the ADR mission planning problem, a hybrid encoding is chosen. In our case, we need to represent a series of transfers of the form ΔV_{ijkm}, which encapsulates information of debris selection, order of removal and transfer selection.

Debris order $\boldsymbol{o}$	3	8	4	2	5	1	7	6
Transfer duration $\boldsymbol{m}$	1	1	2	1	3	5	4	$\varnothing$

Figure 6: Hybrid encoding with parallel chromosomes for order and time

We choose to represent the order of removal and transfer durations separately, with two parallel permutation and value chromosomes [46] forming a hybrid encoding. Figure 6 shows an example solution, where the $\boldsymbol{o}_i$ represents the i^{th} piece of debris to remove and $\boldsymbol{m}_i$ represents the transfer duration between $\boldsymbol{o}_i$ and $\boldsymbol{o}_{i+1}$. $\boldsymbol{m}_n$ is $\varnothing$ because there are no more transfers in the sequence. Once the solutions are found, the encoding $(\boldsymbol{o}, \boldsymbol{m})$ can easily be decoded back into a mission plan $\boldsymbol{M}(\boldsymbol{d}, \boldsymbol{t})$.

One advantage of this encoding scheme is that debris selection and path optimization can be considered separately from transfer durations. This is useful because in the beginning, solutions are spread out uniformly and exploration is more important, making optimization of transfer time pointless, and when the algorithm converges on a set of potential good solutions, exploitation is necessary to select the best possible solution and transfer time becomes significant. Hence, detaching order from time allows finer tuning of genetic operators for ideal exploration and exploitation. However, it must be noted that consideration of time-dependence cannot be completely ignored at any point, as it would easily lead to infeasible solutions to the TDOP.

Another choice that was made relevant to genetic representation was to use a fixed-length permutation encoding for the removal order, which is used for problems such as the TSP but less for the OP and its variants. The main motivation for this was to simplify the implementation and create flexibility in designing the GA operators, especially since many researchers agree that the number of debris to remove per year is fixed [4]. However, future investigations could use a variable-length encoding, especially as the local search process may be able to take advantage of it.

5.1.3 Crossover

Also known as recombination, crossover is the process of combining two "parent" individuals to create new "offspring" [47]. It should aim to retain the desirable genetic qualities of the parents for exploitation, but often also creates opportunities for exploration. Various crossover operators have been proposed for GAs, but for permutation-encoded combinatorial optimization problems, all valid solutions must obey some invariants, such as never removing the same debris twice [48]. The two strategies often used to resolve this are:

1. Using a regular crossover operator, and then *disqualifying* any invalid solutions
2. Using a regular crossover operator, and then performing a *repair* operation on the children to ensure that they are valid
3. Using a specific crossover operator suitable for permutation encoding

We choose the last option, which could use operators such as Order Crossover (OX), Cycle Crossover (CX) and Partially-Mapped Crossover (PMX) [47]. Note that as mentioned in the last section, our genetic representation allows the use of different operators for debris order and transfer times.

For debris order, the PMX operator is selected as it produces offspring which satisfy the invariants of the TDOP, and because it is a well-known, widely accepted operator for non-time-dependent combinatorial optimization problems like the TSP [48]. However, for transfer times, two-point crossover is used as transfer duration uses value encoding. Figure 7 visualizes the PMX operator and both crossover operators are described with pseudocode in Algorithm 2.

5.1.4 Mutation

The mutation operator is applied to a small subset of the population to retain genetic diversity and avoid premature convergence. Several mutation operators were considered in preliminary experimentation, including a "swap" operator and a "replace" operator. The algorithms are described in Algorithm 3.

In practice, the "replace" operator tended to perform better due to the size of the solution space. Even with random initialization of the initial population, solutions tended to converge upon selecting specific high-scoring debris and transfers with low ΔV, so mutations where new debris could be explored increased diversity significantly.

5.1.5 Avoiding premature convergence

In preliminary testing, premature convergence proved to be a severe limiting factor for all GAs when tasked with solving the ADR mission planning problem. To some degree, this is to be expected as premature convergence has been shown to be a severe limiting factor of GAs [19], which is further exacerbated by the uneven nature of the TDOP solution landscape.

As such, some measures to avoid premature convergence have been applied to all GAs, including using random initialization to create the initial population, selection of mutation operators to increase

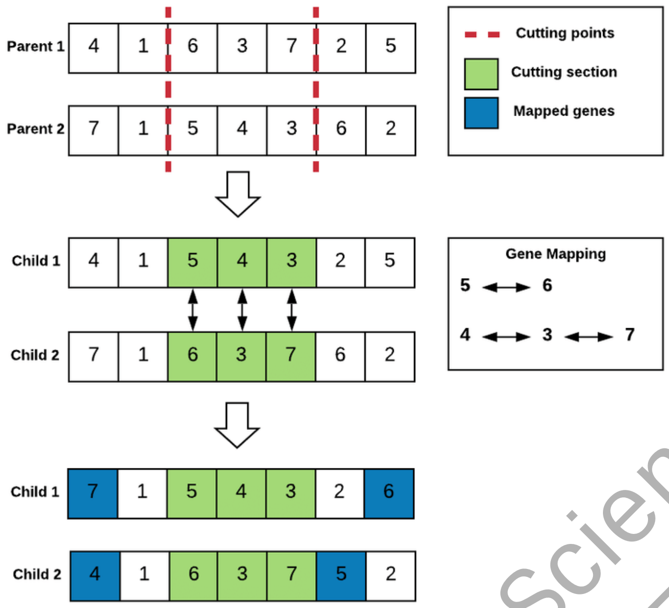


Figure 7: Partially-Mapped Crossover (PMX) [49]

Algorithm 2 Crossover operators

```
procedure TWO-POINT-CROSSOVER($p_1, p_2$)
    $r_1, r_2 \leftarrow$ RANDOM-INTEGER(1, $|x|$), RANDOM-INTEGER(1, $|x|$)
    $l, r \leftarrow \min(r_1, r_2), \max(r_1, r_2)$                ▷ Select two random indices
    $c_1, c_2 \leftarrow p_1, p_2$
    $c_1[l : r] \leftarrow p_2[l : r]$                  ▷ Copy genes from $p_2$ to $c_1$ from $l$ to $r$
    $c_2[l : r] \leftarrow p_1[l : r]$                  ▷ Copy genes from $p_1$ to $c_2$ from $l$ to $r$
    return $c_1, c_2$
end procedure
procedure PMX($p_1, p_2$)
    $c_1, c_2 \leftarrow$ TWO-POINT-CROSSOVER($p_1, p_2$)    ▷ PMX starts by performing two-point crossover
    $m \leftarrow$ mapping created from $p_1[l : r]$ and $p_2[l : r]$
    Legalize $c_1, c_2$ with $m$ by mapping repeated genes into non-repeated genes from the other parent
    return $c_1, c_2$
end procedure
```

Algorithm 3 Mutation operators

```
procedure SWAP($x$)
    $r_1, r_2 \leftarrow$ RANDOM-INTEGER(1, $|x|$), RANDOM-INTEGER(1, $|x|$)
    $l, r \leftarrow \min(r_1, r_2), \max(r_1, r_2)$                ▷ Select two random indices
    $temp \leftarrow x[l]$
    $x[l] \leftarrow x[r]$
    $x[r] \leftarrow temp$
    return $x$
end procedure
procedure REPLACE($x$)
    $i \leftarrow$ RANDOM-INTEGER(1, $|x|$)                          ▷ Select a random index
    $x[i] \leftarrow$ RANDOM-INTEGER(1, $N$)              ▷ Replace it with a random debris
    return $x$
end procedure
```

exploration, a large population size, all of which work in conjunction with each GA's specific diversification strategy, such as the use of crowding distance by NSGA-II [30].

However, precaution was also taken to avoid loss of exploitation abilities, as typical GAs use the same parameters and operators across the entire process, even when exploitation is more favourable at the end. Researchers have demonstrated that GAs could be modified to adapt their configuration during the optimization process to produce a more favorable balance of exploitation and exploration [50], which is a promising future research direction.

5.2 ADR Memetic Algorithm (ADR-MA)

We present a novel application of memetic algorithms to solving the problem of ADR mission planning, which we refer to as ADR-MA (Active Debris Removal Memetic Algorithm). It combines NSGA-II with a novel local search (LS) process capable of exploiting properties of the TDOP specific to ADR mission planning, especially the large number of transfer possibilities between debris.

Memetic algorithms are known to perform particularly well on combinatorial optimization problems, due to the rugged solution landscape and the more complicated permutation-based solution structure [51]. This quality also allows them to perform well on time-dependent problems like the TDOP, which feature a difficult interplay between discrete and continuous optimization.

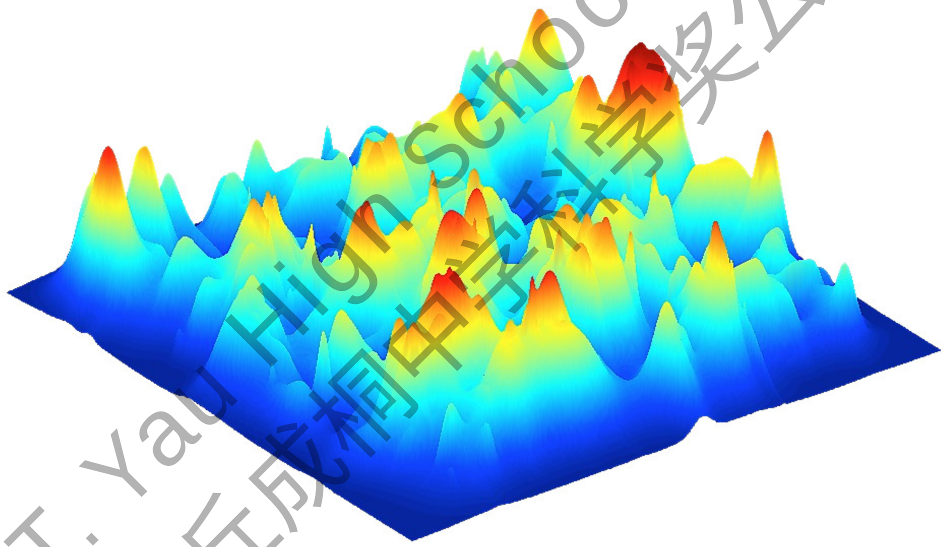


Figure 8: Example of a rugged solution landscape [52]

Figure 8 shows an example of a rugged fitness landscape. This type of solution landscape is common in combinatorial optimization problems such as ADR mission planning and is particularly difficult for EAs to solve due to premature convergence towards local maxima, indicated by the peaks in the figure.

Multiple ways have been proposed for combining EAs with LS techniques, including replacing the mutation operator entirely with LS [51], performing LS on every individual on the population [18], introducing a new parameter $P(\textit{local-search})$ for the probability a solution will be improved [43, 42], and only selecting the best solutions in a population for improvement.

We choose to perform LS on every individual in the population. While initially we chose to define a new parameter, it was found in preliminary experiments that LS had a much larger role in preserving solution diversity than we initially thought, so it was chosen to run on every individual. We also noticed that the mutation operator alone could not create enough genetic diversity to avoid premature convergence using EAs, so an elitist strategy for LS would be likely be suboptimal.

Since ADR-MA is based on NSGA-II, it has the same time complexity of $O(Mn_{\text{pop}}^2)$, assuming the local search process does not have a higher complexity class. As $M = 2$ for our formulation of ADR mission planning, the time complexity can be expressed as $O(2n_{\text{pop}}^2)$, which can then be reduced to $O(n_{\text{pop}}^2)$.

The procedure of ADR-MA is summarized in Algorithm 4, which is shown with a generic LOCAL-SEARCH procedure. We discuss the specific details of LS in ADR-MA in Section 5.3, including our novel LS algorithm which exploits the properties of the ADR problem to produce better results.

Algorithm 4 ADR-MA

```
$P_0 \leftarrow$ INITIALIZE($n_{\text{pop}}$)
$Q_0 \leftarrow$ INITIALIZE($n_{\text{pop}}$)
for $t \leftarrow 1$ to $n_t$ do
    $R_t \leftarrow P_t \cup Q_t$
    for all $x \in R_t$ do
        $f_x \leftarrow$ FITNESS($x$)
    end for
    $F \leftarrow$ NONDOMINATED-SORT($R_t$)    ▷ Sort into non-dominated fronts
    $P_{t+1} \leftarrow \varnothing$
    $i \leftarrow 0$
    while $|P_{t+1}| < n_{\text{pop}}$ do
        $c_i \leftarrow$ CROWDING-DISTANCE($F_i$)    ▷ Crowding distance is used to preserve genetic diversity
        SORT-DESCENDING($F_i, c_i$)    ▷ Keep individuals with higher crowding distance
        $P_{t+1} \leftarrow P_{t+1} \cup \{F_i[0], \ldots, F_i[\min(n - |P_t| - 1, |F_i|)]\}$    ▷ Keep individuals from better fronts
        $i \leftarrow i + 1$
    end while
    $Q_{t+1} \leftarrow \varnothing$
    while $|Q_{t+1}| < n_{\text{pop}}$ do
        $a, b \leftarrow$ BINARY-TOURNAMENT-SELECTION($P_{t+1}$)    ▷ Selects two individuals from $P_{t+1}$
        if RANDOM-PROBABILITY() $\leqslant P(\textit{crossover})$ then
            $a, b \leftarrow$ CROSSOVER($a, b$)
        end if
        if RANDOM-PROBABILITY() $\leqslant P(\textit{mutation})$ then
            $a, b \leftarrow$ MUTATION($a$), MUTATION($b$)
        end if
        $a, b \leftarrow$ LOCAL-SEARCH($a$), LOCAL-SEARCH($b$)    ▷ Perform LS to improve solutions
        $Q_{t+1} \leftarrow Q_{t+1} \cup \{a, b\}$
    end while
end for
```

5.3 Local search techniques

Local search (LS) techniques are a widely researched topic in the field of metaheuristic optimization, with prominent examples of generic LS techniques including random walks, hill climbing and local beam search. Hill climbing is a greedy local search procedure that randomly modifies solutions and evaluates them to see if the change is an improvement, keeping the best solution found [53].

However, the true strength in LS algorithms lies in the fact that compared to global search algorithms like EAs, they can be adapted to be domain and problem-specific. For the TSP, several well-known LS algorithms are widely used, such as k-opt, node insertion and edge insertion. To optimize a TSP tour, the k-opt technique involves selecting a set of k edges to disconnect from the tour, enumerating all possible permutations of those edges, and selecting the solution which produces the minimum travel cost [54]. While this is not computationally efficient enough to be viable for higher values of k, 2-opt and 3-opt are popular LS algorithms for iteratively improving TSP solutions.

Algorithm 5 Local search techniques

```
procedure HILL-CLIMBING($x$)
    $x^* \leftarrow x$                                  ▷ Keep track of the best solution
    for $i \leftarrow 1$ to $n_{\text{iter}}$ do                        ▷ Run for $n_{\text{iter}}$ iterations
        $x' \leftarrow$ MUTATION($x^*$)                      ▷ Perform a mutation on $x'$
        if $x' \succ x^*$ then
            $x^* \leftarrow x'$                        ▷ Greedily keep the best solution found
        end if
    end for
    return $x^*$
end procedure
procedure TWO-OPT($x$)
    $x^* \leftarrow x$
    for $i \leftarrow 1$ to $n_{\text{iter}}$ do
        $x' \leftarrow x^*$
        $e_1, e_2 \leftarrow$ randomly disconnect 2 edges from $x'$
        Reconnect $e_1, e_2$ to $x'$ in the other possible order
        if $x' \succ x^*$ then
            $x^* \leftarrow x'$
        end if
    end for
    return $x^*$
end procedure
```

In the LS techniques shown in Algorithm 5, the criterion for keeping a solution is whether or not it dominates the previous best solution, requiring it to be better in at least one objective and not worse in the other. This avoids accepting solutions that may be better in one objective, but significantly worse in another, leading to an overall decrease in quality; however, this does limit the exploration qualities of LS.

Furthermore, a novel LS procedure for ADR-MA is presented, based on the concept proposed by Verbeeck [55] for solving the TDOP efficiently with ant colony optimization (ACO). The algorithm is a variant of hill climbing parameterized over an integer constant $1 < k < n-1$. It begins by iterating over all contiguous subsequences of both the order and time chromosome of length k, starting from $c[1:k+1]$ and ending at $c[n-k-1:n-1]$ for $c \in \{\boldsymbol{o}, \boldsymbol{m}\}$, and calculates the ratio of total score to total ΔV within the subsequence. The subsequence with the lowest ratio of $s : \Delta V$ is removed.

Next, the algorithm selects all the debris that are not in the sequence and inserts them into the optimal position in the sequence one-by-one in random order. This position is found by considering every position the debris can be inserted into except for the first and last debris, and then finding the minimal transfer cost for this position, finally choosing that with the best transfer cost. The pseudocode for the full procedure is shown in Algorithm 6.

The entire LS procedure requires $O(n_{\text{iter}} \cdot k \cdot n \cdot m_{\max})$ calculations of ΔV, and since n_{iter}, n, m, and

Algorithm 6 Novel ADR-specific local search procedure

```
procedure ADR-Local-Search($\boldsymbol{o}, \boldsymbol{m}, n_{\text{iter}}, k$)
    $\boldsymbol{o}^*, \boldsymbol{m}^* \leftarrow \boldsymbol{o}, \boldsymbol{m}$
    for $i \leftarrow 1$ to $n_{iter}$ do
        $\textit{worst-ratio} \leftarrow \infty$
        $\textit{worst-start} \leftarrow 1$
        for $j \leftarrow 1$ to $n - k$ do                    ▻ Find the worst $k$-length subsequence
            $l, r \leftarrow j, j + k$
            $s \leftarrow \sum_{i=l}^{r}$ Score($\boldsymbol{o}_i$)
            $\Delta V \leftarrow \sum_{i=l}^{r-1}$ Delta-V($\boldsymbol{o}_i, \boldsymbol{o}_{i+1}, \boldsymbol{m}_i$)
            if $\frac{s}{\Delta V} < \textit{worst-ratio}$ then       ▻ Compare subsequences by score-to-$\Delta V$ ratio
                $\textit{worst-ratio} \leftarrow \frac{s}{\Delta V}$
                $\textit{worst-start} \leftarrow j$
            end if
        end for
        $l, r \leftarrow \textit{worst-start}, \textit{worst-start} + k$
        Delete($\boldsymbol{o}[l : r]$)
        Delete($\boldsymbol{m}[l : r]$)
        $\boldsymbol{o}', \boldsymbol{m}' \leftarrow$ Insert-Debris($\boldsymbol{o}, \boldsymbol{m}, k$)
        if $\boldsymbol{o}', \boldsymbol{m}' \succ \boldsymbol{o}^*, \boldsymbol{m}^*$ then
            $\boldsymbol{o}^*, \boldsymbol{m}^* \leftarrow \boldsymbol{o}', \boldsymbol{m}'$
        end if
    end for
    return $\boldsymbol{o}^*, \boldsymbol{m}^*$
end procedure
procedure Insert-Debris($\boldsymbol{o}, \boldsymbol{m}, k$)                      ▻ Inserts $k$ debris
    $\Omega \leftarrow \{x : 1 \leqslant x \leqslant N, x \notin \boldsymbol{o}\}$        ▻ Select all debris *not* in the sequence
    Random-Shuffle($\Omega$)
    for $i \leftarrow 1$ to $k$ do
        $\textit{best-ratio} \leftarrow \infty$
        $\textit{best-insert} \leftarrow 1$
        for $j \leftarrow 1$ to $n - 1$ do
            Consider inserting $\Omega_i$ at $\boldsymbol{o}_j$
            $s \leftarrow$ Score($\Omega_i$)
            $\Delta V^* \leftarrow \min \Delta V$ to transfer from $\boldsymbol{o}_j$ to $\boldsymbol{o}_{j+1}$
            if $\frac{s}{\Delta V^*} > \textit{best-ratio}$ then
                $\textit{best-ratio} \leftarrow \frac{s}{\Delta V^*}$
                $\textit{best-insert} \leftarrow j$
            end if
        end for
        Insert $\Omega_i$ at $\boldsymbol{o}_{\textit{best-insert}}$
        Insert the corresponding transfer duration into $\boldsymbol{m}$
    end for
    return $\boldsymbol{o}, \boldsymbol{m}$
end procedure
```

k are constant parameters with reasonably low values in most situations, the performance cost of LS is tolerable.

5.4 Simulation-based Debris Threat Index (SDTI)

We also present a novel methodology for estimating the risk proposed by space debris based on results obtained from simulations with the `cascade` [56] library developed by the ESA Advanced Concepts Team. Previous work in debris threat evaluation include indices based on the physical and orbital properties of debris, such as the Criticality of Spacecraft Index (CSI) [36] and the Hazard Criticality Index (HCI) [29]. However, an approach based on LEO environment simulations has yet to be developed.

The motivation behind this methodological contribution is to increase the applicability of theoretical approaches of solving the problem to ADR mission planning in practice, making the obtained results more realistic. A key limiting factor in existing ADR research is the lack of technology to track and provide accurate data on space debris [57], including necessary factors such as mass that are required for calculation of indices like the CSI. Researchers have circumvented these limitations by using arbitrary debris datasets [20, 21], such as GTOC 9 [58].

5.4.1 Simulation

We demonstrate our methodology on a sample dataset of the Iridium 33 debris cluster [59]. The source of the data and preprocessing techniques are discussed in more detail in Section 6.3. The dataset is used in a `cascade` simulation that predicts collisions and conjunctions between debris, as well as orbital decay of debris.

Definition 5 *A **collision** is when two debris collide while in orbit around Earth.*

Definition 6 *A **conjunction** between two debris is when the shortest distance between them is below a minimum threshold.*

The simulation is run over a time period of one year, simulating the entire LEO population. Data for 3 collisions and 2 decay events was obtained, however none involved the Iridium 33 cluster. A total of 1.05×10^7 conjunction events with a distance of $0 < d \leqslant 5$ km were recorded, of which 3.13×10^5 events involved the Iridium 33 debris.

5.4.2 Fragmentation damage estimation

Space debris that threatens to create a large amount of fragmentation should be prioritized for removal, as mitigating the Kessler syndrome requires avoiding further fragmentation causing a cascading effect [2]. We decide to use the total momentum involved in a collision to estimate the potential damage it may cause, as the momentum of the colliding debris will be transferred into the LEO environment due to the law of conservation of momentum.

Using the formula for momentum $p = mv$ requires the mass and velocity of each object. Because mass is not tracked in the Satellite Catalog (SATCAT), we use an approximation based on the radar cross section (RCS) of debris objects. RCS is given as an area in m^2, so it is first converted to a radius in m:

$$r = \sqrt{\frac{\text{RCS}}{\pi}} \tag{28}$$

Some SATCAT objects do not have an exact RCS, but are instead categorized into "SMALL", "MEDIUM" and "LARGE", which we assume to be $r = 0.15$ m, $r = 0.55$ m and $r = 2.00$ m respectively. We then use two assumptions to relate the mass of a debris object to its radius, which are:

1. Debris objects are spherical in shape
2. The density of debris is constant

Neither assumption is valid in practice, but for the task of estimating debris threat level they can be used for approximations. Under these assumptions, mass is directly proportional to the cube of the radius:

$$V = \frac{4}{3}\pi r^3 \tag{29}$$
$$\rho = \frac{m}{V} \tag{30}$$
$$m = \frac{4}{3}\pi \rho r^3 \tag{31}$$
$$m \propto r^3 \tag{32}$$

Again, this formula cannot give the mass of debris, but can be used to compare the mass of two debris based on their radii through proportionality. The total momentum in a conjunction between a and b can be calculated as:

$$m_a = r_a^3 \tag{33}$$
$$m_b = r_b^3 \tag{34}$$
$$p_x = m_a v_x(a) + m_b v_x(b) \tag{35}$$
$$p_y = m_a v_y(a) + m_b v_y(b) \tag{36}$$
$$p_z = m_a v_z(a) + m_b v_z(b) \tag{37}$$
$$p = \sqrt{p_x^2 + p_y^2 + p_z^2} \tag{38}$$

5.4.3 Collision probability

Simulations like `cascade` propagate the orbits of debris from a series of initial conditions to create predictions about the future [56]. Any error in the initial conditions would grow in magnitude as the orbits are propagated, so although the simulation may only predict a conjunction, it is possible that a collision occurs instead. This happened to Iridium 33, which did not perform avoidance maneuvers since the closest conjunction distance was predicted to be around 117 m [59].

The distance d between a and b during a conjunction affects the probability of collision. We model the error with a sphere of radius ϵ around each debris, and the probability of collision during a conjunction to be proportional to the intersecting volumes $V^{(2)}$ of the error spheres [60]. There are also more sophisticated models for predicting collisions in orbit [61], but we use a simpler model in this investigation before refining the methodology in future work.

$$V^{(2)} = \frac{\pi}{12d}(r_1 + r_2 - d)^2(d^2 + 2d(r_1 + r_2) - 3(r_1 - r_2)^2), d \geqslant r_1 + r_2 \tag{39}$$
$$r_1 = r_2 = \epsilon \tag{40}$$
$$V^{(2)} = \frac{\pi}{12d}(2\epsilon - d)^2(d^2 + 4d\epsilon) \tag{41}$$
$$V_{\text{error}} = \frac{4}{3}\pi r^3 \tag{42}$$
$$P = \begin{cases} 1 & d = 0 \\ \frac{V^{(2)}}{V_{\text{error}}} & \text{otherwise} \end{cases} \tag{43}$$

5.4.4 Time of conjunction

A simple exponential decay model is used to decrease the prioritization of conjunctions that occur further into the future.

$$T = 2e^{-t} \tag{44}$$

where t is the time of conjunction after the start of simulation in years.

5.4.5 Threat level

Finally, the threat level or criticality of a conjunction is given by:

$$c = p \times P \times T \tag{45}$$

The total score, threat level or criticality of a piece of space debris is the sum of the values of c in which it is involved. After calculation, score values are normalized within the range of 0–100 through min-max scaling. Figure 9 shows how debris scores are distributed after data processing.

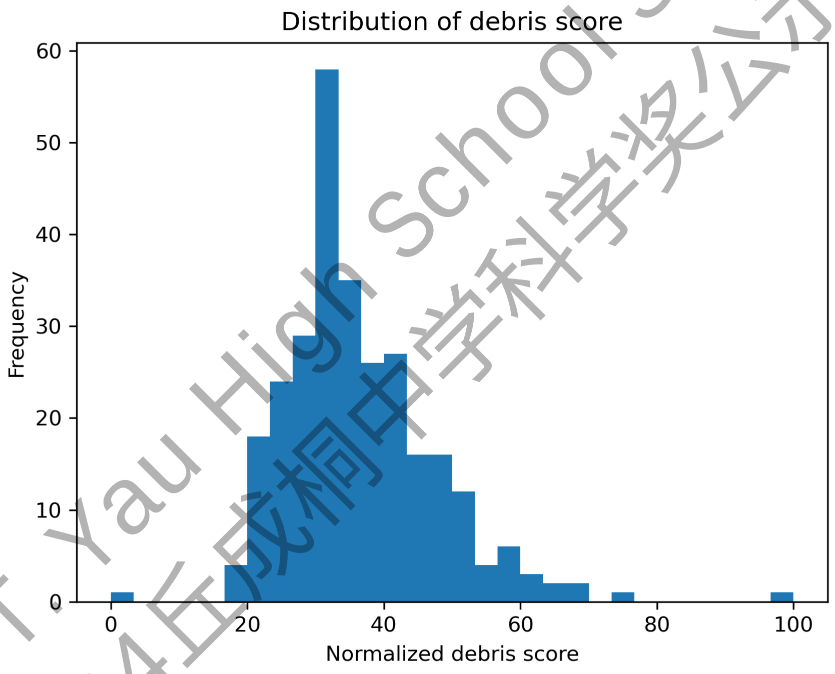


Figure 9: Score distribution for the Iridium 33 cluster

We name our debris threat evaluation methodology the Simulation-based Debris Threat Index (SDTI). We believe that developing this model is a first step towards reliable and practical simulation-based debris scoring; however, there is still much potential for improvement in terms of the model itself.

6 Experiment

6.1 Comparison

We compare our proposed memetic algorithm for ADR mission planning with several other state-of-the-art approaches. As NSGA-II provides a framework that forms the basis of ADR-MA, it intuitively is a good baseline comparison.

Strength Pareto Evolutionary Algorithm 2 (SPEA2) [62] was proposed as an improvement to the original SPEA MOEA, and is widely regarded as a benchmark for MOEA experiments. It employs an "archive" accompanying the typical population, copying the Pareto set to the archive without duplication on each iteration, and is known for its ability to maintain a well-distributed set of non-dominated solutions, achieved through a unique fitness assignment scheme and its archive-based approach.

Furthermore, we also compare ADR-MA to multi-objective memetic algorithms (MOMAs) [51] based on NSGA-II that use other local search techniques described earlier, namely hill climbing and 2-opt. Table 1 summarizes the algorithms we compare in our experiment.

Algorithm	Description
ADR-MA	Proposed algorithm for ADR mission planning.
NSGA-II	MOEA known for speed and elitist selection process.
SPEA2	MOEA known for its fitness assignment scheme and archiving of Pareto optimal solutions.
MOMA (Hill Climbing)	MOMA based on NSGA-II combined with Hill Climbing local search.
MOMA (2-opt)	MOMA based on NSGA-II combined with TSP 2-opt local search.

Table 1: Summary of algorithms compared in the experiment

All algorithms were implemented using the `jMetalPy` multi-objective metaheuristic optimization library [63]. Algorithms NSGA-II and SPEA2 were entirely implemented with `jMetalPy` components, while MOMA and ADR-MA were implemented by extending the algorithms provided by the framework.

6.2 Experiment design

The independent variable of the experiment is the algorithm used to solve the ADR mission planning problem. The algorithms in Table 1 will be compared. To minimize the effect of random error caused by pseudorandomness inherently present in EAs, each algorithm will be run three times and the Pareto set across all runs of each algorithm will be used for comparison.

The dependent variable of the experiment will be the quality of solutions produced by the algorithms, measured by quantitative comparisons of the set of non-dominated solutions produced by each algorithm, as well as qualitative observations on the populations generated by each algorithm at the final generation. The results will be visualized on a Pareto front in a two-dimensional objective space, and performance indicators for evaluation of MOP solutions will be used for quantitative comparisons.

Aside from changing the algorithm, other GA/EA parameters will be set as control variables and kept constant across all comparisons where possible, as different algorithms inevitably require different parameter configurations. These parameters are listed in Table 2.

While the GAs listed in this report are described with termination criteria based on a maximum number of generations, it has been shown that this can easily lead to unfair comparisons, due to the fact that some EAs may perform more evaluations than others even when the number of generations is the same [64]. Hence, our parameters are defined in terms of maximum number of *evaluations* n_{eval}, which occur when the EA computes the fitness of an individual using the objective function. `jMetalPy` has built-in support for number of evaluations as a termination criterion [63].

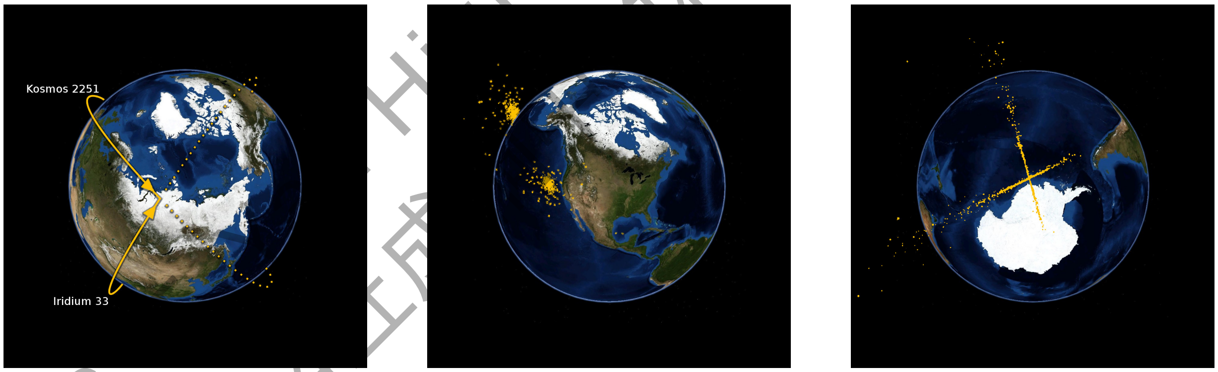
Parameter	Algorithm	Description	Value
n_{eval}	All	Number of evaluations	10^5
n_{pop}	All	Population size	1000
$P(crossover)$	All	Probability of crossover	0.8
$P(mutation)$	All	Probability of mutation	0.01
$P(local\text{-}search)$	MOMA	Probability of local search	0.05
n_{iter}	MOMA, ADR-MA	Number of local search iterations	50
k	ADR-MA	Length of the contiguous debris subsequence replaced during local search	2

Table 2: EA parameters

A higher mutation rate of up to 10% was experimented with in an attempt to avoid premature convergence, but it did not have any affect on the convergence rate of the algorithms, so the standard recommended rate of 1% [19] was used instead to avoid affecting exploitation.

6.3 Dataset

In 2009, the operational Iridium 33 communications satellite accidentally collided with the defunct Cosmos 2251 communications satellite [59] — the first time two satellites have collided in orbit. The US Space Surveillance Network (SSN) has catalogued and tracked 386 debris originating from Iridium 33 and 927 debris from Cosmos 2251, a fraction of which has since decayed, re-entered the atmosphere. Figure 10 depicts the collision and debris fields generated following the collision.



(a) Point of collision (b) Debris fields after 20 minutes (c) Debris fields after 50 minutes

Figure 10: The Iridium 33 and Cosmos 2251 collision [65]

In our experiment, we investigate the Iridium 33 debris field, a moderately-sized debris cluster. The data is provided by CelesTrak[1] and Space-Track[2], most of which originates from the SSN. The data is preprocessed by converting it from the SATCAT format into a format readable by the `cascade` simulator, for running simulations to create the SDTI. After preprocessing, the dataset contains orbital elements, debris characteristics, and SDTI scores for 285 Iridium 33 debris. Table 3 describes the dataset and parameters of the ADR problem.

[1]`https://celestrak.org`
[2]`https://space-track.org`

Parameter	Description	Value
N	Number of debris in cluster	285
n	Target number of debris to collect	20
$t_{\max}$	Maximum length of mission (days)	300
N_t	Number of time epochs	100
Δt	Length of one time epoch (days)	3
$m_{\max}$	Maximum length of singular transfer (epochs)	5
$\Delta V_{\max}$	Maximum ΔV $(\mathrm{m\,s^{-1}})$	1.5×10^5

Table 3: Dataset and ADR problem parameters

6.4 Performance indicators

We use several performance indicators developed for evaluation of solution quality for MOPs, as surveyed in [66] and used in [51]. $X \subset \mathbf{\Omega}$ is used to denote a set of feasible solutions.

Before evaluation with the following performance metrics, the objective values including total score and total ΔV were normalized between 1.0 and 2.0 using min-max scaling. The directions of the objectives were also removed, by making 1.0 indicate the best performance among all solutions and 2.0 the worst performance, regardless of where the objective to be maximized (score) or minimized (ΔV).

6.4.1 Hypervolume indicator

The *hypervolume indicator* I_H is a measure of the size of the hypervolume in the objective space dominated by the set of solutions X, where the *hypervolume* can be thought of as a generalization of volume to higher dimensions. In our formulation, the MOP is bi-objective, so the hypervolume is also an area. Figure 11 visualizes the hypervolume of two Pareto fronts.

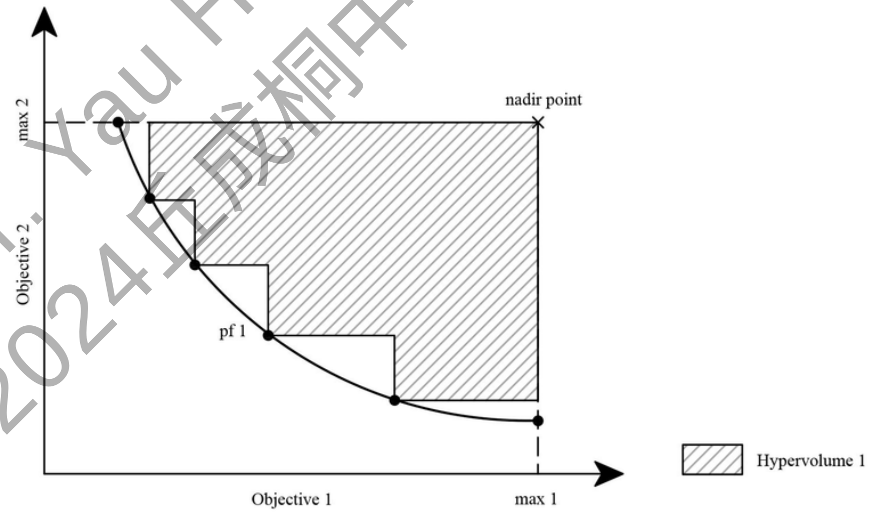


Figure 11: Hypervolume of two Pareto fronts [67]

Since our solutions are normalized between 1.0 and 2.0, the reference or nadir point is at (2.0, 2.0). Algorithm 7 shows how I_H is calculated.

Algorithm 7 Calculation of I_H

```
$\textsc{Sort}(X, f_1)$
$h(X) \leftarrow f_1(x_1) \cdot f_2(x_1)$
for $i \leftarrow 2$ to $|X|$ do
    $h(X) \leftarrow h(X) + (f_1(x_i) - f_1(x_{i-1})) \cdot f_2(x_i)$
end for
$I_H(X) \leftarrow 4.0 - h(x)$
```

The area under the Pareto fronts is calculated, and is subtracted from the area to the reference point $2.0 \times 2.0 = 4.0$. A larger value of I_H indicates that X dominates a larger area in the objective space and hence is a better solution set.

6.4.2 Unary epsilon indicator

The *unary epsilon indicator* I_ϵ is a metric an extension of the dominance relation, known as the ϵ-dominance relation. It indicates how close a solution front is to the Pareto front.

Definition 7 *A solution x^* **ϵ-dominates** another solution x, i.e. $x^* \succ_\epsilon x$ iff $\forall f \in F : \epsilon \cdot f(x^*) \leqslant f(x) \wedge \exists f \in F : \epsilon \cdot f(x^*) < f(x)$ where F is the set of objectives in a minimization MOP and $x, x^* \in \mathbf{\Omega}$.*

The value for I_ϵ can then be calculated as follows:

$$I_\epsilon(X, R) = \inf_{\epsilon \in \mathbb{R}} \{\forall x \in R : \exists x^* \in X, x^* \succ_\epsilon x\} \tag{46}$$

A smaller value of I_ϵ indicates that the solution set X is closer to a given reference Pareto front R and hence is better. Since a reference front R is required for this indicator, a Pareto front approximation was computed by combining all non-dominated solutions found by all algorithms. Without the true Pareto front available for comparison, this is the best approximation that we could use.

6.4.3 Spacing indicator

The *spacing indicator* I_S describes the distribution of the solution set X in the objective space, given by the calculation:

$$I_S(X) = \sqrt{\frac{1}{|X| - 1} \sum_{i=1}^{|X|} (D(x_i) - \bar{D})} \tag{47}$$

where $D(x)$ is the Euclidean distance between x and its nearest neighbor in the objective space and it is assumed that $|X| \geqslant 2$. A smaller value of I_S indicates a more uniformly distributed set and hence is better.

6.4.4 Range cover indicator

The *range cover indicator* I_R represents the spread of the set X in the objective space, and is calculated with the formula:

$$I_R(X) = \frac{1}{M} \sum_{i=1}^{M} \left(\max_{x \in X} f_i(x) - \min_{x \in X} f_i(x) \right) \tag{48}$$

for a MOP with M objectives. A higher value of I_R implies a wider spread and is better.

7 Results

7.1 Solution quality

Figure 12 shows the solutions produced by the algorithms in the form of Pareto fronts in an objective space. The fronts are convex towards the bottom-right direction due to the direction of the objectives, with score (x-axis) to be maximized and ΔV (y-axis) to be minimized. The lines drawn represent the Pareto front and the population of individual solutions is plotted as a scatter plot on the objective space.

It is clear from the figure that ADR-MA exhibits superior performance on the task of ADR mission planning when compared to the other algorithms, with its Pareto front positioned to the right of and below most of the other fronts. This is likely attributed to the ability of the ADR-MA LS procedure to properly perform exploration and exploitation, a weakness of the other EAs when solving this problem.

Figure 13 depicts the distribution of the final population of each algorithm over all three trials in the experiment. Evidence of premature convergence is visible in all algorithms except ADR-MA, and is speculated to be the primary reason why they perform worse. This is clearly visible in the population distributions of the algorithms at the final generation, where all algorithms except ADR-MA have converged on multiple local maxima in both score and ΔV due to selecting the same debris or orbital transfers, indicative of a lack of exploration.

Apart from ADR-MA, all of the other EAs appear to perform similarly, with similar Pareto fronts and the issue of premature convergence. The 2-opt MOMA variant appears to have benefited from the LS process in terms of exploitation as it achieves marginally better solutions, but still faces the same issue of lack of exploration.

7.2 Performance indicators

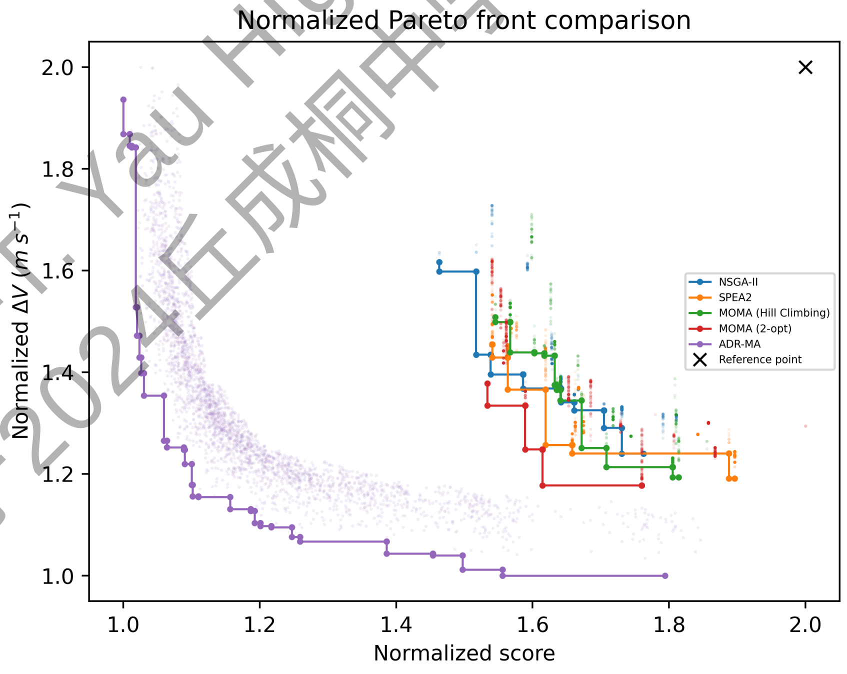


Figure 14: Normalized Pareto front comparison

As described in Section 6.4, min-max scaling was used to normalized the objective values between 1.0 (best) and 2.0 (worst). Figure 14 depicts the normalized Pareto fronts used for calculation of the perfor-

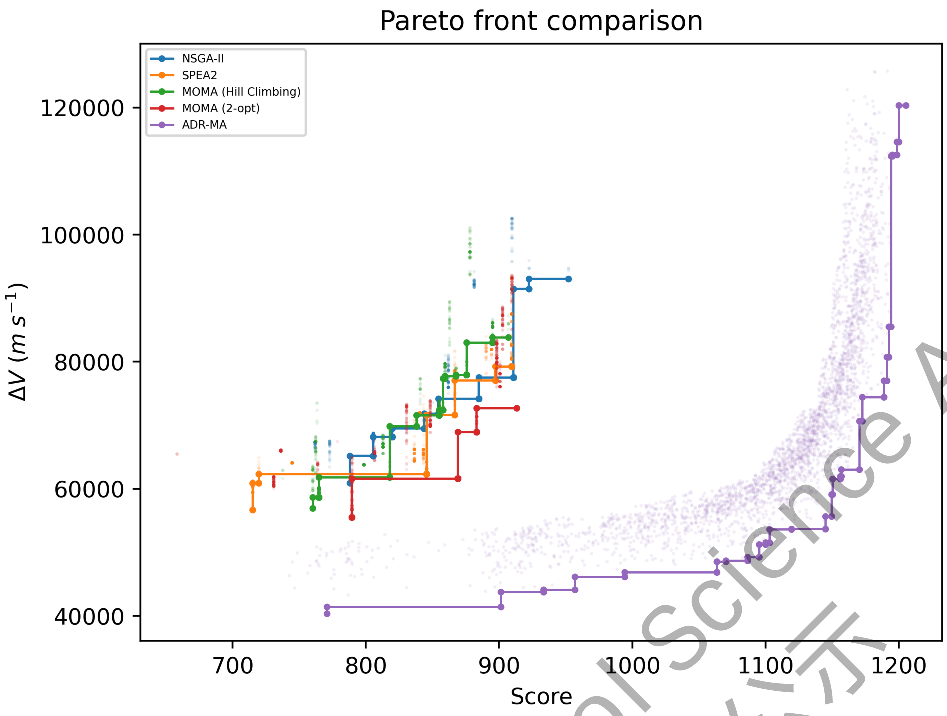


Figure 12: Comparison of Pareto fronts between algorithms

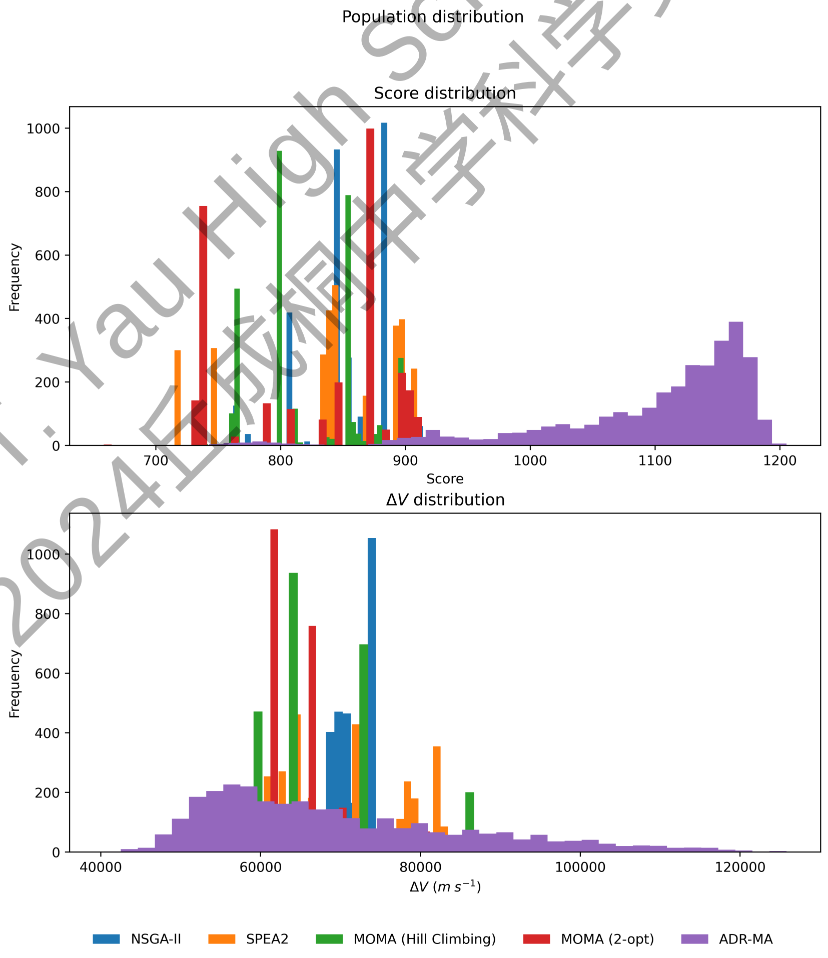


Figure 13: Comparison of final population distribution between algorithms

mance indicators, which are convex towards the origin as both normalized objectives are to be minimized. The reference point for the hypervolume indicator I_H is marked on the objective space.

	Hypervolume (I_H)	Unary epsilon (I_ϵ)	Spacing (I_S)	Range cover (I_R)
NSGA-II	2.242	0.463	0.109	16161.145
SPEA2	2.206	0.541	0.172	11363.503
MOMA (Hill Climbing)	2.199	0.545	0.067	13532.226
MOMA (2-opt)	2.219	0.558	**0.017**	8634.158
ADR-MA	**3.088**	**0.000**	0.162	**40209.826**

Table 4: Algorithm performance indicators

Table 4 contains the values of the four proposed performance indicators, with the best-performing algorithm indicator value displayed in bold. ADR-MA is observed to have better values for the hypervolume indicator I_H and significantly better values for the unary epsilon indicator I_ϵ and range cover indicator I_R. Since ADR-MA makes up the entirety of the Pareto front approximation, its I_ϵ values is 0. However, ADR-MA performs poorly on the spacing indicator I_S, with NSGA-II and both MOMA variants achieving better results. Figure 15 depicts the performance indicators across algorithms visually.

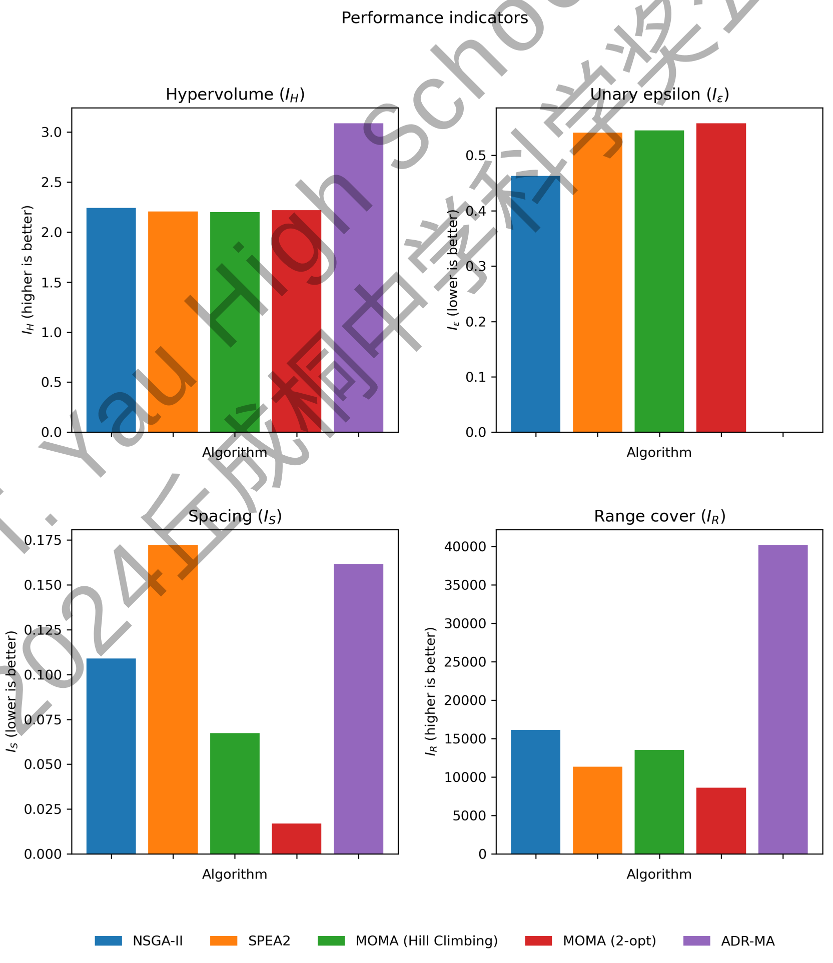


Figure 15: Comparison of performance indicators between algorithms

7.3 Discussion

The results and performance of ADR-MA were presented, showing the capability of ADR-MA to perform well on the task of ADR mission planning and suggesting the overall suitability of MAs over traditional EAs for solving combinatorial optimization problems.

It is evident that the largest limiting factor of an EA-based approach is the balance between exploration and exploitation of the solution space, where insufficient exploration leads to premature convergence and insufficient exploitation leading to poor solution quality. Although it was shown that MAs based on generic LS procedures such as hill climbing and 2-opt are able to marginally improve this balance, the performance of ADR-MA clearly indicates the high potential of domain-specific LS algorithms designed with the characteristics of the problem in mind.

8 Conclusion

8.1 Summary

In this report, the problem of ADR mission planning was discussed. Relevant background topics and existing methodology for ADR mission planning was surveyed. A problem statement based on a multi-objective variant of the Time-Dependent Orienteering Problem (TDOP), a discretized representation of time and an impulsive transfer strategy between debris was presented.

Various approaches were proposed to solve the ADR mission planning problem, including multi-objective evolutionary algorithms (MOEAs) such as NSGA-II, a novel application of memetic algorithms with local search techniques including hill climbing and the Travelling Salesman Problem (TSP) 2-opt search, as well as a novel ADR-specific local search procedure. These algorithms were compared against the state-of-the-art in MOEAs, TDOP solutions and ADR mission planning, and were shown to consistently outperform accepted approaches when measured with multiple performance indicators.

A novel methodology for quantification of the threat level posed by space debris with `cascade` simulations of the LEO environment was developed and used in the ADR mission planning algorithms. The method was shown to be viable in practice for analysis and mission planning for the Iridium 33 cluster of debris, and the further research necessary to improve and refine the simulation-based methodology for wider application to ADR mission planning has been discussed.

8.2 Potential impact

This study discusses a wide range of topics, including ADR, evolutionary computation and multi-objective combinatorial optimization. The potential impact of this study on the field of space sustainability may include an improvement on the state-of-the-art in ADR mission planning through memetic algorithms such as ADR-MA, as well as the introduction and refinement of simulation-based techniques for prioritizing debris to remove even in the absence of certain data on space debris.

This study highlights a significant limitation of evolutionary computation and EAs, which is the challenge in finding a balance between exploration and exploitation suitable for the problem the EA is used to solve. It also suggests that memetic algorithms are a more suitable metaheuristic for complicated combinatorial optimization problems such as ADR mission planning and that MAs are an effective methodology for combining the strengths of EAs with those of local search.

Furthermore, the MA-based methodology could be further developed to make impactful contributions to the wider field of combinatorial optimization, particularly in problems where time-dependence is a significant obstacle for application of typical combinatorial optimization algorithms, or when the solution space features a combination of discrete and continuous structures. This could also take the form of more problem-specific memetic algorithms and local search techniques for other Orienteering Problem or TDOP variants. Given that time-dependence is often the case in real-world scenarios, further developments in this field may also lead to practical benefit.

8.3 Limitations

One major limitation to this study is that its scope was constrained to EAs and EA-based approaches such as MAs. While EAs have been shown to be suitable for combinatorial optimization [8], the trade-off between exploration and exploitation becomes more difficult to achieve on more complex problems such as the TDOP, which could be described as a weakness of EAs as a metaheuristic in general. Although MAs have been shown to outperform other metaheuristics such as ACO and EAs on the TDOP [51], comparison against a wider range of algorithms may benefit this study.

Furthermore, the comparison presented in our results are based on a specific debris dataset — the Iridium 33 cluster. While it is a moderately-sized dataset and shows the scalability of our methodology to larger-scale scenarios, evaluation against more debris datasets may increase the reliability of the results and ensure that properties specific to Iridium 33 do not affect the performance of ADR-MA or the other EAs compared.

Another limitation of this report is the limited evaluation of the SDTI, which is naturally hard to compare as, to the best of our knowledge, similar methodology has not yet been developed in other research. Evaluating the SDTI on a simulation is susceptible to confirmation bias and limitations in openly-available data prevent comparison with the CSI [36] and HCI [29]. However, we still believe that the SDTI is a valuable methodological contribution and is a good first step towards more simulation-based LEO space debris threat estimation.

8.4 Further investigation

Further investigations in this topic may consider alternative novel problem formulations for ADR mission planning, such as that presented by Missel and Mortari [31], investigation of other heuristic techniques for ADR mission planning such as ACO [55] or SA [24], a variable-length hybrid encoding, methods to avoid premature convergence [50], continued development of simulation-based methodology for evaluating the threat level of space debris, or investigation of other local search techniques in memetic algorithms, such as shortest-path algorithms such as Dijkstra's algorithm or the A* graph search algorithm [20].

References

[1] *IADC Space Debris Mitigation Guidelines.* Sept. 2007. URL: https://www.unoosa.org/documents/pdf/spacelaw/sd/IADC-2002-01-IADC-Space_Debris-Guidelines-Revision1.pdf.

[2] Donald J. Kessler and Burton G. Cour-Palais. "Collision frequency of artificial satellites: The creation of a debris belt". en. In: *Journal of Geophysical Research: Space Physics* 83.A6 (June 1978), pp. 2637–2646. ISSN: 0148-0227. DOI: 10.1029/JA083iA06p02637. URL: https://agupubs.onlinelibrary.wiley.com/doi/10.1029/JA083iA06p02637 (visited on 08/03/2024).

[3] *ESA's Annual Space Environment Report.* LOG GEN-DB-LOG-00288-OPS-SD. European Space Agency, July 2024. URL: https://www.sdo.esoc.esa.int/environment_report/Space_Environment_Report_latest.pdf.

[4] J. -C. Liou, N. L. Johnson, and N. M. Hill. "Controlling the growth of future LEO debris populations with active debris removal". In: *Acta Astronautica* 66.5 (Mar. 2010), pp. 648–653. ISSN: 0094-5765. DOI: 10.1016/j.actaastro.2009.08.005. URL: https://www.sciencedirect.com/science/article/pii/S0094576509003981 (visited on 08/04/2024).

[5] Christophe Bonnal, Jean-Marc Ruault, and Marie-Christine Desjean. “Active debris removal: Recent progress and current trends”. In: *Acta Astronautica* 85 (Apr. 2013), pp. 51–60. ISSN: 0094-5765. DOI: 10.1016/j.actaastro.2012.11.009. URL: https://www.sciencedirect.com/science/article/pii/S0094576512004602 (visited on 08/04/2024).

[6] PengYuan Zhao, JinGuo Liu, and ChenChen Wu. “Survey on research and development of on-orbit active debris removal methods”. In: *Science China Technological Sciences* 63.11 (2020), pp. 2188–2210.

[7] Edwin KP Chong and Stanislaw H Żak. *An introduction to optimization*. Vol. 75. John Wiley & Sons, 2013.

[8] Heinz Mühlenbein, Martina Gorges-Schleuter, and Ottmar Krämer. “Evolution algorithms in combinatorial optimization”. In: *Parallel computing* 7.1 (1988), pp. 65–85.

[9] Rajesh Matai, Surya Prakash Singh, and Murari Lal Mittal. “Traveling salesman problem: an overview of applications, formulations, and solution approaches”. In: *Traveling salesman problem, theory and applications* 1.1 (2010), pp. 1–25.

[10] Matej Črepinšek, Shih-Hsi Liu, and Marjan Mernik. “Exploration and exploitation in evolutionary algorithms: A survey”. In: *ACM computing surveys (CSUR)* 45.3 (2013), pp. 1–33.

[11] Yair Censor. “Pareto optimality in multiobjective problems”. In: *Applied Mathematics and Optimization* 4.1 (1977), pp. 41–59.

[12] Kalyanmoy Deb, Karthik Sindhya, and Jussi Hakanen. “Multi-objective optimization”. In: *Decision sciences*. CRC Press, 2016, pp. 161–200.

[13] *Français : Optimum de Pareto*. May 2006. URL: https://commons.wikimedia.org/wiki/File:Front_pareto.svg (visited on 08/14/2024).

[14] Aimin Zhou et al. “Multiobjective evolutionary algorithms: A survey of the state of the art”. In: *Swarm and evolutionary computation* 1.1 (2011), pp. 32–49.

[15] Musatafa Abbas Albadr et al. “Genetic algorithm based on natural selection theory for optimization problems”. In: *Symmetry* 12.11 (2020), p. 1758.

[16] Melanie Mitchell. *An introduction to genetic algorithms*. MIT press, 1998.

[17] David W Corne and Michael A Lones. “Evolutionary algorithms”. In: *arXiv preprint arXiv:1805.11014* (2018).

[18] Melanie Mitchell. “LD Davis, handbook of genetic algorithms”. In: (1998).

[19] Aymeric Vie, Alissa M Kleinnijenhuis, and Doyne J Farmer. “Qualities, challenges and future of genetic algorithms: a literature review”. In: *arXiv preprint arXiv:2011.05277* (2020).

[20] Lorenzo Federici, Alessandro Zavoli, and Guido Colasurdo. “On the use of A* search for active debris removal mission planning”. In: *Journal of Space Safety Engineering* 8 (2021), pp. 245–255. URL: https://api.semanticscholar.org/CorpusID:237692711.

[21] Danilo Zona et al. “Evolutionary Optimization for Active Debris Removal Mission Planning”. In: *IEEE Access* 11 (2023), pp. 41019–41033. DOI: 10.1109/ACCESS.2023.3269305.

[22] Tianjiao Zhang et al. “Ant colony optimization-based design of multiple-target active debris removal mission”. In: *Transactions of the Japan Society for Aeronautical and Space Sciences* 61.5 (2018), pp. 201–210.

[23] Hong-Xin Shen et al. “Optimization of Active Debris Removal Missions with Multiple Targets”. In: *Journal of Spacecraft and Rockets* 55 (Oct. 2017), pp. 1–9. DOI: 10.2514/1.A33883.

[24] Lorenzo Federici, Alessandro Zavoli, and Guido Colasurdo. “A time-dependent TSP formulation for the design of an active debris removal mission using simulated annealing”. In: *arXiv preprint arXiv:1909.10427* (2019).

[25] Yingjie Xu et al. “Active Debris Removal Mission Planning Method Based on Machine Learning”. In: *Mathematics* 11.6 (2023), p. 1419.

[26] Jianan Yang et al. “A reinforcement learning scheme for active multi-debris removal mission planning with modified upper confidence bound tree search”. In: *IEEE Access* 8 (2020), pp. 108461–108473.

[27] Dalal Madakat, Jérôme Morio, and Daniel Vanderpooten. “Biobjective planning of an active debris removal mission”. In: *Acta Astronautica* 84 (2013), pp. 182–188.

[28] Yong Liu et al. “Multi-objective optimal preliminary planning of multi-debris active removal mission in LEO”. In: *Science China Information Sciences* 60 (2017), pp. 1–10.

[29] Jin Haeng Choi, Chandeok Park, and Jinah Lee. “Mission planning for active removal of multiple space debris in low Earth orbit”. In: *Advances in Space Research* 73.9 (2024), pp. 4800–4812.

[30] Kalyanmoy Deb et al. “A fast and elitist multiobjective genetic algorithm: NSGA-II”. In: *IEEE transactions on evolutionary computation* 6.2 (2002), pp. 182–197.

[31] Jonathan Missel and Daniele Mortari. “Path optimization for Space Sweeper with Sling-Sat: A method of active space debris removal”. In: *Advances in Space Research* 52.7 (2013), pp. 1339–1348.

[32] J-C Liou et al. “LEGEND–a three-dimensional LEO-to-GEO debris evolutionary model”. In: *Advances in Space Research* 34.5 (2004), pp. 981–986.

[33] Heiner Klinkrad et al. “An introduction to the 1997 ESA MASTER Model”. In: *EUROPEAN SPACE AGENCY-PUBLICATIONS-ESA SP* 393 (1997), pp. 217–224.

[34] Marco M Castronuovo. “Active space debris removal—A preliminary mission analysis and design”. In: *Acta Astronautica* 69.9-10 (2011), pp. 848–859.

[35] Adam E White and Hugh G Lewis. “An adaptive strategy for active debris removal”. In: *Advances in Space Research* 53.8 (2014), pp. 1195–1206.

[36] A Rossi, GB Valsecchi, and EM Alessi. “The criticality of spacecraft index”. In: *Advances in Space Research* 56.3 (2015), pp. 449–460.

[37] Pieter Vansteenwegen, Wouter Souffriau, and Dirk Van Oudheusden. “The orienteering problem: A survey”. In: *European Journal of Operational Research* 209.1 (2011), pp. 1–10.

[38] Harvey M Salkin and Cornelis A De Kluyver. “The knapsack problem: a survey”. In: *Naval Research Logistics Quarterly* 22.1 (1975), pp. 127–144.

[39] Aldy Gunawan, Hoong Chuin Lau, and Pieter Vansteenwegen. “Orienteering Problem: A survey of recent variants, solution approaches and applications”. en. In: *European Journal of Operational Research* 255.2 (Dec. 2016), pp. 315–332. ISSN: 03772217. DOI: 10.1016/j.ejor.2016.04.059. URL: https://linkinghub.elsevier.com/retrieve/pii/S037722171630296X (visited on 08/04/2024).

[40] Max Cerf. “Multiple space debris collecting mission—debris selection and trajectory optimization”. In: *Journal of Optimization Theory and Applications* 156.3 (2013), pp. 761–796.

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

[42] Pablo Moscato et al. “Memetic algorithms”. In: *New optimization techniques in engineering* 141 (2004), pp. 53–85.

[43] Ferrante Neri and Carlos Cotta. “Memetic algorithms and memetic computing optimization: A literature review”. In: *Swarm and Evolutionary Computation* 2 (2012), pp. 1–14.

[44] Natalio Krasnogor and James Smith. “A tutorial for competent memetic algorithms: model, taxonomy, and design issues”. In: *IEEE transactions on Evolutionary Computation* 9.5 (2005), pp. 474–488.

[45] Tobias Blickle. “Tournament selection”. In: *Evolutionary computation* 1 (2000), pp. 181–186.

[46] Anit Kumar. “Encoding schemes in genetic algorithm”. In: *International Journal of Advanced Research in IT and Engineering* 2.3 (2013), pp. 1–7.

[47] Padmavathi Kora and Priyanka Yadlapalli. “Crossover operators in genetic algorithms: A review”. In: *International Journal of Computer Applications* 162.10 (2017).

[48] Göktürk Üçoluk. “Genetic algorithm solution of the TSP avoiding special crossover and mutation”. In: *Intelligent Automation & Soft Computing* 8.3 (2002), pp. 265–272.

[49] Maritzol Tenemaza et al. “Improving itinerary recommendations for tourists through metaheuristic algorithms: an optimization proposal”. In: *IEEE Access* 8 (2020), pp. 79003–79023.

[50] Martin Serpell and James E Smith. “Self-adaptation of mutation operator and probability for permutation representations in genetic algorithms”. In: *Evolutionary Computation* 18.3 (2010), pp. 491–514.

[51] Yi Mei, Flora D Salim, and Xiaodong Li. “Efficient meta-heuristics for the multi-objective time-dependent orienteering problem”. In: *European Journal of Operational Research* 254.2 (2016), pp. 443–457.

[52] Bjørn Østman. “2D rugged fitness landscape”. In: (Nov. 2013). DOI: `10.6084/m9.figshare.848622.v1`. URL: `https://figshare.com/articles/figure/2D_rugged_fitness_landscape/848622`.

[53] Bart Selman and Carla P Gomes. “Hill-climbing search”. In: *Encyclopedia of cognitive science* 81.333-335 (2006), p. 10.

[54] Matthias Englert, Heiko Röglin, and Berthold Vöcking. “Worst case and probabilistic analysis of the 2-Opt algorithm for the TSP”. In: *Algorithmica* 68.1 (2014), pp. 190–264.

[55] Cédric Verbeeck et al. “A fast solution method for the time-dependent orienteering problem”. In: *European Journal of Operational Research* 236.2 (2014), pp. 419–432.

[56] Francesco Biscani and Dario Izzo. *cascade*. Version 0.1.8. European Space Agency, 2023.

[57] Giacomo Muntoni et al. “Crowded space: a review on radar measurements for space debris monitoring and tracking”. In: *Applied Sciences* 11.4 (2021), p. 1364.

[58] Dario Izzo and Marcus Märtens. “The Kessler run: On the design of the GTOC9 challenge”. In: *Acta Futura* 11.1 (2018), pp. 11–24.

[59] TS Kelso et al. “Analysis of the Iridium 33-Cosmos 2251 collision”. In: *Advances in the Astronautical Sciences* 135.2 (2009), pp. 1099–1112.

[60] R Pavani and G Ranghino. “A method to compute the volume of a molecule”. In: *Computers & Chemistry* 6.3 (1982), pp. 133–135.

[61] Lei Chen et al. “Calculation of collision probability”. In: *Orbital Data Applications for Space Objects: Conjunction Assessment and Situation Analysis* (2017), pp. 135–183.

[62] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. “SPEA2: Improving the strength Pareto evolutionary algorithm”. In: *TIK report* 103 (2001).

[63] Antonio Benítez-Hidalgo et al. “jMetalPy: A Python framework for multi-objective optimization with metaheuristics”. In: *Swarm and Evolutionary Computation* 51 (2019), p. 100598.

[64] Miha Ravber et al. “Maximum number of generations as a stopping criterion considered harmful”. In: *Applied Soft Computing* 128 (2022), p. 109478.

[65] *English: Collision between Iridium 33 and Kosmos 2251. Globe by NASA Worldwind in the public domain.* Feb. 2014. URL: `https://commons.wikimedia.org/wiki/File:Collision-1a1.jpg` (visited on 08/14/2024).

[66] Eckart Zitzler et al. “Performance assessment of multiobjective optimizers: An analysis and review”. In: *IEEE Transactions on evolutionary computation* 7.2 (2003), pp. 117–132.

[67] Nikolaos S Thomaidis and Alexios-Ioannis Moukas. “Designing Efficient Renewable Energy Portfolios for Optimal Coverage of European Power Demand under Transmission Constraints”. In: *Energies* 15.24 (2022), p. 9375.

[68] Vladimir A Chobotov. *Orbital mechanics.* Aiaa, 2002.

[69] *English: Diagram illustrating and explaining various terms in relation to Orbits of Celestial bodies.Español: Elementos orbitales de un cuerpo celeste.* Oct. 2007. URL: `https://commons.wikimedia.org/wiki/File:Orbit1.svg` (visited on 08/14/2024).

[70] Walter Hohmann. *The attainability of heavenly bodies.* 44. National Aeronautics and Space Administration, 1960.

[71] *English: Illustration of Hohmann transfer orbit.* Mar. 2007. URL: `https://commons.wikimedia.org/wiki/File:Hohmann_transfer_orbit.svg` (visited on 08/14/2024).

[72] *J2 Perturbation.* URL: `https://ai-solutions.com/_freeflyeruniversityguide/j2_perturbation.htm` (visited on 08/10/2024).

A Orbital mechanics

Orbital mechanics is the study of the motion of bodies in orbit under the influence of gravity, atmospheric drag, and thrust forces [68]. It is an important consideration for ADR missions as the chaser must perform orbital maneuvers to rendezvous with debris, and is necessary for accurate tracking of the location and movement of space debris.

A.1 Orbital elements

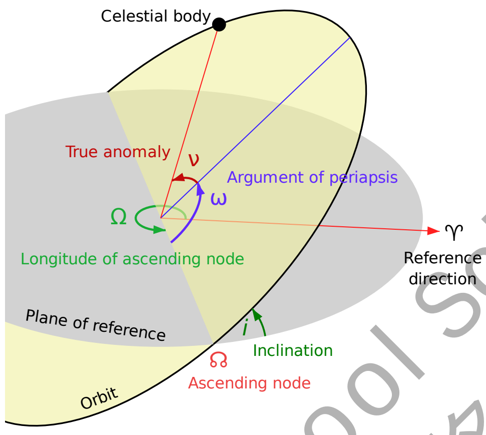


Figure 16: Orbital elements [69]

Any orbit can be fully specified with a set of parameters known as orbital elements [68]. Figure 16 depicts how the elements specify the shape and position of the orbital plane relative to a reference plane and reference direction, and specify the specific location of the orbiting body at any point in time. Table 5 describes the six classical orbital elements as originally described by Kepler.

Orbital element	Symbol	Unit	Description
Eccentricity	e	Unitless	Characterizes the shape of the conic section of the trajectory. For circular orbits $e = 0$ and for elliptical orbits $0 < e < 1$.
Right ascension of ascending node (RAAN)	Ω	°	The ascending node is defined as the intersection of the orbital plane with the reference plane when the satellite is moving from below the plane to above the plane. Ω shows the angle measured eastwards from the vernal equinox.
Inclination	i	°	The inclination specifies the angle between the reference plane and the orbital plane.
Semi-major axis	a	m	Specifies the size of the conic section. For a circle a is the radius and for an ellipse a is the longest radius.
Argument of periapsis	ω	°	The argument of periapsis is an angle measured on the orbital plane from the reference plane at the ascending node to the periapsis of the orbit — when the satellite is closest to the central body.
True anomaly	ν	°	The true anomaly is the angle measured on the orbital plane between the current position of the satellite and the periapsis of the orbit.

Table 5: Orbital elements

A.2 Hohmann transfers

The most efficient transfer between two non-intersecting orbits requires a minimum of two impulsive maneuvers. Assuming that both orbits are circular, the Hohmann transfer [70] can be used, which connects the opposite sides of the initial and target orbits with an ellipse, as shown in Figure 17.

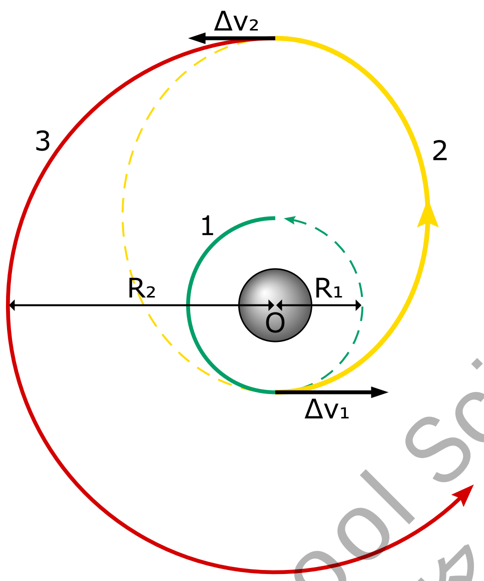


Figure 17: Hohmann transfer [71]

Δv_1 can be calculated as the difference in orbital velocity between the initial orbit and the transfer orbit, and Δv_2 as the difference in orbital velocity between the transfer orbit and target orbit.

A.3 J_2 perturbation

The Earth is not a perfect sphere; it is oblate, bulging around the equator due to the centrifugal effect caused by its rotation. The J_2 zonal coefficient is the second term in the series expansion determining the effect of oblateness on Earth orbits, and the magnitude of its effect is over a thousand times greater than other zonal coefficients [72].

The J_2 perturbation will affect the RAAN of an orbit at a constant rate, depending on the size, shape and inclination of the orbit. The equation for J_2 nodal precession is as follows [72]:

$$\dot{\Omega} = -\frac{3}{2} J_2 \sqrt{GM_\oplus} R_\oplus^2 a^{-\frac{7}{2}} \cos i \tag{49}$$

where the value of the J_2 constant is 1.08262668×10^-3. The J_2 perturbation will be used in Section 4.3 to correct the RAAN of the ADR chaser during a drift orbit.