CHALMERS
UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF GOTHENBURG

Multi-objective optimization for placing airspace surveillance observers

Master's thesis in Computer science and engineering

AMANDA ANDERSON

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2021

Master's thesis 2021

Multi-objective optimization for placing airspace surveillance observers

AMANDA ANDERSON

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2021

Multi-objective optimization for placing airspace surveillance observers

AMANDA ANDERSON

Supervisor: Ulf Assarsson, Department of Computer Engineering
Advisor: Gabriel Tigerström, Carmenta AB
Examiner: Erik Sintorn, Department of Computer Engineering

Master's Thesis 2021
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2021

Multi-objective optimization for placing airspace surveillance observers

AMANDA ANDERSON
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Reconnaissance is an important aspect of military planning. Tools that help analysts monitor and make informed choices are vital for avoiding costly situations. The use of ground-based radar sensors is a common method for monitoring for both land-based and airborne threats. Manually finding optimal locations to install sensors within an area of terrain can be difficult and time intensive, particularly when multiple objectives exist. The purpose of this thesis is to implement and compare two heuristic algorithms for automatically generating a set of optimal locations for airspace surveillance sensors. The algorithms seek to find solutions that maximize both total area coverage and coverage of a specific area of interest. They also seek solutions that minimize sensor overlap and price. The research problem was formulated into a multi-objective optimization. The two algorithms tested include the NSGA-II and a multi-objective Ant Colony Algorithm (MOACO). A population-halving augmentation and the Multi-resolution Approach (MRA) developed by Heyns [1] were also applied to see if algorithm run time could be reduced without impacting final solution quality. The NSGA-II outperformed the MOACO algorithm with respect to diversity of the final solution set, however the algorithms performed similarly with respect to run time and convergence. It was found that population-halving and the MRA could result in computation time reduction for the tested scenario, however not at a significant level.

Keywords: Facility location, multi-objective optimization, ant colony algorithm, genetic algorithm, pareto optimal, observer placement, 3D viewshed

Acknowledgements

I wish to express my sincere gratitude to both of my supervisors. First, to Gabriel Tigerström at Carmenta, thank you for your support and feedback during both the practical implementation and writing of the thesis. To Ulf Assarsson, thank you for your research guidance and advice. I would also like to thank Mark Legh-Smith of Operational Solutions Ltd. for invaluable insight into CUAS.

Thank you to all my friends and family for their ongoing support throughout my years of studying.

Finally, thank you to Adam Bonnycastle, whose inspirational teaching led me to pursue this path.

Amanda Anderson, Gothenburg, June 2021

Contents

List of Figures

List of Tables

1 Introduction

Reconnaissance has long been an important aspect in military planning. Tracking where threats are located provides a significant advantage when taking military action. In modern times, military bases and infrastructure require protection not only from terrain-based threats, but also airborne. Military drones and aircraft are often employed as means of combat in the 21st century, and in many cases are used in asymmetrical operations. Asymmetrical warfare typically involves a group with fewer resources opposing an adversary with superior armed forces [2]. The tactics used in asymmetrical combat are often unconventional or *radically different* from what is expected, and can be difficult to predict [2]. The September 11 attacks on the Twin Towers is one of the most well-known examples of asymmetrical warfare. The American Air Force was completely unprepared, and an Air Force General later admitted to never considering that possibility of attack [2]. Developing strategies for countering asymmetric attacks can be challenging, and decisions must often be made quickly and in the face of uncertainty. In many cases, a wrong decision results in significant loss of human life. As such, tools to help analysts monitor threats and make informed choices are vital.

Computers are often employed to aid in the military decision-making process. Consider the example of an analyst tasked with setting up five radar sensors around a base to detect hostile aircraft flying overhead. The terrain around the base contains hills, a lake, and sheer cliffs. The sensors cannot be placed on the cliffs or in the lake, and must be placed in the remaining terrain such that their signals cover as much as possible of the overhead airspace. Any areas left uncovered by the signal could allow for hostile drones or aircraft to enter undetected. If the five sensors all have different levels of coverage, choosing the optimal location for each sensor without a computer-based visualization aid is challenging. Airspace sensors are both expensive to purchase and install, so placing them in the correct locations from the outset is desirable.

The topic of airspace sensor placement for surveillance is the focus of the following thesis. Sensors are formally referred to as *airspace observers*, and the volume of airspace that each sensor's signal covers is called the *viewshed*. The goal of the analyst is to then maximize the combined viewshed of all observers. This problem can be complex, and finding optimal positions for sensors means *every* possible solution has to be checked. To illustrate, consider two sensors that must be placed in 1km^2 terrain. Assuming a sensor can be placed in one of 1500 possible locations, checking every possible solution means we need to check all combinations for placing two sensors in the 1500 terrain points. For each combination we would then check the resulting combined viewshed in order to determine which placement of the sensors is the best. This leaves us with 1500^2 (2,250,000) solutions to assess. Even for this simple scenario, a standard computer generally takes over an hour to compute the optimal sensor placement [3].

The previous scenario is simple. However, many scenarios are far more complicated. Terrain sizes are larger, more sensors are being placed, and additional objectives may exist when placing sensors. Furthermore, observers with different field-of-view or range characteristics might need to be placed simultaneously. As such, finding solutions for these scenarios using brute-force is not feasible. Methods for reducing the complexity of the computation are needed. A number of heuristic methods exist for solving this problem, and the focus of this thesis is to build upon two previously used heuristic algorithms to solve for more complicated scenarios.

1.1 Objective

There are two main objectives of this thesis. The first objective is to construct two methods for finding optimal placement of airspace observers in a multi-objective context. The methods should return a set of near-optimal locations for the observers given the input of an elevation model, Area of Interest (AoI) and the number and type of observers to be placed. The methods should handle cases with multiple types of observers and function well in scenarios with natural or built environments. The second objective is to reduce computation time without sacrificing solution quality. Faster computation is more favourable to end-users and is particularly important for time-sensitive decision-making within the military context.

1.2 Research Question

The research question is formalized as, "Given n observers of specific types, how can they be placed in a terrain to achieve maximum viewshed coverage of the surrounding airspace while minimizing costs?". This type of problem is referred to as an *Optimal Multiple Viewpoints* (OMV) problem [3]. The research question can be broken down into a set of four optimization goals:

1. Maximize cover of an overhead airspace.
2. Maximize cover of any specific Area of Interest (AoI) within the terrain.
3. Minimize overlap between sensor signals (Cost I).
4. Minimize price of the sensor solution (Cost II).

1.3 Limitations

There are a number of limitations placed on this project to maintain the scope. First, this thesis does not include work on the 3D airspace viewshed calculation. Efficient viewshed calculation is a broad and ongoing field in itself, and as such the focus for this thesis is solely on the observer placement algorithms. A tool provided by the principal (Carmenta AB) is used for all airspace viewshed calculation. Additionally, only the context of ground-based airspace observers will be investigated. Airborne and submerged observers have different characteristics which are not addressed in this thesis. As the primary goal of this thesis is to achieve solutions with greater accuracy (rather than solely improving computation speed), methods that seek to simplify the search space at the expense of solution quality will not be assessed. For instance, the frequently used landform classification technique is not studied as it generally results in a loss of final solution quality.

1.4 Scientific Contribution

While a number of analogous problems in the broader domain of camera sensing have related studies (e.g. building surveillance camera placement), the specific combination of the airspace observation context, optimization goals and constraints in this thesis have not been studied previously. Furthermore, searches for research assessing Multi-objective Ant Colony Optimization algorithms (MOACO) for viewshed or observer placement optimization were not successful, leading to the assumption that this will be a novel approach for this particular research problem. Additionally, the results of this research could impact the broader domain of camera sensing and facility location problems.

The implementation of the NSGA-II genetic algorithm and multi-resolution technique in this thesis will also help validate earlier findings that these techniques significantly improve both solution quality and computation time [1]. The effectiveness of the multi-resolution technique with the MOACO algorithm will also be a novel addition to the research field.

Lastly, the outcomes of this research will also help to identify future research opportunities and may illuminate any airspace-specific characteristics of the problem which must be considered.

2
Theoretical Background

This chapter presents the theoretical background for the thesis. The chapter begins by providing an introduction to geospatial data in Section 2.1. The concept of visibility and viewshed analysis is explained in Section 2.2. The problem of geospatial siting and location analysis and its historical development are outlined in Section 2.3. Finally, multi-objective optimization and relevant optimization algorithms are discussed in Section 2.4.

2.1 Geospatial Data

Geospatial data refers to the digital representations of spatially-referenced, real-world objects. As an example, the digital representation of road networks is often portrayed as a network of lines covering specific coordinates in digital maps. To represent real-world objects digitally, a data model is needed. There are two main geographical data models: vector and raster.

2.1.1 Vector Data

Vector data contains three main types of objects: points, lines and polygons. Points contain an X and Y coordinate, and can also have a Z coordinate if an elevation attribute is included, see fig. 2.1a. Objects such as radar sensor locations can be represented as points on a map. Line objects are constructed by two or more connected points, see fig. 2.1b. A series of points connected by a line can be used to represent objects such as roads and rivers. Polygons can be used to represent objects with an area value, see fig. 2.1c. For example, city boundaries, lakes, and building footprints can be represented using polygons. In the context of this thesis project, observer locations are represented by points with X, Y and Z location data. 3D polygons are used to represent the volume of overhead airspace (viewshed) that is covered by the sensors.

2.1.2 Raster Data

The raster data model represents objects using a grid of pixels. Each pixel is associated with a specific geographical location. Examples of raster data can include photographs (whereby pixels hold a colour value), continuous structures (e.g. temperature or elevation data) or categorical structures (e.g. land use) [4]. For elevation data, each pixel corresponds to a geographical location and holds the height at that

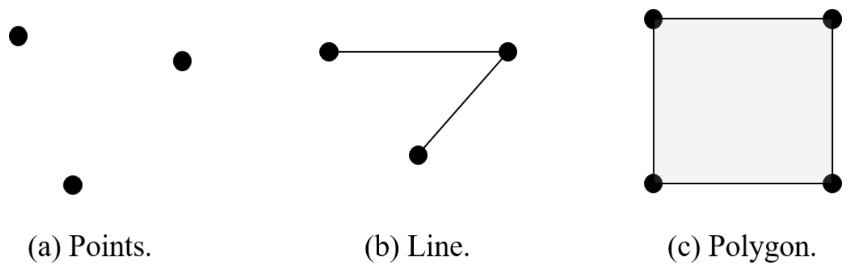

Figure 2.1: Vector data objects.

location, see Figure 2.2. Grid-based raster data also has a specific spatial resolution. The resolution relates to the size of the pixel (i.e. the real-world distance that one pixel represents). A raster dataset with pixels representing $1m^2$ is thus finer in resolution than a dataset with pixels representing $5m^2$.

Figure 2.2: 1m resolution elevation raster for a $5km^2$ area. Lighter pixels correspond to a higher elevation value.

2.1.3 Elevation Models

Digital Elevation Models (DEMs) act as a model of the Earth's surface [5]. They are typically represented in one of two ways: raster or Triangulated Irregular Network (TIN). Each pixel in a raster DEM holds a height value associated with that location in the real-world. Tools can be used to convert raster DEMs into 3D models [5], see Figure 2.3. TINs represent a 3D mesh by using a network of connected triangles. In this thesis, a raster-based DEM is used.

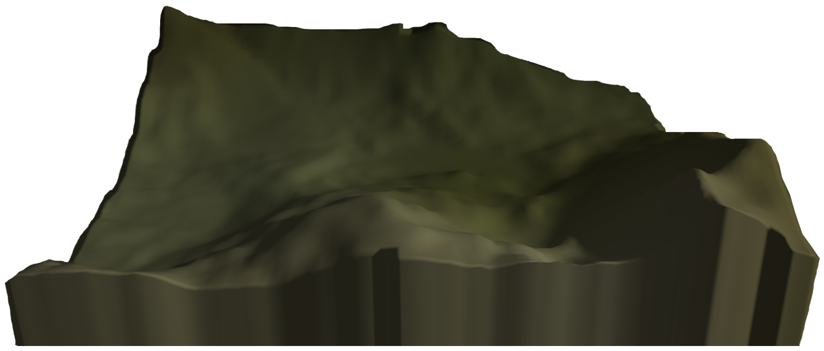

Figure 2.3: 3D visualization of elevation raster shown in Figure 2.2.

2.2 Visibility

Visibility analysis is frequently used in the field of geomatics and Geographical Information Systems (GIS). In the context of digital terrain, visibility refers to whether two objects on the terrain are *visible* from one another [6]. *Line of Sight* (LoS) and *viewshed* are two important concepts in visibility. LoS refers to the visibility between two points, see Figure 2.4. Viewshed refers to the total volume that can be seen from a point.

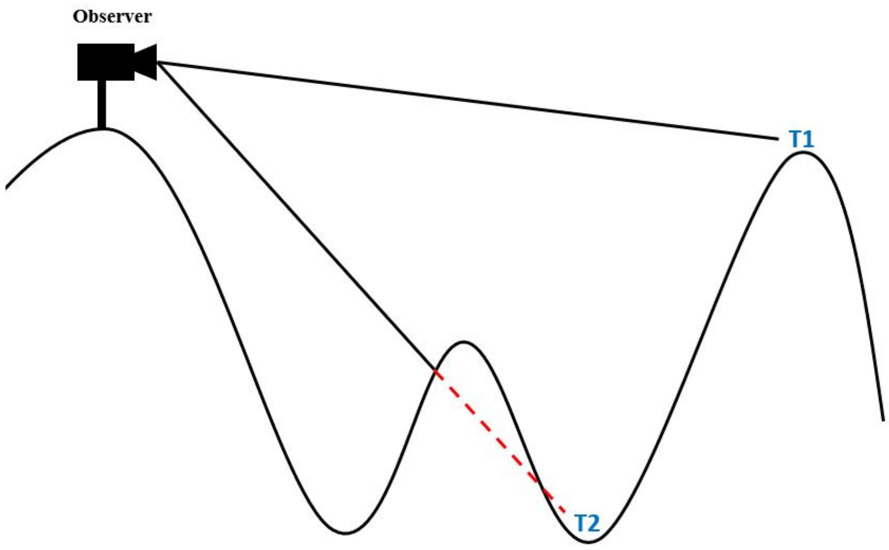

Figure 2.4: LoS between an observer and two targets (T1 & T2). The observer only sees T1 as T2 is obstructed by the terrain.

2.2.1 3D Airspace

In the context of this thesis, the viewshed from a point is considered to be the 3D volume of air that an observer can see. Each observer is thus required to specify the sensor range and both vertical and horizontal field of view so a viewshed can be calculated. In the case of multiple observers being placed in an Area of Interest

(AoI), the combined viewshed is the total 3D volume of air that *at least one* observer can see. All 3D viewshed calculation and analysis used in this thesis is provided by the principal. As such, the specific implementation used for calculating 3D viewshed is not a focus in this project.

2.3 Geospatial Siting

The field of location theory began in 1909 when Alfred Weber sought to select an optimal location for a new warehouse that minimized the geographical distance to customers [7]. A number of related problems have since been identified after the inception of the field, but all focus on the basic problem of finding an optimal location for a facility considering a set of objectives. Elevation models are continuous structures, and placing an object in a DEM means that it can be placed anywhere on an infinite plane [8]. As such, testing all locations is not possible. Selecting only certain points to use as possible siting locations (i.e. a discrete set of locations) is often done for simplification. However, a large number of discrete points can still be computationally challenging. Small scenarios (e.g. 1500 discrete siting locations) can still take over an hour for a standard computer to find optimal locations [3].

Nowadays, siting is not only limited to buildings. Wind farms [9], solar panels [10], and telecommunication networks [11] are also placed in terrain, and often have a number of objectives for optimal placement. Indoor and built environment location siting has also been explored. Namely, finding optimal locations for surveillance sensors or cameras [12]. Brute-force testing of all possible siting configurations in a problem is not normally feasible, and as such multi-objective heuristics to optimize the process are often employed. These heuristics and relevant optimization concepts are discussed in the following section.

2.4 Multi-objective Optimization

This section will describe concepts for multi-objective optimization problems as well as common performance metrics. The research question stated in Section 1.2 was broken down into four separate objectives. Each objective thus represents an independent *objective function*, see Definition 2 (notation from Li [13]). The *decision variables* are the information used when calculating the values of objectives, see Definition 1 (notation from Riquelme [14]). In the context of this thesis, the decision variables are the sensor locations and configuration (direction and sensor type) for a solution. In multi-objective problems, the objective functions conflict with one another. For example, a solution with a low price (perhaps through incorporating cheaper sensors with smaller ranges) will not be able to provide maximal airspace coverage compared to a more expensive solution with longer-range sensors. Likewise, a more expensive solution will not be able to be made cheaper without sacrificing airspace coverage. As such, these objectives are in conflict with one another - price and airspace coverage cannot be simultaneously optimal in a single solution. Multi-objective problems thus do not have a single solution. Instead, they have a number of solutions that are equally optimal with regard to the different objectives. An optimal solution is found when none of the objectives can be improved without degrading one of the others. A solution that is more optimal than another is said to be *dominating*, see Definition 3 (notation from Collette [15]). A non-dominated solution is referred to as a *Pareto optimal* solution. All optimal solutions for a problem are collected into a set of solutions called a *Pareto front* or *Pareto set*. The analyst is then tasked with selecting which Pareto optimal solution from the set to use.

Definition 1 (**Decision Variables** [14])
Decision variables can be denoted as x_j, $j = 1{,}2{,}...n$. A vector containing n decision variables can be represented by: $x = [x_1, x_2..., x_n]^T$. The solution space S is the vector space containing all possible decision variables.

Definition 2 (**Objective function** [13])
Let S be the solution space. For k objective functions f_1,...,f_k, f_i : S $\mapsto \mathbb{R}$. The decision variables yield the value for an objective function.

Definition 3 (**Domination** [15])
Vector x = $(x_1, x_2, ...x_n)$ dominates vector y = $(y_1, y_2, ...y_n)$ iff $\forall i \in [1, ...n], f_i(x) \leq f_i(y)$ and $\exists i \in [1, ...n], f_i(x) < f_i(y)$.

The Pareto front for a problem will have a diverse range of solutions that maximize or minimize the different objectives. As such, algorithms that seek to approximate the Pareto front should try to find diverse solutions that cover the full extent of the front (not only solutions that are *close* to the front) [16].

2.4.1 Performance Metrics

To measure how well an algorithm is able to approximate the Pareto front, a number of performance metrics exist. These metrics are also required to be able to compare the performance of algorithms with one another. Metrics typically consider one of three aspects of a solution set [14]:

- Convergence or closeness of the solutions to the Pareto front
- Diversity of solutions across the front
- Number of solutions

Four commonly used metrics include: computation time, spread, hyperarea, and inverted generational distance. Computation time refers to the total time required to complete one run of an algorithm. Spread (or spacing) measures the diversity of solutions across the Pareto front, see Definition 4 (notation from Tigerström [17]). A low spacing value is preferred as it means the solutions are more evenly dispersed along the front [17]. Hyperarea refers to the area of the objective space that the solution set covers, see Definition 5 (notation from Tigerström [17]). When objectives are to be minimized, a larger area is preferred as it means more of the objective space is covered by the solutions. Inverted generational distance is used to measure how far a solution set is from the Pareto front (i.e. convergence to the front), see Definition 6 (notation from Liu [18]). Solution sets with a lower inverted generational distance value are thus a closer approximation of the Pareto front.

Definition 4 (**Spread [17]**)

$$S = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\bar{d} - d_i)^2}$$

where

$$d_i = \min_j(|f_1^i(\vec{x}) - f_1^j(\vec{x})| + |f_2^i(\vec{x}) - f_2^j(\vec{x})|)$$

and

$$\bar{d} = \frac{\sum_i^n d_i}{n}$$

Definition 5 (**Hyperarea [17]**)

$$H = \sum_i a_i | v_i \in PF_{known}$$

where a_i is the area of dominated objective space under solution v_i, in the estimated Pareto front PF_{known}.

Definition 6 (**Inverted Generational Distance [18]**)

$$IGD = \frac{\sum_{u \in U} dist(u, V)}{|U|}$$

where u is an element in U (the objective vector of a reference point). V is the set of all objective vectors, and dist(u,V) is the nearest distance from u to V.

2.4.2 Genetic Algorithms

Genetic algorithms are based on the theory of evolution, which is the process of change in a species' genes over time. Evolution does not occur to a single individual, but instead occurs during reproduction and over generations. Individuals that are better suited to surviving in their environment have a greater chance of reaching the reproductive stage and thus transferring their genes to offspring. As such, these individuals' genes have a greater chance of accumulating in a population. This concept is called *natural selection*, and the measure of an individual's suitability in their environment is called *fitness* [19].

Random mutation of chromosomes during reproduction is an important feature of evolution. Mutations can result in new traits for individuals that may improve or worsen their fitness. Mutations can sometimes result in special adaptations for an environment, which may then become more prevalent in populations due to natural selection. These concepts serve as the backbone for genetic algorithms.

Genetic algorithms have been successfully applied to many different types of problems. The process begins by initializing a random population of individuals, with each individual representing a potential solution to the problem at hand. Each individual is associated with a measure of fitness for solving the problem, and those with greater fitness levels are then permitted to reproduce. The parents and offspring are then used to create the population for the following generation of the algorithm. The goal of genetic algorithms is to converge to near optimal solutions over a number of generations. Each subsequent generation should result in better solutions to the problem.

A number of researchers simultaneously began working on genetic (or evolutionary) algorithms in the 1960s and 1970s [20, 21, 22]. A vast number of genetic algorithms have been developed since the 1960s, and one of particular interest for multi-objective problems is the non-dominated sorting genetic algorithm II (NSGA-II), see Algorithm 1 [23]. Compared to previously used algorithms, the NSGA-II has improvements in computational complexity, required parameters, and includes the concept of elitism for faster convergence [23]. The NSGA-II algorithm employs a fast non-dominated sorting method, which reduces the computational complexity from O(MN^3) to O(MN^2), where M is the number of objectives and N is population size [23].

Algorithm 1: NSGA-II [23]

```
Data: $genMax, popSize, pMut$
Result: Set of non-dominated solutions.

$P \leftarrow$ Random population of size $popSize$ * 2
$Q \leftarrow \emptyset$
$t \leftarrow 0$
while $t < genMax$ do
    $R_t = P_t \bigcup Q_t$ ;              // Combine parent and children population
    $F$ = fast-non-dominated-sort($R_t$) ;     // Sort into domination fronts
    while $/P_{t+1}/ < N$ do
        crowding-distance-assignment($Fi$) ;       // Assign crowding values
        $P_{t+1} = P_{t+1} \bigcup F_i$
    end
    Sort($P_{t+1}$, >= n) ;            // Sort population by rank and crowding
    $P_{t+1} = P_{t+1}$[0:N] ;                          // Fill parent population
    $Q_{t+1}$ = make-new-pop($P_{t+1}$) ;   // Selection, crossover and mutation
    t=t+1
end
```

2.4.3 Ant-Colony Optimization

Ant Colony Optimization (ACO), first proposed by Dorigo *et al.* [24], is an optimization strategy based on simulating the behaviour of ants in a colony searching for food [24]. Many different types of ACO algorithms exist, however the basic idea is that a group of ants use heuristic data and information from past exploration to make decisions as they search the solution space for optimal solutions. When high-quality solutions are explored, they are assigned "pheromone", which then attracts more ants to explore in that direction. Worse solutions have their pheromone evaporated over time, discouraging further exploration. This collective pheromone information is shared between the ants, and over time leads to convergence to a near optimal solution.

Multi-objective versions of ACO are diverse, and many different configurations exist. Some common design considerations include a single versus multi-colony approach, using weighted versus non-weighted objectives, sharing information between colonies, and augmenting the algorithm with an additional local search strategy. Each design choice can greatly impact the effectiveness of the algorithm. As such, implementation of a MOACO algorithm requires greater consideration compared to the straight-forward implementation of the NSGA-II. The lack of a standardized design framework for MOACO algorithms has led to a diverse range of implementations which can make comparisons between MOACO algorithms challenging [16]. A specific configuration of MOACO is implemented in this thesis for the particular research context, which is described in greater detail in Section 4.3.2 of the Methods chapter.

Algorithm 2: Standard Multi-objective ACO [25]

```
Data: maxIterations
Result: Set of non-dominated solutions.

$T \leftarrow$ Initialize pheromone trails
$n \leftarrow$ Initialize heuristic matrix
$P \leftarrow$ Pareto set as $\emptyset$
while $i < maxIterations$ do
    construct-ant-solution();
    apply-local-search();                          // Optional
    update-pareto-set();
    update-pheromone-matrix();
    i=i+1
end
```

The construction of an ant solution in Algorithm 2 involves the movement of an ant from a starting point to a new point through consideration of the *decision policy* [25]. Consideration of both the pheromone data and heuristics is included in the decision policy. A local search may then be conducted to try and improve the solution. Once all solutions have been generated, the dominating solutions are added to the Pareto set. Pheromone is then added to the well-performing solutions and old pheromone is evaporated according to a predefined rate. After a set number of iterations have run, the final Pareto set of solutions is returned.

2.4.4 Multi-resolution Technique

Facility location problems such as the problem presented in this thesis have been shown to be computationally expensive [3, 26, 27]. Techniques for reducing complexity and computation time are vital particularly for time-sensitive use cases. Algorithms with faster computation-time will thus be more valuable for a military analyst compared to those with longer run times. The multi-resolution technique (MRA) as described by Heyns [1] has previously been used with great success for improving both computation time and final solution quality [1]. Before the introduction of this technique, multi-objective observer placement problems used only a single spatial resolution of candidate points (e.g. a grid of candidates 10m apart). As such, one could reduce computation time by using a coarser grid of points, but at the expense of final solution quality. Heyns [1] sought to eliminate this trade-off between computation time and solution quality, and results from the initial 2014 article show improvements in both computation time *and* final solution quality by using the MRA. Improvement in solution quality occurred as poorly-performing areas were identified and discarded earlier. As such, more time was able to be spent exploiting well-performing locations.

The MRA as described by Heyns [1] is of particular interest given its specific success with the NSGA-II algorithm which is implemented in this thesis [1]. The goal of the MRA is to reduce the search of poorly-performing candidate points, thereby reducing total computation time [1]. The technique works by first running an optimization algorithm with a coarser spatial resolution of candidate points. After a solution set is generated from coarse points, the algorithm is run again with a set of finer resolution points — but only those within a neighborhood of the coarser solution points. This is then repeated for finer resolutions until the finest resolution of points is assessed.

3
Related Work

This chapter discusses the related work with regard to multi-objective observer placement in both terrain and airspace contexts. Previous research has addressed the OMV problem using two main strategies:

1. Using only a subset of the DEM as candidate observer locations
2. Using heuristic algorithms for finding optimal or near-optimal solutions

Both of these strategies have been shown to greatly reduce viewshed optimization computing time while still resulting in near-optimal solutions [26, 3, 28, 27].

3.1 Subset of DEM

Using only a subset of a DEM typically involves using a landform classification scheme that assigns each elevation point in the DEM a particular type. These can include landforms such as peaks, pits, passes, ridges, and numerous others. Specific landforms (e.g. ridges) are often correlated with areas with high visibility of the surrounding area [3, 26]. The main idea for this strategy is to then find a solution using only candidate observer points that are classified as high-visibility landforms. This has been shown to result in nearly-optimal solutions (typically within 10% of the optimal brute-force solution) with significantly reduced runtimes [3, 26]. Specifically, Rana [26] saw a reduction in computation time by 3 orders of magnitude (from approximately 8000 seconds to 10 seconds).

Research pertaining to using landform classification for candidate subsets specifically for the airspace context was not found, however it is assumed that certain landforms would also result in increased visibility of overhead airspace (e.g. peaks/ridges with 360° views as opposed to valleys surrounded by high peaks). The main drawback to this method is some trade-off between solution quality and computation time. Certain scenarios require solutions of the highest quality, and any trade-off to improve computation time is not permitted.

Another subset technique is to do viewshed analysis using a TIN as opposed to a grid-based raster DEM, as there are fewer points in a TIN [29]. When using a TIN, homogenous terrain with little variation (e.g. a flat plain) will use fewer points (thus larger triangles) to represent the area. In grid-based DEMs, evenly spaced points are used regardless of the variation in the terrain, meaning that flat areas may become over-sampled, and thus contain extra points which are not necessary. However, care must be taken if TINs are used. If a TIN simplifies terrain too much, the loss of quality can introduce errors into the viewshed computation.

As described previously in subsection 2.4.4, the more modern multi-resolution approach (MRA) has shown promising results, without any loss of solution quality [1]. However, no studies by other researchers were found that corroborated the results found by Heyns [1]. As such, it is unclear how this method performs when used for other scenarios, datasets, and heuristic algorithms.

3.2 Heuristic Algorithms

The second strategy involves using heuristic algorithms to efficiently find near-optimal solutions. Heuristics offer a faster alternative to the traditional brute-force approach by using more sophisticated techniques to search the solution space. However, it is important to note that heuristics do not always result in optimal solutions. More often, they result in near-optimal ones. There are many different heuristic approaches, however genetic algorithms have been shown to be particularly successful for the OMV problem [27, 17]. Variations of Simulated Annealing Algorithms, Swap Heuristics, and Ant Colony Algorithms have also been successfully applied to multi-objective problems [3, 27, 30, 31, 32].

In the particular context of airspace surveillance, it was found that research on this topic often investigates the problem from the point of view of a drone or Unmanned Aerial Vehicle (UAV) trying to avoid hostile sensors or obstacles (generally called "Seek and Avoid") [33, 34]. However, the problem as described in this thesis is from the point of view of an analyst setting up the ground sensors for detecting hostile aircraft. Thus, the context is opposite to a large portion of related airspace surveillance research, leaving a noticeable gap in the literature.

3.3 Multi-objective Optimization

For multi-objective viewshed optimization, a more limited selection of research exists. Optimization with distance constraints on observer locations can be found in research investigating signal tower placement, as signal towers have transmission ranges and overlap requirements [30]. For example, Kim *et al.* [30] investigates signal tower placement and provides a case where a genetic algorithm is successfully implemented with distance constraints. Tigerström [17] researches a multi-objective scenario where there are three distinct criteria: observer cover/safety, maximizing viewshed, and minimizing required number of observers. In Tigerström's [17] study, three heuristic algorithms are assessed, with the NSGA-II genetic algorithm being the best performing. While this study focuses on observing surrounding terrain (as opposed to airspace), the optimization method and core problem are the same and as such applicable to the problem context focused on in this thesis.

The NSGA-II algorithm has been particularly successful and widely applied to multi-objective problems. The study by Tigerström [17] found that the NSGA-II performed better than both Tabu Search and Multi-objective Simulated Annealing

(MOSA) for a multi-objective observer placement problem. In a water-reservoir optimization study, the NSGA-II significantly outperformed a multi-objective particle swarm optimization (MOPSO), resulting in both a better diversity of solutions and approximation of the Pareto front [35]. When applied to a vehicle suspension problem, the NSGA-II performed marginally better than the SPEA-II and PESA-II algorithms in terms of minimizing the objectives, however the final solution set had poorer diversity [36]. It is highlighted in this study the importance of maintaining diversity in the population for NSGA-II [36]. When applied to a centrifugal pumps problem, the MOPSO outperfomed the NSGA-II in finding the *extremes* of the Pareto front [37]. The results suggest that in the specific scenario the NSGA-II population lacked diversity and converged more to compromise (balanced) solutions, rather than extremes for each objective. It is evident from prior research that the NSGA-II has performed quite well for a large number of different optimization problems.

The work by Heyns [38] whereby a population-halving augmentation was used with the NSGA-II is of particular interest to this thesis. By measuring convergence between populations, this technique decreases the population by half for all remaining generations when a certain threshold is achieved. This is meant to decrease computation time while still allowing exploitation of the best solutions [38]. Heyns [38] found that using a similarity threshold of 80% was able to significantly reduce run time with no effect on solution quality.

MOACO algorithms are a diverse group of algorithms that have been applied to a wide range of optimization problems. While often used to solve path-planning problems, they are effective for a number of different problems [39]. Comparison of the different configurations is challenging, and multiple efforts have been made to create a taxonomy for MOACO algorithms [40, 16, 41]. In-depth explanation of each taxonomy and the numerous ACO configurations is outside the scope of this thesis. However, an important finding from literature is that in some applications of MOACO the algorithms are competitive with or outperform the highly-regarded NSGA-II [42, 43, 44]. No relevant studies using MOACO for the observer placement problem as described in this thesis were found. Previous studies often used tailored or custom MOACO algorithms for the specific research context, and as such it is unclear how effective MOACO will be for the research problem investigated in this thesis. Furthermore, selecting the best configuration requires identifying recommendations in literature and potentially creating a custom implementation of a MOACO algorithm. The analysis performed and final algorithm used is described in the Methods section.

With regard to placing different *types* of observers in the same terrain, no current research was located within the context of terrain or airspace viewshed optimization. However, in the broader geospatial domain, research in surveillance camera planning can offer additional insight. A number of different camera types can be used when setting up building surveillance systems. As a result, to optimize the locations of these cameras, the specific camera types and corresponding properties (e.g. fixed vs rotating, field-of-view, view angle) must also be considered. This problem is thus analogous to airspace viewshed optimization where observers have different field of view characteristics. Gonzalez *et al.* [45] investigates this problem using a binary integer programming algorithm and is able to optimally place a mixture of binary and omni-directional surveillance cameras in a scenario.

4 Methods

This chapter describes the method of the thesis. Section 4.1 reiterates and describes the research problem from an implementation perspective. Section 4.2 describes the experimental setup used for testing the algorithms. Section 4.3 outlines algorithm design and implementation. Section 4.4 describes implementation of the multi-resolution technique. Section 4.5 describes the evaluation metrics and data collection procedures.

4.1 Problem specification

As described in Section 1.2, the multi-objective research problem has four main objectives. To make analysis and visual plotting simpler, it was decided that henceforth the minimization type would be used for all objectives. This was done by using the *complement* of the airspace coverage. The complement is calculated by subtracting the 3D volume of airspace covered from the total volume of the input Area of Interest. In other words, the complement is the volume that none of the observers can see, thus leaving it uncovered. As such, a lower complement value represents a higher coverage of the area. Minimization was also used for price and sensor overlap. In this thesis, sensor overlap for redundancy was not considered a goal for sensor placement. However, the objective was included as for other use cases consideration of overlap may be very important (e.g. placing signal towers), and well-spaced sensor solutions were preferred for this project. The four objectives were thus restated as:

1. Minimize the complement of the cover of an overhead airspace.
2. Minimize the complement of the cover of any specific Area of Interest.
3. Minimize overlap between sensor signals.
4. Minimize price of the sensor solution.

The decision variable as described in Section 2.4 was chosen to be a vector of sensor locations following the advice presented by Tigerström [17]. As a small number of observers are being placed in a large solution space, keeping a fixed-length vector representing all candidate locations (and then marking which has a sensor) is more costly than using a variable-length vector that keeps only the points where sensors are located [17]. The 3D visibility calculations including coverage, overlap, and clipping to the AoIs were all completed using the principal's product Carmenta Engine.

Table 4.1: Sensor specifications used in experiment.

Sensor Type	Horizontal FoV	Vertical FoV	Range (m)	Price
A	360°	80°	750	\$200k
B	40°	80°	1000	\$100K

4.2 Experiment setup

To test the algorithms, input data representing a realistic use case was required. Section 4.2.1 describes how the input geodata was used, Section 4.2.2 describes how the sensor types were specified, and Sections 4.2.3 and 4.2.4 describe the experimental scenario used and AoIs.

4.2.1 Data preparation

A high-resolution digital elevation model was required as input for both the algorithm implementation and multi-resolution technique. The 2013 USFS Lake Tahoe 0.5m DEM from ©OpenTopography was selected given it's large areal coverage, fine resolution, and mountainous terrain type. Additionally, usage of Open Source data was preferred for reproducibility. The Lake Tahoe DEM served as input to the principal's airspace coverage tool and was used to generate a grid of candidate locations at a range of resolutions. The finest resolution (0.5m) DEM contains approximately 22 million candidate points.

4.2.2 Sensor specifications

Two models of sensors were used in this thesis. Each model is based off a general specification for real-world, low-range sensors. Low range sensors were selected as they were most appropriate for the terrain size and resolution used in the experiment. As the focus of the optimization is on combining a number of different objectives, variation within sensor types is limited to just two distinct sensors. This allowed for the price objective of any solution to fall within one of a smaller number of groups, and made interpretation of the resulting 3D Pareto plots clearer. No additional information pertaining to the research objective outlined in Section 1.1 would be elucidated if additional sensor types were included. Table 4.1 contains the field of view, range, and price specifications for the two sensor types A and B. Given that sensor B has a horizontal field of view less than 360°, sensor direction as well as placement was important. Direction was handled by using a random initial value, with possibility for change during the algorithm run based on random mutations.

4.2.3 Experimental Scenario

A small-scale scenario was created and modelled after a real-life civilian use case for testing the algorithms. A civilian use case was selected as opposed to a military one as civilian cases tend to be less complicated and cover a smaller area. A small scenario was better suited for data collection within the time constraints of this thesis. The area selected was a 5km^2 plot south of Lake Tahoe, California. The area contains a flat valley near the center of the DEM, which in civilian use cases could have an industrial complex or small airport which requires surveillance. The surrounding area contains mountains and hills with a number of valleys and passes between them. This scenario is meant to mimic the use case of protecting a base or building of interest from surrounding threats. A mountainous environment represents a challenging area for airspace surveillance given the number of obstructions to line of sight, and as such was selected as an appropriate way to test the strengths of the optimization algorithms.

4.2.4 Areas of Interest

In the 5km^2 scenario described above, the valleys between the mountains in the terrain represent specific Areas of Interest (AoIs). Airborne threats such as civilian drones and small aircraft would be more likely to fly through these passes, and as such monitoring of these areas should be prioritized. The final scenario used with the total area to be monitored as well as two AoIs is shown in Figure 4.1.

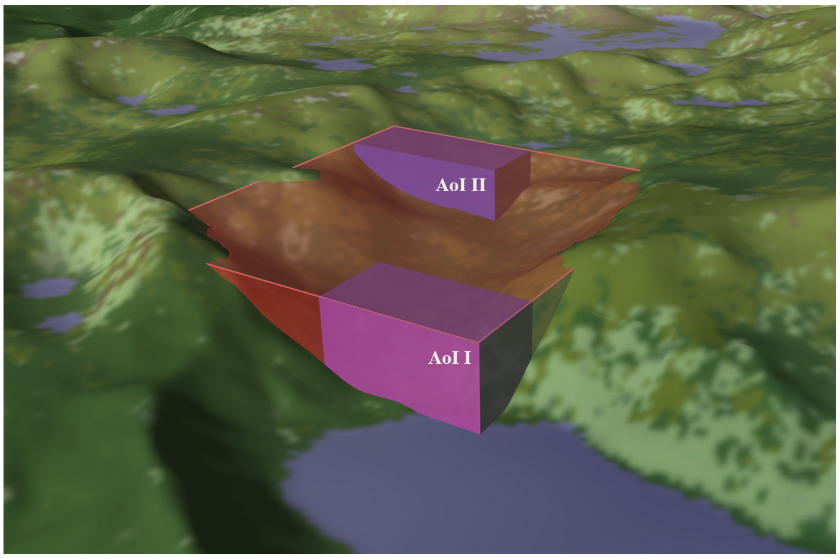

Figure 4.1: Scenario with area to be monitored and two AoIs.

4.3 Algorithm implementation

The following sections describe the implementation for both the NSGA-II and MOACO algorithms. General descriptions for both algorithms were described in Section 2.4.2 and Section 2.4.3 respectively.

4.3.1 NSGA-II

The NSGA-II implementation presented by Deb *et al.* [23] was used. The custom crossover function described by Tigerström [17] was used given the similarity of the research problems. Crossover between two parents was thus implemented by combining all parent sensor locations in a pool and randomly selecting (without repetition) locations to generate child solutions [17]. NSGA-II has three parameters that can be set by the user: population size, number of generations, and mutation rate. Sensitivity analysis was performed to ascertain the best values for these parameters for the specific context.

A population-halving augmentation to the NSGA-II has been successfully used in a previous study [38]. As such, it was decided to implement and test this technique to see if improvements in computation time (without sacrificing solution quality) could be made. Population halving involves measuring the convergence of populations between generations. Once a specific convergence threshold is met, the population size is halved for the remaining generations. To measure convergence, the generational distance metric was used given that it is faster to calculate compared to hypervolume. This lightweight nature means that its use in the heuristic algorithms will not result in drastic increases in computation time. While generational distance can be less accurate than hypervolume in terms of assessing the diversity of solutions, it is still currently the most well-used measure for convergence [14]. The appropriate convergence threshold for this technique was found through testing a range of low and high thresholds and comparing the final solution set quality.

4.3.2 MOACO

The ACO paradigm is a generic algorithm structure that can be tailored to solve a wide range of problems [46]. As no previous examples of using ACO in the observer placement context were found, it was decided to first assess what characteristics the stated research problem has. Recommendations from literature were then located regarding what components and ACO configuration is best suited to this type of problem. An existing algorithm that best matched this configuration was then used as a base for implementation. The main recommendations found from literature included using:

1. Multiple colonies
2. Additional colony for compromise solutions
3. Local search
4. Information sharing between colonies

The use of multiple colonies with independent pheromone and heuristic matrices was recommended for problems where the true Pareto front or solution types needed by a decision-maker are unknown [16]. The use of multiple colonies has shown to be competitive with the other MOACO techniques and in some cases performs better [47, 48, 49]. Furthermore, multiple colonies can reduce the bias that weighted objective schemes have and promote solution diversity [16]. As shown through experimental analysis by López-Ibáñez *et al.* [50], more colonies results in greater exploitation. This is important as the traditional single-colony Pareto-ACO can have bias towards compromise solutions (and disregard edges of the Pareto front) [40]. One downside of heterogeneous multi-colony algorithms is the tendency to focus on the extremes of the Pareto front, while missing more balanced solutions [16]. The mACO-1 algorithm which uses ant groups addresses this problem by adding an additional group that seeks to find compromise solutions according to a weighted sum of the objective-specific groups [39]. An additional colony was thus incorporated into the MOACO algorithm in this thesis which seeks to find compromise solutions. This is achieved by using a non-dominated sorting scheme and crowding comparator to find well-spaced solutions. Thus, colonies exist that seek to find solutions in all areas of the Pareto front.

The local search technique is listed as *optional* in the original metaheuristic. This was added on the recommendation from numerous research articles, which state that ACO algorithms are often strongly enhanced when local search is included [51, 50]. The MOACO algorithm used in this thesis employs a simple k-exchange neighborhood, which is explained in Section 4.3.2.4.

Cooperation has been shown to improve the efficiency of ACO algorithms [39]. Solutions can be shared between colonies through a combined pool. After each colony sorts and selects solutions that maximize their respective objective, the remaining solutions are added to the shared pool. Each colony can then search the shared pool for any solutions found by other colonies that may be an improvement on their solutions.

The general algorithm described by McDonald [52] to solve a multi-objective infrastructure routing problem was found to be most similar to the recommended configuration. MOACO algorithms can be applied to a wide range of problems, and tailoring the algorithm for the observer placement context was feasible. The *unique features* listed by McDonald that can be added to the algorithm for addressing the routing problem (e.g. adding divisions for multiple end points) are not used in this thesis, as they are not applicable to the observer placement problem. The optional local search technique and a simple colony sharing pool were added to the algorithm to address all recommendations from literature.

The algorithm used by McDonald [52] has for every objective O_n, where n is the number of objectives, C_{n+1} colonies. Each colony focuses on solutions for one objective except for colony C_{n+1}, which finds optimal solutions across all objectives [52]. Each colony has a set of ants A_{Cx}, where x is the number of ants in the colony [52]. Each ant uses its colony's pheromone and heuristic information to search for an optimal, non-dominated solution. As the research problem stated in this thesis is not a routing problem with a set end point, each ant explores one new solution for each iteration. Whether the move is permitted depends on both heuristic and pheromone information. The probability of starting a new search from an existing solution is directly related to the amount of pheromone the solution has. If the move does not result in a better solution, the ant enters a state of diversification and moves to a new, random solution in the solution space. Once all ants in a colony have moved to a solution, only the ant at the best solution is permitted to add pheromone.

A parent group Q is used to save all non-dominated solutions found during the algorithm run [52]. The best solution found for each colony is compared after every iteration. If the solution is non-dominated by the parent group, it is added. This parent group of solutions is then returned at the end of the final iteration. The final algorithm implemented employing the stated components is shown in Algorithm 3.

Algorithm 3: Multi-colony MOACO

```
Data: $maxIterations$, $colonySize$, $evaporationRate$
Result: Set of non-dominated solutions.

$T$ ← Initialize pheromone matrix
$n$ ← Initialize heuristic matrix
$C_n$ ← Vector of $n$ colonies
$SP$ ← Shared pool of solutions
$Q$ ← Parent group
while $i < maxIterations$ do
    for $n$ in $C_n$ do
        $C_n$ = get-pareto($C_n$);
        sort-by-objective($C_n$, $n$);
        $V$ = top $C_n$;
        update-Q(top $C_n$;);
        $SP$ = remaining $C_n$;                // Add remaining to shared pool
        $C_n$ = $V$
    end
    for $n$ in $C_n$ do
        $C_n$ = combine($C_n$, SP);            // Combine colony and shared pool
        $C_n$ = get-pareto($C_n$);
        sort-by-objective($C_n$, $n$);
        update-pheromone-matrix($T$);
        apply-local-search($C_n$);;            // Ants in colony exploit or explore
    end
    i=i+1
end
```

4.3.2.1 Decision Policy

The decision policy refers to how ants consider both heuristic and pheromone information as they move through solution space. When choosing to move in a new direction, an ant will combine available data to try and make the best choice. The data available to an ant in this thesis includes colony-specific heuristic information, a pheromone matrix, as well as a local search strategy.

4.3.2.2 Pheromone Update

The pheromone matrix is involved in every iteration when each ant selects a starting point for their search. When an ant explores a new solution, it will do so from the starting point of one of the previously explored solutions with pheromone. The choice of starting solution is probability-based, where solutions with greater pheromone have a greater probability.

The pheromone update strategy used by McDonald [52] was employed in this thesis. Namely, only the top solution in an iteration for each colony has pheromone increased. All other solutions with pheromone then suffer pheromone evaporation. While McDonald [52] uses an evaporation rate of 0.7, a range of values was tested for this thesis to ascertain whether another value was better suited to this research context.

4.3.2.3 Heuristic Information

In this thesis, the heuristic information used in each colony is related to the specific objective that colony seeks to minimize, as well as non-dominance. If a solution an ant is considering moving to is both non-dominated with respect to the other solutions in the colony, and better than at least one solution with respect to the objective of interest, the move is permitted. If the solution does not meet these requirements, the ant enters a "diversification" phase whereby they explore in a new, random direction.

4.3.2.4 Local Search

The local search technique implemented is a simple k-exchange neighborhood. As such, the neighborhood of a solution is considered to be all solutions that differ by only one sensor location. In other words, the neighborhood of a solution with sensor locations A, B and C is all other solutions with at least two locations that are either A, B or C. In the interest of computation time, each ant does not perform a full local search themselves. Each ant explores one solution in the neighborhood of the starting point, and if no better solution is found, they explore in a random direction (to promote diversification). The colony thus explores as many neighborhood solutions as there are ants in the colony in a single iteration. While not an exhaustive list of neighborhood solutions, local searches can be designed to terminate if there is no improvement in a set number of iterations [53]. Thus, the termination in this algorithm is after all ants in the colony have explored at least one neighbor solution.

4.4 Multi-resolution technique

The multi-resolution technique requires two main inputs: resolution levels for candidate points and neighborhood size. Both inputs can have a significant impact on the final solution quality, and as such must be chosen with care. As it was unclear which resolutions and neighborhood size were best, analysis for a range of possibilities was conducted. Candidate point lists were generated for the following resolutions: 50m, 20m, 10m, 5m, & 1m. These resolutions were chosen given the specific scenario of a 5km^2 input area. As used in the original study, an n x n site span around each solution point is used to carry over new candidates [1]. Neighborhood site spans tested included 1, 2, 5 and 10 units as these were seen to represent both small and large neighborhoods for the specific scenario. As the goal of the multi-resolution technique is to reduce total computation time for the heuristic algorithms compared to the single-resolution run, only 2 to 5 resolution levels were applied in a single multi-resolution run for this study. Including a greater number resolutions could result in better solution quality but at the expense of additional computation time. The multi-resolution test results were compared with the results from using only a single, fine resolution list of candidates (1m resolution). The multi-resolution technique uses a convergence measurement to signal when the next run of the algorithm should begin with a finer resolution of points. A convergence value of 20% was used (i.e. the group of individuals had 80% similarity to the previous group), as this was used in the original study by Heyns [1]. When the threshold is reached in the MRA technique, the algorithm is restarted with a finer resolution of candidate points in the neighborhood of the solutions found from the previous run.

4.5 Evaluation

Evaluation of the described algorithms and techniques required both metrics and data. The method used for data collection is described in Section 4.5.1, and the evaluation metrics used are described in Section 4.5.2.

4.5.1 Data collection

Data collection was conducted by running the algorithms with varying parameter values. As parameters can be an infinite number of values, a discrete set of values was required for testing. Each parameter for the algorithms was thus assigned a minimum, maximum, and increment (step) value. The parameters were then changed one at a time to complete a full factorial experiment. Every possible combination of parameter values within the specified ranges was thus tested. A fully crossed factorial experiment was used given that this allows for quantification of interactions between predictor variables with high precision [54]. The precision and detailed analysis that comes from a factorial design was seen as more favourable than alternatives that provide faster data collection. Specifically, the graeco-latin square and fractional factorial designs were also considered [55, 56]. However, no interaction effects between predictor variables can be discovered with these approaches. Furthermore, it is recommended that the graeco-latin square be used for an initial test to find suitable ranges of parameter values, and then to use a full factorial design on a more precise range [54].

One downside to using a fully crossed factorial design was that the long computation time required for data collection allowed for only a single run for each combination of parameter values. As such, robustness through using replicates was not present. However, general trends found in the data are still considered to be reliable given the number of runs tested and step sizes used.

The ranges for each parameter were decided upon through ad-hoc exploratory testing after initial algorithm implementation. Early testing used median parameter values employed by Tigerström [17] in a similar study. A measure of convergence was implemented into the algorithms to see whether these parameters were sufficiently large to result in near optimal solutions. Convergence in this context refers

Table 4.2: Start, stop and step value for parameters in Experiment I.

	Parameter	Start	Stop	Step
NSGA-II				
	genMax	1	20	+2
	popSize	10	100	+10
	pMut	0.0	1.0	+0.1
MOACO				
	maxIterations	1	20	+2
	colonySize	5	50	+5
	evapRate	0.1	1.0	+0.1

to when two generations have populations that have a generational distance within 20% of each other. This convergence threshold was selected given its success in a related research study [1]. It was found that with a population of 50, 10 generations, and a 0.1 mutation rate, convergence generally occurred around generation 7. As such, parameter ranges were chosen to include values both smaller and larger than this median. The maximum values chosen also involved consideration of computation time, as a very high computation time was undesirable, and the presence of convergence suggested that significantly higher parameter values would not be necessary.

Given this initial exploratory work, the final parameter values tested for the initial full factorial experiment (called Experiment I) are listed in Table 4.2. Further analysis was then required for the population-halving technique (Experiment II) and Multi-resolution technique (Experiment III). The population-halving experiment was conducted by using fixed median values for the algorithm parameters, and testing a range of convergence thresholds. As a convergence measure is required for the multi-resolution technique, a *colony*-halving method was also tested for the MOACO algorithm in Experiment II. This was completed by measuring convergence of the parent group Q between iterations. The specific values used are shown in Table 4.3. The multi-resolution experiment functioned similarly to the other experiments, with the parameters and convergence being given a fixed value and the neighborhood span and resolution levels being varied over a number of runs. The values and steps used for this experiment are shown in Table 4.4.

4.5.2 Metrics

A number of metrics were used for comparing the algorithms and techniques. The first metric used was computation time. Lower computation time is favourable for analysts when making time-sensitive decisions, and as such knowing how each algorithm parameter affects computation time was important. The hyperarea, spread, and generational distance metrics described in Section 2.4.1 were used for assessing the quality of the algorithms. The open-source *mco* R package version 1.15.6 was used for hyperarea, spread, and generational distance calculations. A theoretical Pareto front was generated for the scenario by running the NSGA-II algorithm for

Table 4.3: Parameter values for Experiment II.

	Parameter	Start	Stop	Step
NSGA-II				
	genMax	10	-	-
	popSize	50	-	-
	pMut	0.1	-	-
	convergence	5%	30%	+1%
MOACO				
	maxIterations	7	-	-
	colonySize	40	-	-
	evapRate	0.5	-	-
	convergence	5%	30%	+1%

Table 4.4: Parameter values for Experiment III.

	Parameter	Values
NSGA-II		
	genMax	10
	popSize	50
	pMut	0.1
	Convergence threshold	80% similarity
	Resolution levels (m)	Listed in Table 4.5.
	Neighborhood size	1, 2, 5, or 10 unit site span.
MOACO		
	maxIterations	7
	colonySize	40
	evapRate	0.7
	Convergence threshold	80% similarity
	Resolution levels (m)	Listed in Table 4.5.
	Neighborhood size	1, 2, 5, or 10 unit site span.

Table 4.5: Multi-resolution level groups for Experiment III.

Resolution level groups (m)	1, 5
	1, 10
	1, 20
	1, 50
	1, 5, 10
	1, 5, 20
	1, 5, 50
	1, 10, 20
	1, 10, 50
	1, 20, 50
	1, 5, 10, 20
	1, 5, 10, 50
	1, 5, 20, 50
	1, 10, 20, 50
	1, 5, 10, 20, 50

a very large number of generations. Every solution assessed in the algorithm was then collected into a list. Once the algorithm was terminated, the Pareto front of this large set of solutions was found and used as a theoretical "true" Pareto front. It is important to note that this Pareto front of solutions is not the actual collection of optimal solutions, and as such the quality metrics that use this front for calculation have accuracy concerns.

5

Results

This chapter describes the results from the data collection procedure described in Section 4.5.1. Section 5.2 contains the results for Experiment I, Section 5.3 contains results for Experiment II, and Section 5.4 contains results for Experiment III. Lastly, Section 5.5 provides an overall comparison of the two algorithms.

All experiments were conducted on a Dell Precision 5540 with 32 GB of RAM and an Intel i7-9850H processor with six cores of 2.60 GHz.

5.1 Estimated True Pareto Front

As some metrics require a "true" Pareto front for measuring the quality of a solution, an estimated Pareto front was constructed by compiling every solution assessed during a long ($>$ 1 hour) NSGA-II run using a large population and high number of generations, and then retrieving the Pareto front of that set. It should be reiterated that this front is only an estimation, and does not represent the true collection of optimal solutions for the problem. As such, metrics using this estimated front do suffer from accuracy concerns. Figure 5.1 shows the final Pareto front in a 3-dimensional plot. The value of the 4th objective (price) was shown using colour coding to allow for the data to be visualized in a single 3D plot, rather than a set of 2D or 3D plots. A single plot allowed for clearer visual interpretation of the data.

5.2 Experiment I: Metrics

The aim of Experiment I was to investigate any association between the NSGA-II and MOACO algorithm parameters and the time, spread, hypervolume, and generational distance metrics. Furthermore, identification of any interaction effects between the parameters was also of interest. For this purpose, multiple linear regression was used which employed QR decomposition.

A multiple linear regression allows for predicting a response variable y from multiple predictor variables x. In this experiment, the response variable y is one of the quality metrics (time, hyperarea, spread or generational distance), and the predictor variables are the algorithm parameters. The null hypothesis used was that a parameter has no effect on the quality metric. As a full factorial design was employed, multiple regression allowed for investigation of potential interaction effects between the algorithm parameters. Parameters with significant results (i.e. p-value less than

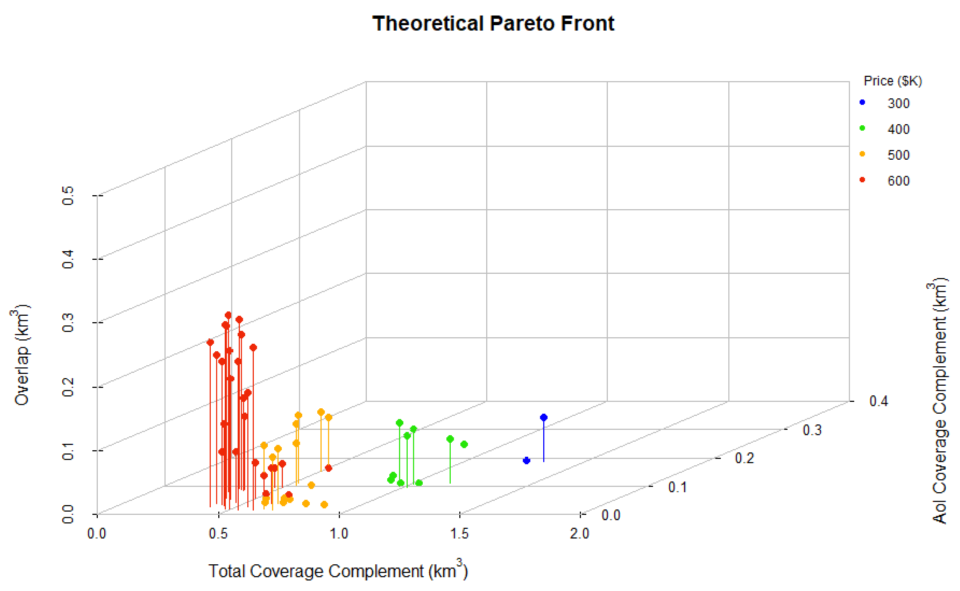

Figure 5.1: Estimated true Pareto front.

or near 0.05) are highlighted in the result tables using an asterisk (*) for clarity. A multiple linear regression for two independent predictor variables results in the following equation:

$$y = b_0 + b_1x_1 + b_2x_2$$

whereby b represents the regression coefficient (association). The regression coefficient is, in other words, the average effect of a one unit increase in x on y [57]. Values further away from 0 thus have a stronger impact on the outcome variable.

For multiple regression considering *interaction* effects between two predictors, the following equation is then used:

$$y = b_0 + b_1x_1 + b_2x_2 + b_3(x_1x_2)$$

where b_3 would represent the increase in effectiveness of x_1 for a one unit increase in x_2 (and vice-versa) [57]. Three-way interactions (e.g. $x_1x_2x_3$) are typically not used as they can be difficult to interpret [57].

5.2.1 Time

Table 5.1 shows the multiple regression results between the independent parameters and computation time. The *Estimate* field represents the regression coefficient (b). The R^2 value (between 0 to 1) was included to indicate how well the line was fitted to the data. Values closer to 1 indicate a better fit. Table 5.2 shows the two-way interaction effects of the parameters associated with computation time.

Table 5.1: Independent time metrics.

Algorithm	Parameter	Estimate	Std. Error	t. value	Pr(>|t|)
NSGA-II					
	popSize	15.67	0.41	38.14	<2e-16 *
	maxGen	63.62	2.05	31.10	<2e-16 *
	pMut	2.33	37.79	0.06	0.951
					R^2 : 0.76
MOACO					
	colonySize	25.17	1.13	22.30	<2e-16 *
	maxIter	64.79	2.80	23.13	<2e-16 *
	evapRate	16.19	50.98	0.32	0.75
					R^2 : 0.50

Table 5.2: Two-way interaction time metrics.

Algorithm	Parameter	Estimate	Std. Error	t. value	Pr(>|t|)
NSGA-II					
	popSize	-1.50	1.05	-1.42	0.16
	maxGen	-24.9	5.22	-4.64	4.06e-06 *
	pMut	43.95	109.34	0.40	0.69
	popSize : maxGen	1.57	0.08	18.83	<2e-16 *
	popSize : pMut	-0.99	1.77	-0.56	0.688
	maxGen : pMut	-4.02	8.81	-0.46	0.648
					R^2 : 0.92
MOACO					
	colonySize	6.04	4.20	1.44	0.15
	maxIter	13.83	10.50	1.32	0.19
	evapRate	5.91	221.82	0.03	0.98
	colonySize : maxIter	1.83	0.35	5.28	2.07e-07 *
	colonySize : evapRate	-1.13	7.06	-0.16	0.87
	maxIter : evapRate	1.79	17.82	0.10	0.92
					R^2 : 0.55

Table 5.3: Independent spread metrics.

Algorithm	Parameter	Estimate	Std. Error	t. value	Pr(>|t|)
NSGA-II					
	popSize	-1.09e-03	7.22e-05	-15.13	<2e-16 *
	maxGen	-4.37e-03	3.60e-04	-12.16	<2e-16 *
	pMut	-4.96e-02	6.62e-03	-7.49	1.71e-13 *
					R^2 : 0.34
MOACO					
	colonySize	-0.003	0.0004	-8.97	<2e-16 *
	maxIter	0.006	0.0009	6.99	5.4e-12 *
	evapRate	0.008	0.02	0.47	0.64
					R^2 : 0.15

5.2.2 Spacing

Table 5.3 shows the association results between the independent parameters and spread. Table 5.4 shows the parameter interaction association with spread.

5.2.3 Hypervolume

Table 5.5 shows the association results between the independent parameters and hypervolume. Table 5.6 shows the parameter interaction association with hypervolume.

5.2.4 Generational Distance

Table 5.7 shows the association results between the independent parameters and generational distance. Table 5.8 shows the parameter interaction association with generational distance.

5.3 Experiment II: Metrics

The aim of Experiment II was to augment the NSGA-II and MOACO algorithms with a population and colony halving technique respectively, and investigate the impact on computation time and solution quality. A linear regression was used to investigate whether halving at certain convergence thresholds resulted in significant changes to the run time or final solution quality. No significant effects were found, however some general trends which could be of interest did appear and are shown in Figures 5.2 and 5.3. The important finding is a general reduction in run time without significant impact on the hypervolume, generational distance, and spread metrics. A table showing the regression summary is located in Appendix A.

Table 5.4: Two-way interaction spread metrics.

Algorithm	Parameter	Estimate	Std. Error	t. value	Pr(>|t|)
NSGA-II					
	popSize	-1.12e-03	2.87e-04	-3.89	0.0001 *
	maxGen	-5.34e-05	1.45e-03	-0.04	0.97
	pMut	-9.02e-02	3.09e-02	-2.92	0.004 *
	popSize : maxGen	-6.50e-05	2.33e-05	-2.79	0.0053 *
	popSize : pMut	1.07e-03	4.87e-04	2.19	0.027 *
	maxGen : pMut	-3.81e-03	2.44e-03	-1.56	0.12
				R^2 : 0.37	
MOACO					
	colonySize	-0.002	0.001	-1.49	0.14
	maxIter	0.007	0.004	1.99	0.05 *
	evapRate	0.05	0.08	0.65	0.52
	colonySize : maxIter	-0.75e-04	0.0001	-0.64	0.53
	colonySize : evapRate	-0.25e-02	0.006	-0.40	0.69
	maxIter : evapRate	0.19e-03	0.0002	0.92	0.36
				R^2 : 0.14	

Table 5.5: Independent hypervolume metrics.

Algorithm	Parameter	Estimate	Std. Error	t. value	Pr(>|t|)
NSGA-II					
	popSize	0.03	0.004	7.31	6.13e-13 *
	maxGen	0.17	0.02	8.51	< 2e-16 *
	pMut	1.33	0.37	3.56	4e-4 *
					R^2 : 0.13
MOACO					
	colonySize	0.007	0.004	1.37	0.17
	maxIter	0.03	0.01	2.34	0.02 *
	evapRate	0.11	0.23	0.48	0.63
					R^2 : 0.03

Table 5.6: Two-way interaction hypervolume metrics.

Algorithm	Parameter	Estimate	Std. Error	t. value	Pr(>|t|)
NSGA-II					
	popSize	0.02	0.02	1.12	0.23
	maxGen	-0.001	0.08	-0.02	0.99
	pMut	0.52	1.71	0.30	0.76
	popSize : maxGen	0.002	0.001	1.50	0.16
	popSize : pMut	-0.01	0.03	-0.41	0.68
	maxGen : pMut	0.19	0.14	1.34	0.18
					R^2 : 0.13
MOACO					
	colonySize	0.02	0.02	0.83	0.41
	maxIter	0.04	0.05	0.73	0.46
	evapRate	-0.16	1.05	-0.15	0.88
	colonySize : maxIter	-0.7e-03	0.002	-0.43	0.67
	colonySize : evapRate	0.28e-03	0.03	0.01	0.99
	maxIter : evapRate	-0.45e-03	0.27e-02	-0.16	0.87
					R^2 : 0.03

Table 5.7: Independent generational distance metrics.

Algorithm	Parameter	Estimate	Std. Error	t. value	Pr(>|t|)
NSGA-II					
	popSize	-3.39e-04	2.19e-05	-15.51	<2e-16 *
	maxGen	-2.41e-03	1.10e-04	-22.00	<2e-16 *
	pMut	2.72e-03	1.98e-03	1.38	0.17
					R^2 : 0.47
MOACO					
	colonySize	-3.39e-04	5.45e-05	-6.21	8.29e-10 *
	maxIter	-4.18e-03	1.36e-04	-30.76	<2e-16 *
	evapRate	1.58e-03	2.45e-03	0.64	0.52
					R^2 : 0.60

Table 5.8: Two-way interaction generational distance metrics.

Algorithm	Parameter	Estimate	Std. Error	t. value	Pr(>|t|)
NSGA-II					
	popSize	-1.78e-04	9.14e-05	-1.73	0.08
	maxGen	-1.94e-03	4.52e-04	-4.23	2.03e-05 *
	pMut	-2.87e-03	9.62e-03	-0.30	0.77
	popSize : maxGen	-1.55e-05	7.18e-06	-2.16	0.03 *
	popSize : pMut	-5.35e-05	1.54e-04	-0.35	0.73
	maxGen : pMut	3.51e-06	1.22e-05	0.29	0.77
					R^2 : 0.47
MOACO					
	colonySize	-6.54e-04	2.14e-04	-3.06	0.002 *
	maxIter	-5.00e-03	5.38e-04	-9.28	<2e-16 *
	evapRate	1.14e-03	1.10e-02	0.10	0.92
	colonySize : maxIter	2.60e-05	1.79e-05	1.45	0.15
	colonySize : evapRate	-7.39e-05	3.56e-04	-0.21	0.84
	maxIter : evapRate	-1.51e-04	2.97e-05	0.48	0.63
					R^2 : 0.60

5.4 Experiment III: Metrics

The aim of Experiment III was to apply the multiresolution approach (MRA) as described by Heyns [1] to both the NSGA-II and MOACO algorithms and assess the impact on computation time and final solution quality. It was unclear what configuration of resolution levels and neighborhood span would best suit the data, and as such a large range was initially tested. The best performing configuration was then used for the final comparison against the single-resolution runs. It was evident after collection that the NSGA-II runs with 4 or more resolution levels performed poorly compared to the baseline (single resolution runs) and those with only 2 or 3 resolution levels (plots included in Appendix C). As such, only runs with 2 or 3 resolution levels were thus considered for comparison against the baseline.

For the MOACO algorithm, there were no obvious performance differences between the resolution levels (plots included in Appendix C). As such, runs from all levels (2 - 5) were included for comparison against the baseline. A multiple linear regression was then conducted to investigate any effects of using MRA on the quality metrics. Table 5.9 shows the regression summary. The only significant effect found was that generational distance worsened when MRA was applied to the MOACO algorithm.

5.5 Algorithm Comparison

To obtain a broader picture of the algorithms' effectiveness, scatterplots were created for each algorithm which show all Pareto solutions found during Experiment I. This provides a graphical view of how well the algorithms perform and their coverage of

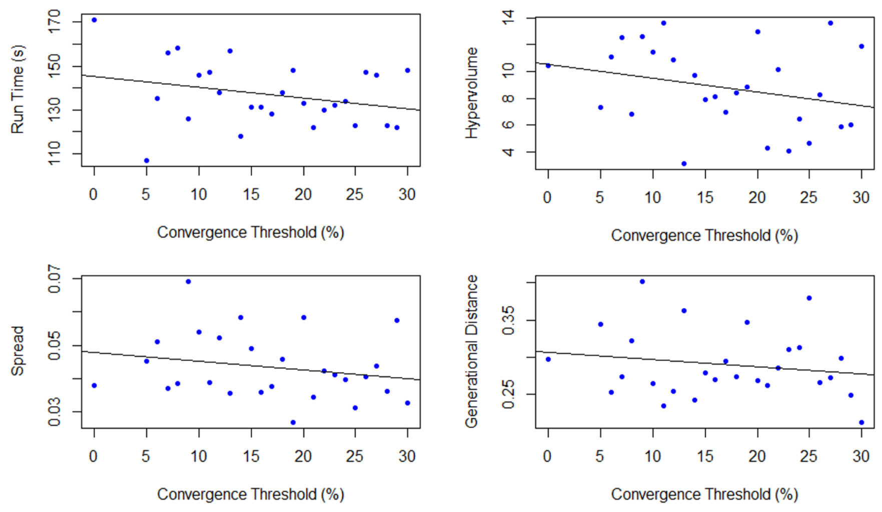

Figure 5.2: Effect of NSGA-II population halving at different convergence thresholds on run time, hypervolume, spread, and generational distance.

the Pareto front. The plots are shown in Figure 5.4. A series of four histograms for each quality metric was then created using the same data from Experiment I to further compare the algorithms. The histograms are included in Figure 5.5.

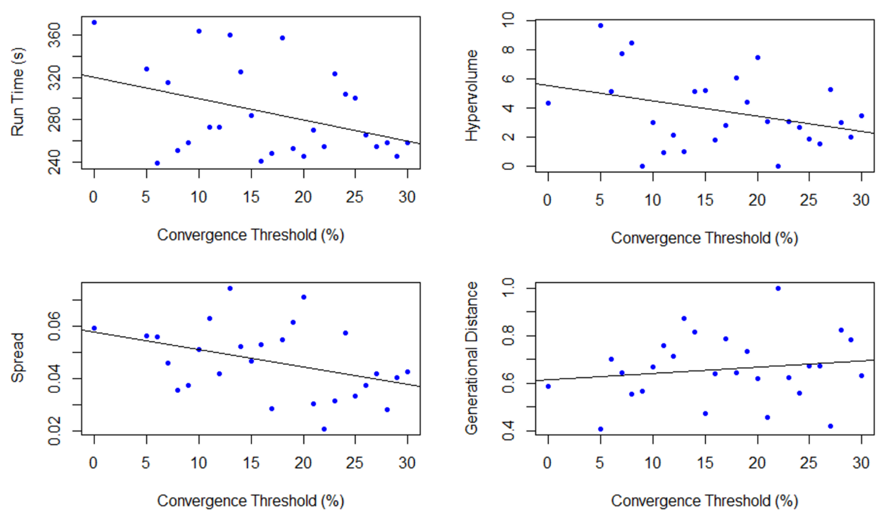

Figure 5.3: Effect of MOACO colony halving at different convergence thresholds on run time, hypervolume, spread, and generational distance.

Table 5.9: Linear regression results between MRA and quality metrics.

Algorithm	**Parameter**	**Estimate**	**Std. Error**	**t. value**	**Pr(>|t|)**
NSGA-II					
	Runtime	4.40e-4	3.54e-4	1.24	0.22
	HV	0.02	0.02	1.44	0.16
	SP	-0.44	0.74	-0.59	0.56
	GD	4.08	3.12	1.31	0.20
					R^2 : 0.16
MOACO					
	Runtime	-1.64e-4	1.97e-4	-0.83	0.41
	HV	0.01	0.01	0.71	0.48
	SP	-0.27	0.28	-0.95	0.35
	GD	3.29	1.39	2.38	0.02 *
					R^2 : 0.14

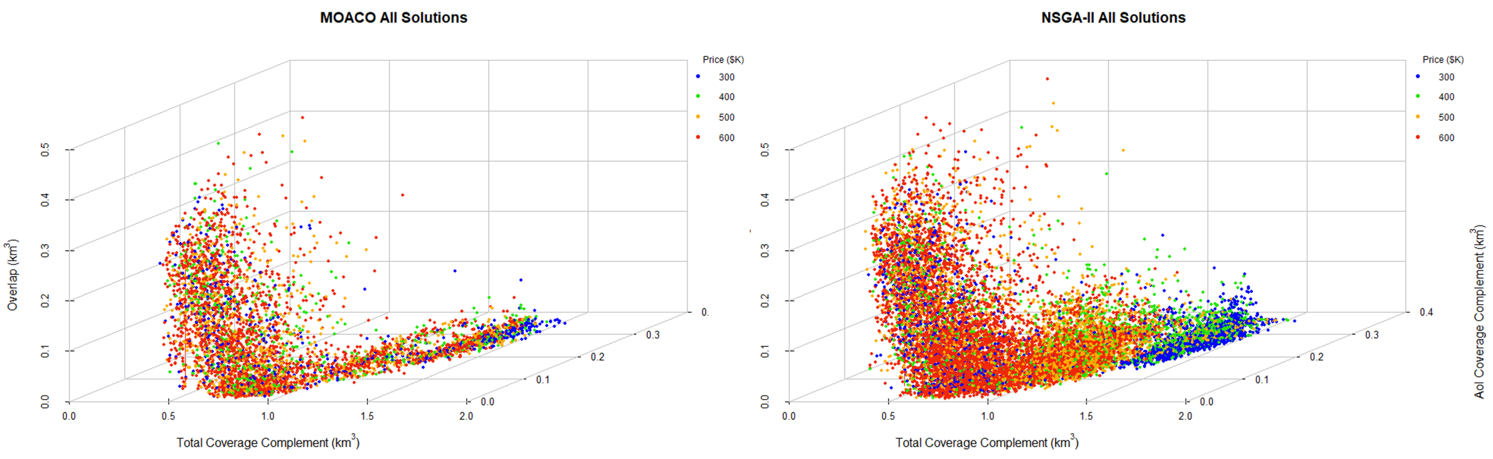

Figure 5.4: Scatterplots showing all Pareto solutions found during Experiment I for NSGA-II and MOACO.

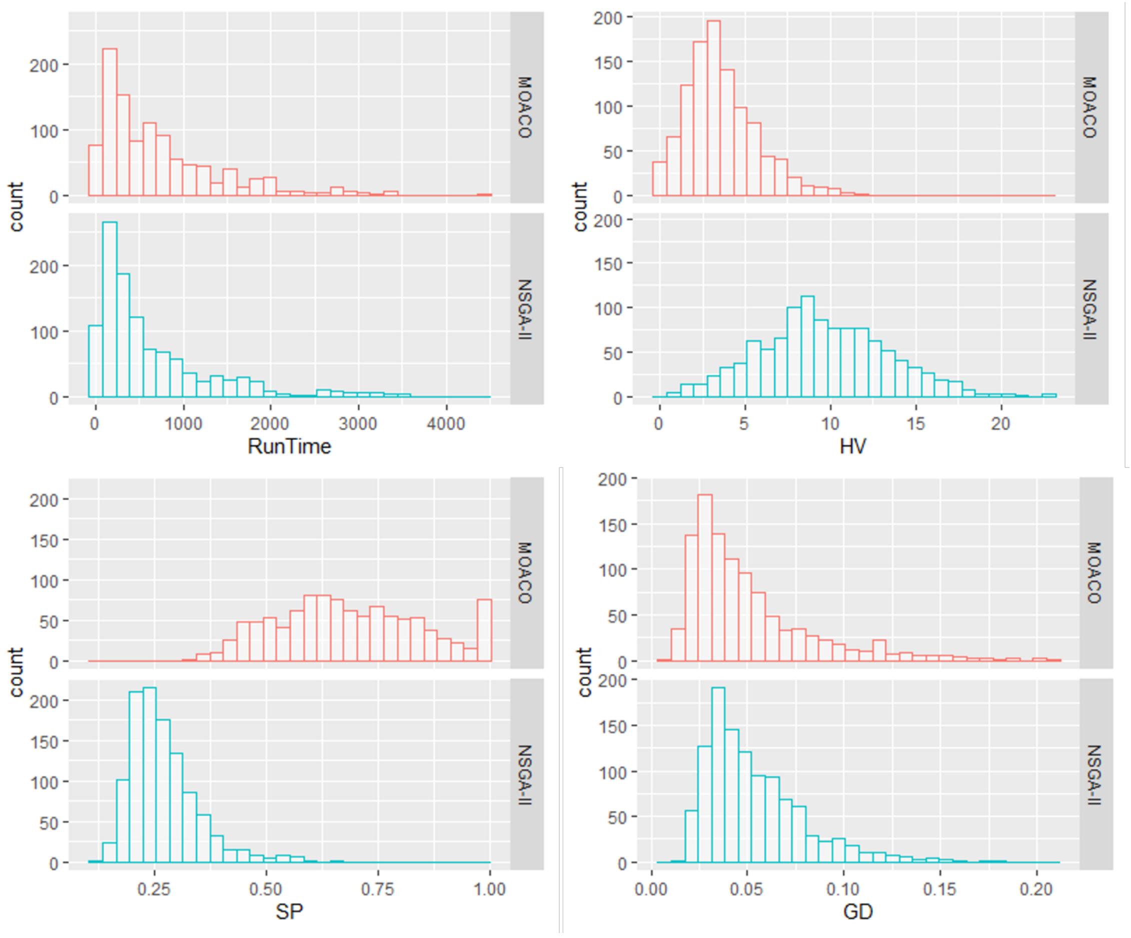

Figure 5.5: NSGA-II and MOACO histogram comparison of quality metrics for all solutions found during Experiment I.

6 Discussion

This chapter contains discussion about the results and research methodology. Section 6.1 describes challenges during implementation. Section 6.2 discusses the experimental results and key findings. Section 6.3 highlights concerns with internal and external validity of the study. Section 6.4 outlines opportunities for future work, and Section 6.5 provides details regarding the ethical concerns of the thesis.

6.1 Implementation

There were a number of challenges during the implementation stage of this thesis.

For the NSGA-II implementation, the concern stated by Tigerström [17] is also relevant in this study. As the same crossover function was used (given the similarity of the studies and desire to compare findings), the fitness of an individual was not the sole factor in how often an individual will mate. During reproduction, four potential parents were chosen from the population at random. The two with the highest fitness were then permitted to reproduce. As such, it is possible that a more sophisticated mating scheme may have resulted in higher quality solutions.

For the MOACO algorithm, it was unclear from the initial planning stages exactly how this should be implemented for the specific research problem. As no previous examples using ACO for this problem were found, implementation decisions were made from more general recommendations found in literature. Furthermore, it was unclear how to handle convergence, as multiple colonies were used. It was decided to measure convergence between the parent group Q in each iteration, however this group often has only small changes between iterations. As such, convergence to the specified thresholds typically occurs quickly (often at only 2 iterations). A more robust convergence mechanism or using a minimum iteration threshold that must be reached before halving can occur could potentially improve the results for Experiment II and III.

6.2 Experimental Results

The following section will outline the main findings, recommendations, and any concerns for each conducted experiment. A concern for all experiments is the lack of replication. Given the large number of runs that had to be completed, replication was not possible for the majority of the tests within the given testing time frame. For instance, a single run of Experiment I for NSGA-II required around 9000 minutes of constant computation time (6.25 days). Given the stochastic and variable nature of the algorithms, replication would have added robustness to the findings and as such this is a concern when making generalizations from the data.

6.2.1 Experiment I

The purpose of Experiment I was to ascertain which algorithm parameters had the greatest effect on run time and the quality metrics and if any interaction effects were present. This knowledge can help fine-tune an algorithm to perform better in a particular way depending on what an analyst desires.

6.2.1.1 NSGA-II

For the NSGA-II, generations had the greatest influence on run time, with population size having the next largest (around half that of generations). There was also a significant interaction effect between these two parameters. As such, ideally they should not both be increased simultaneously. Furthermore, it is recommended to try increasing population size first if trying to improve performance so run times are not increased as drastically.

To improve spread (or diversity) of solutions, increasing the mutation probability provides the greatest impact, followed by an increase in generations. There is an interaction effect between population size and mutation probability, so increasing either of these would also be recommended to improve diversity. Hypervolume (which also measures diversity) is also improved strongly by the mutation rate. Given these results, using a fairly high mutation rate is recommended.

To improve generational distance, increasing generations provides the best improvement. However, there is also an interaction effect between population size and generations, and as such increasing population could also improve generational distance (without the additional run time that another generation adds).

6.2.1.2 MOACO

For the MOACO algorithm, the number of iterations had the greatest impact on run time, with colony size also contributing. These two parameters also have an interaction effect, so ideally both should not be increased simultaneously.

To improve spread or diversity, there is a slight improvement by increasing colony size. However, this is a very small improvement. Adding additional iterations could also worsen the spread. To improve hypervolume, the data suggests that additional iterations could help. However the model had a very poor fit and the p-value (0.02) is fairly high, so this finding is not as reliable. It is evident from these findings that the MOACO configuration used in this thesis suffers from problems with solution diversity.

To improve generational distance, additional iterations have the strongest impact.

6.2.2 Experiment II

The main findings from Experiment II are that the population and colony halving approaches that were applied have no significant impacts on run time, hypervolume, generational distance, or spread. However, reductions in run time between 20s-60s were achieved, and run time was never *increased* as a result of population halving. If this reduction scales with larger parameter values (e.g. additional generations), then at other scales this could result in substantial computation time savings without sacrificing solution quality. However, additional testing would need to be completed to confirm this. Despite the lack of statistical significance, it is still recommended to use this technique at any of the tested convergence thresholds as run time savings were possible with no negative impacts on solution quality.

6.2.3 Experiment III

The results from Experiment III show that the MRA approach has no significant effects for the NSGA-II or MOACO. While a slight ($p = 0.02$) effect of the MRA exists on generational distance in the MOACO algorithm, the regression poorly fits the data and as such the accuracy of this is questionable. Furthermore, the different neighborhood spans (1, 2, 5, or 10 units) had no significant effect on solution quality.

While there were no significant results, there are some trends that are still interesting. Namely, many NSGA-II and MOACO runs using 2 or 3 resolution levels resulted in better hypervolume than the single resolution baseline. The largest improvement was a difference in 10 units, which is fairly substantial. However, it is not guaranteed that every run will have that large of an improvement, and it is also possible that only slight or no improvements will occur.

For the MOACO, it is most notable that when only two resolution steps are used there can be a drastic decrease in run time. For instance, the single resolution run typically requires 600s - 800s, while runs in the 2 resolution level require only 200s - 575s. In the best case, this can mean a much faster computation. However, there is no guarantee that a run will result in a much faster computation.

Given the results for this experiment, the recommendation for the MRA would be that if it is used, 2 or 3 resolution levels would be most appropriate for this data. Furthermore, the quality of the results are unpredictable and could either improve or worsen the solution with respect to run time and solution quality.

6.2.4 Overall Comparison

The scatterplots provided in Section 5.5 visually corroborate the results found in Experiment I. Namely, that the MOACO algorithm suffers from solutions that lack diversity across the full Pareto front. The extremes are found well in both algorithms, however the NSGA-II is better able to find compromise solutions. Furthermore, the comparison histograms show clearly that the NSGA-II performs better in terms of hypervolume and spread (the diversity metrics). An interesting finding from this data is that the MOACO algorithm struggles with objectives that have only a small set of discrete values (e.g. the price objective). We can see in the scatter plot that the NSGA-II finds a greater range of low-cost ($300k) solutions. The MOACO algorithm has far fewer low-cost points, and they are further away from point (0,0,0) (the minimization goal). What seems to occur is that the colony that seeks to find solutions which minimize the price objective find low-cost solutions quickly, and these initial solutions (which are usually not optimal with respect to the other objectives) are still considered "ideal" with respect to price, and kept in the colony. As such, they are carried over to the final solution set (as the colony will only ever consider its specific objective). It is thus evident that this configuration of the MOACO algorithm is best suited for objectives that are more continuous in nature, with solutions almost always having a different value.

Despite clear drawbacks of the MOACO algorithm, it is still competitive in terms of run time and generational distance. Furthermore, it is able to find the extremes of the front well. As such, changes to the algorithm to improve diversity and find compromise solutions could result in an algorithm that competes well with the NSGA-II. However, this requires additional research to confirm.

6.3 Validity

There are some important considerations concerning the validity of the study. Section 6.3.1 discusses internal validity (i.e. issues relating to the experiment itself). Section 6.3.2 discusses external validity (i.e. how the findings relate to the broader research context).

6.3.1 Internal validity

Both the data and experimental method impact internal validity. Only one scenario was used during testing, and as such it is unclear how well the algorithms would perform on terrain with different geographical characteristics or size. The results may also be dependent on the specific sensor specifications used. If the sensor ranges are too large for the area of interest, it's possible that the initial random generation of solutions is enough to find optimal sensor locations (and thus the algorithm itself has less influence). It is assumed that the specifications used in this study are an adequate size for the 5km^2 terrain, however further testing using a range of sensor types would be needed to confirm this. The final area of concern is replication. As stated previously, due to the long computation times and number of tests, only a single run was completed for each test. As such, there is no robustness via replication. Given that the algorithms are stochastic in nature, it is possible that certain runs had a better or worse initial random generation of solutions, thus resulting in unusually good (or bad) results.

6.3.2 External validity

The specific configuration used for the MOACO algorithm has not been implemented in other studies, and as such may be difficult to meaningfully compare with other MOACO algorithms or study results. Furthermore, as the airspace observer placement problem has not previously been solved using MOACO, it is possible that another configuration would be better suited. As such, generalizations that NSGA-II is better than MOACO for this problem should not be made without further testing of additional MOACO algorithms.

All data used in this study were retrieved from Open-source repositories in the interest of reproducibility. However, use of the proprietary geospatial SDK (Carmenta Engine) provided by the principal is required to guarantee the same viewshed calculation results. As such, this introduces some barriers to reproducibility.

6.4 Future work

There are a number of recommendations for future research. This study used only a set number of observers to be placed. However, having a variable number of observers that can be placed is perhaps more realistic, and can be incorporated as an additional objective (e.g. minimizing the number of observers). This may have important impacts on the algorithm implementation and complexity. Furthermore, testing using additional scenarios and various terrain sizes or sensor specifications would be valuable for making more concrete assertions about the performance of the algorithms. Additionally, replication should be used to provide additional robustness. The MOACO algorithm used in this study is an augmentation of one specific algorithm. However there are many other configurations that are vastly different in design. As such, a thorough evaluation of relevant configurations for this research problem should be completed before a generalized statement that NSGA-II performs better than MOACO can be made.

Lastly, the local search has shown to be an important factor in the performance of a MOACO algorithm. As such, trying other types of local search methods rather than a simple k-swap method should be tested to see if improvements can be made.

6.5 Ethics

This thesis project derives primarily from the field of military planning. Military ethics is a complicated topic, however it is undeniable that military efforts have resulted in significant casualties and environmental destruction [58]. Advances in military technology have, in some ways, attributed to this destruction [58]. However, war has also been used in the name of justice (e.g. when human rights are being infringed upon). Furthermore, war might be a result of protecting one's nation, rather than attacking another. As such, the ethics of military efforts should be judged on a case-by-case basis.

An important distinction of this project is that it is not a tool or weapon being developed, but algorithms for defense planning. The product will not directly harm human beings or the environment. The aim instead is to better survey and protect an area from hostile aircraft. As such, the goal is to help reduce the chance of armed combat and casualties. Furthermore, the algorithms developed in this thesis can be applied to other location-allocation problems such as the optimal placement for fire watch towers. The products of this thesis therefore also aid planning that strives to *reduce* casualties and environmental destruction.

The main areas of risk of this project are uncertainty with the analysis and use by unintended persons. The quality of the solution (observer locations) that the algorithms find relate directly to the quality of the input parameters. It is important to note that if out-of-date or less accurate terrain models are used, the solution will not be as well-suited for the real conditions of the area. As such, users may have a false understanding of how much of the airspace is being covered and make planning and defense decisions based on these assumptions. In the worst case these assumptions can result in hostile aircraft entering undetected and causing casualties. "Unintended" users may refer to hostile groups/enemies or non-qualified personnel. Hostile groups could apply optimization algorithms to improve their own security, leading to areas that are more difficult to breach. This could result in enemies being given a greater amount of time to cause damages before they are apprehended. If unqualified personnel use these algorithms for planning, they could use or interpret them incorrectly and make false assumptions regarding security. As previously mentioned, these assumptions could lead to destruction or death in the worst case.

The ethical concerns and risks outlined above were taken into consideration during this project.

7
Conclusions

This thesis project had two main goals. The first was to implement two multi-objective methods to find optimal placement for airspace observers. The second was to apply techniques to reduce the computation time without sacrificing solution quality. The problem was first formalized into a multi-objective optimization problem. Two optimization algorithms, the NSGA-II and a MOACO algorithm, were then implemented. A population halving technique and a multi-resolution approach (MRA) were then applied to try and improve computation time. Evaluation was conducted through parameter tuning, statistical analysis and visual plotting to find trends in the data.

The results suggest that the NSGA-II performs better in terms of solution quality and diversity compared to the MOACO algorithm for the scenario presented in this thesis. However, the algorithms performed similarly with respect to run time and generational distance. The population halving technique provided no significant improvements. However, there was a general trend suggesting a reduction in computation time. Similarly, applying the MRA resulted in no significant improvements, but using 2 or 3 resolution levels could provide a reduction in computation time with no impact on solution quality in best-case scenarios. Given the stochastic nature of these algorithms, it is challenging to generalize or predict the effects of these techniques with certainty. However, the potential for run time improvements for both techniques is present, and as such are seen as viable methods for decreasing computation time without sacrificing solution quality. The two optimization algorithms implemented in this thesis are considered satisfactory methods for finding optimal placements for airspace observers in a multi-objective context.

Bibliography

[1] A. Heyns and J. H. van Vuuren, “A multi-resolution approach towards multi-objective gis-based facility location,” 2014.

[2] R. Thornton, *Asymmetric warfare: Threat and response in the 21st century.* Polity, 2007.

[3] Y.-H. Kim, S. Rana, and S. Wise, “Exploring multiple viewshed analysis using terrain features and optimisation techniques,” *Computers & Geosciences*, vol. 30, no. 9-10, pp. 1019–1032, 2004.

[4] Carpentries", “Introduction to Raster Data – Introduction to Geospatial Concepts.”

[5] L. Croneborg, K. Saito, M. Matera, D. McKeown, and J. van Aardt, “Digital elevation models,” 2020.

[6] F. Hurtado, M. Löffler, I. Matos, V. Sacristán, M. Saumell, R. I. Silveira, and F. Staals, “Terrain visibility with multiple viewpoints,” *International Journal of Computational Geometry & Applications*, vol. 24, no. 04, pp. 275–306, 2014.

[7] D. Fearon, “Alfred weber: theory of the location of industries, 1909,” *Ŕ Center for Spatially Integrated Social Science*, 2006.

[8] A. T. Murray and D. Tong, “Coverage optimization in continuous space facility siting,” *International Journal of Geographical Information Science*, vol. 21, no. 7, pp. 757–776, 2007.

[9] S. Şişbot, Ö. Turgut, M. Tunç, and Ü. Çamdalı, “Optimal positioning of wind turbines on gökçeada using multi-objective genetic algorithm,” *Wind Energy: An International Journal for Progress and Applications in Wind Power Conversion Technology*, vol. 13, no. 4, pp. 297–306, 2010.

[10] J. Macknick, T. Quinby, E. Caulfield, M. Gerritsen, J. Diffendorfer, and S. Haines, “Geospatial optimization of siting large-scale solar projects,” tech. rep., National Renewable Energy Lab.(NREL), Golden, CO (United States), 2014.

[11] H. A. Shehadeh, M. Y. I. Idris, I. Ahmedy, and H. R. Hassen, “Optimal placement of near ground vhf/uhf radio communication network as a multi objective problem,” *Wireless Personal Communications*, vol. 110, no. 3, pp. 1169–1197, 2020.

[12] A. Gupta, K. A. Pati, and V. K. Subramanian, “A nsga-ii based approach for camera placement problem in large scale surveillance application,” in *2012 4th International Conference on Intelligent and Advanced Systems (ICIAS2012)*, vol. 1, pp. 347–352, IEEE, 2012.

[13] L. Li, “Multi-objective a* route optimization for terrain vehicles,” 2020.

[14] N. Riquelme, C. Von Lücken, and B. Baran, "Performance metrics in multi-objective optimization," in *2015 Latin American Computing Conference (CLEI)*, pp. 1–11, IEEE, 2015.

[15] Y. Collette and P. Siarry, *Multiobjective optimization: principles and case studies.* Springer Science & Business Media, 2004.

[16] D. Angus and C. Woodward, "Multiple objective ant colony optimisation," *Swarm intelligence*, vol. 3, no. 1, pp. 69–85, 2009.

[17] G. Lördal Tigerström, "Automated decision support for placing terrain observers," 2019.

[18] Y. Liu, J. Wei, X. Li, and M. Li, "Generational distance indicator-based evolutionary algorithm with an improved niching method for many-objective optimization problems," *IEEE Access*, vol. 7, pp. 63881–63891, 2019.

[19] C. Darwin, *The origin of species.* PF Collier & son New York, 1909.

[20] J. R. Sampson, "Adaptation in natural and artificial systems (john h. holland)," 1976.

[21] L. J. Fogel, A. J. Owens, and M. J. Walsh, "Artificial intelligence through simulated evolution," 1966.

[22] G. Gunter Rudolph, "Parallel approaches to stochastic global optimization," in *Parallel Computing: From Theory to Sound Practice, Proceedings of the European Workshop on Parallel Computing*, pp. 256–267, Citeseer, 1960.

[23] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[24] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 1, pp. 29–41, 1996.

[25] J. S. Angelo and H. J. Barbosa, "On ant colony optimization algorithms for multiobjective problems," *Ant Colony Optimization Methods and Applications*, vol. 53, 2011.

[26] S. Rana, "Fast approximation of visibility dominance using topographic features as targets and the associated uncertainty," *Photogrammetric Engineering & Remote Sensing*, vol. 69, no. 8, pp. 881–888, 2003.

[27] S. Bao, N. Xiao, Z. Lai, H. Zhang, and C. Kim, "Optimizing watchtower locations for forest fire monitoring using location models," *Fire safety journal*, vol. 71, pp. 100–109, 2015.

[28] Y. Wang and W. Dou, "A fast candidate viewpoints filtering algorithm for multiple viewshed site planning," *International Journal of Geographical Information Science*, vol. 34, no. 3, pp. 448–463, 2020.

[29] L. De Floriani, P. Marzano, and E. Puppo, "Line-of-sight communication on terrain models," *International Journal of Geographical Information Systems*, vol. 8, no. 4, pp. 329–342, 1994.

[30] K. Kim, A. T. Murray, and N. Xiao, "A multiobjective evolutionary algorithm for surveillance sensor placement," *Environment and Planning B: Planning and Design*, vol. 35, no. 5, pp. 935–948, 2008.

[31] S. V. Magalhaes, M. V. Andrade, and W. R. Franklin, "An optimization heuristic for siting observers in huge terrains stored in external memory," in *2010 10th*

International Conference on Hybrid Intelligent Systems, pp. 135–140, IEEE, 2010.

[32] M. MORIN, "Multi-criteria path planning with terrain visibility constraints," 2010.

[33] L. Babel, "Three-dimensional route planning for unmanned aerial vehicles in a risk environment," *Journal of Intelligent & Robotic Systems*, vol. 71, no. 2, pp. 255–269, 2013.

[34] W. Zhan, W. Wang, N. Chen, and C. Wang, "Efficient uav path planning with multiconstraints in a 3d large battlefield environment," *Mathematical Problems in Engineering*, vol. 2014, 2014.

[35] A. Hojjati, M. Monadi, A. Faridhosseini, and M. Mohammadi, "Application and comparison of nsga-ii and mopso in multi-objective optimization of water resources systems," *Journal of Hydrology and Hydromechanics*, vol. 66, no. 3, pp. 323–329, 2018.

[36] B. Gadhvi, V. Savsani, and V. Patel, "Multi-objective optimization of vehicle passive suspension system using nsga-ii, spea2 and pesa-ii," *Procedia Technology*, vol. 23, pp. 361–368, 2016.

[37] H. Safikhani, S. Nourbakhsh, A. Bagheri, and M. M. Abadi, "Multi-objective optimization of centrifugal pumps using particle swarm optimization method,"

[38] A. Heyns and J. van Vuuren, "Multi-objective optimisation of discrete gis-based facility location problems," *Optimization and Engineering*, 2015.

[39] J. Ning, C. Zhang, P. Sun, and Y. Feng, "Comparative study of ant colony algorithms for multi-objective optimization," *Information*, vol. 10, no. 1, p. 11, 2019.

[40] C. García-Martínez, O. Cordón, and F. Herrera, "A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria tsp," *European journal of operational research*, vol. 180, no. 1, pp. 116–148, 2007.

[41] G. Leguizamón and C. A. Coello Coello, "Multi-objective ant colony optimization: A taxonomy and review of approaches," in *Integration of swarm intelligence and artificial neural network*, pp. 67–94, World Scientific, 2011.

[42] A. Garcia-Najera and J. A. Bullinaria, "Extending acor to solve multi-objective problems," in *Proceedings of the UK Workshop on Computational Intelligence (UKCI 2007), London, UK*, 2007.

[43] J. G. Falcón-Cardona and C. A. C. Coello, "A new indicator-based many-objective ant colony optimizer for continuous search spaces," *Swarm Intelligence*, vol. 11, no. 1, pp. 71–100, 2017.

[44] A. Afshar, F. Sharifi, and M. Jalali, "Non-dominated archiving multi-colony ant algorithm for multi-objective optimization: Application to multi-purpose reservoir operation," *Engineering Optimization*, vol. 41, no. 4, pp. 313–325, 2009.

[45] J.-J. Gonzalez-Barbosa, T. García-Ramírez, J. Salas, J.-B. Hurtado-Ramos, *et al.*, "Optimal camera placement for total coverage," in *2009 IEEE International Conference on Robotics and Automation*, pp. 844–848, IEEE, 2009.

[46] D. Angus, "Niching for ant colony optimisation," in *Biologically-inspired optimisation methods*, pp. 165–188, Springer, 2009.

[47] M. Jalali, A. Afshar, and M. Mariño, “Multi-colony ant algorithm for continuous multi-reservoir operation optimization problem,” *Water Resources Management*, vol. 21, pp. 1429–1447, 09 2007.

[48] L. Melo, F. Pereira, and E. Costa, “Mc-ant: A multi-colony ant algorithm,” vol. 5975, pp. 25–36, 10 2009.

[49] L. Gambardella, E. Taillard, and G. Agazzi, “Macs-vrptw: A multiple ant colony system for vehicle routing problems with time windows in new ideas in optimization, 1999, edited by d.”

[50] M. López-Ibánez, L. Paquete, and T. Stützle, “On the design of aco for the biobjective quadratic assignment problem,” in *International Workshop on Ant Colony Optimization and Swarm Intelligence*, pp. 214–225, Springer, 2004.

[51] M. López-Ibáñez and T. Stützle, “An experimental analysis of design choices of multi-objective ant colony optimization algorithms,” *Swarm Intelligence*, vol. 6, no. 3, pp. 207–232, 2012.

[52] W. McDonald, *A Multi-Objective Ant Colony Optimization Algorithm for Infrastructure Routing.* PhD thesis, Texas A & M University, 2012.

[53] H. H. Hoos and T. Stützle, “Sls methods,” *Stochastic Local Search, The Morgan Kaufmann Series in Artificial Intelligence, Morgan Kaufmann, San Francisco*, pp. 61–112, 2005.

[54] R. Myers and E. R. Hancock, “Empirical modelling of genetic algorithms,” *Evolutionary computation*, vol. 9, no. 4, pp. 461–493, 2001.

[55] *Graeco-Latin Square Design*, pp. 235–236. New York, NY: Springer New York, 2008.

[56] G. E. Box and J. S. Hunter, “The 2 k—p fractional factorial designs,” *Technometrics*, vol. 3, no. 3, pp. 311–351, 1961.

[57] A. Kassambara, *Machine learning essentials: Practical guide in R.* Sthda, 2018.

[58] M. A. Hersh, “The ethics of military work: A guide for scientists and engineers,” *IFAC Proceedings Volumes*, vol. 33, no. 8, pp. 95–106, 2000.

A

Appendix 1. Regression Data for Experiment II

Table A.1: Linear regression results between convergence and quality metrics for Experiment II.

Algorithm	Parameter	Estimate	Std. Error	t. value	Pr(>|t|)
NSGA-II					
	Runtime	-0.08	0.15	-0.54	0.60
	HV	-0.45	0.69	-0.66	0.52
	SP	-176.96	209.04	-0.85	0.41
	GD	-62.89	39.51	-1.59	0.13
					R^2 : 0.50
MOACO					
	Runtime	-0.05	0.05	-1.03	0.31
	HV	-0.73	0.73	-0.99	0.35
	SP	-103.53	135.11	-0.77	0.45
	GD	0.33	13.10	0.03	0.98
					R^2 : 0.63

B

Appendix 2. Regression Data for Experiment III

Table B.1: Linear regression results between number of resolution levels and quality metrics for Experiment III.

Algorithm	**Parameter**	**Estimate**	**Std. Error**	**t. value**	**Pr(>|t|)**
NSGA-II					
	Runtime	4.40e-4	3.54e-4	1.24	0.22
	HV	0.02	0.02	1.44	0.16
	SP	-0.44	0.74	-0.59	0.56
	GD	4.08	3.12	1.31	0.20
					R^2 : 0.16
MOACO					
	Runtime	-1.64e-4	1.97e-4	-0.83	0.41
	HV	0.01	0.01	0.71	0.48
	SP	-0.27	0.28	-0.95	0.35
	GD	3.29	1.39	2.38	0.02 *
					R^2 : 0.14

C

Appendix 3. Resolution Level vs Metric Plots

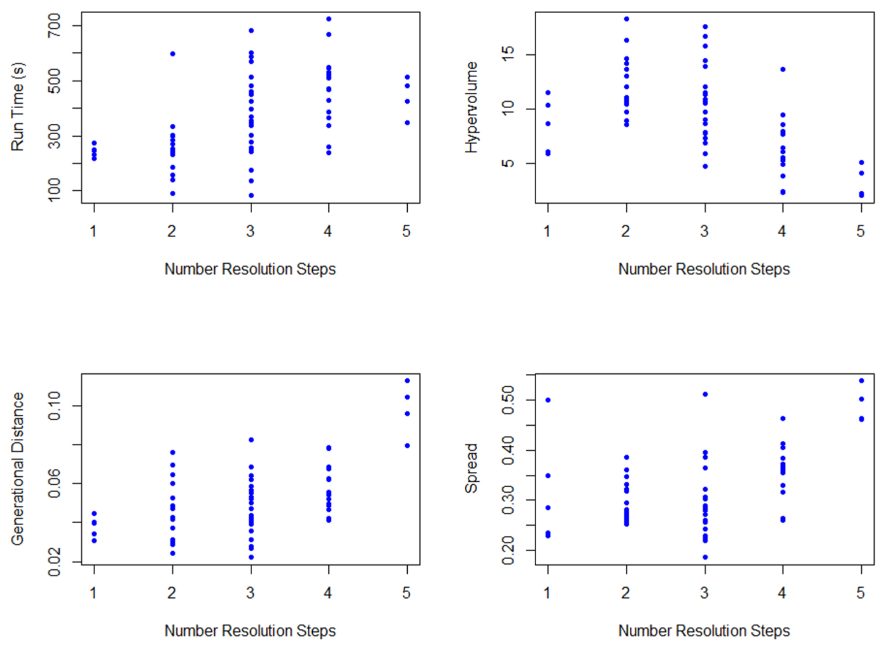

Figure C.1: Effect of NSGA-II and MRA using various resolution levels on run time, hypervolume, spread, and generational distance.

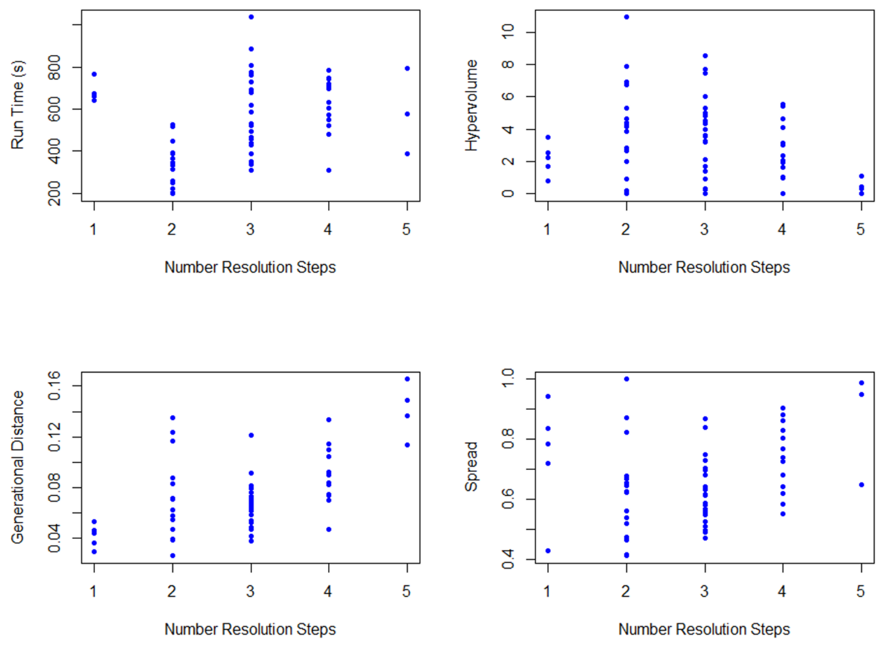

Figure C.2: Effect of MOACO and MRA using various resolution levels on run time, hypervolume, spread, and generational distance.