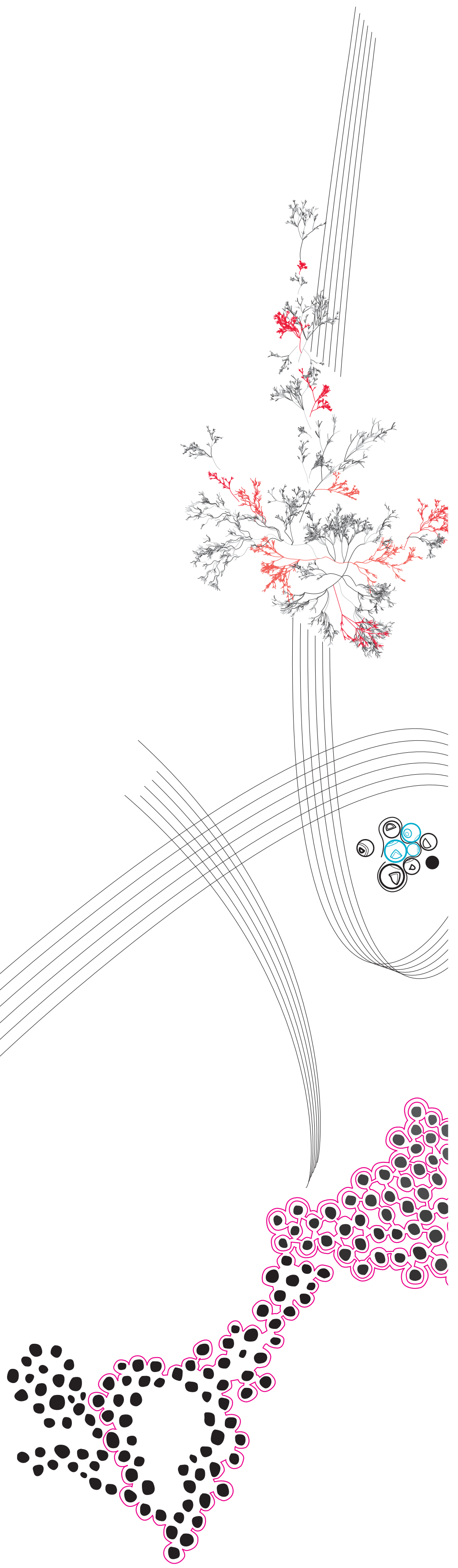


MSc Business Information Technology
Final Project

Preference-based multi-objective optimization: a comparative case study in the Dutch steel manufacturing industry

Ewout van der Wal

Supervisors:
Dr. Renata Guizzardi-Silva Souza (University of Twente)
Dr. Luís Ferreira Pires (University of Twente)
Jochem Verburg (Voortman Steel Group)

July 11, 2024

UNIVERSITY OF TWENTE.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Economic and environmental factors are putting pressure on the business model of the steel manufacturing industry. Companies are exploring ways to improve their processes and use of materials to remain competitive. One approach focuses on optimizing which parts are cut from which stock material, known as the *Nesting Problem*, a sub-problem of the family of *Cutting Stock Problems*. However, current algorithms that focus on a single objective do not meet the demand of the industry. In this context, we investigate a multi-objective optimization approach to the nesting problem. Our approach focuses on incorporating the decision-maker's knowledge and preferences in the optimization. We first present a review of the state-of-the-art literature and an investigation into the factors that affect how a decision-maker evaluates solutions to the nesting problem. Then, we select two evolutionary algorithms based on this review and compare these on the ZDT and DTLZ test sets. From there, we apply one of these algorithms to a case study using cases from Dutch steel construction companies. We show that the multi-objective optimization outperforms the single-objective optimization, and gives the decision-makers more control over which solution they want to accept. We also show that, in the context of the nesting problem, the local optimization and repair steps in the evolutionary algorithm are profoundly impactful on which solutions the algorithm finds. Based on these observations, we postulate that some techniques that work on theoretical test sets may not have a significant impact when applied to real-world nesting cases, and propose various research directions based on this notion.

Keywords: Multi-objective optimization, decision-maker, preference modelling, cutting stock problem, case study

ACKNOWLEDGEMENTS

The first page of this document lists one man as the author of this thesis. However, this does not do justice to the battalion of people who have helped me during the 10 long months it took to write this thesis. Without their support, words of advice, dedication, and love I would not have made it here. In this section, I want to thank everyone who has been here along with me and highlight some in particular.

First, I want to thank Dr. Renata Guizzardi-Silva Souza and Dr. Luís Ferreira Pires for their invaluable support, motivation, and feedback. As my supervisors, they were always ready to answer my questions, ask questions of their own, provide excellent feedback, and guide me forward in the moments when I was lost. Their supervision has shaped my work tremendously.

Second, everyone at Voortman Steel Group and DIGI-STEEL, particularly Jochem Verburg. As my daily supervisor at DIGI-STEEL, he helped me navigate the world of steel machinery and steel construction, asked many deep and domain-specific questions, and kept me on the right track when I inevitably got lost in the steel jungle. Although his comments usually resulted in more work, rather than less, they have been a terrific asset to this work.

Third, my family, in particular my parents and brothers, for their love and support, and for reminding me to take a break now and then. They were the ones that I celebrated my achievements with, but also the ones who carried me through my times of frustration when nothing would work the way I wanted it to. The lessons they have taught me over the years, and their absolute conviction that I would succeed, have helped me get to where I am.

Fourth, everyone at Huize Boslust for listening to my ramblings, having dinner ready when I was too preoccupied with optimization problems to function like a normal person, and creating an amazing place to live. Our house in the forests of the university campus will always have a special place in my heart, along with the people who made it home.

Fifth, Dispuut Yorinf, for the support, reality checks, and great friendships. The drinks on Monday evenings, the vacations, the fun activities, and the many discussions have been a great pleasure to be a part of. Our discussions have been an excellent way to check if my ideas would hold up to scrutiny, because ”With friends like these, who needs enemies?”

Finally, I want to thank everyone else I have had the pleasure of meeting during my career as a student. Including, but not limited to, the friends I have made along the way, the committee members I had the pleasure of working with, the people in the bands I have played in, and the teachers who have guided me to this point. You know who you are.

The first page of this document lists one man as the author, and he couldn’t have done it alone.

1 INTRODUCTION

The Netherlands produced over 6 million tonnes of raw steel in 2022 (Worldsteel (2023b)). Steel frames are an essential part of large construction projects such as offices, warehouses, and various production facilities. Material costs are the biggest cost factor in a steel construction project, with even a small project requiring somewhere in the order of 100 tonnes of steel. Table 1 lists the price per kilogram of steel for various H-profiles. Exposure to movements in price also makes this a big risk factor given that a construction project may take months or years to complete. At the same time, projects have a huge variability in the profiles, dimensions, and lengths of steel beams required. Therefore, most steel manufacturers order steel on a per-project basis and do not hold any general inventory.

Table 1. Steel prices for HEA profiles. Data collected from `https://www.limtrade.nl/` on March 12th, 2024

Profile	Price per meter (€)	Kilograms per meter	Price per kilogram (€)
HEA100	20,76	17,1	1,21
HEA200	53,13	43,2	1,23
HEA450	186,09	142,7	1,30
HEA700	278,28	207,9	1,34
HEA1000	371,04	277,2	1,34

Besides the economic risks, energy usage, carbon emissions, and waste reduction are pressing concerns which need to be addressed for the sustainability and future growth of the steel manufacturing industry. Steel production is an energy-intensive process that requires a great deal of heat. Transporting steel between production, manufacturing, and construction facilities uses trucks that typically run on diesel. All in all, the carbon footprint of steel is large, at 1.89 kilograms of carbon dioxide per kilogram of steel according to Worldsteel (2023a). Given the increased effort of the European Union[1] to reduce the carbon footprint of Europe, the steel industry will have to become more efficient.

Typically, steel suppliers offer beams to the manufacturing industry in length increments of 1 metre, ranging from 6 to 24 metres. However, projects often require parts in varying lengths that do not fit neatly into the supplied lengths. Given this limitation, manufacturers will attempt to cut their parts out of the given beams with as little waste as possible. Optimizing which parts to cut from which beam is known as *nesting*, belonging to the more generic class of *Cutting and Packing problems* Oliveira and Ferreira (1993). Specifically, when nesting into beams it is the *1-Dimensional Multiple Stock Size Cutting Stock Problem (1D-MSSCSP)* (Wäscher et al. (2007)). An alternative name sometimes used in literature is the *Paper Trim Problem*.

1.1 Problem statement

The one-dimensional cutting stock problem, and by extension the nesting process, is known to be NP-Hard (Fang et al. (2023)). The search space tends to become too large to explore exhaustively for problems larger than only a handful of parts and beam lengths. Therefore, finding optimal nestings requires an approach that efficiently explores the search space.

Over the years, various software packages have been created to automate the nesting process. These include DIGI-STEEL (Voortman (2024)), Tekla PowerFab (Tekla (2024)), ConstruSteel (ConstruSteel (2024)), STRUMIS (STRUMIS (2024)), LiemarX (Liemar (2024)), and Steel Projects (FICEP (2024)). These software packages each use a proprietary algorithm to minimize the amount of wasted material in a nesting.

However, there are common problems with the algorithms. First, they only optimize the amount of wasted material without regard for other factors that may influence the evaluation of a nesting. Second, many of these algorithms lack contextual understanding of the problem and give the decision-maker no control over the outcome of the nesting process. This leads to situations where the produced nesting may appear optimal but causes problems during production. Consequently, these problems reduce the effectiveness and acceptance of these nesting algorithms. In this context, the steel manufacturing industry requires a generalized, multi-objective solution to the 1D-MSSCSP that can adapt to the preferences of a decision-maker.

[1]For example, the *Fit for 55* legislation: `https://commission.europa.eu/strategy-and-policy/priorities-2019-2024/european-green-deal/delivering-european-green-deal_en`

1.2 Research questions

The goal of this research is to create a proof of concept implementation of a multi-objective algorithm with preference modelling to optimize nestings. To develop this proof of concept, we answer the following research questions:

RQ1 What factors influence a decision-maker's evaluation of a nesting, and how do we model these factors as the objectives, constraints, decision variables, and parameters of a multi-objective optimization problem?

RQ2 Given a small selection of state-of-the-art multi-objective optimization algorithms that include decision-maker preferences, which algorithm is most suitable to solve the nesting problem?

RQ3 How does the proposed nesting algorithm compare to the existing nesting algorithm in real-world cases, based on nesting performance and decision-maker evaluation?

1.3 Methodology

To answer **RQ1** we perform semi-structured interviews with several expert decision-makers. Based on these interviews, we compile an extensive list of factors influencing the nesting process. From there, we show how these factors may be used to define the objectives, constraints, decision variables, and parameters of a multi-objective optimization problem by creating a proof of concept that covers a subset of the identified factors.

In preparation for this research project, we perform a structured literature review of the field of multi-objective optimization. We select two multi-objective optimization algorithms from the studies collected in this literature review. We implement the algorithms to answer **RQ2** by comparing their performance on various test sets found in the literature and selecting the most suitable algorithm based on their performance.

To answer **RQ3** we apply the evolutionary algorithm selected as a result of answering **RQ2** to the proof of concept problem resulting from **RQ1**. We compare the performance of the algorithm against an existing benchmark on various real-world nesting cases. We then ask domain experts and expert decision-makers to evaluate the algorithm.

1.4 Report structure

The remainder of this report is further organised as follows. Chapter 2 describes the case study context of this report. Chapter 3 describes the structured elicitation of state-of-the-art studies as a background to our work, and our analysis of these literature sources. Chapter 4 describes the interview process, and Chapter 5 describes the implementation and testing of the chosen algorithms. Chapter 6 discusses the evaluation by the domain experts and decision-makers. Finally, Chapter 7 discusses our findings and presents possible directions for future research.

2 CASE STUDY

Our research aims to apply multi-objective optimization to the one-dimensional multi-stock-size cutting stock problem in a real-world setting. In this research, we perform a case study at Voortman Steel Group (VSG). This chapter analyses the organization to give context to our research. We describe the relevant stakeholders, processes, and technologies at Voortman Steel Group and their clients.

Although this analysis is done in the context of one company and its partners, many elements of the case study are generally applicable, since many of the challenges faced by Voortman and their clients represent the challenges of the steel manufacturing sector. The case study is introduced into this research to capture real-world requirements and to serve as an environment for validation

The models and descriptions presented in this chapter are based on multiple conversations with the staff and management at VSG, and information gathered from examining the public-facing websites of VSG and their clients.

2.1 Voortman Steel Group

Voortman Steel Group[2] is a family-owned business that operates in the steel manufacturing industry. Their primary business is configure-to-order production lines consisting of multiple steel manufacturing machines. These machines are connected by automatic rollers that can move steel parts between these machines to enable the automated production of steel parts that require multiple processing steps. Until early 2023, the company had a steel construction division, a machine manufacturing division, and a software development division. To focus on the machine line and software development divisions, the steel construction division was sold to Severfield[3].

The machine manufacturing division produces machine lines consisting of multiple steel processing machines connected by roller belts. The proprietary *Voortman Automation Computer-Aided Manufacturing (VACAM)* software controls these machine lines locally. When presented with a CAM file, VACAM instructs the machine line to automatically produce the requested part(s) from the provided stock material(s).

More recently, in 2018, the Voortman subsidiary DIGI-STEEL started to produce software running in the cloud that can replace and extend the production logic currently implemented in VACAM. This cloud solution aims to create a digital twin of the steel manufacturing line that integrates inventory management, production planning, and manufacturing control in one place. The advantage of the proposed solution is that the VACAM software will only be responsible for controlling the machines and ensuring the correct execution of a production order. At the same time, the business logic is run in the cloud, making it easier to connect to various data sources and introduce new functionality.

2.2 Client companies

VSG primarily serves clients in Western Continental Europe, the United Kingdom, and the United States of America. These client companies are steel manufacturers that own one or more machines built by Voortman Machinery and use the existing automatic beam nesting algorithm in their production planning. These companies are primarily concerned with making parts from steel stock material as efficiently as possible.

Client companies are not only concerned with making steel products as efficiently as possible. An important driver for organizations is the impact of their operations on the people and environment around them. Tighter safety laws, higher consumer sustainability awareness, and the impact of sustainability laws mean that companies are moving towards operating their business more sustainably. They understand that for the business model to survive, they must consider their impact on their surroundings. Along with this, costs are positively driving an overall reduction of waste.

2.3 Production planning

The process we are targeting with our research is production planning, which is the main responsibility of the production planner. Figure 1 is an overview of how a production planning process functions. In practice, each production planner may deviate slightly from what is modelled here, but the overall flow of the process remains the same.

Production planning starts when a project is accepted. Depending on the size of the projects, multiple projects may be planned at the same time to benefit from economies of scale. This is especially useful when multiple small projects require a stock of the same shape, size, and material. The process does not change fundamentally when multiple projects are planned at the same time.

[2] https://www.voortman.net/en/
[3] https://www.severfield.com/

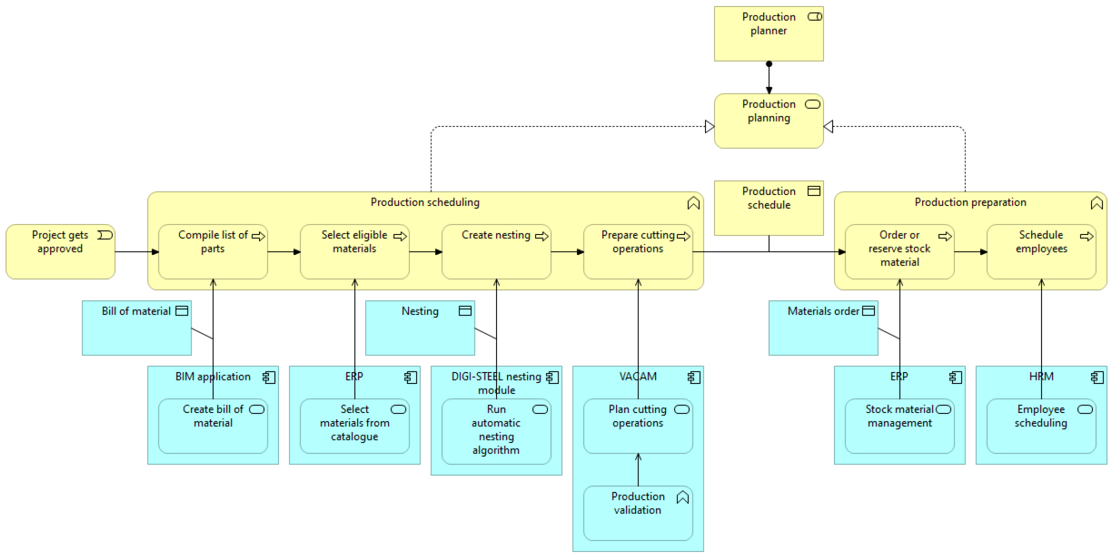


Figure 1. General production planning process.

The size of a steel construction is generally defined by the weight of steel required. Large projects with a mostly homogeneous selection of beam specifications have a different process. Above certain volume requirements, steel beams can be rolled on demand from a steel mill. In this case, steel can be ordered in large batches of one specific length, typically in increments of 10 centimetres. Ordering these rolled steel beams can save 5% to 10% on material costs compared to ordering standard-length stock. At the moment, nesting for these large projects is done by hand, with the help of validation and efficiency calculation tools in the DIGI-STEEL software.

2.4 Production planner

The production planner is responsible for stock material purchasing, human resource planning, and production scheduling at the client company. Production planning often happens on a per-project basis.

Purchasing is done for the project as a whole or in large batches, corresponding to the several phases of a large project. In some cases, the purchasing for multiple small projects is done at once to benefit from economies of scale.

Human resources planning consists of the daily scheduling of production personnel on the production lines. A manufacturing facility with a fully connected and automated Voortman production line only needs a handful of operators manning the machines. Most of the human resources are required at the final manufacturing step, where smaller components are welded to the large beam components that are produced by the automated production line. This assembly step in the production process is the most labour-intensive because it is also the most specialized.

To model the production planner, we use the goal modelling language *i* 2.0*[4] defined in Dalpiaz et al. (2016). This is an update of the original i* language presented in Yu (1997) that consolidates the core concepts of the language. Figure 2 is an i* 2.0 model of the production planner, created using piStar (Pimentel and Castro (2018)).

A production planner's ultimate goal is to create efficient, viable production plans for the workshop. For the sake of this research, we are interested in how the nesting affects this goal, as well as how efficiency is quantified.

Creating a nesting is a required step to complete the production planning. A nesting may be created by hand or by using an automated nesting solution. Ideally, we reduce the number of situations in which the production planner needs to resort to manual nesting by improving the automated nesting solution. An important requirement of the nesting process is that the nesting it produces is valid for the machine line that will ultimately create the parts.

We find that the production planning process is influenced by various cost factors, including material costs, labour costs, and the cost of lost production due to an idle machine. These costs are influenced by many factors, which will often influence multiple costs at the same time. Sometimes, multiple costs are reduced by improving a certain factor. For example, purchasing less steel leads to lower material costs and requires less labour to move

[4]Often written as iStar 2.0 for SEO purposes.

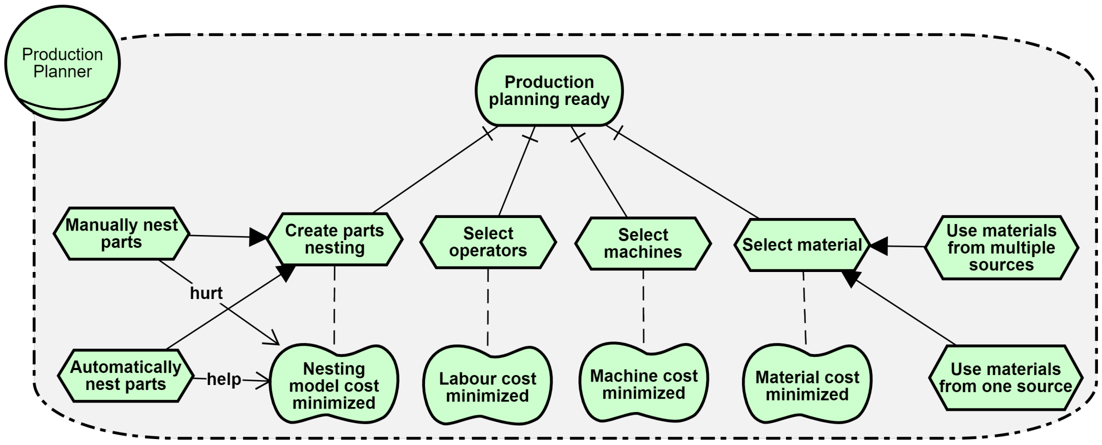


Figure 2. i* goal model of the production planner

the steel in the workshop. Other factors may positively affect one of the costs but negatively impact another. This leads to trade-off decisions for the production planner, where the knowledge and preferences of the production planner will influence their nesting and planning decisions.

3 BACKGROUND

As part of the work that was done leading up to this graduation project, we performed a structured literature review. In this section, we briefly describe the literature review process and the outcomes. Many of the details of the review process have been omitted for the sake of brevity.

The first step in a literature review is to consider the research questions we want to answer. From the problem context, we arrive at the following research questions.

Q1 What techniques are being used to perform requirement prioritization in multi-objective optimization problems?

Q2 What multi-objective optimization techniques have been applied to the cutting stock problem?

Q3 How do these techniques aid in the decision-making process?

A commonly accepted methodology for a literature review is to create a search string that encodes our research questions (Kitchenham (2007)). We performed our search based on the following search string:

> (”requirement* priorit*” OR ”preference* model*” OR ”preference* priorit*”) AND (”multi* objective*” OR ”multiobjective*” OR ”multi* criteri*” OR ”multicriteri*” OR ”multi* attribute*” OR ”multiattribute*”) AND (optim* OR csp OR ”cutting stock” OR ”paper trim” OR moo)

Digital libraries collect, index, and link published research. Most of these libraries also enable researchers to perform queries on their research database using a search string. For our literature review, we queried Scopus, IEEEXplore, and the ACM Digital Library. Together, these libraries represent a breadth of research in general and depth in the relevant fields of study.

As part of the literature review process, we defined the criteria to accept and reject studies. Table 2 details the acceptance and rejection criteria that we used.

Table 2. Acceptance and rejection criteria

Acceptance criteria	**Rejection criteria**
The study is of a relevant topic/subject area.	The study is incomplete.
The study relates to methodologies or techniques. We want to explore existing techniques in the state-of-the-art.	The study is duplicated compared to another selected study or extended in another study. In this case, the most complete study is selected.
The study is published in a venue, a conference or a journal, that requires peer review.	The study presents a technique where the decision maker is not a domain expert.
The study is written in English.	To the best of our abilities, the full-text study cannot be acquired.
The study was published after 2003.	

The final step to collect works from the state-of-the-art is to perform a snowballing step. Wohlin (2014) define snowballing as “using the reference list of a paper [. . .] to identify additional papers”, which we do for one level of references. Like the initial search, we apply the acceptance and rejection criteria in Table 2 to the works resulting from snowballing.

By applying the described search methodology, we arrived at 170 studies on which we performed our analysis. Figure 3 shows the number of studies at each step in the search process.

From here, we analysed the collected work. The goal of the review is to present a comprehensive overview of the breadth of the state-of-the-art in multi-objective optimization with decision-maker preference modelling from 2004 until 2023[5].

The rest of this chapter is structured into sections discussing our findings regarding multi-objective optimization, preference modelling, and validation. Appendix A lists the studies and some of the collected data. Throughout the analysis, we include some examples of relevant studies.

[5]The literature review was performed in October 2023, the newest included work is from September 2023.

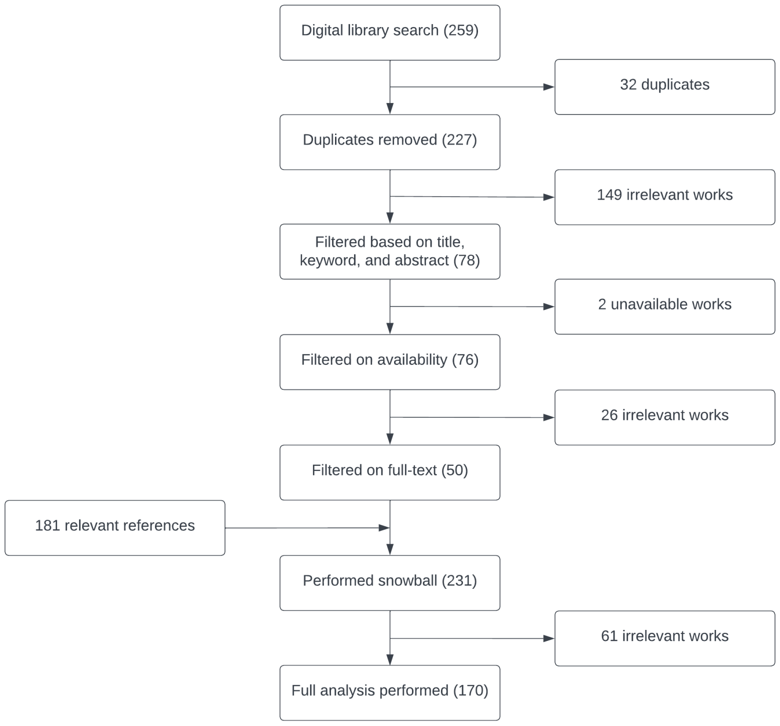

Digital library search (259)
32 duplicates
Duplicates removed (227)
149 irrelevant works
Filtered based on title, keyword, and abstract (78)
2 unavailable works
Filtered on availability (76)
26 irrelevant works
Filtered on full-text (50)
181 relevant references
Performed snowball (231)
61 irrelevant works
Full analysis performed (170)

Figure 3. The full research elicitation process.

3.1 Multi-objective optimization

The general structure of a multi-objective optimization problem consists of a decision vector $\mathbf{x}$ of n decision variables. Without loss of generality, we assume that all decision variables and function outputs are an element of $\mathbb{R}$.

$$\mathbf{x} = \langle x_1, x_2, \ldots, x_n \rangle, \quad \mathbf{x} \in \mathbb{R}^n$$

We define the function $\mathbf{f}(\mathbf{x})$ which maps the vector of decision variables to a vector of m objective functions. In multi-objective optimization, we are concerned with finding the decision vectors that correspond with the most optimal objective vector.

$$\mathbf{f}(\mathbf{x}) = \langle f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x}) \rangle, \quad \mathbf{f}(\mathbf{x}) \in \mathbb{R}^n \mapsto \mathbb{R}^m$$

In optimization, the problem space is bound by p constraints that must be satisfied for the solution to be admissible.

$$g_k(\mathbf{x}) \leq 0, \quad k = 1, \ldots, p, \quad g_k(\mathbf{x}) \in \mathbb{R}^n \mapsto \mathbb{R}$$

Each of the decision functions $f_1, f_2, \ldots, f_m$ can be maximized or minimized, depending on the problem that we model. An optimal (or non-dominated) decision vector results in an objective vector with the property that an improvement in one objective has to come with a loss in at least one other objective. This trade-off property between the objective functions is the foundation of the Pareto dominance principle. Without loss of generality, we define the Pareto dominance relation where $\mathbf{x}$ dominates $\mathbf{y}$ for the case when all objectives are being minimized[6].

[6]We use the notation described by Simon (2013). However, in the literature, we see that both $\mathbf{x} \prec \mathbf{y}$ and $\mathbf{x} \succ \mathbf{y}$ are used as notation for the case where $\mathbf{x}$ dominates $\mathbf{y}$. For example, Rudolph et al. (2014).

$$\mathbf{x} \succ \mathbf{y} \iff \forall i \in \{1,\ldots,n\} : f_i(\mathbf{x}) \leq f_i(\mathbf{y}) \wedge \exists j \in \{1,\ldots,n\} : f_j(\mathbf{x}) < f_j(\mathbf{y})$$

A decision vector is considered non-dominated, or part of the Pareto front, when there exists no other decision vector with a strict improvement in one of the objective values and no loss in all other objective values.

A special kind of Pareto dominance exists between two decision vectors when there is no strict improvement between the vectors. This weak Pareto dominance only requires that a non-dominated decision vector is not strictly worse than any other decision vector in all objectives.

$$\mathbf{x} \succeq \mathbf{y} \iff \forall i \in \{1,\ldots,n\} : f_i(\mathbf{x}) \leq f_i(\mathbf{y})$$

When the size of an optimization problem increases, in terms of decision variables, objective functions, input size or constraints, the Pareto front quickly becomes infeasible to calculate as a whole. Instead, many multi-objective optimization techniques aim to generate a subset of non-dominated decision vectors as a proxy for (a region in) the Pareto front.

In the rest of this section, we describe the size classifications, types of output, search methods, and most used algorithms for performing multi-objective optimization.

3.1.1 Problem dimensions

The objective vector $\mathbf{f}(\mathbf{x})$ in a multi-objective optimization contains two or more objective functions. The literature differentiates three classes of multi-objective optimization problems based on the number of objectives. *Bi-objective* problems have two objective functions (Rudolph et al. (2014); Baykasoğlu (2005)), *tri-objective* problems have three objective functions (Pedro and Takahashi (2014); Ruiz et al. (2020)), and any problem with at least four objectives is considered a *many-objective* problem (Rivera et al. (2022a); Li et al. (2018b)). In extreme cases, the many-objective problems described in the literature can have up to 20 objectives (Goulart and Campelo (2016); Gong et al. (2017)). The widely used synthetic test set DTLZ is often configured for up to 10 objectives (Li et al. (2019); Abouhawwash and Deb (2021)).

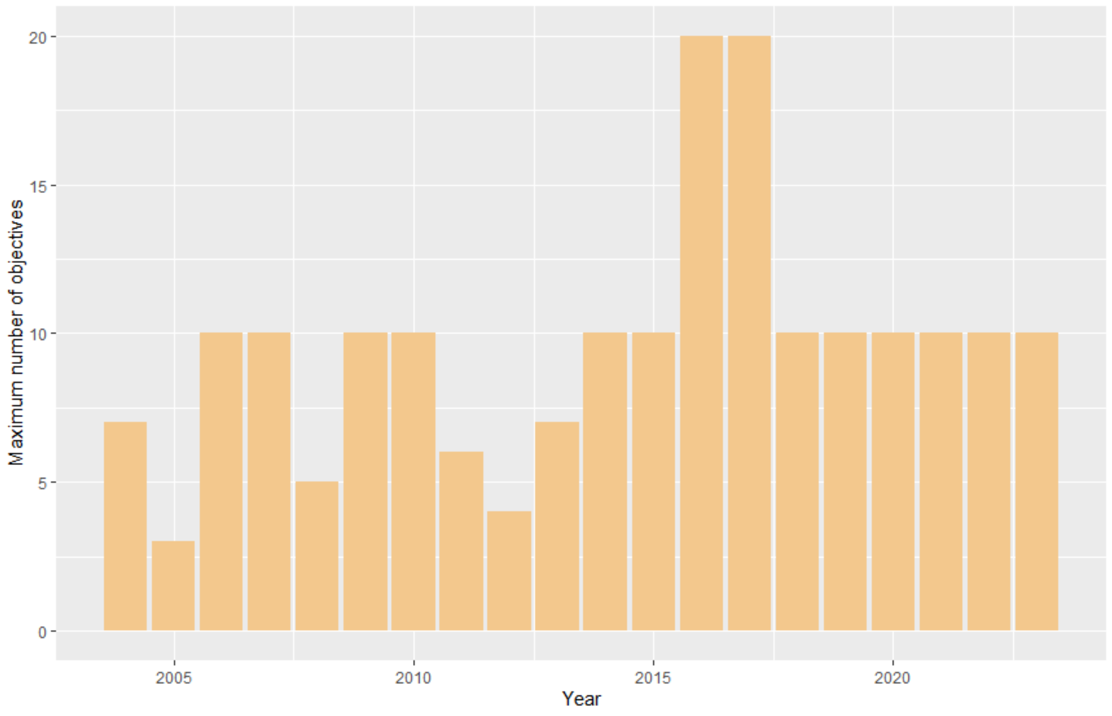


Figure 4. The maximum number of objectives studied per year.

Figure 4 shows the maximum number of objectives considered by the studies in our dataset for each year. Based on this plot, we conclude that researchers have been consistently investigating optimization problems with higher numbers of objectives in recent years.

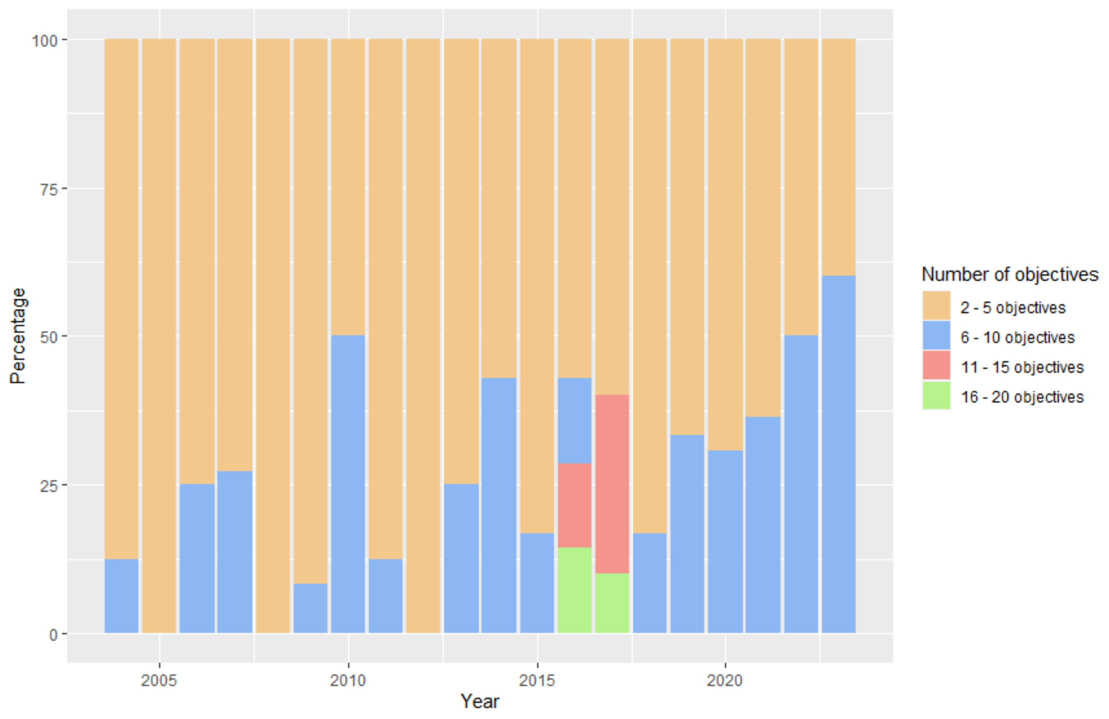


Figure 5. Distribution of the number of objectives per study, per year.

Figure 5 shows the distribution of the maximum number of objectives in studies per year. The graph demonstrates that the percentage of studies that consider at least six and up to ten objectives has increased over the last five years.

Together, Figure 4 and Figure 5 suggest that the state-of-the-art is moving towards multi-objective optimization with at least six and up to ten objective functions. However, studies that consider more than ten objectives are rarely being carried out. A potential explanation for this upper limit is that 10 objectives may be the limit of real-world applicability. A decision-maker may not be able to effectively deal with choices that consider more than 10 objectives.

3.1.2 Model output

The optimization techniques also differ in the output that is presented to the decision-maker at the end of the optimization. Some techniques present one decision vector as the final solution, which has been automatically selected from the Pareto front based on the preferences of the decision maker (Benabbou et al. (2020)). These methods do not require any cognitive effort from the decision-maker in the selection process. However, this method is sensitive to errors in the preference elicitation process.

As an improvement to finding a single solution, researchers started developing search algorithms that instead try to find a set of non-dominated solutions to approximate the Pareto front. The result of the search is a set of Pareto optimal solutions for the consideration of the decision-maker. The decision maker uses a multi-objective decision-making technique to select the preferred solution from this set. The advantage over single-solution methods is that the decision-maker can inspect the solution space to make their final decision, which gives them a better frame of reference for their choice. Approximating the entire Pareto front has been losing popularity (Branke et al. (2016)).

With the problem sizes and the number of objectives increasing, it becomes infeasible to find a set of non-dominated solutions to approximate the complete Pareto front in a reasonable time. In addition, the decision-maker is often not interested in many of these solutions. Many multi-objective techniques search for the region of interest on the Pareto front that closely matches the preferences of the decision maker (Fernandez et al. (2015)). The final set of non-dominated solutions presented to the decision-maker is selected from this region of interest, which ensures that many of the solutions are relevant to the decision-maker, increasing the efficacy of the search. Only approximating a subset of solutions during the search also improves the rate of convergence on the Pareto front.

3.1.3 General search methods

The literature boasts a wide array of search algorithms for multi-objective optimization. These algorithms can be classified into several general methodologies.

The exact methods are inherited from single-objective optimization (Brafman and Chernyavsky (2005); Hunt et al. (2004)). These methods enumerate the search space to find the optimal solution or set of solutions. In bi-objective optimization problems with few variables and constraints, these algorithms can cover the full search space, ensuring that the resulting solution is Pareto optimal. However, many real-world problems cannot be solved by exact methods because of the size of the problem.

Heuristic methods use rule-based logic to explore the search space (Galand and Spanjaard (2007); Kania et al. (2022)). These algorithms create a single solution that is not guaranteed Pareto optimal. The heuristic search depends on the quality of the rules that govern solution construction. This means it requires extensive knowledge of the solution space to create a heuristic model that produces an admissible, near-optimal solution. In many cases, this knowledge is not available.

Local search is an alternative to the heuristic method that does not use rule-based logic (Wiecek et al. (2009); Luque et al. (2009)). Instead, it starts on a (semi-)random admissible point in the search space. It then tries to find the lowest (or highest, in the case of a maximization objective) point in the search space by moving towards the neighbouring solution with the best objective values. This search strategy is effective in convex search spaces, where the search space has a clear global minimum or maximum. However, the algorithms struggle in non-convex search spaces in which they can get stuck in local optima that are not globally non-dominating.

Meta-heuristics solve the issues of the exact, heuristic, and local search methods (Wickramasinghe and Li (2008, 2009)). These search methods use a pseudo-random search strategy to move through the search space. Meta-heuristic implementations do not require any knowledge about the search space to function. This makes them easier to implement compared to heuristic methods. In addition, they do not rely on enumeration of the full search space, which makes them applicable to larger optimization problems than the exact methods. A downside to meta-heuristics is that many implementations approach the search space sequentially. This means the search outcome becomes increasingly dependent on the starting location when optimizing large problems. Many meta-heuristics incorporate a random element that can escape local minima, making it more suitable for non-convex search spaces than local search.

Evolutionary algorithms model the process of evolution from nature (Sudeng and Wattanapongsakorn (2015b); Chugh et al. (2015)). A population of individuals is generated in the search space. Each individual performs an optimization routine and is evaluated according to a fitness function. Then a selection of the fittest individuals from a generation is used to breed new individuals for the next generation. This process continues until a stopping condition is reached. Genetic algorithms are a class of evolutionary algorithms widely used in the literature (Molina et al. (2009); Braun et al. (2017a)). The advantage of evolutionary algorithms over meta-heuristics is the parallelism of having a population of individuals. This enables the evolutionary algorithms to search large problem spaces with high measures of diversity while still being able to converge on the Pareto front in reasonable time frames. On the other hand, some studies have shown that parallelizing the meta-heuristic approach can benefit from parallel execution as well.

The search strategies are not disjoint in the literature. One research direction investigates applying multiple search strategies in a hyper-heuristic (Rivera et al. (2023b); Filho et al. (2018)). These methods use multiple search algorithms in sequence to take advantage of the strengths of the individual algorithms. For example, a heuristic solution with simple rules may be used to create the initial solution for a meta-heuristic to speed up convergence on a high-quality solution.

Another method to combine the strengths of the algorithms is to use a local search step in a meta-heuristic or evolutionary algorithm (Dias et al. (2008)). This local search step converges on the local optimum, increasing the global pressure towards Pareto optimal solutions.

An upcoming research area is applying the Bayesian method to optimization problems (Astudillo and Frazier (2020); Feliot et al. (2019). The advantage of the Bayesian method is the ability to learn one or more black-box functions with no prior bias for the shape of the function. This allows researchers to elicit the preference model of the decision-maker without having to make assumptions about the shape of the underlying preference function.

Figure 6 shows the distribution of the search methods over the reviewed literature[7]. The category *Evolutionary algorithms* contains the evolutionary and genetic methods, and the category *Other* contains local search, Bayesian search, heuristics, and hyper-heuristics. We observe that the evolutionary algorithms are the most-studied search method. The exact methods make up a large portion of the earlier work. However, the state-of-the-art has largely

[7]The bar for 2012 is split 50/50 between *Exact* and *Other* because for this year we only found two studies in the review.

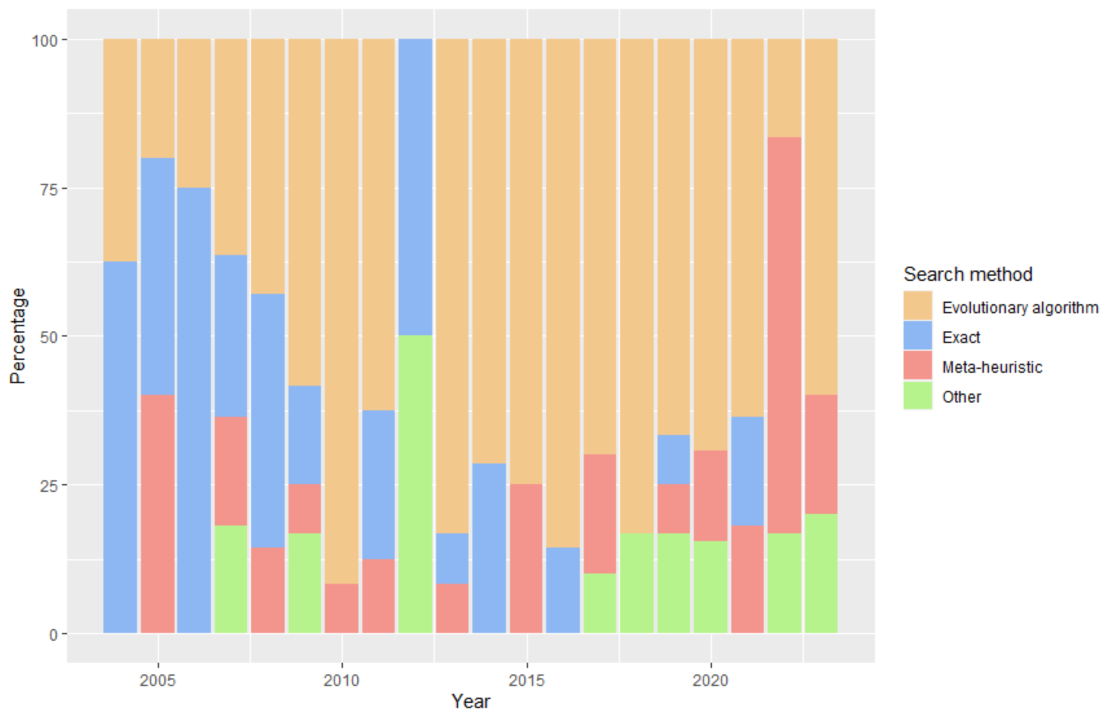


Figure 6. Distribution of the search methods per study, per year.

departed from investigating exact solutions to optimization problems in recent years in favour of evolutionary and meta-heuristic search.

3.1.4 Commonly used search algorithms

The general classes of search methods identified in the previous section consist of various techniques. In this section, we discuss these techniques and the differences between them.

Studies that apply exact search methods to a multi-objective optimization problem show a preference for mathematical programming and goal programming techniques. These studies often present the optimization problem as a linear program or a goal program (Wiecek et al. (2009); Abd El-Wahed and Lee (2006)). These mathematical problems are then solved using the exact algorithms previously developed for single-objective optimisation problems. These algorithms include the Simplex method, branch-and-bound, and cutting planes.

This notion of applying existing single-objective methods to a multi-objective problem persists in the studies that use meta-heuristics. Early work into multi-objective meta-heuristics uses *Simulated Annealing (SA)* to do multi-objective optimization (Aggelogiannaki and Sarimveis (2007)). Later studies show a preference for *Particle Swarm Optimization (PSO)* and *Ant Colony Optimization (ACO)* (Rivera et al. (2023a); Saldanha et al. (2020)). These algorithms, together with the *Firefly Algorithm (FA)* (Trachanatzi et al. (2020)) and several other algorithms, are known as nature-inspired meta-heuristics. These techniques model population-based search methods found in nature. Their biggest advantage over simulated annealing is that the population of search agents can achieve a higher diversity over the search space.

Evolutionary algorithms extend population-based search by repeatedly improving the population in a way that mimics evolution in nature. The state-of-the-art has a few reference algorithms that have been used as the basis for most other studies that present a multi-objective optimization technique. The evolutionary algorithm used most widely as a reference in the literature is the *Multi-objective Evolutionary Algorithm by Decomposition (MOEA/D)* (Hu et al. (2021); Liu et al. (2016)).

An often-seen subset of evolutionary algorithms is the genetic algorithms. The studies we investigated generally favour the *Non-dominated Sorting Genetic Algorithm (NSGA)* (Deb and Kumar (2007a); Braun et al. (2017a)), which is one of the most widely extended and compared-against family of reference algorithms. State-of-the-art algorithms in this family are based on the NSGA-II and NSGA-III reference algorithms.

Other notable reference evolutionary algorithms are the *Necessary-preference-enhanced Evolutionary Multi-objective Optimizer (NEMO)*, *Strength Pareto Evolutionary Algorithm (SPEA)*, *Indicator-based Evolutionary*

Algorithm (IBEA), and *Territory Defining Evolutionary Algorithm (TDEA)* (Branke et al. (2010); Friedrich et al. (2013); Zitzler and Künzli (2004); Pedro and Takahashi (2013)).

Hyper-heuristics are presented in only a handful of studies. We cannot identify a hyper-heuristic that has gained enough traction in the state-of-the-art to be a reference algorithm for many other hyper-heuristics.

3.2 Preference modelling

So far, we have discussed the methods that are available to perform a search over the solutions to a multi-objective problem. In this section, we describe the methods for incorporating the preferences of the decision-maker into the search. We consider the stage of the process when we ask the decision maker for their preferences, the way that preferences are presented to the search algorithms, and the methods by which we may elicit the preference model from a decision maker.

3.2.1 Preference timing

The literature describes three methods by which preference information can be elicited from the decision-maker. These are the *a posteriori*, *a priori*, and *interactive* methods.

Performing a posteriori preference modelling is the most straightforward method to incorporate decision-maker preferences in the search for a solution (Jia et al. (2013); Perera et al. (2013)). First, we use a search algorithm to find an approximation of the Pareto front. Using this approximation of the Pareto optimal front, the decision-maker chooses the solution that best fits their preferences. The decision-making phase can incorporate a decision-making framework to assist the decision-maker in selecting their preferred solution. The advantage of the a posteriori method is that the decision-maker can use the knowledge of the whole Pareto front to select a solution. This can improve the decision-maker's understanding of the trade-offs between objectives. However, the a posteriori method is difficult to apply to problems with a large, complex Pareto front. In addition, the decision-maker may find it difficult to choose from the entire Pareto front.

A priori preference modelling considers the decision-maker's preferences before starting the search for the Pareto front (Park and Koh (2004); Galand et al. (2013)). This method uses preference information from the decision-maker to guide the search algorithm to a single preferred solution or a region of interest on the Pareto front. The advantage of this technique is that the solutions presented to a decision-maker at the end of the search come from a region on the Pareto front that is likely to be relevant to the decision-maker. This method can ignore a large portion of the search space, allowing it to tackle larger problems than the a posteriori method. However, the a priori method requires the decision-maker to have well-defined preferences before the search begins. The literature describes the weakness of the a priori method that in many real-world applications, even when the decision-maker is an expert, the preferences are not known beforehand or are inaccurate.

The interactive preference modelling method tackles the disadvantage of the a priori methods by eliciting the decision-maker's preferences multiple times while searching for the Pareto front (Krettek et al. (2009); Abd El-Wahed and Lee (2006)). Every so often, the decision-maker is presented with information about the best solutions found so far. Based on this information, they can review and adapt their preferences. The interactive method retains the advantages of the a priori method by guiding the search to the region of interest on the Pareto front. At the same time, it also incorporates the knowledge of the search space from the a posteriori method by presenting the decision maker with intermediate solutions to base their preferences on.

3.2.2 Preference encoding

An important aspect of preference modelling is how we determine which solution is most preferred by the decision-maker. The literature knows many methods for encoding preferences over multiple objectives to rank the solutions found during the search for the Pareto front.

Weights are the simplest and one of the oldest methods for reducing multiple objectives to a single objective that can be optimized by classical single-objective means (Hunt et al. (2004); Friedrich et al. (2013)). By some means, the decision maker defines a weight for each objective that can be used to combine the objectives into a weighted-sum, single-objective optimization problem. Then, we can use a single-objective optimization technique to find the optimal solution.

Goal programming is an early method to solve a multi-objective optimization problem without defining a weighted sum that creates a single-objective proxy problem (Wang and Liang (2005); Abd El-Wahed and Lee (2006)). Rather, goal programming encodes the decision-maker's preferences as goal points for each objective. The goal programming methods differ in how these goal points are then used. Some methods try to minimize the total distance between the objective values of the solution and the goal points. Other methods use lexicographic ordering to reach the most important goal(s) first, before optimizing the other objectives.

Decision trees are used occasionally as a preference encoding method (Hu et al. (2021); Cheng and Jia (2021). These are most useful when the components of a solution can be divided into a small, finite set of classes. The decision-maker can define a binary preference relation over the power set of the classes, based on which the technique creates a decision tree. An application of this is when searching for multi-objective spanning trees in finite graphs. Decision trees are difficult to apply to many generalized multi-objective optimization problems because it can be difficult to define a complete set of decision rules accurately.

Besides weights, there are other methods for reducing a multi-objective problem down to a single-objective problem. The techniques using weights implicitly assume that the decision-maker's preference function is linear. However, this is not always the case. Methods that use value functions and utility functions do not need to make this assumption (Ozbey and Karwan (2014); Fowler et al. (2010)). The decision-maker is assumed to follow some function for their preferences. These methods use preference information gathered from the decision-maker to approximate their preference function. This approximation is used to guide the search or select the most optimal solution. A disadvantage of this technique is that the approximation may not be accurate enough in cases with sparse preference information to learn from.

Another option is to define a reference solution or set of reference solutions that the decision-maker considers optimal given their preference model (Deb and Kumar (2007b); Vesikar et al. (2018)). The search for a set of Pareto optimal solutions can be guided towards this reference location. Techniques that incorporate such a reference, often a reference point, need to be aware that the reference is not necessarily Pareto optimal or even feasible. Search methods using a reference point are commonly found in the literature. These techniques incorporate projecting the reference point onto the (approximate) Pareto front to define the region of interest.

Finally, rather than defining explicit preferences, we may only require the decision-maker to produce a ranking of a small set of potential solutions (Cruz-Reyes et al. (2017); Braun et al. (2011)). From there, we create decision rules that are used to rank newly found solutions. These implicit rankings can be used to define the relative fitness between solutions. Higher-ranked solutions are used to guide the search process.

3.2.3 Preference elicitation

The way that preferences are encoded is in part determined by the chosen preference elicitation framework. Reviewing the literature, we identify several methods for preference elicitation, which are often referred to by their acronym (for simplicity, we will refer to these methods as the *acronym methods*). Many of the acronym methods are multi-criteria decision analysis methods that have been applied to multi-objective optimization. In addition to the acronym methods, the literature also describes preference elicitation methods that do not have an acronym-based name. In this section, we describe these findings.

The *Analytic Hierarchy Process (AHP)* by Saaty (1990) uses pairwise comparison to determine the importance between all pairs of criteria (Cordone et al. (2007); Wang et al. (2004)). Relative importance is described by the ratios between the criteria. The method requires a full elicitation of the relative importance between all objectives.

ELECTRE was first described by Benayoun et al. (1966) and has been extended to include the I, II, III, IV, TRI, and IS variants (Rivera et al. (2022b); Fernández et al. (2022)). The ELECTRE III and ELECTRE IV methods create a fuzzy outranking between the solutions. The ranking is based on the strength of the assertion that *a outranks b* for any two solutions *a* and *b* (Figueira et al. (2016)). The decision-maker can set weights, preference thresholds, indifference thresholds, and veto thresholds to influence the ranking.

INTERCLASS is an extension of ELECTRE TRI by Fernández et al. (2020) that can accommodate decision-maker preferences that are described as an interval (Castellanos-Alvarez et al. (2021); Castellanos et al. (2022)). The INTERCLASS method requires the decision-maker to define classes for actions or solutions in the model. Allowing the decision maker to describe the boundaries between classes with intervals accommodates uncertainty.

The *Multi-Atribute Utility Theory (MAUT)* by Keeney and Raiffa (1993) uses information gathered from the decision maker to construct a utility function (Le Huédé et al. (2006); Fouchal et al. (2011)). This utility function captures the total utility of a potential solution to the decision-maker.

The *Preference Ranking Organization Method for Enrichment Evaluations (PROMETHEE)* by Brans et al. (1986) also uses an outranking approach to reach a valued outranking graph (Saldanha et al. (2020); Fernandez et al. (2009)). This allows us to specify a partial pre-order (PROMETHEE I) or a complete pre-order (PROMETHEE II) on the set of possible actions.

TOPSIS by Hwang and Yoon (1981) uses the distance from the ideal solution and the nadir solution to rank the potential solutions to an optimization problem (Yao et al. (2011); Jia et al. (2013)). The solutions are ranked based on the ratio between these distances. The decision maker can influence the ranking by specifying which objective has a higher weight in the distance calculations.

The UTASTAR method by Siskos and Yannacopoulos (1985) indirectly infers preference information from the decision-maker (Trachanatzi et al. (2020)). According to Trachanatzi et al. (2020) the decision-maker is required to provide decision examples on a reference set of solutions that can be used to determine an approximation of their decision function.

The final acronym method is the *Weighted Ordered Weighted Averaging (WOWA)* by Torra (1997), an extension of the OWA method (Ogryczak and Śliwiński (2009); Ogryczak (2008)). The OWA method uses a set of weights to aggregate several objective values into one single objective. The WOWA extension applies the theory of the weighted mean to this method, which takes into account the reliability of the source of the information.

To understand which of the acronym methods are most prevalent in the state-of-the-art, we count the number of studies that discuss each of the methods. Table 3 shows the acronym methods and how often they appear in the state of the art. This includes mentions in the background or related work sections. We observe that AHP, ELECTRE, MAUT, and PROMETHEE are the most prevalent in the studies we review.

Table 3. The number of studies that mention each of the acronym methods.

Elicitation method	Number of studies
AHP	18
ELECTRE	36
INTERCLASS	3
MAUT	14
PROMETHEE	25
TOPSIS	6
UTASTAR	1
WOWA	6

Many of the reviewed studies do not elicit preferences by specifically using one of the techniques in Table 3. One method that is used particularly often is to encode the decision-maker's preferences as a reference point, vector, or set. This method, often referred to as the *Reference Point Method*, does not require the decision-maker to set weights, describe relative importance, or determine threshold values. Instead, the decision-maker is asked for their ideal values for the optimization problem's objectives. The search for a set of optimal solutions is guided towards the region on the Pareto front that is closest to the reference point (or line, or set). Distance metrics used in the literature include Euclidean distance, Tchebycheff distance, and Hausdorff distance. The advantage of the reference point method over many other preference modelling methods is that it is less sensitive to minor imperfections in the decision-maker's preference. The method works for reference points that are feasible and infeasible, as well as dominated and non-dominated. The reference point method can be used in a priori and interactive optimization models.

Many of the preference elicitation techniques have a similar workflow. The first step is to perform some comparison, classification, or ranking of objective functions or a small set of candidate solutions. From there a weight matrix, value/utility function, or set of decision rules is used to evaluate the fitness of the full set of candidate solutions. This produces a ranking that can be used to choose the best *k* solutions[8]. These best solutions are then presented to the decision-maker or used to guide the next iteration of a search algorithm.

3.3 Validation

We have identified many methods for preference modelling and multi-objective optimization. In this section, we consider the methods described in the literature for the validation of optimization methods. We first cover the numerical and case-based validation methods. Then, we cover the most widely used standardized test sets. Finally, we describe the metrics by which we can evaluate an optimization method.

[8]In many evolutionary algorithms, the ranking also takes a diversity metric into account. Including a diversity measure in the fitness calculation avoids local optima by covering a larger portion of the search space.

3.3.1 Numerical example

A common method for validating an optimisation technique is to apply it to a numerical example. In this test method, researchers define an example problem against which they evaluate their technique. This can be a problem-specific optimization problem or a general optimization problem. The numerical example has the advantage of demonstrating the technique's performance on a representative optimization problem. However, these tailored examples make comparing techniques from different studies difficult. Furthermore, studies that use custom-built example problems frequently do not apply the technique to a broad range of optimization problems. This makes it impossible to make claims about the technique's general applicability.

3.3.2 Case study

Numerical examples are generally done on synthetic problems in a lab setting. In some studies, researchers apply their technique to a real-world case study. This shows the applicability of the technique to optimization problems with realistic numbers of variables, preference models, decision-makers, and constraints. This also enables a qualitative evaluation of user acceptance, usability, and running time. Like with the numerical examples, the disadvantage of a case study is the difficulty in making comparisons between studies. This is especially difficult in studies using case studies as their main validation method because the mathematical model underlying the problem is not always disclosed. In the literature, case studies are not used very often.

3.3.3 Standardized test sets

Researchers have developed standard test sets to improve the comparability of the validation of different multi-objective optimization techniques. The test sets ZDT (Zitzler et al. (2000)), DTLZ (Deb et al. (2005)), and WFG (Huband et al. (2006)) are widely used in the state-of-the-art to validate optimization techniques. Each of the test sets has different properties, strengths, and weaknesses.

ZDT is the oldest of these widely used test sets. It was created in response to the growing number of evolutionary algorithms for solving multi-objective optimizations. Zitzler et al. (2000) describe six reasons why an evolutionary algorithm may have difficulty achieving convergence on the Pareto front or maintaining diversity over the Pareto front. The set contains six tests that expose the technique under evaluation to the six identified challenges. All of the problems in ZDT are strictly bi-objective.

Deb et al. (2005) identify other challenges that a multi-objective may encounter and point out that ZDT is not scalable to increasing numbers of objective functions. DTLZ was developed to measure the convergence and diversity performance of evolutionary algorithms for any number of objectives and decision variables. The DTLZ test set is defined in two separate works, as noted by Huband et al. (2006). The original technical report describes the test problems DTLZ1-9 (Deb et al. (2005)), whereas a later conference paper only describes DTLZ1-7 (Deb et al. (2002b)). The tests DTLZ5 and DTLZ9 from the technical report are not found in the conference paper. In our work, we use the problems DTLZ1-9 presented in the original technical report by Deb et al. (2005).

Huband et al. (2006) analyse earlier multi-objective test sets, including ZDT and DTLZ, to conclude that these test sets do not adequately cover several newly established criteria for multi-objective test sets. WFG incorporates new criteria such as deceptive objective spaces, flat Pareto fronts, and separability. However, the authors note that, despite these shortcomings, ZDT and DTLZ are still effective test sets.

In the literature, multiple test sets are often used in tandem to cover as many of the known challenges as possible.

3.3.4 Evaluation metrics

Another aspect of the comparisons between multi-objective optimization techniques is the measurements that we use to describe the performance of the technique. In this section, we discuss convergence and diversity, composite Pareto fronts, and qualitative metrics.

The quantitative measurements discussed in this section require some prior knowledge of the true Pareto front of a problem to determine how the approximation by a technique compares. In the standardized test sets discussed above, the function that determines the Pareto front is known. When the true Pareto front is not known, researchers often opt to create a close approximation of the true Pareto front by doing one expensive, long run with an optimization method with good convergence and diversity properties such as (unmodified implementations of) NSGA-II or NSGA-III.

Convergence is the measure of how closely an optimization technique can approximate the true Pareto front of a given problem. In their 2015 review, Riquelme et al. (2015) concludes that hyper-volume is the most used convergence metric. Other notable measurements for convergence are the generational distance, the inverted generational distance, and the epsilon indicator.

Diversity is the measure of how evenly the solutions obtained by a technique are distributed over the Pareto front. The review by Yan et al. (2007) concludes that there are several diversity metrics, all of which are flawed in some way. They conclude that studies using diversity metrics should be aware of the drawbacks of each metric. Diversity measurements include clustering, hyper-volume, spacing, chi-square-like deviation, and Δ.

When the true Pareto front is not known, these metrics require us to be able to approximate the true Pareto front by some technique. A downside of this is that we are dependent on the accuracy of the approximation of the true Pareto front for our metrics.

Alternatively, when comparing the techniques against each other, it is possible to create a Pareto front using the combined approximations of the Pareto front found by each technique. The percentage contribution to this composite Pareto front measures how good the technique is compared to the other technique(s) in the composite front. IGD-CF and R-IGD are convergence measures that measure the relative contributions of each technique to the solutions in the region of interest on the composite Pareto front.

The literature also presents some qualitative measurements. The quality of a technique can be determined according to the mental load on the decision maker as well as the computational load needed. The number of interactions with the decision-maker and the amount of information per interaction are defining traits for the mental load on the decision-maker, especially in interactive optimization techniques. Generally, it is considered better to limit the number of interactions and amount of information. Finding techniques with low computational load is important in online use cases where it is not possible to wait a long time for a computation to finish.

3.4 Conclusion

This research concerns a multi-objective optimization problem in which the decision-maker has a strong preference when choosing between multiple Pareto-optimal solutions. In Section 1.3, we mention that we selected two algorithms for our methodology to answer **RQ2**. In this chapter, we describe how we chose our algorithms given our requirements and our literature analysis.

Conversations with domain experts have shown that a decision-maker performs in the order of 10s of nesting runs per day. This makes interactive algorithms prohibitively intrusive for the decision-makers. The literature has shown that a posteriori algorithms perform significantly worse than interactive or a priori algorithms. For these reasons, we considered only a priori algorithms.

In the last 10 years, the state-of-the-art has largely moved to researching evolutionary algorithms. Research has shown that these algorithms can deal with many problem types, handle higher numbers of objectives, incorporate preference information, and are simple to implement. The two most popular evolutionary algorithms researchers have modified or expanded to suit their particular requirements are NSGA-2 and MOEA/D. We chose one algorithm based on NSGA-2 and one based on MOEA/D for our research.

A common method for preference encoding is to use a reference point that guides the search to a small area on the Pareto front. The advantage of this method is that the decision-maker does not need to be very accurate to still obtain good results, compared to using a method based on weights. On top of that, the reference point can be saved between optimization runs to reduce the mental load on the decision-maker, and algorithms using reference points are generally easy to adapt with preference learning techniques.

Finally, this research considers the state-of-the-art. Therefore, we chose algorithms that have been published relatively recently. Given the requirements we have outlined so far, we investigated ar-NSGA-2 (Yi et al. (2019), originally known as ar-MOEA) and r-MOEA/D, (Qi et al. (2019)). Both algorithms meet the requirements and have the advantage that they can be used a priori and interactively. This would enable us to investigate interactive optimization compared to a priori optimization using the same implementations of the algorithms in future work.

4 IDENTIFYING NESTING FACTORS

Our first research question concerns the factors influencing how a decision-maker evaluates a nesting and how we model these factors in a multi-objective optimization model.

In this chapter, we describe the process of eliciting the factors by interviewing domain experts at DIGI-STEEL, interviewing expert decision-makers at the client companies, compiling the lists of factors from these interviews, and creating the multi-objective optimization model using a representative subset of these factors.

4.1 Interviews

To prepare for the interviews with the decision-makers, we first met with several experts on nesting from DIGI-STEEL. During the unstructured interviews with these experts, we compiled an initial list of factors to guide our questions in the interviews with the decision-makers. Based on these meetings we determined that the factors influencing nestings can be broadly categorized into *cost*, *time*, and *labour*.

Next, we performed semi-structured interviews with the expert decision-makers at three client companies operating in the steel manufacturing industry in the Netherlands. The interviews were conducted in two sessions. In session one, we focused on eliciting as many factors as possible from the decision-maker. In session two, we dove deeper into how the decision maker chooses a nesting when confronted with trade-offs between these factors. Appendix B shows a complete list of the guiding questions we prepared for the interviews.

4.1.1 Interview session 1

The first interview session was split into two phases: factor collection and factor prioritization.

During the collection phase, we focused on identifying the factors the decision-maker pays attention to when assessing a nesting. To do so, we asked them to describe how they evaluate a nesting. During the expert interviews at DIGI-STEEL, we were informed that many of these considerations normally happen subconsciously. Therefore, we prepared our semi-structured interview with guiding questions, such as:

> ”In previous projects, what aspects of a nesting (besides the waste it produced) did you consider and why?”

The meetings with internal experts at DIGI-STEEL yielded an initial list of factors and categories. During the initial elicitation phase of the interview, we did not share this list with the decision-maker. However, we used the identified categories to ask questions such as the following:

> ”What are situations in which [costs/time/labour] would be important to consider when creating a nesting?”
>
> ”What aspect of the nesting or the situation makes it important to consider [cost/time/labour]?”

Before the prioritization phase, we showed the decision-maker the full list of factors. The decision-maker was then given time to read through this list and give their comments. These comments could include any factors they were still missing and situations in which a factor may be influential.

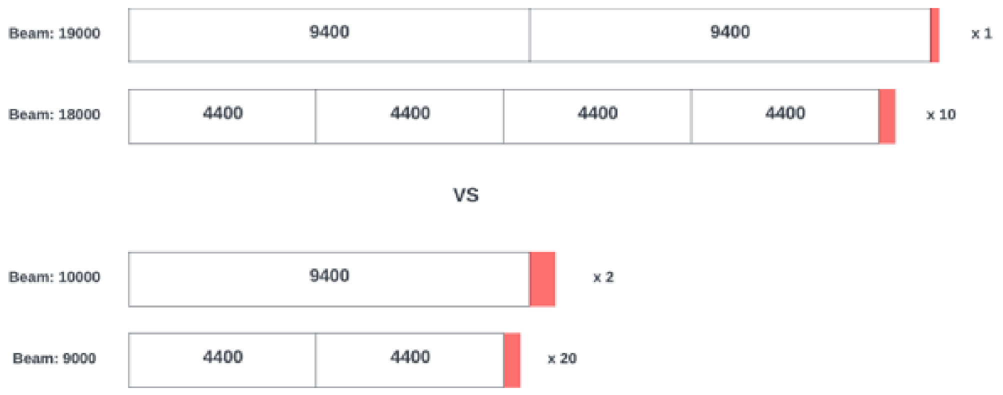
During the prioritization phase, the decision-maker was asked to rank the factors in order of importance. During the interview session, the decision-makers were able to create an ordering to the factors. However, the decision-makers all reached a point in the prioritization where ordering the factors became impossible because the factors were incomparable to each other or only situationally useful to the decision-makers.

In Section 3.2.3 we identified decision-making frameworks such as AHP (Saaty (1990)) that rely on comparisons between all objectives in a multi-objective decision-making problem. Given that many factors were considered equally important to the decision-makers, these decision-making frameworks may struggle to find reasonable weights for the objectives in the nesting context.

4.1.2 Interview session 2

In the second interview session, we delved deeper into the interactions between the nesting factors. During this session, the interviewees were asked to compare two nestings that contain the same parts. These nestings differed according to one or more factors identified in session one.

Figure 7 is an example of a comparison between two nestings. At the top of the example, we show the two nestings for the comparison. All parts, waste, and beam sizes are to scale to give the decision-maker an intuitive understanding of the relations between these lengths. Below the examples, we included information to place the choice in context. This information is the total length of the steel required for the nesting, the total amount of waste in metres, and the cost of this waste for the HEA300 and HEA800 profiles.



	Option 1	Option 2
Total length (m)	199	200
Waste (m)	4.8	5.8
Waste cost HEA300 @ 30.48 (€)	146.31	176.79
Waste cost HEA800 @ 88.23 (€)	423.51	511.74

Figure 7. Nesting comparison shown to the decision-maker in interview 2

The decision-makers were each presented with 6 to 8 comparisons, created based on the factors they considered important in interview one. With these comparisons, we could discuss the relations between these factors more deeply. On top of that, the second interview session also led to identifying more factors that had been missed during session one.

All in all, the second round of interviews did not lead to any drastic changes in our understanding of the nesting factors.

4.2 Interview Results

To simplify reasoning about the factors that influence the nesting process, they were grouped into the categories of money, process, time, and producibility. In reality, the factors often influence the nesting process in multiple ways, which makes it difficult to separate them neatly into groups or categories.

4.2.1 Money

The most straightforward measure for the performance of a nesting is how much it would cost to produce. Factors in this category include the cost of waste and the delivery costs of purchased steel.

Many of the quantifications of the factors relating to cost can be expressed in multiple measures. For example, the amount of waste a nesting produces can be expressed in Euros, millimetres, and kilograms. This offers us flexibility in how we define our optimization model. We may prefer to show the decision-maker the cost of waste in Euros, but the price of steel may not always be known at the time of nesting[9]. In this case, we could express the cost of waste in millimetres or kilograms. A decision-maker can still use this information to evaluate a nesting.

4.2.2 Process

In many steel production facilities, the process can be heavily influenced by a nesting. Production planners prefer to keep the process as streamlined as possible, to reduce chaos in the process. Chaos will lead to production delays, accidents, lost inventory, and many other problematic situations. Factors in this category include the number of handling operations performed on a beam and phase clustering.

The factors related to the process are mostly quantified by counting the relevant occurrences. The flexibility of these factors lies in the methods we could choose to define an occurrence. For example, when considering the quantification of phase clustering we could choose to count the number of beams that contain multiple phases. Alternatively, we could count the number of distinct phases per beam or the largest spread within a beam. Using a different method to quantify the factor could lead to other outcomes when nesting.

[9]The price of steel has fluctuated a lot over the past few years, and nestings are sometimes done months before the start of a project.

4.2.3 Time
Time is money, and this is no different in steel manufacturing. Production planners seek to use the time of their workforce and machines as effectively as possible. Factors in this category include operation duration and the number of manual operations.

The factors related to time are all relatively straightforward. Using time as the quantification for these factors does introduce the difficulty of estimating the nominal time for an action. For example, the operation duration factor is sensitive to the accuracy of the time estimation for each operation. Using this factor in a nesting evaluation model may become infeasible if the estimation is too inaccurate or unknown.

4.2.4 Producibility
The steel manufacturers that purchase their machines at Voortman do so to produce parts automatically. The degree to which parts can be produced automatically is determined in part by reproducibility. Factors in this category include the number of operations that cannot be produced automatically.

The producibility factors may be most suitable as configurable constraints, rather than as objectives for the nesting algorithm. A decision-maker could choose to set a minimum level of producibility. Some decision-makers may choose to accept some non-automated operations, while other may want to fully constrain the nesting to ensure automated production. Constraints may also include a deviation to allow a decision-maker to consider the trade-off between producibility and other factors. The deviation can be used as a minimization objective for the nesting algorithm.

4.3 Proof of Concept Model
In this section, we describe the factors that can influence the evaluation of a nesting. The goal is to implement a nesting algorithm that can take these factors into account. As an example of what such an implementation would look like, we created a proof of concept implementation. In the rest of this section, we explain the proof of concept model, the choices of factors, and how this model is representative of the complete list of factors we identify. For a rigorous mathematical definition, we refer to Appendix C.

4.3.1 Parameters
As input to the model, we define the set of parts P that we intend to nest, the set of possible beam lengths A, and the set of phases F. The set of beams B always contains enough beams to fit all parts. For this research, we do not consider any nesting problems with a bounded number of beams.

The cost parameters are the cost of steel (C_{steel}) and the cost of an oversized load ($C_{oversized}$). The threshold parameters are the discount threshold for the remnant ($T_{discount}$), the Short Piece Removal System threshold (T_{sprs}), and the threshold for which beam lengths count as an oversized load ($T_{oversized}$). Other parameters are the length of a part (len_p), the discount factor we apply to a remnant (d), and whether a part belongs to a phase ($phase_{pf}$).

Most of these parameters are configurable by the decision-maker before executing a nesting.

4.3.2 Decision Variables
The nesting model has two decision variables we can perturb to generate new nestings. The first decision variable determines whether a part is nested into a beam (x_{pb}), while the second decision variable determines the length of each beam (y_b).

4.3.3 Auxiliaries
The auxiliaries are variables and functions that we use to support the calculations for the objectives and constraints. If a beam is used, z_b equals one, and 0 otherwise. The waste generated by a beam is given by W_b and the estimated number of waste pieces the SPRS cannot remove from a beam is given by S_b. The function D_b returns the discount factor if the discount can be applied and 1 otherwise. To check if a beam contains parts from multiple phases we check whether a phase is nested into a beam (q_{bf}) and use this information to determine if a beam contains multiple phases (Q_b). Finally, the extra delivery costs for an oversized load are given by E.

4.3.4 Objectives
We define three objectives that represent the factors in the cost, process, and time categories, respectively.

To represent the cost category, we minimize the total cost of waste and the costs for an oversized load. In the cost of waste, we apply the discount factor to any piece of waste that is long enough to be a remnant piece.

To represent the process category, we minimize the number of beams that contain multiple phases. We do not consider the number of phases per beam or the distances between phases in this model. With the available information, it would be trivial to implement this in the future.

Lastly, we minimize the estimated number of waste pieces that the Short Piece Removal System cannot remove to represent the time category.

4.3.5 Constraints

To begin with, each part needs to fit into the beam it is nested into and each part should be nested once. In addition, a beam that contains parts needs to be considered in use. Furthermore, a part may only belong to one phase. Lastly, do not consider a beam in use when it does not contain parts.

5 COMPARING AR-NSGA-2 AND R-MOEA/D

This chapter compares the two algorithms we selected in section 3.4 to determine their performance on test sets from the literature. The results of this comparison are used to decide which algorithm would be used for the case study evaluation. We describe the implementation of the algorithms, the test setup, and the performance metric. After that, we present the test results and our findings.

5.1 Algorithms

We base our implementation of ar-NSGA-2 on the original study by Yi et al. (2019). The core of their work is the ar-dominance relationship, which considers the angle and distance to the reference point for each solution. The algorithm uses an adaptive weight to vary the importance of the angle and distance information. At the beginning of the optimization run, the distance is more influential to the preference relation. Towards the end of the optimization run, the angle becomes more influential.

Importantly, the authors of the original work present a general methodology without explicitly mentioning NSGA-2. To select the individuals for the next generation, the authors use the ar-dominance relation with an elitist sorting strategy. However, they give no further details for the specific implementation of this sorting strategy. Based on the given information, we decided to use the elitist non-dominated sorting strategy from NSGA-2 (Deb et al. (2002a)). Instead of using crowding distance, we apply the ar-dominance principle to sort solutions belonging to the same non-dominated set. Therefore, we name this implementation ar-NSGA-2.

The algorithm has two parameters that influence the ar-dominance relation. These parameters are adaptive, changing automatically with the number of elapsed iterations. Yi et al. (2019) provide reference bounds for these parameters, which we have used in this work. Our lower bound for the dominance threshold differs from the original implementation[10]. The adaptive weight ξ determines the trade-off between distance and angle in the ar-dominance relationship. We set the interval for the adaptive weight to $\xi \in [0,1]$. The adaptive dominance threshold δ determines how much more a solution must be preferred over another Pareto-indifferent solution to be considered ar-dominant. We set the interval for the adaptive dominance threshold to $\delta \in [0.2,1]$.

Our implementation of r-MOEA/D follows the original work by Qi et al. (2019). This algorithm is based on MOEA/D, first proposed by Zhang and Li (2007), incorporating the distance to a reference point as the preference structure.

The implementation of r-MOEA/D follows directly from the implementation of the original MOEA/D. The authors introduce a new scalarizing function that incorporates the reference point which replaces the scalarizing function in MOEA/D. This new scalarizing function is a direct replacement of the original scalarizing function and has the added benefit that it is not dependent on an estimation of the ideal point.

The algorithm has two parameters that influence the selection and replacement of neighbouring solutions. Qi et al. (2019) provide reference values for the parameters, which we have used in this research. The neighbourhood size T determines how many weight vectors are considered neighbours to any given weight vector. This neighbourhood is used to generate new solutions in the recombination phase. We set the neighbourhood size to $T = 3$. The maximum number of replacements n_r determines how many solutions can be replaced by any candidate solution. We set the maximum number of replacements to $n_r = 3$.

5.2 General considerations

Central to any evolutionary algorithm is the recombination, mutation, and repair mechanics used in child generation. In the literature, authors generally expect the reader to have a prior understanding of the algorithms used for recombination, mutation, and repair. The authors of both papers do not reference any previous work for implementations of these mechanics.

The recombination, mutation, and repair that are applicable also depend on the underlying model of the optimization. Many optimization problems, such as bin packing, have specialized child generation algorithms. Given that our work concerns a real-world nesting problem and testing on ZDT and DTLZ, we use two sets of child generation algorithms.

The decision variables in ZDT1-4, ZDT6, and DTLZ1-9 use an identical data structure. The genome is a list of real-valued decision variables with known upper and lower bounds. To perform recombination, we implement the *cut and splice* operator FCO on the individual variables instead of bins of variables (Mellouli et al. (2019)). For the mutation, we implement polynomial mutation (Carles-Bou and Galán (2023)). On this genome, the chosen recombination and mutation strategies always yield feasible results, so a repair mechanism is not required.

[10]In the original paper, the authors use a lower bound of 0 for the dominance threshold. Due to a misunderstanding, we set the lower bound to 0.2 in our implementation.

For a more detailed explanation of the genomes and the effects of the recombination and mutation, see Appendix Section D.1.

Our representation of a nesting uses beams as bins with an ordered list of parts. For the recombination step, we implement the GCO algorithm (Falkenauer (1996)). The GCO algorithm does not take into account variable bin sizes, so in our implementation whenever we create a new bin we use a semi-random procedure to select the bin size. For the mutation strategy, Falkenauer (1996) suggests destroying some bins based on the mutation factor. We implement this mutation strategy and the first fit descending strategy with local optimization as our repair mechanism. For a more detailed explanation of the genomes and the effects of the recombination and mutation, see Appendix Section D.2.

5.2.1 Generational Distance

In Chapter 3 we identified several performance metrics for multi-objective optimization algorithms. For this research, we are mostly concerned with how well an algorithm can converge on optimal solutions. The measure for convergence we use is the *generational distance (GD)* as described by Carles-Bou and Galán (2023), which is a measure of the distance between the non-dominated front found by the optimization algorithm and the true Pareto front.

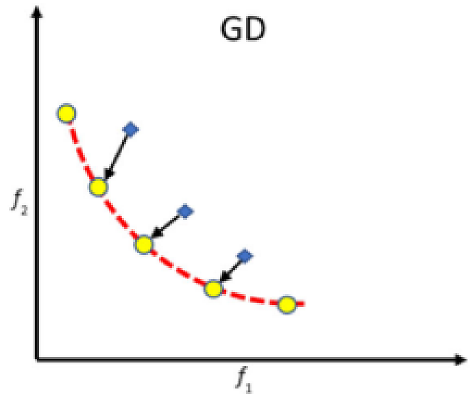


Figure 8. An illustration of generational distance from Carles-Bou and Galán (2023)

Figure 8 illustrates how we calculate the distances used in generational distance. The arrows show the distance between each solution found during a testing run (blue) and a known solution on the Pareto front (yellow). For ZDT and DTLZ the Pareto fronts are known, so we can use the shortest distance from each point to the front instead.

With these distances, we can use the following equation to calculate the generational distance for the set of solutions S to the Pareto front PF:

$$GD(S) = \frac{1}{|S|}\sum_{s\in S}(distance(s,PF)) \tag{1}$$

5.3 Test configurations

In this section, we discuss the distance calculations we used for the ZDT (Zitzler et al. (2000)) and DTLZ (Deb et al. (2005)) test sets, and the parameters we changed to affect the performance of the algorithms.

Of the six tests in the test set, ZDT1-4 and ZDT6 use the same genome structure, where the genome is a list of real-valued decision variables. ZDT5 uses a genome that consists of a list of decision variables that contain a binary string (see Appendix Section D.3 for the genome structure). The downside of this approach is that ZDT5 requires a different recombination, mutation, and repair mechanism than ZDT1-4 and ZDT6. Therefore, to simplify the implementation and ensure we compare performance on the same child generation mechanisms, we do not consider ZDT5 in our testing.

The GD performance metric requires knowing the distance to the optimal front for each solution in the result of the optimization run. ZDT defines the function $g(\mathbf{x})$ with a known optimal value for each test instance. We use the difference between the value of $g(\mathbf{x})$ and the optimal value as the distance for all ZDT tests, such that for each solution the distance is $d = g(\mathbf{x}) - g^*$.

For DTLZ1 the optimal front is a linear hyper plane with $\sum_{i=1}^{M}(f_i^*) = 0.5$. For each solution, the distance to this hyperplane is:

$$d = \sum_{i=1}^{M}(f_i(\mathbf{x})) - 0.5 \tag{2}$$

The optimal front for DTLZ2-6 is the positive quadrant of a unit sphere. The distance to the optimal front for each individual is:

$$d = \sqrt{\sum_{i=1}^{M}(f_i^2(\mathbf{x}))} - 1 \tag{3}$$

The authors do not describe an equation for the optimal front of DTLZ7. Rather, the Pareto-optimal solution corresponds to $x_M = \mathbf{0}$. In the literature, the Tchebychev distance is an accepted method to find the distance between two vectors. Given that our optimal vector is the zero vector, the distance calculation simplifies to:

$$d = max(x_M) \tag{4}$$

We could not establish the optimal front for DTLZ8 from the original work. Therefore, we opt to use the cumulative distance to the line component of the optimal front as an approximation of optimality[11]:

$$d = \sum_{i=2}^{M-1}(|f_1(\mathbf{x}) - f_i(\mathbf{x})|) \tag{5}$$

Finally, DTLZ9 also has an optimal plane that is difficult to establish from the original work. For any objective function $f_i(\mathbf{x}),\ where\ i \neq M$ plotted against $f_M(\mathbf{x})$ the optimal front is the positive quadrant of the unit circle. As an approximation of the performance of the distance we use:

$$d = \sqrt{f_1^2(\mathbf{x}) + f_M^2(\mathbf{x})} - 1 \tag{6}$$

To compare the theoretical performances of the algorithms, we run them on the ZDT and DTLZ test sets. The algorithms and test sets have various parameters that affect the performance of the algorithms. The general parameters are the number of iterations (I), the population size ($|P|$), the mutation probability (p_m), the mutation index (η_m), and the number of objectives (m, only for DTLZ).

The polynomial mutation operation requires a mutation index η_m. This mutation index determines the width of the distribution function we use to perturb the genome. A lower value makes the distribution wider. Previous work focusing on the polynomial mutation operation has shown that a mutation index between 5 and 40 will show similar performance (Carles-Bou and Galán (2023)). For the rest of this testing, we set the mutation index to $\eta_m = 5$ unless specified otherwise.

We set the other parameters based on testing data, starting with the mutation probability, then the population size, the number of iterations, and finally the number of objectives (only for DTLZ).

5.4 ZDT Results

We first analysed the performance of the algorithms on the ZDT test set (Zitzler et al. (2000)). As described in Section 5.2.1, we based our analysis on the generational distance (GD) metric. To adjust for the randomness in the algorithms, we averaged the GD value over 20 runs for all tests[12].

[11]This approximation is not the most effective method to determine the distance to the optimal front. Generally, it overestimates the distance to the front, and this overestimation error becomes bigger for points further away. A more accurate distance metric may have been the Tchebychev distance. We note that this choice has not affected the results of our testing, given that DTLZ8 did not show any major differences between our algorithms.

[12]We chose to run the algorithm 20 times to average out much of the variation while still keeping the total runtime of the tests manageable.

5.4.1 Mutation probability

The mutation probability determines the likelihood that a decision variable is changed in the mutation step of an evolutionary algorithm. To determine the mutation factor we used for the remainder of the tests, we observed the performance of the algorithms with a mutation chance of 1%, 2.5%, 5%, 7.5%, 10%, 50%, and 100%. Previous unstructured testing and the literature (De Jong (1975)) favour lower mutation chances, so most of the chosen mutation chances are low percentages. We included 50% and 100% to show the algorithms' behaviours at higher mutation chances as an indication of robustness.

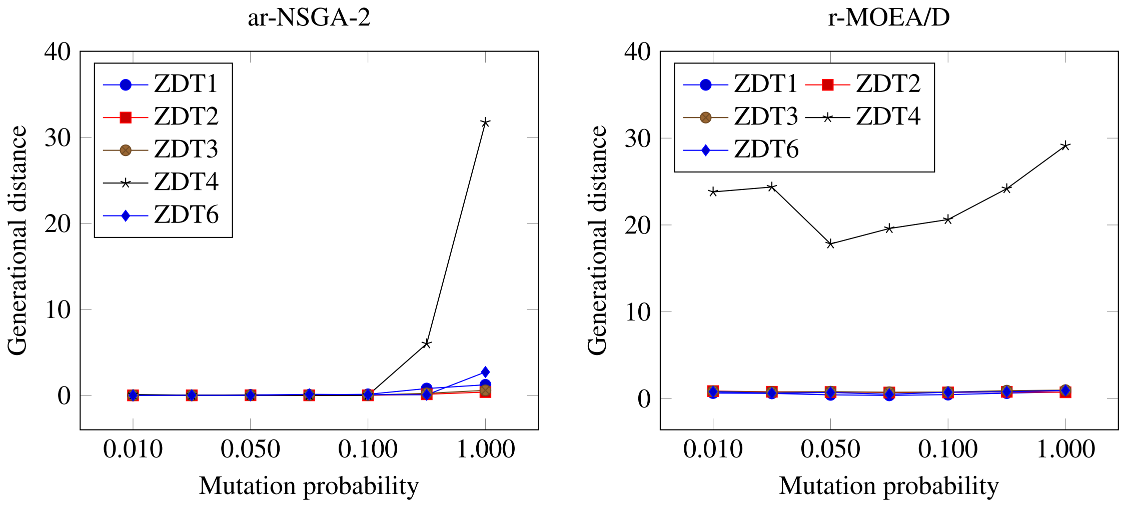


Figure 9. Varying mutation probabilities on ZDT1-4 and ZDT6. $I = 1000$, $|P| = 50$, and $\eta_m = 5$.

Figure 9 shows the performance of ar-NSGA-2 and r-MOEA/D on the ZDT test set at the given mutation probabilities. We observe that ar-NSGA-2 shows near-optimal convergence for mutation probabilities up to 10% across all tests. On the other hand, r-MOEA/D does not approach optimality on ZDT4 for any mutation probability.

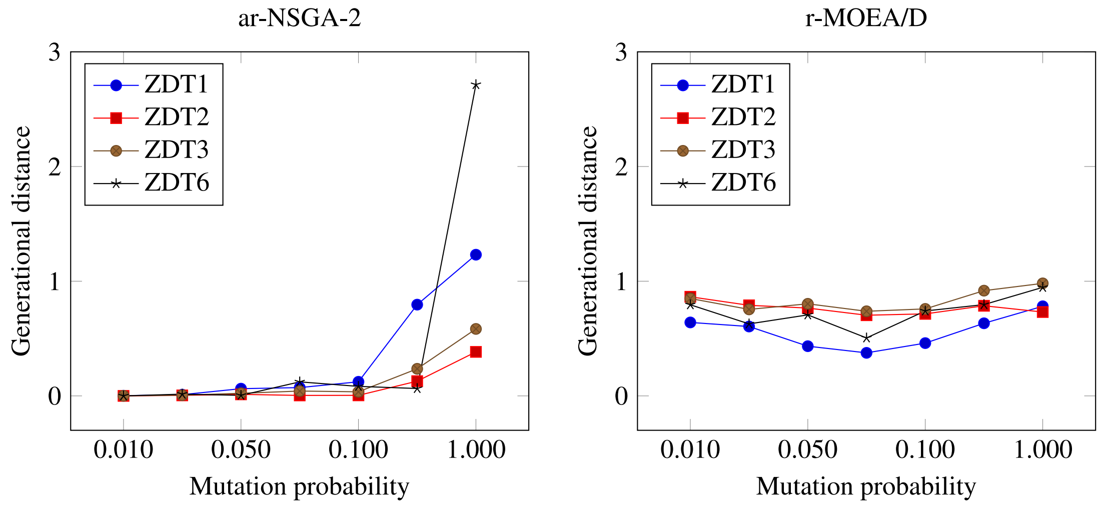


Figure 10. Varying mutation probabilities on ZDT1-3 and ZDT6. $I = 1000$, $|P| = 50$, and $\eta_m = 5$.

Figure 9 does not show the relative performance when the algorithms approach optimality. The high GD values as a result of ZDT4 cramp the graphs at the low end. To show the difference in performance more clearly, Figure 10 shows the performance of the algorithms on ZDT1-3 and ZDT6. Clearly, ar-NSGA-2 performs better for mutation probabilities up to 10%, whereas r-MOEA/D struggles to reach optimality at any mutation probability.

Given these results, we conclude that ar-NSGA-2 outperforms r-MOEA/D, especially at mutation probabilities below 10%. The algorithms show the best performance at a mutation probability of 5%. We will use this mutation probability for the rest of the ZDT testing.

5.4.2 Population size

The population size determines the number of individuals in the population and the number of children generated for the next generation. Previous research has shown that increasing the population size tends to lead to better convergence to optimality (Benecke and Mostaghim (2021)). However, the downside of increasing the population size is an increase in computation time. Work has been done that investigates evolutionary algorithms that vary the population size depending on the stage of the optimization (Guan et al. (2017)). Adapting the population size during the optimization run looks promising. For this research, we used a fixed population size.

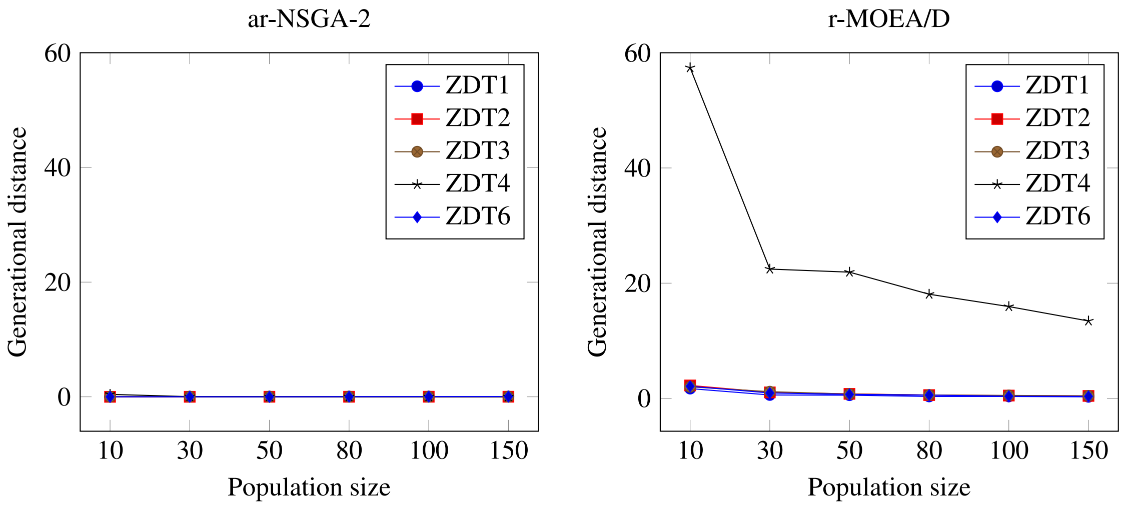


Figure 11. Varying population sizes on ZDT1-4 and ZDT6. $I = 1000$, $p_m = 0.050$, and $\eta_m = 5$.

Figure 11 shows the performance of the algorithms on various population sizes. The behaviour shown by the algorithms when exposed to varying population sizes looks similar to what we observe when varying the mutation probabilities. Once again, r-MOEA/D does not come close to optimal convergence on ZDT4. On all other tests, the population size does not appear to meaningfully affect the convergence performance.

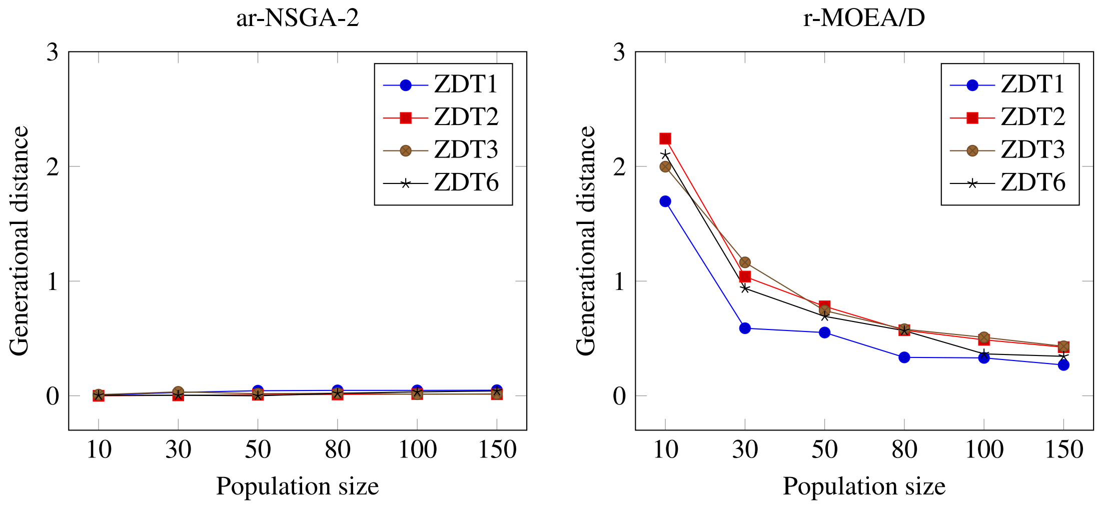


Figure 12. Varying population sizes on ZDT1-3 and ZDT6. $I = 1000$, $p_m = 0.050$, and $\eta_m = 5$.

However, Figure 11 is too zoomed out to show the differences between the algorithms on ZDT1-3 and ZDT6. To show the difference between the algorithms more clearly, we looked at the performance of the algorithms on only ZDT1-3 and ZDT6. Figure 12 shows the performance of the algorithms on these tests. The graphs show that r-MOEA/D does not converge to optimality as closely as ar-NSGA-2, even at higher population sizes. Interestingly, the performance of ar-NSGA-2 is not meaningfully affected by the population size.

The performance of r-MOEA/D improved only marginally at population sizes higher than 80. Therefore, we set the population size to 80 for the remainder of the ZDT tests.

5.4.3 Number of iterations

The final parameter is the number of iterations. Generally speaking, more iterations give the algorithms more time to converge. We expect the performance of the algorithms to improve as the number of iterations is increased. For this analysis, we ran the algorithms on the test set for 100, 250, 500, 750, 1000, 1500, and 2000 iterations.

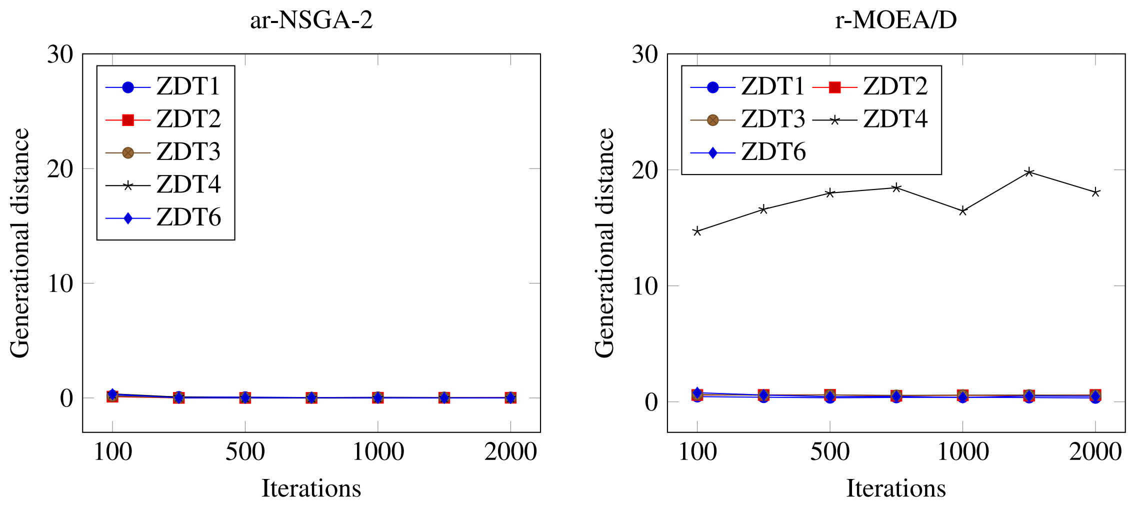


Figure 13. Varying numbers of iterations on ZDT1-4 and ZDT6. $|P| = 80$, $p_m = 0.050$, and $\eta_m = 5$.

Figure 13 shows the convergence performance for the algorithms over varying numbers of iterations. As we observed in the other tests, r-MOEA/D struggles with ZDT4. For all other tests, the performance appears equal based on these graphs. What does stand out is the performance of r-MOEA/D on ZDT4. We expect the performance to improve at higher iterations, but this is not the case for this test. ZDT4 tests an algorithm's performance when presented with a multi-modal problem, and it appears that r-MOEA/D struggles to overcome this multi-modality.

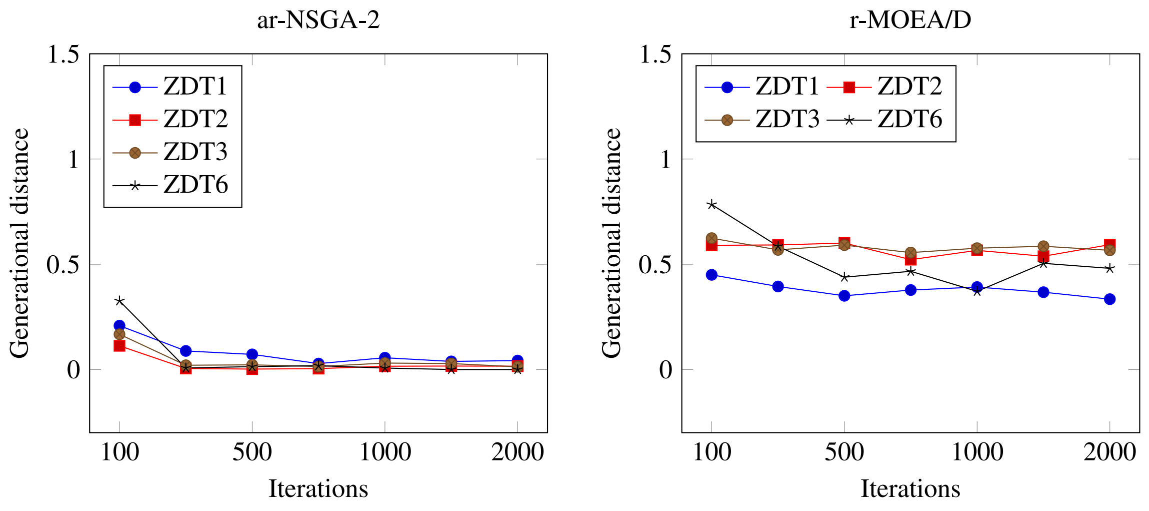


Figure 14. Varying numbers of iterations on ZDT1-3 and ZDT6. $|P| = 80$, $p_m = 0.050$, and $\eta_m = 5$.

To compare the near-convergence performance, we plot the performance on ZDT1-3 and ZDT6. Figure

14 shows this performance in detail. We observe that both algorithms perform the worst at 100 iterations and this performance improves between 250 and 500 iterations. Overall, ar-NSGA-2 outperforms r-MOEA/D at all numbers of iterations.

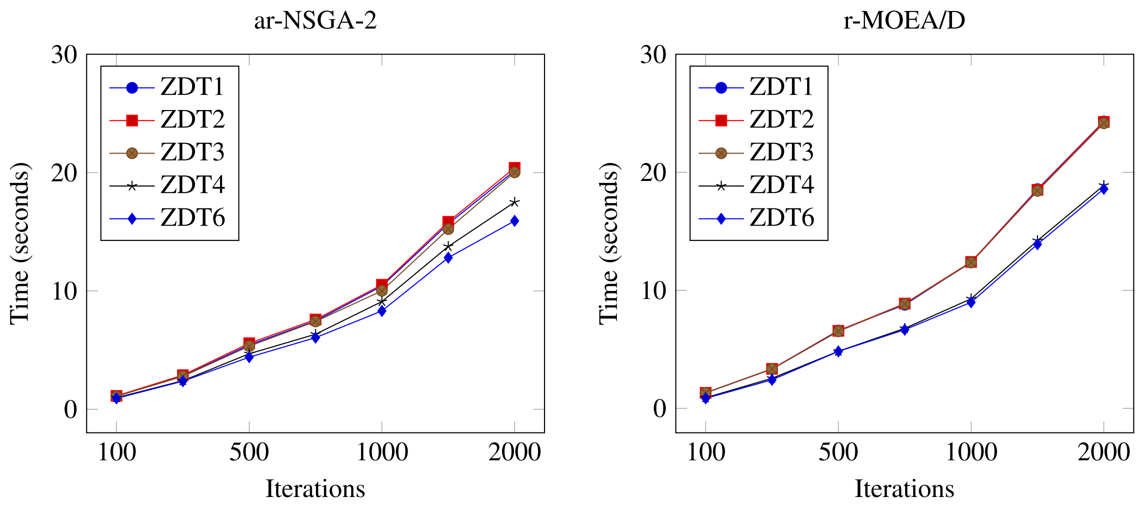


Figure 15. Varying numbers of iterations on ZDT1-4 and ZDT6. $|P| = 80$, $p_m = 0.050$, and $\eta_m = 5$.

Figure 15 shows the time the algorithms took to complete the optimization tests for the varying numbers of iterations. We observe that, as we would logically expect, a higher number of iterations leads to a longer time to complete. The algorithms perform similarly in terms of time complexity.

5.5 DTLZ Results

We perform the same analysis as above on the DTLZ test set (Deb et al. (2005)). In this analysis, we also fix the mutation probability, population size, and the number of iterations, after which we analyse the performance of the algorithms over a varying number of objectives.

5.5.1 Mutation probability

The ZDT tests showed that the algorithms tend to become unstable at high mutation probabilities. Since the DTLZ set contains more tests and is intended to test performance at higher numbers of objectives, we only test the performance for fewer mutation chances. To determine the mutation factor used for the remainder of the tests, we observed the performance of the algorithms with a mutation chance of 1%, 2.5%, 5%, 7.5%, and 10%.

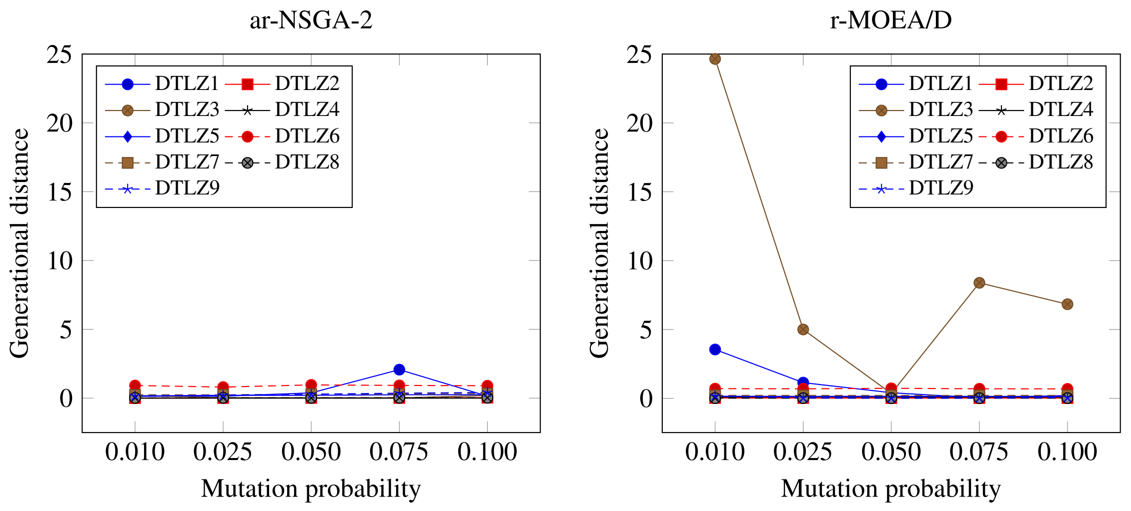


Figure 16. Varying mutation probabilities on DTLZ1-9. $I = 1000$, $|P| = 80$, $\eta_m = 5$, and $m = 3$.

Figure 16 shows the performance of the algorithms on DTLZ1-9. Much like the ZDT testing, ar-NSGA-2 shows good performance at mutation probabilities up to and including 10%. On the other hand, r-MOEA/D is far less performant. At 1% mutation chance, the performance degrades on DTLZ1, and the performance on DTLZ3 varies wildly across the mutation probabilities. The performance of both algorithms appears most stable at a 5% mutation probability. Therefore, we set the mutation probability to 5% for the DTLZ tests.

5.5.2 Population size

The algorithms did not show any differentiating behaviour at population sizes over 100 during testing on the ZDT tests. Therefore, we compared the performance of the algorithms on population sizes of 10, 30, 50, 80, and 100.

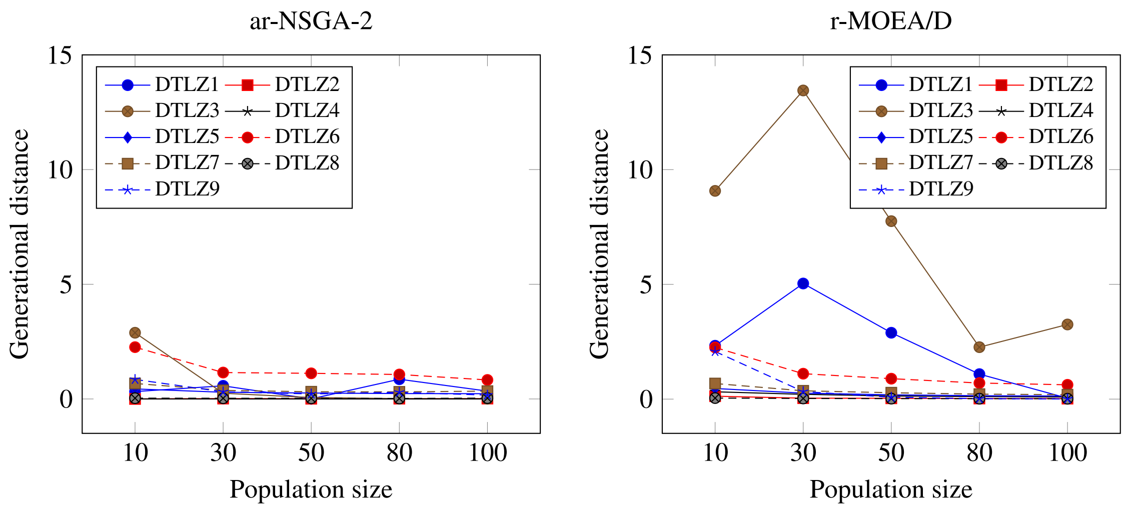


Figure 17. Varying population sizes on DTLZ1-9. $I = 1000$, $p_m = 0.050$, $\eta_m = 5$, and $m = 3$.

Figure 17 shows the performance at the different population sizes. The behaviour of ar-NSGA-2 is similar to the ZDT set. The algorithm appears agnostic to changes in the population size. At a population size of at least 30, the performance does not meaningfully change. On the other hand, as population size increases, so does the performance of r-MOEA/D. The algorithm struggles specifically on DTLZ1 and DTLZ3, with a large GD value at lower population sizes. For the remainder of the DTLZ tests, we set the population size to 80.

5.5.3 Number of iterations

The ZDT testing showed that the algorithms did not perform meaningfully differently at the highest numbers of iterations. To show the performance on DTLZ, we ran the algorithms for 100, 250, 500, 750, 1000, and 1500 iterations.

Figure 18 shows the generational distance at varying numbers of iterations. At less than 500 iterations, ar-NSGA-2 performs poorly on DTLZ3, which is gone when the number of iterations rises to at least 500. This poor performance on DTLZ3 with fewer iterations also happens with r-MOEA/D. However, DTLZ3 remains difficult for r-MOEA/D even at higher iteration counts.

As we did for the ZDT tests, we plot the performance of the algorithms close to optimal convergence. Figure 19 shows the performance on DTLZ1-2 and DTLZ4-9 over the various iteration counts. In general, the graph shows the expected improvement in performance when the algorithms are run for more iterations. Notably, some of the tests show no improvements with more iterations. In the original DTLZ paper, the authors show that unmodified NSGA-2 and SPEA2 do not converge on the true Pareto front on DTLZ6 (Deb et al. (2005)). The performance we observe in Figure 19 suggests that ar-NSGA-2 and r-MOEA/D are similarly unable to reach the true Pareto front.

Like with the ZDT tests, the algorithms perform similarly in time complexity. Figure 20 shows the time taken for each test at the different numbers of iterations. In some cases, r-MOEA/D will run quicker than ar-NSGA-2 on our testing setup. But rather than an algorithmic inefficiency, this may be the result of an inconsistent testing setup.

The tests were run in parallel on one machine and were started in the order ar-NSGA-2 test 1 to 9 and then r-MOEA/D test 1 to 9. This would consistently put the r-MOEA/D runs at the end of the execution queue. What

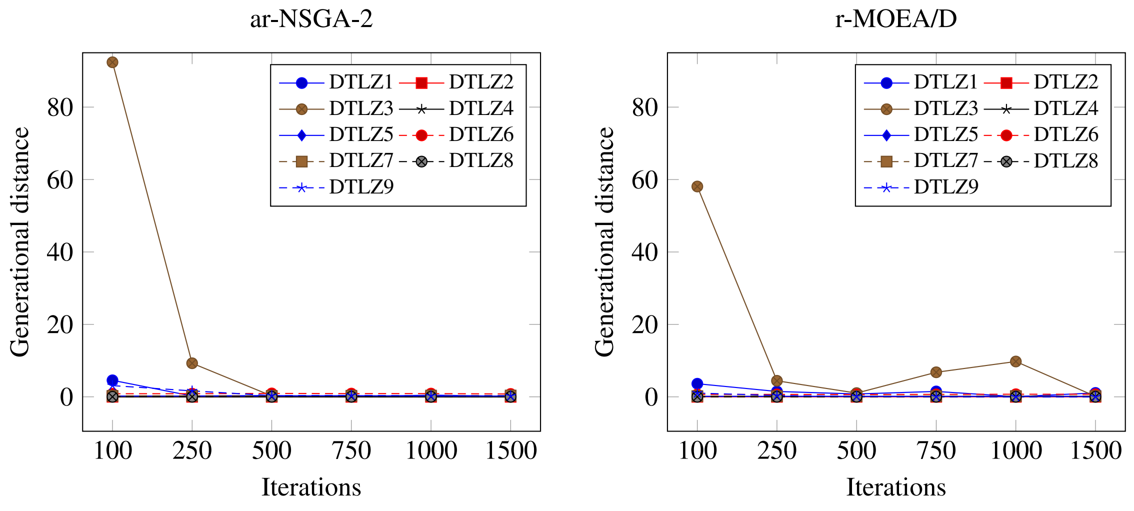


Figure 18. Varying numbers of iterations on DTLZ1-9. $|P| = 80$, $p_m = 0.050$, $\eta_m = 5$, and $m = 3$.

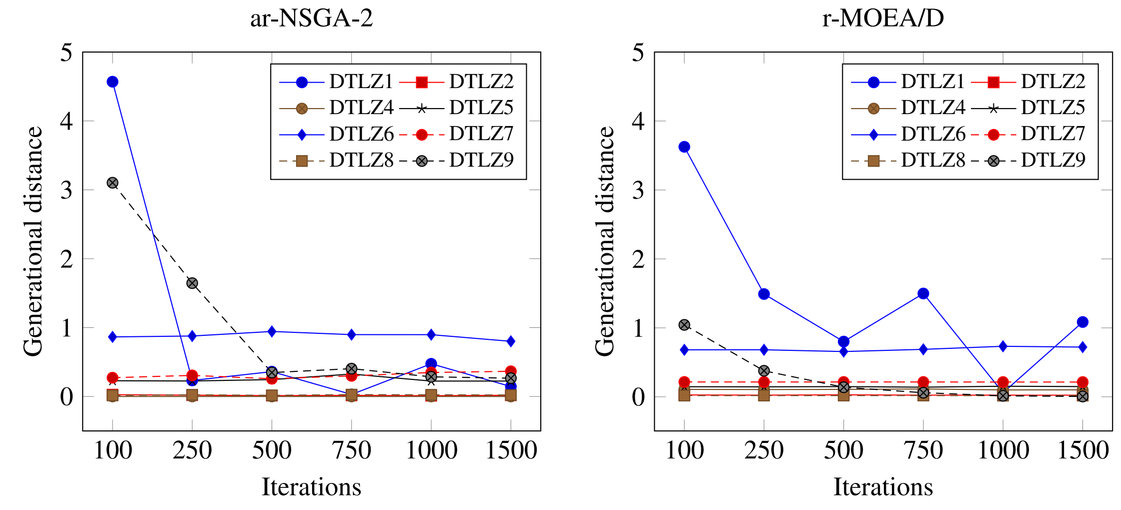


Figure 19. Varying numbers of iterations on DTLZ1-2 and DTLZ4-9. $|P| = 80$, $p_m = 0.050$, $\eta_m = 5$, and $m = 3$.

we may be seeing is that computer resources are being freed up, and more processing power is going to the later tests. This could explain why DTLZ6-9 is faster for r-MOEA/D. Given that the speed difference between the algorithms did not contribute to the comparison and reworking the testing infrastructure would take prohibitively long, we did not rerun the tests to adjust for this issue.

5.5.4 Number of objectives

The ZDT tests are strictly bi-objective optimization problems. One of the improvements made by the DTLZ tests is that all tests are defined for any number of objectives greater than 1. In the literature, DTLZ was mostly used for 3 up to 10 objectives, although there are studies that have gone up to 20 objectives. To test the performance of the algorithms, we run them on 3, 5, 7, and 10 objectives.

Figure 21 shows the performance at the different numbers of objectives. Like in the previous tests, DTLZ3 has proven to be a difficult problem for these algorithms. For r-MOEA/D specifically, DTLZ5 also leads to poor performance at higher objective counts.

To analyse the performance close to optimality more accurately, we plot the performance on DTLZ2 and DTLZ4-9. Figure 22 shows how the algorithms perform on these tests. Besides the poor performance of ar-NSGA-2, and the slightly worsening performance of r-MOEA/D on DTLZ7, the graphs show that the algorithms come close to reaching optimality over many numbers of objectives.

Finally, Figure 23 shows the time average time taken by the algorithms to complete each test. As expected, the

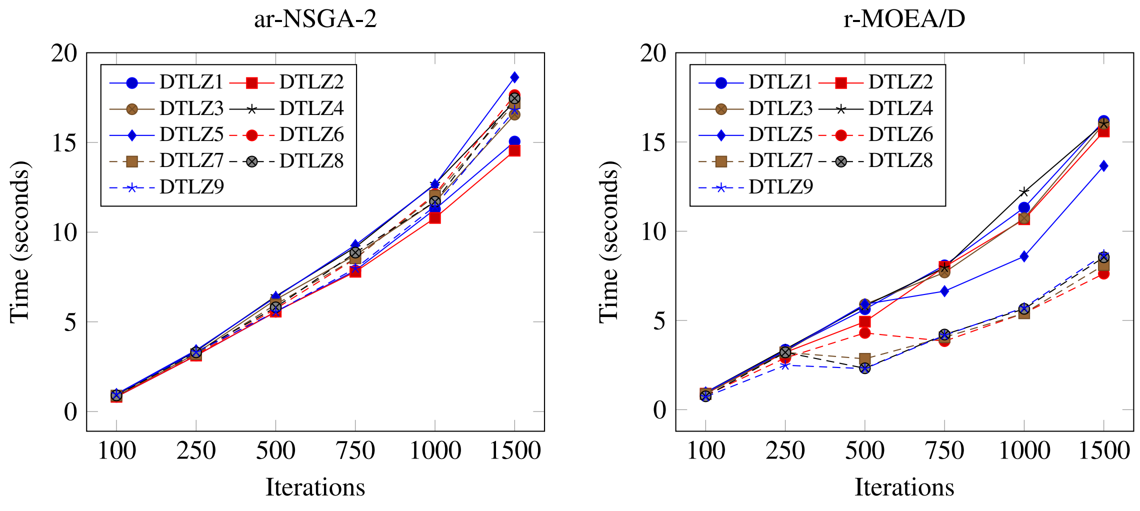


Figure 20. Varying numbers of iterations on DTLZ1-9. $|P| = 80$, $p_m = 0.050$, $\eta_m = 5$, and $m = 3$.

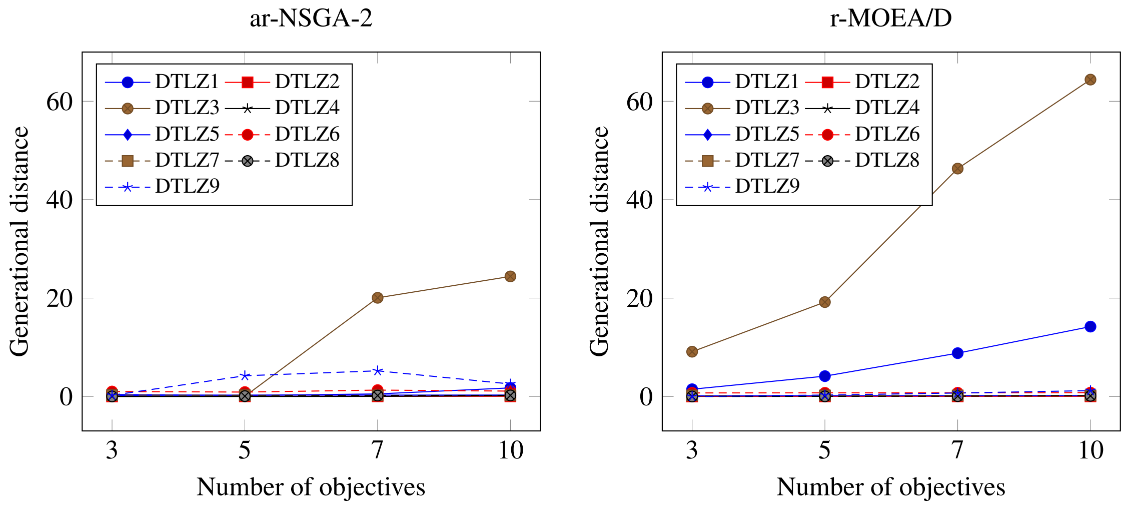


Figure 21. Varying numbers of objectives on DTLZ1-9. $I = 1000$, $|P| = 80$, $p_m = 0.050$, $\eta_m = 5$, and $m = 3$.

time taken goes up with the number of objectives. The increase in complexity also leads to longer computational times.

5.6 Conclusion

In this chapter, we have described the implementation of ar-NSGA-2 and r-MOEA/D and covered their performance on the ZDT and DTLZ test sets. To perform the case study, we chose one of these algorithms to implement the proof of concept model we defined in Chapter 4 and Appendix C. We based this choice on the performance of the algorithm on the test set, as well as how straightforward the algorithm is to understand and implement. This allowed us to use a quantitative and qualitative measure when deciding which algorithm to implement for the proof of concept.

The performance of the algorithms follows from the analysis of the testing runs we described above. Based on this data, we conclude that ar-NSGA-2 performs better overall. In many of the tests, it comes closer to the optimal frontier and is shown to be more robust against changes in mutation factor, population size, and number of iterations. In most of the test cases, it clearly outperforms r-MOEA/D. The robustness against changes in parameters makes it plausible that ar-NSGA-2 is easier to implement in various situations, because it may not be as dependent on perfect parameter tuning as r-MOEA/D.

Research has shown that maintaining a software system can account for a large portion of the costs of

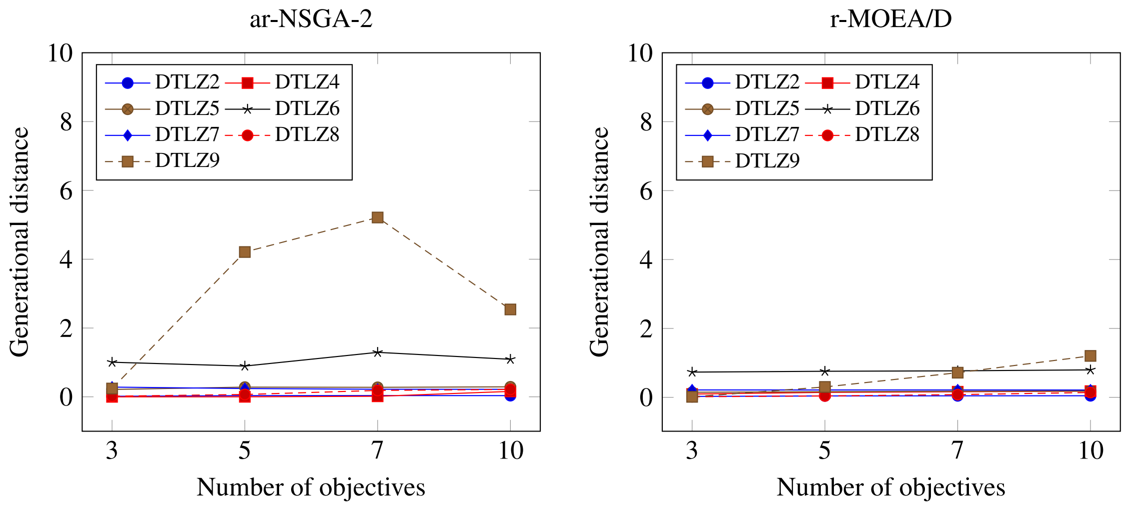


Figure 22. Varying numbers of objectives on DTLZ2 and DTLZ4-9. $I = 1000$, $|P| = 80$, $p_m = 0.050$, $\eta_m = 5$, and $m = 3$.

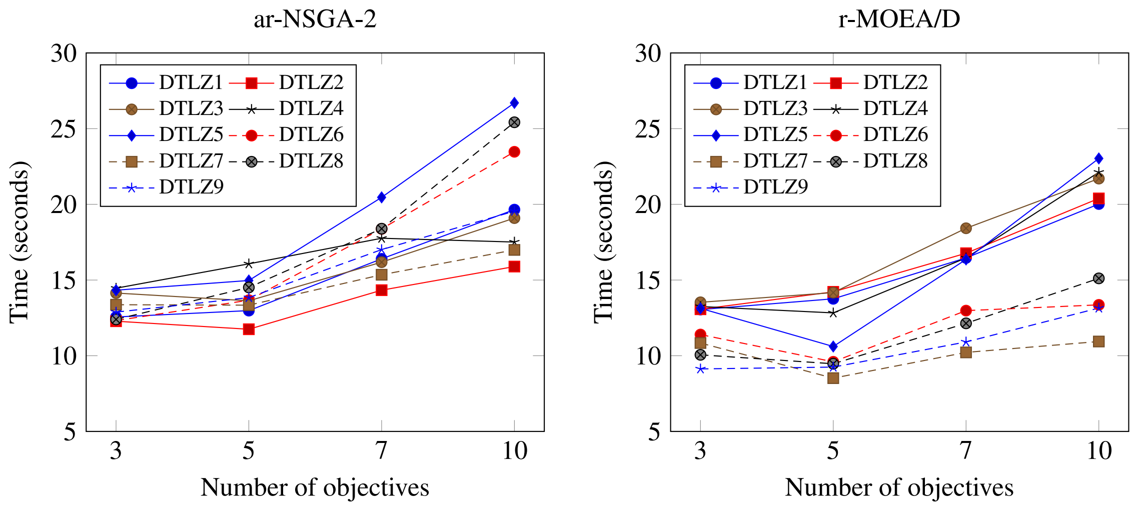


Figure 23. Varying numbers of objectives on DTLZ1-9. $I = 1000$, $|P| = 80$, $p_m = 0.050$, $\eta_m = 5$, and $m = 3$.

a software project (Heričko and Šumak (2023)). Given that we are exploring the practical application of an optimization algorithm, we must consider the maintainability of the algorithm. One aspect of this maintainability is how straightforward the algorithm is to understand. An algorithm that is easier to understand and implement may save costs on implementation, improvement, and maintenance. We find that ar-NSGA-2 uses a far more straightforward strategy to update the population compared to r-MOEA/D. In general, NSGA-2 uses a simple non-dominated searching strategy to update the population and archive, while MOEA/D uses a more convoluted local replacement.

Our analysis of the algorithms has shown that ar-NSGA-2 performs better and is preferable in terms of understanding and implementation. Therefore, we performed the case study with the proof of concept on realistic data using the ar-NSGA-2 algorithm.

6 CASE STUDY VALIDATION AND EVALUATION

In this chapter, we describe how we evaluate the algorithm we selected in Chapter 5. First, we describe how we collected the real-world nesting cases and show some characteristics of each case. Then, we explain the tests that we perform on these cases and show the results of these tests. Finally, we consider the evaluation with the decision-makers and draw our conclusions.

6.1 Cases

For this case study we collected nestings for steel construction projects from one of the client companies we interviewed for the work described in Chapter 4. They provided 19 nesting cases (C1-19) from their previous steel construction projects. On top of this, we also collected 9 nesting cases (C20-28) which the current algorithm is known to perform poorly on from various client companies.

Table 4. Case characteristics

Case ID	Unique Parts	Total Parts	Phases
C1	53	75	1
C2	8	41	2
C3	1	7	1
C4	19	26	2
C5	43	53	4
C6	5	6	1
C7	25	49	3
C8	5	53	1
C9	22	36	3
C10	1	12	1
C11	10	11	2
C12	2	10	1
C13	76	115	7
C14	3	3	1
C15	2	2	1
C16	43	78	7
C17	49	82	4
C18	5	6	1
C19	5	12	1
C20	1	4	1
C21	8	17	1
C22	27	37	1
C23	2	6	1
C24	6	34	1
C25	4	11	1
C26	4	4	1
C27	3	41	1
C28	5	9	1

Table 4 shows the cases and the characteristics such as the unique number of parts, the total number of parts, and the number of phases. Each case contains several unique parts, which may be produced more than once, resulting in the total number of parts in a case. Multiple production phases may be nested together to improve the efficiency of the nesting solution by leveraging economies of scale. In general, cases with a higher number of parts or more phases will be more difficult to optimize.

To make testing simpler, we saved each case in a JSON file. The case file contains information about the profile, the parts, and the available beam lengths. Listing 1 shows an example of the structure of the case files.

Listing 1. Case File Example

```
{
    "Profile": "IPE240",
    "Parts": [
        {
            "Length": 6700,
            "Count": 10,
            "Phase": 1
        },
        {
            "Length": 4400,
            "Count": 4,
            "Phase": 2
        }
    ],
    "BeamLengths": [
        11000,
        15000,
        17000,
        22000
    ]
}
```

These JSON files enabled us to perform many of the tests automatically. By storing the data in this way, we also ensured that each testing run used the same information.

6.2 Comparative testing

The first step in our testing is to evaluate how the nesting algorithms perform when presented with the same real-world cases. To do this, we ran comparative tests to determine which algorithms provided the dominant solutions under various circumstances.

Although we selected one algorithm in Section 5.6, we performed the case study evaluation on two versions of the ar-NSGA-2 algorithm. One of the factors we chose to implement in the proof of concept was the number of beams with multiple phases. We noticed that using the local optimization and repair by Falkenauer (1996) was generating solutions with many mixed-phase beams. In Section 4.2.2, we explained that beams with multiple phases increase handling and storage costs. By reducing the number of mixed-phase beams, we reduce these costs. Therefore, we also implemented a version of the Falkenauer (1996) local optimization and repair that does not produce mixed-phase beams. We have named this the *phase preserving* local optimization and repair.

As a result, we performed our tests on three algorithms: the single-objective benchmark (SO), the multi-objective ar-NSGA-2 implementation (MO), and the multi-objective ar-NSGA-2 implementation with phase preservation (MO-PP). For the MO and MO-PP algorithms, we use the following values for the thresholds and costs:

- Remnant discount threshold: $T_{discount} = 6000$
- Oversized load threshold: $T_{oversized} = 18000$
- SPRS threshold: $T_{sprs} = 25$
- Cost of an oversized load: $C_{oversized} = 100$
- Remnant discount factor: $d = 0.15$

These values were chosen based on conversations with decision-makers and internal experts at DIGI-STEEL. The cost of steel (C_{steel}) depends on the profile defined for each case[13].

The first test compares the algorithms without introducing any restrictions. We ran the 28 cases on the algorithms, collected the resulting solutions, and evaluated these solutions on the objectives defined in the proof of concept.

Using the solutions found by the three algorithms, we created a combined Pareto front with all of the non-dominating solutions. This combined Pareto front contains all solutions with the property that any other solutions better in one objective must be worse in another. From there, we determined which algorithm(s) had contributed solutions to this combined Pareto front.

If all of the solutions in the Pareto front have been contributed by one algorithm, we consider this algorithm to be fully dominant over the others. If an algorithm provided none of the solutions in the Pareto front, we regard it as fully dominated by the other algorithms. For this analysis, we use the definition of Pareto dominance as described in Section 3.1.

Table 5. Domination characteristics

Outcome	Number of cases	Percentage of cases
SO dominates	1	3.57%
MO dominates	1	3.57%
MO-PP dominates	3	10.71%
Both MO dominate	13	46.43%
Equal	7	25.00%
Indifferent	3	10.71%

Table 5 shows the number and percentage of cases for which the algorithms were dominant. When the algorithms produced solutions that were exactly equal in terms of their objective values, we noted them as *Equal*. When the single-objective algorithm and at least one of the multi-objective algorithms produced Pareto-indifferent solutions, we describe these algorithms as *Indifferent* in the table.

Table 6. Domination characteristics

Outcome	Number of cases	Percentage of cases
SO dominates	1	3.57%
Any MO dominates	17	60.71%
No dominant algorithm	10	35.71%

Table 6 shows the outcome of the comparison when we combine the performances of the two multi-objective variants. From these tables, we conclude that in most cases, the multi-objective implementations, and more specifically the MO-PP implementation, perform equally well or better compared to the SO benchmark.

However, while the multi-objective algorithms are intended to optimize for the objectives in the proof of concept model, the single-objective benchmark algorithm only optimizes for the waste produced by the nesting. To make claims about the efficacy of adding more objectives to the optimization, we must also consider the performance in the single-objective domain.

To test the relative performance in the single-objective domain, we investigated how our ar-NSGA-2 implementation would perform when presented with one objective. We defined two objectives that we could use for this test: a cost objective and an efficiency objective. The cost objective is the same as the cost objective in the proof of concept model. The efficiency objective measures the amount of waste material created by a nesting solution, compared to the total length of steel used. The benchmark SO algorithm uses the efficiency objective.

Table 7 shows the performance of the multi-objective algorithm when it is presented with one objective. We ran the benchmark algorithm and our algorithm on the 28 cases and evaluated the outcomes based on the expected cost of the solution and the total length of steel purchased. For both measures, we counted the cases where each algorithm outperforms the other in cost and total length.

In single-objective performance, the MO outperforms or matches the SO benchmark in almost all cases. In both comparisons, we observe that ar-NSGA-2 finds lower-cost solutions that the benchmark algorithm misses.

[13]For each profile, we collected the price per meter excluding taxes from `https://www.limtrade.nl` on May 6th, 2024.

Table 7. Performance on single-objective optimization

	Cost objective		Efficiency objective	
Outcome	**Cost**	**Total Length**	**Cost**	**Total Length**
SO outperforms	1	4	2	1
Equal performance	8	14	13	17
MO outperforms	19	10	13	10

The SO algorithm does find more solutions with a lower total length when ar-NSGA-2 is given only the cost objective. In three of the four cases where the benchmark finds a lower total length, the cost of the solution found by ar-NSGA-2 is lower than that of the benchmark solution. This suggests that the benchmark solution is unknowingly introducing extra costs. Indeed, looking at the lengths of the longest beams for the solutions, the SO algorithm tends to use beam lengths above the oversized load threshold, whereas the MO solution tends to use beam lengths below this threshold.

Finally, we compared the single-objective performance of ar-NSGA-2 to the multi-objective performance of ar-NSGA-2. By comparing the performance of our implementation in the single-objective and multi-objective domains, we can make an apples-to-apples comparison of the effect that adding more objectives has on the performance of the algorithm. Like with the first test, we compare the solutions generated by each algorithm on the cost, phase, and manual operation objectives defined in the proof of concept model.

Table 8. Comparing single-objective and multi-objective ar-NSGA-2

Case ID	Pareto Front Members
C1	MO, MO-PP
C2	MO-PP, COST
C3	MO, MO-PP, COST
C4	MO, MO-PP
C5	MO-PP, COST, EFF
C6	MO, MO-PP, COST
C7	MO-PP, COST, EFF
C8	MO, COST, EFF
C9	MO-PP, COST, EFF
C10	Equal
C11	MO, MO-PP
C12	Equal
C13	MO, MO-PP, COST, EFF
C14	Equal
C15	Equal
C16	MO-PP, COST, EFF
C17	MO, MO-PP
C18	Equal
C19	MO, MO-PP, COST
C20	Equal
C21	Equal
C22	Equal
C23	Equal
C24	MO, MO-PP
C25	MO, MO-PP
C26	Equal
C27	MO
C28	MO, MO-PP, COST

Table 8 shows the composition of the Pareto front for each case. For the cases where all algorithms produce the same solutions, the table notes *Equal*. The single-objective ar-NSGA-2 was run twice, once with the cost objective, and once with the efficiency objective. These are labelled as *COST* and *EFF* in the table. Based on this table, how the algorithms compare is not immediately obvious.

Table 9. Domination summary

Outcome	Count	Percentage
Single-objective dominates	0	0.00%
Multi-objective dominates	7	25.00%
Indifference	11	39.29%
Equality	10	35.71%

Therefore, we have compiled Table 9 as a summary of the results shown in Table 8. Based on this summary, we observe that the multi-objective approach matches or outperforms the single-objective approach in all cases. This shows that adding more objectives to the optimization is not noticeably detrimental to the performance of the other objectives.

This analysis shows that the MO and MO-PP algorithms match or outperform the existing SO nesting algorithm in almost all real-world cases we collected. Based on this numeric performance, we conclude that the newly proposed nesting algorithms (MO and MO-PP) have the potential to be effective improvements to the existing SO algorithm.

6.3 Decision-maker evaluation

An aspect of multi-objective optimization that has been a driving factor throughout this project is incorporating decision-maker preferences in the optimization process. Our review of the literature in Chapter The final step in evaluating the proposed implementation of ar-NSGA-2 is to determine the effects of adding more objectives to the optimization process with a decision-maker.

In our previous evaluation of the performance of the algorithms, we established that there are various cases where the algorithms produce incomparable solutions. We have selected 11 cases to evaluate with a decision-maker based on the following criteria:

- The SO and MO/MO-PP solutions have equal cost, the MO/MO-PP solution dominates in the other objectives.
- The SO and MO/MO-PP solutions are Pareto-indifferent
- The SO solution has a higher cost, and the MO/MO-PP solution has a higher total length

For each case, we selected one solution solution from the SO, MO, and MO-PP algorithms for the decision-makers to compare. For each solution, we show them the efficiency (percentage of steel used for parts), the total purchased length, the number of purchased beams, the cost objective, the phase objective, the manual operation objective, the shortest beam, and the longest beam. In addition, we provided information about the steel profile, number of parts, and number of phases. Based on a review by the internal experts, we added information about remnant steel to some of the cases where relevant.

We asked two internal experts at DIGI-STEEL and two decision-makers at a client company to evaluate these cases. The cases were presented to the internal experts and the decision-makers with no indication of which solution came from which algorithm and no indication of how these algorithms function. The order of the solutions was randomized[14] to prevent a situation where the choice is made based on a pattern or preference for one particular algorithm.

Table 10 shows each decision-maker's preferred solution. The decision-makers could discuss the examples with each other and would prefer the same nesting solution in most cases. They chose different solutions for Case 10 based on personal preferences. We observed that they prioritized certain nesting factors differently, therefore reaching a different conclusion regarding which solution was the best.

Table 10 shows that the internal experts preferred the SO solution more often than the decision-makers. This suggests that their choices may be influenced by their previous understanding of the nesting problem. To understand the impact of this potential bias, we decided to redo the interview with internal expert 1 (who had the biggest difference from the decision-makers). Given that the SO algorithm's objective is efficiency, we suspected

[14]We used the free List Randomizer from `https://www.random.org` to randomize the order of the solutions for each case.

Table 10. Preferred solutions

Case #	Internal expert 1	Internal expert 2	DM 1	DM 2
1	MO-PP	MO-PP	MO	MO
2	SO	MO/MO-PP	MO/MO-PP	MO/MO-PP
3	MO	MO	MO	MO
4	SO	MO-PP	MO-PP	MO-PP
5	SO	MO-PP	MO-PP	MO-PP
6	MO-PP	MO-PP	MO	MO
7	SO	SO	SO	SO
8	SO	SO	MO-PP	MO-PP
9	SO	SO	MO-PP	MO-PP
10	MO	MO	MO	MO-PP
11	SO	SO	SO	SO

that removing the efficiency of the solutions from the presented information may influence the expert's choices. In the repeated interview, we showed internal expert 1 the same 11 cases without the efficiency information.

Table 11. Preferred solutions for the repeated interview

Case #	Original choice	New choice
1	MO-PP	PP
2	SO	MO/MO-PP
3	MO	MO
4	SO	SO
5	SO	MO-PP
6	MO-PP	MO-PP
7	SO	SO
8	SO	SO
9	SO	MO-PP
10	MO	MO
11	SO	MO

Table 11 shows the solutions originally preferred by internal expert 1 and their preferred solutions when shown the solutions without the efficiency information. Given no further instruction, internal expert 1 displayed a shift towards the solutions found by MO and MO-PP when they could not choose a solution based on efficiency. This brings their choices much more in line with the decision-maker's choices.

Besides evaluating the cases, we also discussed the setup of the interview and what requirements they might have for a new nesting implementation. This discussion led to two interesting observations: the importance of context and the importance of control.

Throughout the evaluation, the decision-makers often remarked that when choosing between the given options, the option which was the best nesting solution could change drastically depending on the context. For most of the 11 cases, they would name one or more factors that would influence the evaluation of the nesting which had not been captured in the presented data. These factors were captured in the tables in Section 4.2, but were not included in the proof of concept model. On the whole, both decision-makers stated that being shown more information offered them more context to make their choice between the solutions. In this thesis, we claimed that giving a decision-maker more information to make a decision would enable them to make decisions that better fit their preferences. This feedback from the decision-makers validates this claim. Furthermore, this feedback shows that the factors that we captured during the interview process are relevant to the nesting process.

The control discussion is summarized by decision-maker 2 as ”I just want to be able to choose between a couple of solutions”. A critical point of feedback on the existing algorithm is that it only ever produces one solution. The decision-makers were adamant that having multiple options to choose from would improve their interaction with the nesting algorithm. More specifically, they wanted to be presented with a ’best effort’ solution up front and then given the option to review other candidate solutions should they want to do so. This indicates that an a priori preference model may fit the requirements of high-volume nesting workflows better than an interactive preference model. The desire for multiple options also validates the generally accepted understanding that presenting a set of Pareto-indifferent solutions to the decision-makers will improve their interaction with the nesting algorithm.

During the initial interviews with internal experts, we were warned that the decision-makers may not be receptive to the idea of evaluating nestings on other criteria besides efficiency. However, the interviews in Chapter 4 and the discussion described in this section have shown that decision-makers understand the context of their nesting decisions very well, and can reason about how the factors that influence a nesting interact and lead to trade-off decisions. In the context of the steel manufacturing industry, we believe decision-makers can play an important role in the development of effective preference models to support their nesting decisions.

6.4 Conclusion

In this chapter, we have comparatively tested the performance of our ar-NSGA-2 implementations against the single-objective benchmark and evaluated implementing multiple objectives with expert decision-makers using real-world cases.

The comparative tests have focused on the relative performance of the algorithms. First, we showed that the benchmark is matched or outperformed in almost all cases when evaluated according to the objectives defined in the proof of concept model. Next, we showed that the proposed ar-NSGA-2 implementations outperform the benchmark in almost all cases when run in a single-objective mode. Finally, we have shown that multi-objective ar-NSGA-2 matches or outperforms single-objective ar-NSGA-2 in all cases.

Based on these results, we conclude that our implementation of ar-NSGA-2 is an improvement on the benchmark. The testing results and the evaluations with the internal experts and the decision-makers show that a multi-objective approach to the nesting problem gives the decision-maker more control over their preferred solution to a nesting. The evaluations have also shown that the way information is presented, as well as which information is presented, can influence the nesting decision. Therefore, we conclude that using a multi-objective algorithm that can include more contextual information in the objectives and constraints is a valuable asset to a decision-maker for finding solutions to the nesting problem.

7 CONCLUSION

The goal of this research was to implement a proof of concept of a multi-objective optimization algorithm for the nesting problem. To develop this proof of concept, we answered the three research questions **RQ1-3** described in Section 1.2. In this chapter, we draw our conclusions to these research questions, discuss the various aspects of multi-objective optimization we discovered during our research, provide directions for future work, and acknowledge the invaluable help we have had along the way.

RQ1 asks which factors influence a decision-maker's evaluation of a nesting, and how we use these to construct an optimization model. In Chapter 4, we interviewed expert decision-makers from the steel manufacturing industry and compiled an extensive list of factors. We identified four categories that these factors could be loosely sorted into, which could be used to guide future investigations into these factors. Finally, we created a proof of concept model for a multi-objective optimization of the nesting problem that incorporates a representative selection of these factors.

RQ2 asks which algorithm from a small selection of state-of-the-art algorithms is most suited to solve the nesting problem. In Chapter 5, we used test sets and a performance metric from the literature to compare the ar-NSGA-2 and r-MOEA/D algorithms. Based on these tests and our experiences implementing the algorithms, we concluded that ar-NSGA-2 performed better and is preferable in terms of understandability and ease of implementation. We selected ar-NSGA-2 to implement the proof of concept optimization model.

RQ3 asks how the selected algorithm compares to the existing benchmark. In Chapter 6 we compared the performance of these algorithms in 28 real-world cases. We also asked internal experts and decision-makers to evaluate the solutions found by the algorithms in 11 of these cases. Based on the performance of the algorithms and the evaluations by the internal experts and decision-makers, we concluded that our implementation of the ar-NSGA-2 algorithm is preferable over the benchmark algorithm currently used in production because of the improved performance and by providing the decision-maker with multiple solutions.

To conclude, in this research we analysed the factors that influence a decision-maker's choices when evaluating solutions to the nesting problem and showed that a multi-objective optimization algorithm that incorporates these factors improves nesting performance and gives the decision-maker more control over the outcome of the nesting process.

7.1 Discussion

Throughout our work, we discovered various aspects of multi-objective optimization that, while we did not get the chance to explore them further, are worth discussing.

In Chapter 3 we performed a structured review of the literature regarding multi-objective optimization with preference modelling. The relevant studies for this review belong to two research domains: *multi-objective optimization (MOO)* and *multi-criteria decision-making (MCDM)*. We found that these domains do not share the same keywords, leading to a fractured field of research. For example, when considering optimization problems with multiple objectives, studies may use the keywords *multi-objective*, *multi-criteria*, and even *multi-attributre* interchangeably. This makes it difficult to find relevant studies because there is no single definition or keyword for many of the concepts used in the field.

On top of this, our literature review showed that the state-of-the-art is moving towards interactive preference modelling techniques. The idea behind these techniques is to give the decision-maker multiple opportunities to guide the optimization process. However, we found that in the steel construction industry, decision-makers perform many optimizations per day, often in parallel. Therefore, requiring them to interact with each of these optimization runs puts a lot of strain on the decision-makers. In our study, we focused on the factors that influenced the evaluation of a nesting and proposed an approach to modelling such an optimization. Our work does not answer which preference modelling technique could best be applied to this optimization.

In Chapter 6 we noted that we implemented two versions of ar-NSGA-2. We found that the local optimization and repair step in the evolutionary algorithm had a profound impact on the solutions it would produce. In this work, we did not explore this behaviour in depth, although the good performance of the MO-PP implementation of ar-NSGA-2 shows that finding more problem-specific local optimization and repair mechanisms may improve the performance of multi-objective optimization algorithms in real-world, high-volume contexts.

Another aspect of nesting that we discussed in Chapter 4 is that each decision-maker has different ideas about which factors are relevant to the evaluation of a nesting. One research direction that we have not found in the state-of-the-art is whether the ability to turn objectives or constraints on and off at will could improve the usability of the optimization for the decision-maker.

Finally, in our evaluation of the nestings with the internal experts and the decision-makers, we found that the way that information was presented greatly influenced the choices the subjects made. This phenomenon was too far outside the scope of this work to investigate it any further.

7.2 Future work

Based on our discussion, we have identified several topics for future work.

Unification of the terminology used in the research into multi-objective optimization, specifically the studies concerning preference modelling, via a mapping study. Such a study could provide an in-depth understanding of the terminology and a proposition for a unified terminology may aid future literature reviews.

Which preference modelling techniques may apply to high-volume optimization environments. Further investigations could improve our understanding of how multi-objective optimization techniques could be applied to real-world optimization problems, especially in industries where the decision-maker does not have the time or mental capacity to interactively guide a high number of optimizations.

A study to define best practices for local optimization and repair mechanics, especially for problems with more complex models. We observed that a local optimization and repair that is not aware of the global objectives could lead to sub-par results. A study that investigates this connection further could improve the local optimization and repair steps for complex real-world problems.

Modularization of the optimization model for an evolutionary algorithm as an extension to the investigation into local optimization and repair best practices. By giving the decision-maker the option to turn on or off the objectives and constraints, and by creating local optimizations and repair mechanics that support this, they could adapt the optimization model to their current needs on a case-by-case basis.

Finally, an investigation into how information should be presented when making a decision. This could lead to improvements in the ways that decision-makers are informed about the trade-offs between various solutions to an optimization problem. Based on our investigation, this could lead to more informed choices, which in turn could lead to a better fit between the chosen solution and the business processes that rely on this solution.

7.3 Contribution

Our contribution to the state-of-the-art is a structured review of the literature regarding preference modelling and multi-objective optimization from 2004 until 2023, the comparison of two multi-objective optimization algorithms from the state-of-the-art, and the application of one state-of-the-art algorithm to a real-world case study. Furthermore, we have shown that local optimization and repair mechanics have a profound impact on the performance of a multi-objective optimization algorithm when applied to a real-world problem. Our investigation has also shown that the dominant understanding that interactive preference modelling is preferable cannot be applied to situations where many optimizations are run each day. Finally, we have shown the importance of applying multi-objective optimization algorithms to real-world cases as well as test sets from the literature. In our work, we found that some accepted notions from the state of the art, such as the effectiveness of interactive preference modelling, cannot be applied to all real-world scenarios.

REFERENCES

Abd El-Wahed, W. and Lee, S. (2006). Interactive fuzzy goal programming for multi-objective transportation problems. *Omega*, 34(2):158–166. Publisher: Elsevier BV.

Abdolshah, M., Shilton, A., Rana, S., Gupta, S., and Venkatesh, S. (2019). Multi-objective Bayesian optimisation with preferences over objectives. In *Advances in Neural Information Processing Systems*, volume 32. Neural information processing systems foundation. ISSN: 10495258.

Abouhawwash, M. and Deb, K. (2021). Reference point based evolutionary multi-objective optimization algorithms with convergence properties using KKTPM and ASF metrics. *Journal of Heuristics*, 27(4):575–614. Publisher: Springer.

Aggelogiannaki, E. and Sarimveis, H. (2007). A simulated annealing algorithm for prioritized multiobjective optimization - Implementation in an adaptive model predictive control configuration. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 37(4):902–915.

Astudillo, R. and Frazier, P. (2020). Multi-attribute Bayesian optimization with interactive preference learning. In Chiappa S., C. R., editor, *Proceedings of Machine Learning Research*, volume 108, pages 4496–4507. ML Research Press. ISSN: 26403498.

Augeri, M. G., Greco, S., Distefano, N., and Leonardi, S. (2021). Road Safety Resource Allocation Using Interactive Multiobjective Optimization. *European Transport - Trasporti Europei*, 0(84). Publisher: Institute for Transport Studies in the European Economic Integration Type: Article.

Balderas, F., Fernandez, E., Gomez-Santillan, C., Rangel-Valdez, N., and Cruz, L. (2019). An Interval-Based Approach for Evolutionary Multi-Objective Optimization of Project Portfolios. *International Journal of Information Technology and Decision Making*, 18(4):1317–1358. Publisher: World Scientific Publishing Co. Pte Ltd.

Battiti, R. and Passerini, A. (2010). Brain–Computer Evolutionary Multiobjective Optimization: A Genetic Algorithm Adapting to the Decision Maker. *IEEE Transactions on Evolutionary Computation*, 14(5):671–687. Conference Name: IEEE Transactions on Evolutionary Computation.

Baykasoğlu, A. (2005). Preemptive goal programming using simulated annealing. *Engineering Optimization*, 37(1):49–63.

Bemporad, A. and Piga, D. (2021). Global optimization based on active preference learning with radial basis functions. *Machine Learning*, 110(2):417–448. Publisher: Springer.

Ben Said, L., Bechikh, S., and Ghedira, K. (2010). The r-Dominance: A New Dominance Relation for Interactive Evolutionary Multicriteria Decision Making. *IEEE Transactions on Evolutionary Computation*, 14(5):801–818. Conference Name: IEEE Transactions on Evolutionary Computation.

Benabbou, N., Leroy, C., and Lust, T. (2020). An interactive regret-based genetic algorithm for solving multi-objective combinatorial optimization problems. In *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, pages 2335–2342. AAAI press.

Benayoun, R., Roy, B., and Sussman, B. (1966). ELECTRE: Une méthode pour guider le choix en présence de points de vue multiples. *Note de travail*, 49:2–120.

Benecke, T. and Mostaghim, S. (2021). The Impact of Population Size on the Convergence of Multi-objective Evolutionary Algorithms. In *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8.

Bezoui, M., Olteanu, A.-L., and Sevaux, M. (2023). Integrating preferences within multiobjective flexible job shop scheduling. *European Journal of Operational Research*, 305(3):1079 – 1086. Publisher: Elsevier B.V. Type: Article.

Borges, J., Garcia-Gonzalo, J., Bushenkov, V., McDill, M., Marques, S., and Oliveira, M. (2014). With addressing multicriteria forest management Pareto frontier methods: An application in Portugal. *Forest Science*, 60(1):63–72. Publisher: Society of American Foresters.

Brafman, R. and Chernyavsky, Y. (2005). Planning with Goal Preferences and Constraints.

Branke, J., Corrente, S., Greco, S., Słowiński, R., and Zielniewicz, P. (2016). Using Choquet integral as preference model in interactive evolutionary multiobjective optimization. *European Journal of Operational Research*, 250(3):884–901. Publisher: Elsevier Type: Article.

Branke, J. and Deb, K. (2005). Integrating User Preferences into Evolutionary Multi-Objective Optimization. In Jin, Y., editor, *Knowledge Incorporation in Evolutionary Computation*, Studies in Fuzziness and Soft Computing, pages 461–477. Springer, Berlin, Heidelberg.

Branke, J., Greco, S., Słowiński, R., and Zielniewicz, P. (2010). Interactive evolutionary multiobjective optimization driven by robust ordinal regression. *Bulletin of the Polish Academy of Sciences: Technical Sciences*, 58(3):347–358.

Branke, J., Greco, S., Słowiński, R., and Zielniewicz, P. (2015). Learning Value Functions in Interactive Evolutionary Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation*, 19(1):88–102. Publisher: Institute of Electrical and Electronics Engineers Inc. Type: Article.

Brans, J., Vincke, P., and Mareschal, B. (1986). How to select and how to rank projects: The Promethee method. *European Journal of Operational Research*, 24(2):228–238.

Braun, M., Heling, L., Shukla, P., and Schmeck, H. (2017a). Multimodal Scalarized Preferences in Multi-Objective Optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '17, pages 545–552, New York, NY, USA. Association for Computing Machinery. event-place: Berlin, Germany.

Braun, M., Shukla, P., and Schmeck, H. (2011). Preference ranking schemes in multi-objective evolutionary algorithms. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6576 LNCS:226–240. ISBN: 9783642198922 Place: Ouro Preto.

Braun, M., Shukla, P., and Schmeck, H. (2015). Obtaining optimal Pareto front approximations using scalarized preference information. In S, S., editor, *GECCO 2015 - Proceedings of the 2015 Genetic and Evolutionary Computation Conference*, pages 631–638. Association for Computing Machinery, Inc.

Braun, M., Shukla, P., and Schmeck, H. (2017b). Angle-based preference models in multi-objective optimization. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10173 LNCS:88–102. ISBN: 9783319541563 Publisher: Springer Verlag.

Brockhoff, D., Bader, J., Thiele, L., and Zitzler, E. (2013). Directed Multiobjective Optimization Based on the Weighted Hypervolume Indicator. *Journal of Multi-Criteria Decision Analysis*, 20(5-6):291–317. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/mcda.1502.

Carles-Bou, J. L. and Galán, S. F. (2023). Self-adaptive polynomial mutation in NSGA-II. *Soft Computing*, 27(23):17711–17727.

Castellanos, A., Cruz-Reyes, L., Fernández, E., Rivera, G., Gomez-Santillan, C., and Rangel-Valdez, N. (2022). Hybridisation of Swarm Intelligence Algorithms with Multi-Criteria Ordinal Classification: A Strategy to Address Many-Objective Optimisation. *Mathematics*, 10(3). Publisher: MDPI.

Castellanos-Alvarez, A., Cruz-Reyes, L., Fernandez, E., Rangel-Valdez, N., Gómez-Santillán, C., Fraire, H., and Brambila-Hernández, J. A. (2021). A Method for Integration of Preferences to a Multi-Objective Evolutionary Algorithm Using Ordinal Multi-Criteria Classification. *Mathematical and Computational Applications*, 26(2).

Chen, L., Xin, B., and Chen, J. (2017). A tradeoff-based interactive multi-objective optimization method driven by evolutionary algorithms. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 21(2):284–292. Publisher: Fuji Technology Press.

Cheng, F. and Jia, J. (2021). Quantitative Interactive Investment Algorithm Based on Machine Learning and Data Mining. In *2021 4th International Conference on Data Science and Information Technology*, DSIT 2021, pages 396–401, New York, NY, USA. Association for Computing Machinery. event-place: Shanghai, China.

Cheng, R., Jin, Y., Olhofer, M., and Sendhoff, B. (2016). A Reference Vector Guided Evolutionary Algorithm for Many-Objective Optimization. *IEEE Transactions on Evolutionary Computation*, 20(5):773–791. Conference Name: IEEE Transactions on Evolutionary Computation.

Chica, M., Cordón, O., Damas, S., and Bautista, J. (2015). Interactive preferences in multiobjective ant colony optimisation for assembly line balancing. *Soft Computing*, 19(10):2891–2903. Publisher: Springer Verlag.

Chugh, T., Sindhya, K., Hakanen, J., and Miettinen, K. (2015). An interactive simple indicator-based evolutionary algorithm (I-SIBEA) for multiobjective optimization problems. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9018:277–291. ISBN: 9783319159331 Publisher: Springer Verlag.

ConstruSteel (2024). Bar nesting.

Cordone, R., Milesi, M., and Salani, M. (2007). A decision support tool to plan shifts in a home for the aged. In *2007 IEEE International Conference on Service Operations and Logistics, and Informatics, SOLI*, pages 1–6. Type: Conference paper.

Cruz-Reyes, L., Fernandez, E., Sanchez, P., Coello Coello, C., and Gomez, C. (2017). Incorporation of implicit decision-maker preferences in multi-objective evolutionary optimization using a multi-criteria classification method. *Applied Soft Computing Journal*, 50:48–57. Publisher: Elsevier Ltd.

Cruz-Reyes, L., Fernandez, E., Sanchez-Solis, J., Coello Coello, C., and Gomez, C. (2020). Hybrid evolutionary multi-objective optimisation using outranking-based ordinal classification methods. *Swarm and Evolutionary Computation*, 54. Publisher: Elsevier B.V.

Dalpiaz, F., Franch, X., and Horkoff, J. (2016). iStar 2.0 Language Guide. arXiv:1605.07767 [cs].

Dasdemir, E., Köksalan, M., and Tezcaner Öztürk, D. (2020). A flexible reference point-based multi-objective evolutionary algorithm: An application to the UAV route planning problem. *Computers and Operations Research*, 114. Publisher: Elsevier Ltd.

De Jong, K. A. (1975). An Analysis of the Behavior of a Class of Genetic Adaptive Systems. *Doctoral Thesis, University of Michigan.*

Deb, K. and Kumar, A. (2007a). Interactive evolutionary multi-objective optimization and decision-making using reference direction method. In *Proceedings of GECCO 2007: Genetic and Evolutionary Computation Conference*, pages 781–788, London.

Deb, K. and Kumar, A. (2007b). Light beam search based multi-objective optimization using evolutionary algorithms. In *2007 IEEE Congress on Evolutionary Computation, CEC 2007*, pages 2125–2132.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002a). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197. Conference Name: IEEE Transactions on Evolutionary Computation.

Deb, K. and Sundar, J. (2006). Reference point based multi-objective optimization using evolutionary algorithms. In *GECCO 2006 - Genetic and Evolutionary Computation Conference*, volume 1 of *GECCO '06*, pages 635–642, New York, NY, USA. Association for Computing Machinery (ACM).

Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2002b). Scalable multi-objective optimization test problems. In *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*, volume 1, pages 825–830 vol.1.

Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2005). Scalable Test Problems for Evolutionary Multiobjective Optimization. In Abraham, A., Jain, L., and Goldberg, R., editors, *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*, Advanced Information and Knowledge Processing, pages 105–145. Springer, London.

Dias, J., Captivo, M., and Clímaco, J. (2008). A memetic algorithm for multi-objective dynamic location problems. *Journal of Global Optimization*, 42(2):221–253.

Ehrgott, M., Klamroth, K., and Schwehm, C. (2004). An MCDM approach to portfolio optimization. *European Journal of Operational Research*, 155(3):752–770.

Ertuğul, I. and Güeş, M. (2007). Fuzzy goal programming and an application of production process. *Advances in Soft Computing*, 41:649–659. ISBN: 9783540724315.

Falkenauer, E. (1996). A hybrid grouping genetic algorithm for bin packing. *Journal of Heuristics*, 2(1):5–30.

Fang, J., Rao, Y., Luo, Q., and Xu, J. (2023). Solving One-Dimensional Cutting Stock Problems with the Deep Reinforcement Learning. *Mathematics*, 11(4):1028. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

Feliot, P., Bect, J., and Vazquez, E. (2019). User preferences in bayesian multi-objective optimization: The expected weighted hypervolume improvement criterion. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11331 LNCS:533–544. ISBN: 9783030137083 Publisher: Springer Verlag.

Fernandez, E., Felix, L., and Mazcorro, G. (2009). Multi-objective optimisation of an outranking model for public resources allocation on competing projects. *International Journal of Operational Research*, 5(2):190–210. Publisher: Inderscience Publishers.

Fernandez, E., Gomez, C., Rivera, G., and Cruz-Reyes, L. (2015). Hybrid metaheuristic approach for handling many objectives and decisions on partial support in project portfolio optimisation. *Information Sciences*, 315:102 – 122. Publisher: Elsevier Inc. Type: Article.

Fernandez, E., Lopez, E., Mazcorro, G., Olmedo, R., and Coello Coello, C. (2013). Application of the non-outranked sorting genetic algorithm to public project portfolio selection. *Information Sciences*, 228:131–149.

Fernandez, E., Navarro, J., Solares, E., and Coello, C. C. (2019). A novel approach to select the best portfolio considering the preferences of the decision maker. *Swarm and Evolutionary Computation*, 46:140–153. Publisher: Elsevier B.V. Type: Article.

Fernández, E., Figueira, J. R., and Navarro, J. (2020). Interval-based extensions of two outranking methods for multi-criteria ordinal classification. *Omega*, 95:102065.

Fernández, E., Rangel-Valdez, N., Cruz-Reyes, L., Gomez-Santillan, C., and Coello-Coello, C. (2022). Preference incorporation in MOEA/D using an outranking approach with imprecise model parameters. *Swarm and Evolutionary Computation*, 72. Publisher: Elsevier B.V.

FICEP (2024). Structural steel nesting software, steel fabrication bar nesting software.

Figueira, J. R., Mousseau, V., and Roy, B. (2016). ELECTRE Methods. In Greco, S., Ehrgott, M., and Figueira,

J. R., editors, *Multiple Criteria Decision Analysis: State of the Art Surveys*, International Series in Operations Research & Management Science, pages 155–185. Springer, New York, NY.

Filho, H., Ferreira, T., and Vergilio, S. (2018). Multiple objective test set selection for software product line testing: Evaluating different preference-based algorithms. In T, G., editor, *ACM International Conference Proceeding Series*, pages 162–171. Association for Computing Machinery.

Fliedner, T. and Liesiö, J. (2016). Adjustable robustness for multi-attribute project portfolio selection. *European Journal of Operational Research*, 252(3):931–946. Publisher: Elsevier B.V.

Fouchal, H., Gandibleux, X., and Lehuédé, F. (2011). Preferred solutions computed with a label setting algorithm based on Choquet integral for multi-objective shortest paths. In *IEEE SSCI 2011 - Symposium Series on Computational Intelligence - MCDM 2011: 2011 IEEE Symposium on Computational Intelligence in Multicriteria Decision-Making*, pages 143–150. Type: Conference paper.

Fowler, J. W., Gel, E. S., Köksalan, M. M., Korhonen, P., Marquis, J. L., and Wallenius, J. (2010). Interactive evolutionary multi-objective optimization for quasi-concave preference functions. *European Journal of Operational Research*, 206(2):417–425.

Friedrich, T., Kroeger, T., and Neumann, F. (2013). Weighted preferences in evolutionary multi-objective optimization. *International Journal of Machine Learning and Cybernetics*, 4(2):139–148.

Galand, L., Ismaili, A., Perny, P., and Spanjaard, O. (2013). Bidirectional preference-based search for multiobjective state space graph problems. In *Proceedings of the 6th Annual Symposium on Combinatorial Search, SoCS 2013*, pages 80 – 88. Type: Conference paper.

Galand, L. and Perny, P. (2007). Search for Choquet-optimal paths under uncertainty. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence, UAI 2007*, pages 125–132, Vancouver, BC.

Galand, L. and Spanjaard, O. (2007). OWA-based search in state space graphs with multiple cost functions. In *Proceedings of the Twentieth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2007*, pages 86–91, Key West, FL.

Gong, D., Sun, F., Sun, J., and Sun, X. (2017). Set-based many-objective optimization guided by a preferred region. *Neurocomputing*, 228:241–255. Publisher: Elsevier B.V.

Gong, D., Sun, J., and Ji, X. (2013). Evolutionary algorithms with preference polyhedron for interval multi-objective optimization problems. *Information Sciences*, 233:141–161.

Gong, M., Liu, F., Zhang, W., Jiao, L., and Zhang, Q. (2011). Interactive MOEA/D for multi-objective decision making. In *Genetic and Evolutionary Computation Conference, GECCO'11*, pages 721–728, Dublin.

Goulart, F. and Campelo, F. (2016). Preference-guided evolutionary algorithms for many-objective optimization. *Information Sciences*, 329:236–255.

Greco, S., Matarazzo, B., and Słowiński, R. (2010). Interactive evolutionary multiobjective optimization using dominance-based rough set approach. In *2010 IEEE World Congress on Computational Intelligence, WCCI 2010 - 2010 IEEE Congress on Evolutionary Computation, CEC 2010*, Barcelona.

Guan, Y., Yang, L., and Sheng, W. (2017). Population Control in Evolutionary Algorithms: Review and Comparison. In He, C., Mo, H., Pan, L., and Zhao, Y., editors, *Bio-inspired Computing: Theories and Applications*, pages 161–174, Singapore. Springer.

Guo, Y.-N., Zhang, X., Gong, D.-W., Zhang, Z., and Yang, J.-J. (2020). Novel Interactive Preference-Based Multiobjective Evolutionary Optimization for Bolt Supporting Networks. *IEEE Transactions on Evolutionary Computation*, 24(4):750–764. Publisher: Institute of Electrical and Electronics Engineers Inc.

Hakanen, J. and Knowles, J. (2017). On using decision maker preferences with ParEGO. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10173 LNCS:282–297. ISBN: 9783319541563 Publisher: Springer Verlag.

He, Y., He, Z., Kim, K.-J., Jeong, I.-J., and Lee, D.-H. (2021). A Robust Interactive Desirability Function Approach for Multiple Response Optimization Considering Model Uncertainty. *IEEE Transactions on Reliability*, 70(1):175–187. Publisher: Institute of Electrical and Electronics Engineers Inc.

He, Y., Sun, J., Song, P., Wang, X., and Usmani, A. (2020). Preference-driven Kriging-based multiobjective optimization method with a novel multipoint infill criterion and application to airfoil shape design. *Aerospace Science and Technology*, 96. Publisher: Elsevier Masson SAS.

Heričko, T. and Šumak, B. (2023). Exploring Maintainability Index Variants for Software Maintainability Measurement in Object-Oriented Systems. *Applied Sciences*, 13(5):2972. Number: 5 Publisher: Multidisciplinary Digital Publishing Institute.

Hu, C. and Li, S. (2006). Enhanced interactive satisfying optimization approach to multiple objective optimization with preemptive priorities. *International Journal of Information Technology and Decision Making*, 5(1):47–63.

Publisher: World Scientific Publishing Co. Pte Ltd.

Hu, C.-F., Teng, C.-J., and Li, S.-Y. (2007). A fuzzy goal programming approach to multi-objective optimization problem with priorities. *European Journal of Operational Research*, 176(3):1319–1333.

Hu, J., Yu, G., Zheng, J., and Zou, J. (2017). A preference-based multi-objective evolutionary algorithm using preference selection radius. *Soft Computing*, 21(17):5025–5051. Publisher: Springer Verlag.

Hu, S., Li, D., Jia, J., and Liu, Y. (2021). A self-learning based preference model for portfolio optimization. *Mathematics*, 9(20). Publisher: MDPI Type: Article.

Huang, H.-Z., Tian, Z., and Zuo, M. J. (2005). Intelligent interactive multiobjective optimization method and its application to reliability optimization. *IIE Transactions*, 37(11):983–993. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/07408170500232040.

Huband, S., Hingston, P., Barone, L., and While, L. (2006). A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506.

Hunt, B. J., Wiecek, M. M., and Fadel, G. (2004). Matrices as preference modeling tools in bi-criteria engineering design. In *Collection of Technical Papers - 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, volume 1, pages 212–220, Albany, NY. American Institute of Aeronautics and Astronautics Inc. Type: Conference paper.

Hwang, C.-L. and Yoon, K. (1981). Methods for Multiple Attribute Decision Making. In Hwang, C.-L. and Yoon, K., editors, *Multiple Attribute Decision Making: Methods and Applications A State-of-the-Art Survey*, Lecture Notes in Economics and Mathematical Systems, pages 58–191. Springer, Berlin, Heidelberg.

Iniestra, J. and Gutiérrez, J. (2009). Multicriteria decisions on interdependent infrastructure transportation projects using an evolutionary-based framework. *Applied Soft Computing Journal*, 9(2):512–526.

Jaimes, A., Montaño, A., and Coello, C. (2011). Preference incorporation to solve many-objective airfoil design problems. In *2011 IEEE Congress of Evolutionary Computation, CEC 2011*, pages 1605–1612, New Orleans, LA.

Jia, S., Yi, J., Yang, G., Du, B., and Zhu, J. (2013). A multi-objective optimisation algorithm for the hot rolling batch scheduling problem. *International Journal of Production Research*, 51(3):667–681.

Junker, U. (2004). Preference-based search and multi-criteria optimization. *Annals of Operations Research*, 130(1-4):75–115. Publisher: Kluwer Academic Publishers.

Kaddani, S., Tamby, S., Vanderpooten, D., and Vanpeperstraete, J.-M. (2016). Partial preference models using translated cones in discrete multi-objective optimization. In *2016 IEEE Symposium Series on Computational Intelligence, SSCI 2016*, pages 1–8. Institute of Electrical and Electronics Engineers Inc. Type: Conference paper.

Kadziński, M. and Słowiński, R. (2012). Interactive robust cone contraction method for multiple objective optimization problems. *International Journal of Information Technology and Decision Making*, 11(2):327–357.

Kadziński, M., Tomczyk, M., and Słowiński, R. (2020). Preference-based cone contraction algorithms for interactive evolutionary multiple objective optimization. *Swarm and Evolutionary Computation*, 52. Publisher: Elsevier B.V.

Kania, A., Sipilä, J., Misitano, G., Miettinen, K., and Lehtimäki, J. (2022). Integration of lot sizing and safety strategy placement using interactive multiobjective optimization. *Computers & Industrial Engineering*, 173:108731.

Karahan, I. and Köksalan, M. (2010). A territory defining multiobjective evolutionary algorithms and preference incorporation. *IEEE Transactions on Evolutionary Computation*, 14(4):636–664.

Keeney, R. L. and Raiffa, H. (1993). *Decisions with Multiple Objectives: Preferences and Value Trade-Offs*. Cambridge University Press.

Kharrat, A., Chabchoub, H., and Aouni, B. (2010). Decision-maker's preferences modelling within the interactive imprecise goal programming model. *International Journal of Innovative Computing and Applications*, 2(3):150 – 169. Publisher: Inderscience Publishers Type: Article.

Kiriş, S. and Ustun, O. (2012). An integrated approach for stock evaluation and portfolio optimization. *Optimization*, 61(4):423–441.

Kitchenham, B. (2007). *Kitchenham, B.: Guidelines for performing Systematic Literature Reviews in software engineering. EBSE Technical Report EBSE-2007-01*. Unknown.

Klamroth, K. and Miettinen, K. (2008). Integrating Approximation and Interactive Decision Making in Multicriteria Optimization. *Operations Research*, 56(1):222–234. Publisher: INFORMS.

Krettek, J., Braun, J., Hoffmann, F., Bertram, T., Ewald, T., Schubert, H.-G., and Lausch, H. (2009). Interactive evolutionary multiobjective optimization for hydraulic valve controller parameters. In *IEEE/ASME*

International Conference on Advanced Intelligent Mechatronics, AIM, pages 816–821, Singapore.

Köksalan, M. and Karahan, I. (2010). An Interactive Territory Defining Evolutionary Algorithm: iTDEA. *IEEE Transactions on Evolutionary Computation*, 14(5):702–722. Conference Name: IEEE Transactions on Evolutionary Computation.

Le Huédé, F., Grabisch, M., Labreuche, C., and Savéant, P. (2006). Integration and propagation of a multi-criteria decision making model in constraint programming. *Journal of Heuristics*, 12(4-5):329–346.

Li, K., Chen, R., Min, G., and Yao, X. (2018a). Integration of preferences in decomposition multiobjective optimization. *IEEE Transactions on Cybernetics*, 48(12):3359–3370. Publisher: Institute of Electrical and Electronics Engineers Inc.

Li, K., Chen, R., Savic, D., and Yao, X. (2019). Interactive decomposition multiobjective optimization via progressively learned value functions. *IEEE Transactions on Fuzzy Systems*, 27(5):849–860. Publisher: Institute of Electrical and Electronics Engineers Inc.

Li, K., Lai, G., and Yao, X. (2023). Interactive Evolutionary Multiobjective Optimization via Learning to Rank. *IEEE Transactions on Evolutionary Computation*, 27(4):749–763. Publisher: Institute of Electrical and Electronics Engineers Inc. Type: Article.

Li, L., Chen, H., Li, J., Jing, N., and Emmerich, M. (2018b). Preference-Based Evolutionary Many-Objective Optimization for Agile Satellite Mission Planning. *IEEE Access*, 6:40963–40978. Conference Name: IEEE Access.

Li, L., Yao, F., Jing, N., and Emmerich, M. (2017). Preference incorporation to solve multi-objective mission planning of agile earth observation satellites. In *2017 IEEE Congress on Evolutionary Computation, CEC 2017 - Proceedings*, pages 1366–1373. Institute of Electrical and Electronics Engineers Inc. Type: Conference paper.

Li, S. and Hu, C. (2009). Satisfying optimization method based on goal programming for fuzzy multiple objective optimization problem. *European Journal of Operational Research*, 197(2):675–684.

Li, Z. and Liu, H.-L. (2015). Integrating preferred weights with decomposition based multi-objective evolutionary algorithm. In *Proceedings - 2014 10th International Conference on Computational Intelligence and Security, CIS 2014*, pages 58–63. Institute of Electrical and Electronics Engineers Inc.

Liemar (2024). LiemarX ERP-software.

Lin, C.-C. (2004). A weighted max-min model for fuzzy goal programming. *Fuzzy Sets and Systems*, 142(3):407–420.

Liu, R., Wang, R., Feng, W., Huang, J., and Jiao, L. (2016). Interactive Reference Region Based Multi-Objective Evolutionary Algorithm Through Decomposition. *IEEE Access*, 4:7331–7346. Publisher: Institute of Electrical and Electronics Engineers Inc.

Liu, R., Zhou, R., Ren, R., Liu, J., and Jiao, L. (2020). Multi-layer interaction preference based multi-objective evolutionary algorithm through decomposition. *Information Sciences*, 509:420–436. Publisher: Elsevier Inc.

Lu, J., Wu, F., and Zhang, G. (2007). On a generalized fuzzy goal optimization for solving fuzzy multi-objective linear programming problems. *Journal of Intelligent and Fuzzy Systems*, 18(1):83–97.

Luque, M., Yang, J.-B., and Wong, B. (2009). PROJECT method for multiobjective optimization based on gradient projection and reference points. *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, 39(4):864–879.

López-Jaimes, A. and Coello Coello, C. (2014). Including preferences into a multiobjective evolutionary algorithm to deal with many-objective engineering optimization problems. *Information Sciences*, 277:1–20. Publisher: Elsevier Inc.

Ma, X., Liu, F., Qi, Y., Li, L., Jiao, L., Deng, X., Wang, X., Dong, B., Hou, Z., Zhang, Y., and Wu, J. (2015). MOEA/D with biased weight adjustment inspired by user preference and its application on multi-objective reservoir flood control problem. *Soft Comput.*, pages 1–25.

Macias-Escobar, T., Cruz-Reyes, L., Fraire, H., and Dorronsoro, B. (2020). Plane Separation: A method to solve dynamic multi-objective optimization problems with incorporated preferences. *Future Generation Computer Systems*, 110:864–875. Publisher: Elsevier B.V.

Mellouli, A., Mellouli, R., and Masmoudi, F. (2019). An Innovative Genetic Algorithm for a Multi-Objective Optimization of Two-Dimensional Cutting-Stock Problem. *Applied Artificial Intelligence*, 33(6):531–547. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/08839514.2019.1583857.

Misitano, G. (2023). Exploring the Explainable Aspects and Performance of a Learnable Evolutionary Multiobjective Optimization Method. *ACM Trans. Evol. Learn. Optim.* Place: New York, NY, USA Publisher: Association for Computing Machinery.

Mkaouer, M. W., Kessentini, M., Bechikh, S., and Tauritz, D. R. (2013). Preference-based multi-objective software modelling. In *2013 1st International Workshop on Combining Modelling and Search-Based Software Engineering (CMSBSE)*, pages 61–66.

Mohammadi, A., Omidvar, M. N., Li, X., and Deb, K. (2014). Integrating user preferences and decomposition methods for many-objective optimization. In *Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014*, pages 421–428. Institute of Electrical and Electronics Engineers Inc. ISSN: 1941-0026.

Molina, J., Santana, L. V., Hernández-Díaz, A. G., Coello Coello, C. A., and Caballero, R. (2009). g-dominance: Reference point based dominance for multiobjective metaheuristics. *European Journal of Operational Research*, 197(2):685–692.

Mukhlisullina, D., Passerini, A., and Battiti, R. (2013). Learning to diversify in complex interactive multiobjective optimization. *Proceedings of the 10th Metaheuristics International Conference (MIC 2013)*, pages 230–239.

Ogryczak, W. (2008). WOWA enhancement of the preference modeling in the reference point method. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5285 LNAI:38–49. ISBN: 3540882685; 9783540882688 Place: Sabadell Publisher: Springer Verlag.

Ogryczak, W. and Śliwiński, T. (2009). On efficient WOWA optimization for decision support under risk. *International Journal of Approximate Reasoning*, 50(6):915–928.

Oliveira, J. F. C. and Ferreira, J. A. S. (1993). Algorithms for Nesting Problems. In Vidal, R. V. V., editor, *Applied Simulated Annealing*, Lecture Notes in Economics and Mathematical Systems, pages 255–273. Springer, Berlin, Heidelberg.

Ozbey, O. and Karwan, M. (2014). An Interactive Approach for Multicriteria Decision Making Using a Tchebycheff Utility Function Approximation. *Journal of Multi-Criteria Decision Analysis*, 21(3-4):153–172. Publisher: John Wiley and Sons Ltd.

Park, K.-S. and Koh, H.-M. (2004). Preference-based optimum design of an integrated structural control system using genetic algorithms. *Advances in Engineering Software*, 35(2):85–94. Publisher: Elsevier Ltd.

Parreiras, R. and Vasconcelos, J. (2007). A multiplicative version of Promethee II applied to multiobjective optimization problems. *European Journal of Operational Research*, 183(2):729–740.

Parreiras, R. and Vasconcelos, J. (2009). Decision making in multiobjective optimization aided by the multicriteria tournament decision method. *Nonlinear Analysis, Theory, Methods and Applications*, 71(12):e191–e198.

Pedro, L. and Takahashi, R. (2013). Decision-maker preference modeling in interactive multiobjective optimization. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7811 LNCS:811–824. ISBN: 9783642371394 Place: Sheffield.

Pedro, L. R. and Takahashi, R. H. (2014). INSPM: An interactive evolutionary multi-objective algorithm with preference model. *Information Sciences*, 268:202–219. Publisher: Elsevier Inc. Type: Article.

Perera, A., Attalage, R., Perera, K., and Dassanayake, V. (2013). A hybrid tool to combine multi-objective optimization and multi-criterion decision making in designing standalone hybrid energy systems. *Applied Energy*, 107:412–425. Publisher: Elsevier Ltd.

Pimentel, J. and Castro, J. (2018). piStar Tool – A Pluggable Online Tool for Goal Modeling. In *2018 IEEE 26th International Requirements Engineering Conference (RE)*, pages 498–499, Banff, AB. IEEE.

Qi, Y., Li, X., Yu, J., and Miao, Q. (2019). User-preference based decomposition in MOEA/D without using an ideal point. *Swarm and Evolutionary Computation*, 44:597–611. Publisher: Elsevier B.V. Type: Article.

Rachmawati, L. and Srinivasan, D. (2010). Incorporating the notion of relative importance of objectives in evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 14(4):530–546.

Ramírez, A., Romero, J., and Ventura, S. (2018). Interactive multi-objective evolutionary optimization of software architectures. *Information Sciences*, 463-464:92–109. Publisher: Elsevier Inc.

Riquelme, N., Von Lücken, C., and Baran, B. (2015). Performance metrics in multi-objective optimization. In *2015 Latin American Computing Conference (CLEI)*, pages 1–11.

Rivera, G., Coello Coello, C., Cruz-Reyes, L., Fernandez, E., Gomez-Santillan, C., and Rangel-Valdez, N. (2022a). Preference incorporation into many-objective optimization: An Ant colony algorithm based on interval outranking. *Swarm and Evolutionary Computation*, 69. Publisher: Elsevier B.V.

Rivera, G., Cruz-Reyes, L., Fernandez, E., Gomez-Santillan, C., and Rangel-Valdez, N. (2023a). An interactive ACO enriched with an eclectic multi-criteria ordinal classifier to address many-objective optimisation problems. *Expert Systems with Applications*, 232:120813. Publisher: Elsevier Ltd Type: Article.

Rivera, G., Cruz-Reyes, L., Fernandez, E., Gomez-Santillan, C., Rangel-Valdez, N., and Coello Coello, C. A.

(2023b). An ACO-based Hyper-heuristic for Sequencing Many-objective Evolutionary Algorithms that Consider Different Ways to Incorporate the DM's Preferences. *Swarm and Evolutionary Computation*, 76:101211. Publisher: Elsevier B.V. Type: Article.

Rivera, G., Florencia, R., Guerrero, M., Porras, R., and Sánchez-Solís, J. (2021). Online multi-criteria portfolio analysis through compromise programming models built on the underlying principles of fuzzy outranking. *Information Sciences*, 580:734–755. Publisher: Elsevier Inc.

Rivera, G., Porras, R., Sanchez-Solis, J., Florencia, R., and García, V. (2022b). Outranking-based multi-objective PSO for scheduling unrelated parallel machines with a freight industry-oriented application. *Engineering Applications of Artificial Intelligence*, 108. Publisher: Elsevier Ltd.

Rosen, S. L., Harmonosky, C. M., and Traband, M. T. (2007). A simulation optimization method that considers uncertainty and multiple performance measures. *European Journal of Operational Research*, 181(1):315–330. Type: Article.

Rudolph, G., Schütze, O., Grimme, C., and Trautmann, H. (2014). A Multiobjective Evolutionary Algorithm Guided by Averaged Hausdorff Distance to Aspiration Sets. In Tantar, A.-A., Tantar, E., Sun, J.-Q., Zhang, W., Ding, Q., Schütze, O., Emmerich, M., Legrand, P., Del Moral, P., and Coello Coello, C. A., editors, *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation V*, Advances in Intelligent Systems and Computing, pages 261–273, Cham. Springer International Publishing.

Ruiz, A., Saborido, R., Bermúdez, J., Luque, M., and Vercher, E. (2020). Preference-based evolutionary multi-objective optimization for portfolio selection: a new credibilistic model under investor preferences. *Journal of Global Optimization*, 76(2):295–315. Publisher: Springer.

Ruiz, A. B., Saborido, R., and Luque, M. (2015). A preference-based evolutionary algorithm for multiobjective optimization: the weighting achievement scalarizing function genetic algorithm. *Journal of Global Optimization*, 62(1):101–129. Publisher: Kluwer Academic Publishers.

Saaty, T. L. (1990). How to make a decision: The analytic hierarchy process. *European Journal of Operational Research*, 48(1):9–26.

Saborido, R., Ruiz, A., Luque, M., and Miettinen, K. (2019). IRA-EMO: Interactive method using reservation and aspiration levels for evolutionary multiobjective optimization. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11411 LNCS:618–630. ISBN: 9783030125974 Publisher: Springer Verlag.

Saldanha, W., Ponce Arrieta, F., Machado-Coelho, T., Santos, E., Maia, C., Ekel, P., and Soares, G. (2020). Evolutionary algorithms and the Preference Ranking Organization Method for Enrichment Evaluations as applied to a multiobjective design of shell-and-tube heat exchangers. *Case Studies in Thermal Engineering*, 17. Publisher: Elsevier Ltd.

Sanchis, J., Martínez, M., and Blasco, X. (2008). Integrated multiobjective optimization and a priori preferences using genetic algorithms. *Information Sciences*, 178(4):931–951.

Selim, H. and Ozkarahan, I. (2008). A supply chain distribution network design model: An interactive fuzzy goal programming-based solution approach. *International Journal of Advanced Manufacturing Technology*, 36(3-4):401–418.

Shen, X., Guo, Y., Chen, Q., and Hu, W. (2010). A multi-objective optimization evolutionary algorithm incorporating preference information based on fuzzy logic. *Computational Optimization and Applications*, 46(1):159–188.

Shukla, P., Braun, M., and Schmeck, H. (2013). Theory and algorithms for finding knees. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7811 LNCS:156–170. ISBN: 9783642371394 Place: Sheffield.

Simon, D. (2013). *Evolutionary Optimization Algorithms*. John Wiley & Sons. Google-Books-ID: gwUwIEPqk30C.

Sindhya, K., Ruiz, A., and Miettinen, K. (2011). A preference based interactive evolutionary algorithm for multi-objective optimization: PIE. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6576 LNCS:212–225. ISBN: 9783642198922 Place: Ouro Preto.

Sinha, A., Korhonen, P., Wallenius, J., and Deb, K. (2014). An interactive evolutionary multi-objective optimization algorithm with a limited number of decision maker calls. *European Journal of Operational Research*, 233(3):674–688.

Sinha, A., Malo, P., and Kallio, M. (2018). Convex preference cone-based approach for many objective optimization problems. *Computers and Operations Research*, 95:1–11. Publisher: Elsevier Ltd.

Siskos, Y. and Yannacopoulos, D. (1985). UTASTAR: An ordinal regression method for building additive value functions. *Investigaçao Operacional*, 5(1):39–53.

Solares, E., Coello Coello, C. A., Fernandez, E., and Navarro, J. (2019). Handling uncertainty through confidence intervals in portfolio optimization. *Swarm and Evolutionary Computation*, 44:774 – 787. Publisher: Elsevier B.V. Type: Article.

STRUMIS (2024). Steel Fabrication Software.

Sudeng, S. and Wattanapongsakorn, N. (2015a). Interactive Preference Incorporation in Evolutionary Multi-objective Engineering Design. In *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*, volume 2016-January, pages 1005–1012. IEEE Computer Society. ISSN: 10823409 Type: Conference paper.

Sudeng, S. and Wattanapongsakorn, N. (2015b). Post Pareto-optimal pruning algorithm for multiple objective optimization using specific extended angle dominance. *Engineering Applications of Artificial Intelligence*, 38:221–236. Publisher: Elsevier Ltd.

Sun, J., Gong, D., and Sun, X. (2011). Solving Interval Multi-Objective Optimization Problems Using Evolutionary Algorithms with Preference Polyhedron. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, GECCO '11, pages 729–736, New York, NY, USA. Association for Computing Machinery. event-place: Dublin, Ireland.

Tamssaouet, K., Dauzère-Pérès, S., Knopp, S., Bitar, A., and Yugma, C. (2022). Multiobjective optimization for complex flexible job-shop scheduling problems. *European Journal of Operational Research*, 296(1):87 – 100. Publisher: Elsevier B.V. Type: Article.

Tang, H., Liu, X., Zheng, J., and Chen, T. (2021). A Preference-based Multiobjective Evolutionary Algorithm Based on Weight Vector Adjustment Strategy. In *Proceedings - 2021 6th International Conference on Computational Intelligence and Applications, ICCIA 2021*, pages 53–58. Institute of Electrical and Electronics Engineers Inc. Type: Conference paper.

Taylor, K., Ha, H., Li, M., Chan, J., and Li, X. (2021). Bayesian Preference Learning for Interactive Multi-Objective Optimisation. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '21, pages 466–475, New York, NY, USA. Association for Computing Machinery. event-place: Lille, France.

Tekla (2024). Tekla PowerFab - Steel Fabrication Management Software.

Thiele, L., Miettinen, K., Korhonen, P. J., and Molina, J. (2009). A Preference-Based Evolutionary Algorithm for Multi-Objective Optimization. *Evolutionary Computation*, 17(3):411–436. Conference Name: Evolutionary Computation.

Tiwari, S., Wiecek, M., and Fadel, G. (2008). Inclusion of preferences in the design of evolutionary optimization algorithms: An empirical study. In *12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, MAO*. American Institute of Aeronautics and Astronautics Inc. Type: Conference paper.

Tomczyk, M. K. and Kadziński, M. (2019a). EMOSOR: Evolutionary multiple objective optimization guided by interactive stochastic ordinal regression. *Computers and Operations Research*, 108:134–154. Publisher: Elsevier Ltd Type: Article.

Tomczyk, M. K. and Kadziński, M. (2019b). Robust Indicator-Based Algorithm for Interactive Evolutionary Multiple Objective Optimization. In *GECCO 2019 - Proceedings of the 2019 Genetic and Evolutionary Computation Conference*, GECCO '19, pages 629–637, New York, NY, USA. Association for Computing Machinery, Inc. event-place: Prague, Czech Republic.

Tomczyk, M. K. and Kadziński, M. (2020). On the Elicitation of Indirect Preferences in Interactive Evolutionary Multiple Objective Optimization. In *GECCO 2020 - Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, GECCO '20, pages 569–577, New York, NY, USA. Association for Computing Machinery. event-place: Cancún, Mexico.

Tomczyk, M. K. and Kadziński, M. (2021). Decomposition-based co-evolutionary algorithm for interactive multiple objective optimization. *Information Sciences*, 549:178 – 199. Publisher: Elsevier Inc. Type: Article.

Torra, V. (1997). The weighted OWA operator. *International Journal of Intelligent Systems*, 12(2):153–166. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/%28SICI%291098-111X%28199702%2912%3A2%3C153%3A%3AAID-INT3%3E3.0.CO%3B2-P.

Trachanatzi, D., Rigakis, M., Marinaki, M., and Marinakis, Y. (2020). An interactive preference-guided firefly algorithm for personalized tourist itineraries. *Expert Systems with Applications*, 159. Publisher: Elsevier Ltd Type: Article.

Trautmann, H. and Mehnen, J. (2009). Preference-based Pareto optimization in certain and noisy environments. *Engineering Optimization*, 41(1):23–38.

Tyagi, S. and Verma, A. (2017). Interactive adaptive particle swarm optimization for optimal global supply chain design. *International Journal of Integrated Supply Management*, 11(1):1 – 23. Publisher: Inderscience Publishers Type: Article.

Tyagi, S., Yang, K., Tyagi, A., and Dwivedi, S. (2011). Development of a fuzzy goal programming model for optimization of lead time and cost in an overlapped product development project using a Gaussian Adaptive Particle Swarm Optimization-based approach. *Engineering Applications of Artificial Intelligence*, 24(5):866–879.

Vallerio, M., Hufkens, J., Van Impe, J., and Logist, F. (2015). An interactive decision-support system for multi-objective optimization of nonlinear dynamic processes with uncertainty. *Expert Systems with Applications*, 42(21):7710–7731. Publisher: Elsevier Ltd.

Vesikar, Y., Deb, K., and Blank, J. (2018). Reference Point Based NSGA-III for Preferred Solutions. In S, S., editor, *Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence, SSCI 2018*, pages 1587–1594. Institute of Electrical and Electronics Engineers Inc.

Voortman (2024). Software | Nesting.

Wagner, T. and Trautmann, H. (2010). Integration of preferences in hypervolume-based multiobjective evolutionary algorithms by means of desirability functions. *IEEE Transactions on Evolutionary Computation*, 14(5):688–701.

Wang, G., Huang, S., and Dismukes, J. (2004). Product-driven supply chain selection using integrated multi-criteria decision-making methodology. *International Journal of Production Economics*, 91(1):1–15.

Wang, R., Purshouse, R., Giagkiozis, I., and Fleming, P. (2015). The iPICEA-g: A new hybrid evolutionary multi-criteria decision making approach using the brushing technique. *European Journal of Operational Research*, 243(2):442–453. Publisher: Elsevier B.V.

Wang, R.-C. and Liang, T.-F. (2004). Project management decisions with multiple fuzzy goals. *Construction Management and Economics*, 22(10):1047–1056.

Wang, R.-C. and Liang, T.-F. (2005). Aggregate production planning with multiple fuzzy goals. *International Journal of Advanced Manufacturing Technology*, 25(5-6):589–597.

Wickramasinghe, U. and Li, X. (2009). Using a distance metric to guide PSO algorithms for many-objective optimization. In *Proceedings of the 11th Annual Genetic and Evolutionary Computation Conference, GECCO-2009*, pages 667–674, Montreal, QC.

Wickramasinghe, U. K., Carrese, R., and Li, X. (2010). Designing airfoils using a reference point based evolutionary many-objective particle swarm optimization algorithm. In *IEEE Congress on Evolutionary Computation*, pages 1–8. ISSN: 1941-0026.

Wickramasinghe, W. and Li, X. (2008). Integrating user preferences with particle swarms for multi-objective optimization. In *GECCO'08: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation 2008*, pages 745–752, Atlanta, GA. Association for Computing Machinery (ACM).

Wiecek, M., Blouin, V., Fadel, G., Engau, A., Hunt, B., and Singh, V. (2009). Multi-scenario multi-objective optimization with applications in engineering design. *Lecture Notes in Economics and Mathematical Systems*, 618:283–298. ISBN: 9783540856450.

Wohlin, C. (2014). Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, EASE '14, pages 1–10, New York, NY, USA. Association for Computing Machinery.

Worldsteel (2023a). Climate change policy paper.

Worldsteel (2023b). Total production of crude steel.

Wäscher, G., Haußner, H., and Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research*, 183(3):1109–1130.

Yan, J., Li, C., Wang, Z., Deng, L., and Sun, D. (2007). Diversity Metrics in Multi-objective Optimization: Review and Perspective. In *2007 IEEE International Conference on Integration Technology*, pages 553–557.

Yao, S., Jiang, Z., Li, N., Zhang, H., and Geng, N. (2011). A multi-objective dynamic scheduling approach using multiple attribute decision making in semiconductor manufacturing. *International Journal of Production Economics*, 130(1):125–133.

Yi, J., Bai, J., He, H., Peng, J., and Tang, D. (2019). ar-MOEA: A Novel Preference-Based Dominance Relation for Evolutionary Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation*, 23(5):788–802. Conference Name: IEEE Transactions on Evolutionary Computation.

Yu, E. (1997). Towards modelling and reasoning support for early-phase requirements engineering. In *Proceedings of ISRE '97: 3rd IEEE International Symposium on Requirements Engineering*, pages 226–235.

Yu, G., Zheng, J., Shen, R., and Li, M. (2016). Decomposing the user-preference in multiobjective optimization. *Soft Computing*, 20(10):4005–4021.

Zhang, Q. and Li, H. (2007). MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731. Conference Name: IEEE Transactions on Evolutionary Computation.

Zhao, X., Shen, L. W., Peng, X., and Zhao, W. (2013). Finding Preferred Skyline Solutions for SLA-Constrained Service Composition. In *Proceedings - IEEE 20th International Conference on Web Services, ICWS 2013*, pages 195–202. IEEE Computer Society. Type: Conference paper.

Zheng, J., Yu, G., Zhu, Q., Li, X., and Zou, J. (2017). On decomposition methods in interactive user-preference based optimization. *Applied Soft Computing*, 52:952–973.

Zhou-Kangas, Y. and Miettinen, K. (2019). Decision making in multiobjective optimization problems under uncertainty: balancing between robustness and quality. *OR Spectrum*, 41(2):391–413. Publisher: Springer Verlag.

Zitzler, E., Deb, K., and Thiele, L. (2000). Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195.

Zitzler, E. and Künzli, S. (2004). Indicator-Based Selection in Multiobjective Search. In Yao, X., Burke, E. K., Lozano, J. A., Smith, J., Merelo-Guervós, J. J., Bullinaria, J. A., Rowe, J. E., Tiňo, P., Kabán, A., and Schwefel, H.-P., editors, *Parallel Problem Solving from Nature - PPSN VIII*, Lecture Notes in Computer Science, pages 832–842, Berlin, Heidelberg. Springer.

A TABLE OF STUDIES

We performed our literature review on 170 studies published between 2004 and 2023. In Section 3, we describe the review process and the analysis of these works. To make the review process transparent, we present table 12, which lists the studies that we analysed for our review of the literature. For each of the studies, we include the maximum number of objectives discussed, the preference timing used, the method for encoding preference, the search type, and the test set(s) used during validation.

Table 12: Some of the collected data points for each study in the review.

Citation	# of Obj	Preference timing	Preference encoding	Search type	Test set
Abd El-Wahed and Lee (2006)	2	Interactive	Goal points	Exact	Numerical example
Abdolshah et al. (2019)	3	A priori	Relations	Bayesian search	Schaffer1, MNIST
Abouhawwash and Deb (2021)	10	A priori	Reference locations	Evolutionary algorithm	ZDT, SRN, DTLZ
Aggelogiannaki and Sarimveis (2007)	3	A priori	Relations	Meta-heuristic	State of the art problem sets
Astudillo and Frazier (2020)	5	A priori	Relations	Bayesian search	DTLZ, Numerical example
Augeri et al. (2021)	4	Interactive	Relations	Exact	Numerical example
Balderas et al. (2019)	9	A priori	Relations	Genetic algorithm	Numerical example
Battiti and Passerini (2010)	10	Interactive	Utilities	Evolutionary algorithm	DTLZ
Baykasoğlu (2005)	2	A priori	Goal points	Meta-heuristic	State of the art problem sets
Bemporad and Piga (2021)	6	Interactive	Relations	Meta-heuristic	State of the art problem sets
Ben Said et al. (2010)	10	Interactive	Reference locations	Evolutionary algorithm	Fonseca, DTLZ, ZDT
Benabbou et al. (2020)	7	Interactive	Functions	Genetic algorithm	MTSP
Bezoui et al. (2023)	2	A priori	Reference locations	Genetic algorithm	Numerical example
Borges et al. (2014)	3	A posteriori	Relations	Exact	Case study
Brafman and Chernyavsky (2005)	3	A priori	Goal points	Exact	Numerical example
Branke and Deb (2005)	3	A priori	Reference locations	Evolutionary algorithm	ZDT, DTLZ
Branke et al. (2010)	2	Interactive	Functions	Evolutionary algorithm	ZDT
Branke et al. (2015)	5	Interactive	Functions	Evolutionary algorithm	ZDT, DTLZ, WFG
Branke et al. (2016)	5	Interactive	Utilities	Evolutionary algorithm	ZDT, DTLZ

Continued on next page

Table 12: Some of the collected data points for each study in the review. (Continued)

Braun et al. (2017b)	3	A priori	Utilities	Genetic algorithm	ZDT, DEB2DK, DO2DK
Braun et al. (2017a)	3	A priori	Functions	Evolutionary algorithm	ZDT, DTLZ, WFG, DEB2DK, DEB3DK, DO2DK
Braun et al. (2015)	3	A priori	Functions	Evolutionary algorithm	DEB2DK, DO2DK, DTLZ, ZDT, Lamé
Braun et al. (2011)	3	A priori	Relations	Evolutionary algorithm	CTP, DEB2DK, DEB3DK, DO2DK, DTLZ, LZ09, SZDT, UF, WFG, ZDT
Brockhoff et al. (2013)	7	A priori	Reference locations	Evolutionary algorithm	ZDT
Castellanos-Alvarez et al. (2021)	3	A priori	Relations	Genetic algorithm	DTLZ
Castellanos et al. (2022)	10	A priori	Relations	Meta-heuristic	DTLZ
Chen et al. (2017)	5	Interactive	Relations	Evolutionary algorithm	GLT, DTLZ
Cheng and Jia (2021)	2	Interactive	Decision trees	Evolutionary algorithm	Numerical example
Cheng et al. (2016)	10	A priori	Reference locations	Evolutionary algorithm	DTLZ, SDTLZ, WFG
Chica et al. (2015)	2	Interactive	Reference locations	Meta-heuristic	Numerical example
Chugh et al. (2015)	3	Interactive	Weights	Evolutionary algorithm	DTLZ, ZDT
Cordone et al. (2007)	6	A priori	Weights	Heuristic, meta-heuristic, exact	Case study
Cruz-Reyes et al. (2020)	8	Interactive	Relations	Evolutionary algorithm	DTLZ
Cruz-Reyes et al. (2017)	12	A priori	Relations	Hyper-heuristic	Case study
Dasdemir et al. (2020)	10	Interactive	Reference locations	Evolutionary algorithm	ZDT, DTLZ, welded beam, spring design
Deb and Kumar (2007a)	10	Interactive	Reference locations	Evolutionary algorithm	ZDT, DTLZ, Korhonen and Laakso, Side impact
Deb and Kumar (2007b)	10	Interactive	Reference locations	Evolutionary algorithm	ZDT, KUR, OSY, DTLZ, Welded beam, Spring design, Side impact
Deb and Sundar (2006)	10	A priori	Reference locations	Evolutionary algorithm	ZDT, DTLZ, Welded beam
Dias et al. (2008)	2	Interactive	Reference locations	Genetic algorithm	Generated problem sets

Continued on next page

Table 12: Some of the collected data points for each study in the review. (Continued)

Ehrgott et al. (2004)	7	A priori	Utilities	Exact, meta-heuristic, genetic algorithm	Numerical example
Ertuğul and Güeş (2007)	2	A priori	Goal points	Exact	Numerical example
Feliot et al. (2019)	2	A priori	Reference locations	Bayesian search	BNH
Fernandez et al. (2013)	3	A priori	Relations	Genetic algorithm	Numerical example
Fernandez et al. (2015)	9	A priori	Relations	Meta-heuristic	Case study
Fernandez et al. (2009)	4	A priori	Relations	Evolutionary algorithm	Numerical example
Fernandez et al. (2019)	4	A priori	Relations	Meta-heuristic	Numerical example
Fernández et al. (2022)	10	A priori	Relations	Evolutionary algorithm	DTLZ
Filho et al. (2018)	4	A priori	Reference locations	Hyper-heuristic	James, CAS, WS, E-Shop
Fliedner and Liesiö (2016)	3	A priori	Weights	Exact	Numerical example
Fouchal et al. (2011)	5	A priori	Utilities	Exact	Numerical example
Fowler et al. (2010)	4	Interactive	Functions	Evolutionary algorithm	MO-Knapsack
Friedrich et al. (2013)	2	A priori	Weights	Evolutionary algorithm	ZDT, WFG
Galand et al. (2013)	7	A priori	Functions	Exact	Numerical example
Galand and Spanjaard (2007)	2	A priori	Utilities	Heuristic	Numerical example
Galand and Perny (2007)	3	A priori	Utilities	Heuristic	Numerical example
Gong et al. (2013)	5	Interactive	Functions	Evolutionary algorithm	ZDT, DTLZ
Gong et al. (2011)	3	Interactive	Relations	Evolutionary algorithm	ZDT, DTLZ, welded beam
Gong et al. (2017)	20	Interactive	Reference locations	Evolutionary algorithm	DTZL
Goulart and Campelo (2016)	20	A priori	Reference locations	Evolutionary algorithm	DTLZ
Greco et al. (2010)	2	Interactive	Relations	Evolutionary algorithm	Theoretical analysis
Guo et al. (2020)	3	Interactive	Reference locations	Evolutionary algorithm	Case study
Hakanen and Knowles (2017)	4	Interactive	Functions	Meta-heuristic	DTLZ

Continued on next page

Table 12: Some of the collected data points for each study in the review. (Continued)

He et al. (2020)	4	A priori	Reference locations	Evolutionary algorithm	LZ08, ZDT, DTLZ
He et al. (2021)	4	Interactive	Weights	Meta-heuristic	Tire tread compound
Hu and Li (2006)	5	A priori	Goal points	Exact	Numerical example
Hu et al. (2007)	3	A priori	Goal points	Genetic algorithm	Numerical example
Hu et al. (2017)	15	A priori	Reference locations	Evolutionary algorithm	ZDT, DTLZ
Hu et al. (2021)	2	Interactive	Decision trees	Evolutionary algorithm	Numerical example
Huang et al. (2005)	3	Interactive	Reference locations	Meta-heuristic	Numerical example
Hunt et al. (2004)	2	A priori	Weights	Exact	Theoretical analysis
Iniestra and Gutiérrez (2009)	5	A posteriori	Relations	Evolutionary algorithm	Numerical example
Jaimes et al. (2011)	6	Interactive	Relations	Genetic algorithm	Numerical example
Jia et al. (2013)	2	A posteriori	Reference locations	Meta-heuristic	Case study
Junker (2004)	3	A priori	Relations	Exact	Theoretical analysis
Kaddani et al. (2016)	3	A priori	Functions	Genetic algorithm	Numerical example
Kadziński and Słowiński (2012)	4	Interactive	Relations	Exact	Numerical example
Kadziński et al. (2020)	5	Interactive	Functions	Evolutionary algorithm	DTLZ, WFG
Kania et al. (2022)	4	Interactive	Relations	Heuristic	Case study
Karahan and Köksalan (2010)	5	A priori	Weights	Evolutionary algorithm	ZDT, DTLZ
Kharrat et al. (2010)	5	Interactive	Goal points	Meta-heuristic	Numerical example
Kiriş and Ustun (2012)	3	Interactive	Weights	Heuristic	Numerical example
Klamroth and Miettinen (2008)	5	Interactive	Reference locations	Exact	Numerical example
Köksalan and Karahan (2010)	3	Interactive	Weights	Evolutionary algorithm	ZDT, DTLZ
Krettek et al. (2009)	2	Interactive	Relations	Evolutionary algorithm	Kursawe
Le Huédé et al. (2006)	3	A priori	Utilities	Exact	Numerical example
Li and Liu (2015)	2	A priori	Reference locations	Evolutionary algorithm	ZDT
Li et al. (2018a)	10	Interactive	Reference locations	Evolutionary algorithm	DTLZ, WFG
Li et al. (2019)	10	Interactive	Functions	Evolutionary algorithm	DTLZ

Continued on next page

Table 12: Some of the collected data points for each study in the review. (Continued)

Li et al. (2023)	10	Interactive	Relations	Evolutionary algorithm	DTLZ, MDTLZ, WFG
Li et al. (2018b)	5	Interactive	Relations	Evolutionary algorithm	STK
Li et al. (2017)	3	A priori	Reference locations	Genetic algorithm	STK
Li and Hu (2009)	5	A priori	Goal points	Exact	Numerical example
Lin (2004)	2	A priori	Goal points	Exact	Numerical example
Liu et al. (2016)	3	Interactive	Reference locations	Evolutionary algorithm	ZDT, DTLZ
Liu et al. (2020)	10	Interactive	Relations	Evolutionary algorithm	ZDT, DTLZ
López-Jaimes and Coello Coello (2014)	6	Interactive	Reference locations	Evolutionary algorithm	Case study
Lu et al. (2007)	2	Interactive	Goal points	Exact	Numerical example
Luque et al. (2009)	2	Interactive	Reference locations	Local search	Numerical example
Ma et al. (2015)	10	A priori	Reference locations	Evolutionary algorithm	ZDT, DTLZ, UF
Macias-Escobar et al. (2020)	2	A priori	Reference locations	Hyper-heuristic, Genetic algorithm	FDA, dMOP
Misitano (2023)	3	Interactive	Weights	Evolutionary algorithm	Numerical example
Mkaouer et al. (2013)	3	A priori	Reference locations	Evolutionary algorithm	Case study
Mohammadi et al. (2014)	10	A priori	Reference locations	Evolutionary algorithm	DTLZ
Molina et al. (2009)	2	Interactive	Reference locations	Evolutionary algorithm	ZDT, deb32
Mukhlisullina et al. (2013)	3	Interactive	Reference locations	Evolutionary algorithm	UF
Ogryczak and Śliwiński (2009)	2	A priori	Weights	Exact	Numerical example
Ogryczak (2008)	3	A priori	Reference locations	Exact	Numerical example
Ozbey and Karwan (2014)	6	Interactive	Utilities	Exact	Generated problem sets
Park and Koh (2004)	4	A priori	Functions	Genetic algorithm	Numerical example
Parreiras and Vasconcelos (2009)	3	A posteriori	Relations	Evolutionary algorithm	TEAM benchmark problem 22
Parreiras and Vasconcelos (2007)	4	A posteriori	Relations	Evolutionary algorithm	Numerical example

Continued on next page

Table 12: Some of the collected data points for each study in the review. (Continued)

Pedro and Takahashi (2013)	3	Interactive	Utilities	Evolutionary algorithm	Numerical example
Pedro and Takahashi (2014)	3	Interactive	Relations	Evolutionary algorithm	Numerical example
Perera et al. (2013)	4	A posteriori	Weights	Evolutionary algorithm	Numerical example
Qi et al. (2019)	10	Interactive	Reference locations	Evolutionary algorithm	ZDT, DTLZ, WFG
Rachmawati and Srinivasan (2010)	6	Interactive	Relations	Evolutionary algorithm	ZDT, DTLZ
Ramírez et al. (2018)	3	Interactive	Relations	Evolutionary algorithm	Aqualush, Dataproj4, Java2HTML, JSapar, Marvin, NekoHTML
Rivera et al. (2023b)	10	A priori	Relations	Hyper-heuristic	DTLZ, WFG
Rivera et al. (2023a)	10	Interactive	Relations	Meta-heuristic	DLTZ, WFG
Rivera et al. (2021)	9	Interactive	Relations	Exact	Numerical example
Rivera et al. (2022b)	2	A priori	Relations	Meta-heuristic	Numerical example
Rivera et al. (2022a)	10	A priori	Relations	Meta-heuristic	DTLZ, WFG
Rosen et al. (2007)	4	A priori	Functions	Exact	Numerical example
Rudolph et al. (2014)	2	A priori	Reference locations	Evolutionary algorithm	SPHERE, DTLZ, DENT, ZDT
Ruiz et al. (2015)	3	A priori	Reference locations	Genetic algorithm	ZDT, DTLZ
Ruiz et al. (2020)	3	A priori	Reference locations	Evolutionary algorithm	CMASdL
Saborido et al. (2019)	3	Interactive	Goal points	Genetic algorithm	Numerical example
Saldanha et al. (2020)	2	A posteriori	Relations	Meta-heuristic	Case study
Sanchis et al. (2008)	2	A priori	Relations	Genetic algorithm	Numerical example
Selim and Ozkarahan (2008)	3	A priori	Goal points	Exact	Numerical example
Shen et al. (2010)	7	Interactive	Relations	Evolutionary algorithm	Theoretical analysis, numerical analysis
Shukla et al. (2013)	3	A priori	Functions	Evolutionary algorithm	DTLZ, ZDT, DEB2DK, DEB3DK
Sindhya et al. (2011)	5	Interactive	Weights	Evolutionary algorithm	Numerical example
Sinha et al. (2018)	5	Interactive	Functions	Evolutionary algorithm	DTLZ
Sinha et al. (2014)	5	Interactive	Functions	Evolutionary algorithm	MZDT, MDTLZ
Solares et al. (2019)	2	A priori	Intervals	Evolutionary algorithm	Numerical example

Continued on next page

Table 12: Some of the collected data points for each study in the review. (Continued)

Sudeng and Wattanapongsakorn (2015a)	3	Interactive	Reference locations	Evolutionary algorithm	Numerical example
Sudeng and Wattanapongsakorn (2015b)	3	Interactive	Reference locations	Evolutionary algorithm	ZDT, DTLZ, WFG
Sun et al. (2011)	2	Interactive	Functions	Evolutionary algorithm	ZDT
Tamssaouet et al. (2022)	4	A priori	Relations	Meta-heuristic	Numerical example
Tang et al. (2021)	10	Interactive	Reference locations	Evolutionary algorithm	ZDT, DTLZ
Taylor et al. (2021)	3	Interactive	Reference locations	Evolutionary algorithm	ZDT, DTLZ, RE
Thiele et al. (2009)	5	Interactive	Reference locations	Evolutionary algorithm	ZDT
Tiwari et al. (2008)	2	Interactive	Functions	Evolutionary algorithm	SCH, FON, KUR, POL, BNH, OSY, SRN, TNK, ZDT
Tomczyk and Kadziński (2021)	5	Interactive	Relations	Evolutionary algorithm	WFG
Tomczyk and Kadziński (2020)	5	Interactive	Relations	Evolutionary algorithm	WFG
Tomczyk and Kadziński (2019a)	5	Interactive	Weights	Evolutionary algorithm	WFG, DTLZ
Tomczyk and Kadziński (2019b)	5	Interactive	Relations	Evolutionary algorithm	WFG
Trachanatzi et al. (2020)	3	Interactive	Functions	Meta-heuristic	Benchmark set
Trautmann and Mehnen (2009)	2	A priori	Functions	Evolutionary algorithm	Numerical example
Tyagi et al. (2011)	3	A priori	Goal points	Meta-heuristic	Numerical example
Tyagi and Verma (2017)	2	Interactive	Weights	Meta-heuristic	Numerical example
Vallerio et al. (2015)	5	Interactive	Relations	Meta-heuristic	Case study
Vesikar et al. (2018)	5	A priori	Reference locations	Evolutionary algorithm	DTLZ, WFG, CRASH
Wagner and Trautmann (2010)	10	A priori	Functions	Evolutionary algorithm	ZDT
Wang and Liang (2005)	3	A priori	Goal points	Exact	Case study
Wang et al. (2015)	5	Interactive	Reference locations	Evolutionary algorithm	ZDT, DTLZ
Wang et al. (2004)	2	A priori	Goal points	Exact	Numerical example
Wang and Liang (2004)	3	A priori	Goal points	Exact	Numerical example

Continued on next page

Table 12: Some of the collected data points for each study in the review. (Continued)

Wickramasinghe et al. (2010)	6	A priori	Reference locations	Evolutionary algorithm	Numerical example
Wickramasinghe and Li (2008)	3	Interactive	Reference locations	Meta-heuristic	ZDT, DTLZ
Wickramasinghe and Li (2009)	10	A priori	Reference locations	Meta-heuristic	ZDT, DTLZ, WFG
Wiecek et al. (2009)	2	A priori	Functions	Local search	Numerical example
Yao et al. (2011)	4	A priori	Relations	Exact	Case study
Yi et al. (2019)	10	Interactive	Reference locations	Evolutionary algorithm	ZDT, DTLZ, WFG
…	…	…	…	…	…
Yu et al. (2016)	15	Interactive	Weights	Evolutionary algorithm	ZDT, DTLZ
Zhao et al. (2013)	7	A priori	Relations	Evolutionary algorithm	Numerical example
Zheng et al. (2017)	15	Interactive	Reference locations	Evolutionary algorithm	ZDT, DTLZ
Zhou-Kangas and Miettinen (2019)	4	Interactive	Functions	Exact	Numerical example
Zitzler and Künzli (2004)	4	A priori	Relations	Evolutionary algorithm	EXPO, ZDT, DTLZ, KUR

B SEMI-STRUCTURED INTERVIEW QUESTIONS

B.1 Interview Session 1 - Identifying Factors

- What are factors that you take into account when creating or evaluating a nesting?
- When evaluating the [cost/time/labour] effectiveness of a nesting, what are factors that you take into account?
- What are situations in which [costs/time/labour] would be important to take into account when creating a nesting? What aspect of the nesting or the situation makes it important to take [cost/time/labour] into account?
- In previous projects, what aspects of a nesting did you take into account and why?
- Has there been a project in the past where [costs/time/labour] influenced your decision making? If yes, how did it influence your decision making?
- Examples of cost factors that we have already come up with are material waste and the cost of storing inventory. What other factors are relevant to you?
- What factors do you think could be relevant to other steel manufacturers?

B.2 Interview Session 1 - Ranking Factors

- Given [factor], could you describe a situation where this factor would be influential to your nesting?
- Is there a situation where [factor 1] would be more important to your nesting than [factor 2]? If so, can you give an example? And in what case would it be the other way around?
- Given [important factor], have there been projects in the past where you considered this factor not important? If so, what situation and why?
- Given [unimportant factor], have there been projects in the past where you considered this factor important? If so, what situation and why?

B.3 Interview Session 2 - Evaluating Nestings

- Given [nesting 1] and [nesting 2], which nesting would you prefer and why?
- Is there a situation in which you would choose the other nesting? If so, what would that situation be?
- How would [aspect] have to change for you to choose the other nesting?
- How would you change [non-preferred nesting] to make it more suitable for your production plant?

C POC MODEL

C.1 Parameters

C.1.1 Sets

Set of parts: P
Set of beams: $B \ \ where \ \ |B| \leq |P|$
Set of possible beam lengths: A
Set of phases: F

C.1.2 Costs

Cost of a millimetre of steel: C_{steel}
Oversized load penalty: $C_{oversized}$

C.1.3 Thresholds

Discount threshold: $T_{discount}$
Short piece removal threshold: T_{sprs}
Oversized load threshold: $T_{oversized}$

C.1.4 Other Parameters

Length of a part: len_p
The discount factor for remnants: $d \ \ where \ \ 0 \leq d \leq 1$
Is part p in phase f: $phase_{pf} \in \{0,1\} \ \forall p \in P, \ \forall f \in F$

C.2 Decision Variables

Is part p nested into beam b:

$$x_{pb} \in \{0,1\} \ \forall p \in P, \ \forall b \in B \tag{7}$$

Length of beam b:

$$y_b \in A \ \forall b \in B \tag{8}$$

C.3 Auxiliaries

Beam b is in use:

$$z_b \in \{0,1\} \ \forall b \in B \tag{9}$$

Waste generated by bar b in millimetres:

$$W_b = z_b * (y_b - \sum_{p \in P} (len_p * x_{pb})) \tag{10}$$

An estimation of the number of pieces the SPRS cannot remove from a bar b:

$$S_b = \begin{cases} 2 & if \quad W_b < T_{sprs} \\ 1 & if \quad T_{sprs} \leq W_b < 2 * T_{sprs} \\ 0 & if \quad W_b \geq 2 * T_{sprs} \end{cases} \tag{11}$$

If we have enough waste to make a remnant, apply the discount:

$$D_b = \begin{cases} d & if \quad W_b \geq T_{discount} + T_{sprs} \\ 1 & otherwise \end{cases} \tag{12}$$

Does beam b contain phase f:

$$q_{bf} = \begin{cases} 1 & if \quad \sum_{p \in P}(x_{pb} * phase_{pf}) > 1 \\ 0 & otherwise \end{cases} \tag{13}$$

Check if a beam contains parts from multiple phases:

$$Q_b = \begin{cases} 1 & if \quad \sum_{f \in F}(q_{bf}) > 1 \\ 0 & otherwise \end{cases} \tag{14}$$

Extra delivery costs:

$$E = \begin{cases} C_{oversized} & if \quad \exists b \in B \;\; s.t. \;\; y_b * z_b > T_{oversized} \\ 0 & otherwise \end{cases} \tag{15}$$

C.4 Objectives

Minimize cost:

$$\min \; C_{steel} * \sum_{b \in B}(W_b * D_b) + E \tag{16}$$

Minimize beams with multiple phases:

$$\min \; \sum_{b \in B}(Q_b) \tag{17}$$

Minimize the estimated number of manual removals:

$$\min \; \sum_{b \in B}(S_b) \tag{18}$$

C.5 Constraints

All parts are nested exactly once:

$$\sum_{b \in B}(x_{pb}) = 1 \; \forall p \in P \tag{19}$$

All parts fit in the beam, the beam must be considered used:

$$\sum_{p \in P}(x_{pb} * len_p) \leq y_b * z_b \; \forall b \in B \tag{20}$$

Part can only belong to one phase:

$$\sum_{f \in F}(phase_{pf}) = 1 \; \forall p \in P \tag{21}$$

When a beam contains no parts, do not consider it used:

$$z_b \leq \sum_{p \in P}(x_{pb}) \; \forall b \in B \tag{22}$$

D GENOMES, RECOMBINATIONS, AND MUTATIONS

D.1 ZDT and DTLZ

D.1.1 Genome

For the ZDT (except ZDT5) and DTLZ tests, the genome is an array of real-valued numbers between 0 and 1:

$$[x_1, \cdots, x_i] \;\; s.t. \;\; \forall n \in \mathbb{N} \;\; where \;\; n \leq i, \;\; 0 \leq x_n \leq 1 \tag{23}$$

D.1.2 Recombination

To create a child solution, we perform a crossover between two parent solutions **x** and **y** (Mellouli et al. (2019)):

$$\mathbf{x} = [x_1, \cdots, x_k, x_{k+1}, \cdots, x_i] \;\; s.t. \;\; 1 \leq k < i \tag{24}$$

$$\mathbf{y} = [y_1, \cdots, y_k, y_{k+1}, \cdots, y_i] \;\; s.t. \;\; 1 \leq k < i \tag{25}$$

We can create two children from these parents:

$$\mathbf{c}_1 = [x_1, \cdots, x_k, y_{k+1}, \cdots, y_i] \tag{26}$$

$$\mathbf{c}_2 = [y_1, \cdots, y_k, x_{k+1}, \cdots, x_i] \tag{27}$$

D.1.3 Mutation

For the mutation, we take any child solution **c** and mutate one or more of the values in the genome:

$$c_n = c_n + c_n * n \;\; where \;\; -1 \leq n \leq 1 \tag{28}$$

For the polynomial mutation, the possible values of *n* depend on the chosen distribution width. The value of *n* is determined such that it maintains $0 \leq c_n \leq 1$ (Carles-Bou and Galán (2023)).

D.2 Nesting

D.2.1 Genome

The genome for the nesting problem is an array of beams, each of which is an array of parts:

$$[b_1, \cdots, b_i] \;\; s.t. \;\; \forall n \in \mathbb{N} \;\; where \;\; n \leq i, \;\; b_n = [p_1, \cdots, p_j] \tag{29}$$

D.2.2 Recombination

For the recombination, we use the GCO as it is defined by Falkenauer (1996). Given two parents:

$$\mathbf{x} = [x_1, \cdots, x_k, x_{k+1}, \cdots, x_n, x_{n+1}, \cdots, x_i] \;\; s.t. \;\; 1 \leq k < n < i \tag{30}$$

$$\mathbf{y} = [y_1, \cdots, y_k, y_{k+1}, \cdots, y_n, y_{n+1}, \cdots, y_i] \;\; s.t. \;\; 1 \leq k < n < i \tag{31}$$

We create one child from these parents:

$$\mathbf{c} = [x_1, \cdots, x_k, y_{k+1}, \cdots, y_n, x_{n+1}, \cdots, x_i] \tag{32}$$

However, this will cause the child to have duplicate or missing parts. We remove any beams that came from parent **x** that contain parts also contained in the section of the genome coming from parent **y**. For this example, suppose *a* came from parent **x** and *b* came from parent **y** and both contain the same part:

$$[c_1, \cdots, a, \cdots, b, \cdots, c_i] \;\; becomes \;\; [c_1, \cdots, b, \cdots, c_i] \tag{33}$$

This leaves some parts unnested, so we repair the genome using an adaptation of First Fit Descending as described by Falkenauer (1996). Instead of using one length, we randomly choose a length for each beam.

D.2.3 Mutation

For the mutation, we randomly remove a beam from the child solution:

$$[c_1, \cdots, a, \cdots, c_i] \ \ \textit{becomes} \ \ [c_1, \cdots, c_i] \tag{34}$$

We then repair using the adapted First Fit Descending.

D.3 Unique Case ZDT5

The genome for ZDT5 is different than all other tests in ZDT and DTLZ. Instead of an array of real values, the genome is an array of binary strings:

$$[b_1, \cdots, b_i] \ \ s.t. \ \forall n \in \mathbb{N} \ \ \textit{where} \ \ n \leq i, \ \ b_n = [x_m \mid x_m \in \{0, 1\}] \tag{35}$$

As a result, ZDT5 requires different recombination and mutation operators from the rest of ZDT and DTLZ. For this reason, we chose to focus our efforts on testing with ZDT1-4, ZDT6, and all the tests in DTLZ.