TEL-AVIV UNIVERSITY

The Iby and Aladar Fleischman Faculty of Engineering
The Zandman-Slamer School of Graduate Studies

Evolutionary Many Concept Optimization under Multiple Objectives

A thesis submitted toward the degree of
Master of Science in Mechanical Engineering

by

Barak Samina

February, 2019

TEL-AVIV UNIVERSITY

The Iby and Aladar Fleischman Faculty of Engineering
The Zandman-Slamer School of Graduate Studies

Evolutionary Many Concept Optimization under Multiple Objectives

A thesis submitted toward the degree of
Master of Science in Mechanical Engineering

by

Barak Samina

This research was carried out in the School of Mechanical Engineering
Under the supervision of Amiram Moshaiov Ph.D

February, 2019

Acknowledgments

I would like to thank the following persons for making this work possible. First of all, I would like to thank my mentor, Dr. Ami Moshaiov, who provided the inspiration and guidance during the entire research. I also would like to thank Prof. Yoram Reich for his support of my thesis work. My deep appreciation goes to my dear friend and associate Alon Snir for all of his unprecedented help and support during this research. Also, to Nataly Kuritz and Raz Shraga for their great help and support.

Last but not least, I would like to thank my dear wife Yael, for her unbounded support and tolerance, and to also thank my parents and friends for standing next to me during this challenging endeavor.

Abstract

Computers can search and compare many conceptual solutions (concepts) for multi-objective problems. This may be used to find superior concepts before making a decision on a selected concept and eventually on a particular solution. Yet, such a search commonly involves prohibitive computational cost.

This research is motivated by the need to rationalize the allocation of computational resources during the evolution of Set-based Concepts (SBCs). Such an evolution involves the simultaneous exploration of different design spaces, at both the conceptual and particular solution levels, with respect to multiple optimization objectives. Here, several techniques for an on-line estimation of the inferiority of conceptual solutions are suggested and investigated. Such techniques allow on-line intelligent elimination of inferior concepts, which aims at a reduction of the required computational resources.

To evaluate the effectiveness of the proposed techniques, unique concept-based measures and analysis methodology for evolving SBCs are suggested. Extensive statistical study is carried out to assess and compare the suggested techniques. Based on the statistical results, it is concluded that several of the proposed techniques are effective. It is also shown that the effectiveness varies in accordance with the difficulty level of the solved search problems. These conclusions are in accordance with the well-known optimization theorem of no-free lunch.

Table of Contents

1 INTRODUCTION 1
1.1 The Considered Problem 1
1.2 Research Methodology 2
1.3 Thesis Goals and Outline 3
2 BACKGROUND 4
2.1 Evolutionary Algorithms 4
2.2 The Set-Based Concept Approach 4
2.2.1 Fundamentals 4
2.2.2 Comparing SBC methods 5
2.2.3 Concept-based search by evolutionary algorithms 6
3 PROBLEM DEFINITION 7
4 SEARCH METHODOLOGY 8
4.1 Measures 8
4.1.1 Measure of concept-progress 8
4.1.2 Measures of overall-progress 10
4.1.3 Measure of concept-scalar-performance 10
4.1.4 Measures of concept-advance-rate 10
4.2 Concept Ordering Techniques 11
4.2.1 Scalar-performance based orders 11
4.2.2 Non-domination-based orders 12
4.3 Resource Allocation Heuristics 12
4.4 Methods of Evolution 13
4.5 Evolutionary Search Framework and Variants 14
4.5.1 Main procedure of the evolutionary search framework 14
4.5.2 Sub-procedures of the evolutionary search framework 15
5 TESTING METHODOLOGY 19
5.1 Identification Accuracy Measure of Superior-Concepts 19
5.2 Description of the Test Problems 19
5.3 Scalar versus Non-domination based Experiments 21
5.4 The Evolutionary Runs 22
5.5 The Reference Runs and the Evaluation Count 22
6 TESTING RESULTS AND ANALYSIS 23
6.1 Test Problem #1 23
6.2 Test Problem #2 25

6.3 Test Problem #3 27
6.4 Test Problem #4 29
6.5 Test Problem #5 31
6.6 Test Problem #6 33
7 TOWARDS INTERACTIVE SEARCH 36
7.1 Background on interactive evolutionary computation 36
7.2 The envisioned interactive procedure 36
8 SUMMARY & CONCLUSIONS 38
8.1 Summary 38
8.2 Conclusions 38
8.3 Future work 39
9 REFERENCES 40
10 APPENDIX 43
10.1 Test problem #1 – concepts configuration 43
10.2 Test problem #2 – concepts configuration 44
10.3 Test problem #3 – concepts configuration 45
10.4 Test problem #4 – concepts configuration 46
10.5 Test problem #5 – concepts configuration 47
10.6 Test problem #6 – concepts configuration 48

List of Figures

Figure 1: Illustration of the SBC approach 1
Figure 2: Illustration of the Research Methodology 3
Figure 3: Illustrating s-Pareto vs. Relaxed-Pareto 5
Figure 4: Illustration of concept-progress 9
Figure 5: Block diagram of the evolutionary algorithm 15
Figure 6: Illustration of the concept elimination rule 18
Figure 7: Example of the Pareto-fronts of one of the concept-configurations 20
Figure 8: Pareto-fronts of test problem #1 23
Figure 9: Identification accuracy distribution of GS1, GS2 and GS3 24
Figure 10: Identification accuracy distribution of FS1, FS2 and FS3 24
Figure 11: Pareto-fronts of test problem #2 25
Figure 12: Identification accuracy distribution of GS1, GS2 and GS3 26
Figure 13: Superior-concepts identification accuracy distribution of FS1, FS2 and FS3 26
Figure 14: Pareto-fronts of test problem #3 27
Figure 15: Identification accuracy distribution of GS1, GS2 and GS3 28
Figure 16: Identification accuracy distribution of FS1, FS2 and FS3 28
Figure 17: Pareto-fronts of test problem #4 29
Figure 18: Identification accuracy distribution of GN1 and GN2 30
Figure 19: Identification accuracy distribution of FN1 and FN2 30
Figure 20: Pareto-fronts of test problem #5 31
Figure 21: Identification accuracy distribution of GN1 and GN2 32
Figure 22: Identification accuracy distribution of FN1 and FN2 32
Figure 23: Pareto-fronts of test problem #6 33
Figure 24: Identification accuracy distribution of GN1 and GN2 34
Figure 25: Identification accuracy distribution of FN1 and FN2 34
Figure 26: Scheme of the envisioned interactive search 37

List of Tables

Table 1: Methods of Evolution 13
Table 2: Design-space resolution values for each test-function 20
Table 3: Details of the Experimental Plan 21
Table 4: Performance comparison of the examined methods of evolution 25
Table 5: Performance comparison of the examined methods of evolution 27
Table 6: Performance comparison of the examined methods of evolution 29
Table 7: Performance comparison of the examined methods of evolution 31
Table 8: Performance comparison of the examined methods of evolution 33
Table 9: Performance comparison of the examined methods of evolution 35
Table 10: Best results summary 38
Table 11: Details of test problem #1 concept configuration 43
Table 12: Details of test problem #2 concept configuration 44
Table 13: Details of test problem #3 concept configuration 45
Table 14: Details of test problem #4 concept configuration 46
Table 15: Details of test problem #5 concept configuration 47
Table 16: Details of test problem #6 concept configuration 48

List of Acronyms

DM	Decision-Maker
EA	Evolutionary Algorithm
IEC	Interactive Evolutionary Computation
MOEA	Multi-Objective Evolutionary Algorithm
MOP	Multi-Objective Problem
SBC	Set-Based Concept approach

List of Nomenclature

C_{can}	Set of candidate concepts
C_{evo}	Set of evolving concepts
C_{red}	Set of reduced concepts
C_{sup}	Set of superior concepts
E_{adv}	Event of advancement
F_m	Non-dominated set of solutions of the m-th concept
$\mathbf{I}$	Index vector
N_m	Number of resolution intervals
O	Set of concept order
P_m	Set of population history of feasible solutions of the m-th concept
X_m	Design-space of the m-th concept
acc	Superior-concepts identification accuracy
ar	Current concept-advance-rate
cp	Concept-progress measure
cp_{th}	Concept-progress threshold
e_m	Concept progress-indicator
$\mathbf{f}_m$	Objective function of the m-th concept
i_m	Number of iterations in which the m-th concept was selected
l_r	Number of past records
m	Concept's index
n_{can}	Number of candidate concepts
n_{el}	Number of concepts to be eliminated
n_m	Dimension of the m-th concept's design-space
n_o	Dimension of the objective-space
n_{red}	Number of reduced concepts
n_{sup}	Number of superior concepts
op	Overall-progress
$\widetilde{op}$	Modified overall-progress
op_{el}	Overall-progress at elimination
op_{sig}	Significant overall-progress indicator
s	Particular design
$\mathbf{x}_s$	Design vector of solution s

$\mathbf{y}$	Performance vector
$\mathbf{y}_s$	Performance vector of a particular solution s
$\widetilde{y}$	Scalar-performance
$\boldsymbol{\varepsilon}$	Objective-space resolution vector
$\boldsymbol{\varepsilon}^*$	Design-space resolution vector
$\boldsymbol{\omega}$	Objective-weight vector

1 INTRODUCTION

1.1 The Considered Problem

At early stage of the design process, engineers generate and evaluate different conceptual solutions. Once a concept is selected, the process is continued with preliminary and detailed design. It is well-known that concept selection is a challenging problem. Wrong concept selection commonly results in a large waste of resources, which could have a devastating effect on the product development process [1]–[3].

It has been suggested that the traditional process of evaluating concepts can be supported by computations (e.g., [4]–[6]). This is known as the Set-Based Concept (SBC) approach [6]. In the SBC approach, the solutions are classified into meaningful subsets (concepts). Solutions belonging to a particular subset are characterized by some common features. During a search for superior concepts, the performance vectors of particular designs, from all concepts, are compared in a mutual objective-space. The main goal of such an exploration is concept selection. In addition to supporting concept selection, the SBC approach allows exploration of design-spaces at both the conceptual and particular solution levels [7] , and could be viewed as an alternative to multi-modal optimization [8].

Figure 1 illustrates the SBC approach. Three concepts of aircrafts are shown. The generally different design spaces of the concepts are marked by ellipses of different colors. The associated performance vectors of particular designs are shown in the mutual objective-space (marked by light-gray pluses, gray x-s and white stars).

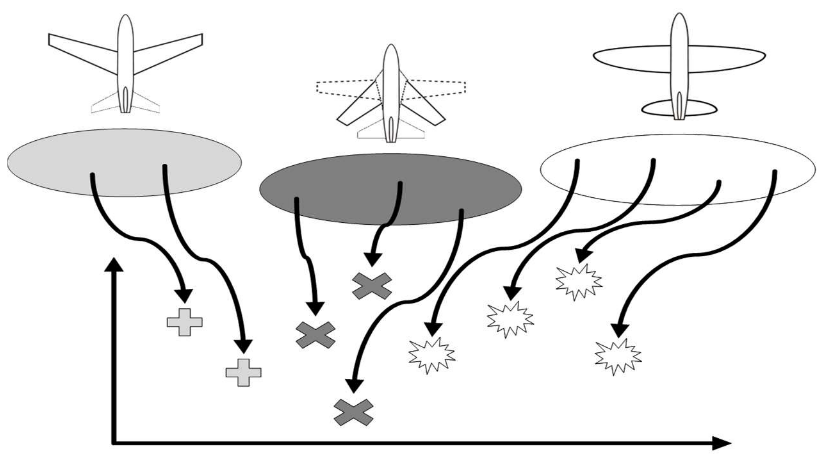

Figure 1: Illustration of the SBC approach

It should be noted that the traditional (manual) process of concept selection by brainstorming is commonly confined to the evaluation of a small number of concepts. In contrast, the SBC approach has no such restriction. In fact, as indicated in [8], the number of explored concepts can be substantial. Namely, designers may divide the set of feasible solutions into a very large number of meaningful subsets.

As the number of explored concepts is increased then the required computational resources could be prohibitive. With this respect, there is a need to devise concept ordering techniques that allow efficient filtering-out of non-promising concepts during the search.

Namely, it would be beneficial to be able to detect the inferior concepts, as early as possible during the search. The on-line detection of such concepts is at the focus of this thesis.

1.2 Research Methodology

Most studies on the SBC approach have not dealt with a large number of concepts as done here. Moreover, while suggesting some heuristics to allocate computational resources among the evolving concept, studies such as in [7] and [9] do not provide an extensive investigation on the effectiveness of the employed heuristics. The current study follows the work in [10], which is based on the assertion that on-line estimation of superiority of concepts can be used to rationalize the on-line allocation of computational resources among the solved concepts. It should be noted, however, that the study in [10] has been confined to single-objective optimization of concepts and the employed demonstration dealt with only a few concepts. In contrast, the current study focuses on multi-objective evolution of a large number of concepts. Hence, it cannot adopt the procedure of [10] as-is.

This study aims to investigate several techniques for concept-ordering to estimate the inferiority of concepts during the process of multi-objective evolution. The suggested concept-ordering techniques are combined with two resource allocation heuristics to form ten methods of evolution, which are examined for their capability to evolve many concepts efficiently. As demonstrated here, employing such methods allows on-line intelligent elimination of concepts. Such eliminations reduce the required computational resources when trying to find the superior concepts. The proposed concept-ordering techniques and resource allocation heuristics are incorporated into a modified version of the evolutionary algorithm of [7]. It should be noted that the employed algorithm is different from those of [7] and [11] as it serves as a framework to be used with the ten methods that are suggested and investigated in this work.

To evaluate the effectiveness of the proposed methods for obtaining the actual superior concepts, a special testing approach is employed. Similar to the testing method in [7], it is based on using a set of bi-objective test functions, where each such function represents a virtual concept and its associated performances. The entire set of such virtual concepts is used to devise six different bi-objective test problems. Each such problem involves one hundred concepts with various numerical difficulties. The entire set of problems is divided into three subsets of two problems each, where each subset is associated with a different level of search difficulty. Using the devised problems, an extensive statistical study is carried out to investigate and compare the effectiveness of the suggested methods.

Figure 2 illustrates the various elements, which are briefly described above, and their combination into the proposed research methodology of this thesis.

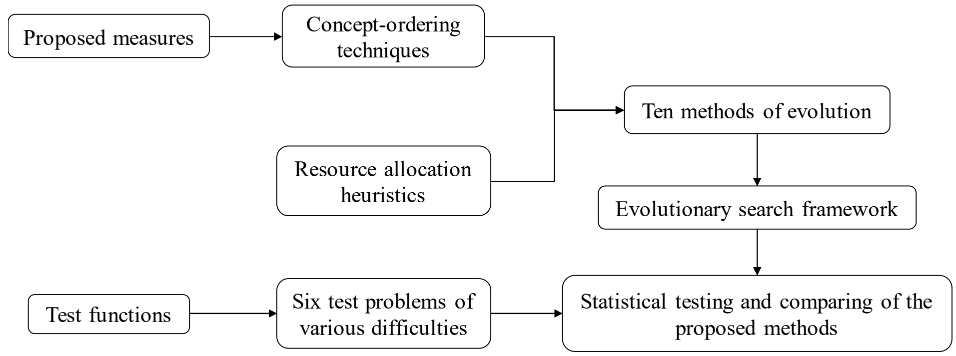

Figure 2: Illustration of the Research Methodology

1.3 Thesis Goals and Outline

This thesis focusses on a preliminary evolutionary search process. The aim of such a preliminary process is an on-line reduction of the amount of the considered concepts into a manageable subset that includes superior concepts. It should be noted that the subset of superior concepts, as found by the preliminary process, is expected to undergo final evaluations to support the selection of a concept and/or a detailed design. Yet, this thesis does not deal with the selection of a concept out of the obtained set of superior concepts.

The main goal of this thesis is to suggest and compare various methods of evolution to support the on-line elimination of inferior concepts during the process of multi-objective evolution of a large number of concepts. While not explicitly performed, a long-term goal of this study is to support the development of interactive techniques for the elimination of inferior concepts during a multi-objective search with a large number of concepts.

The thesis includes the following sub-goals:

1. Devising an evolutionary search framework.
2. Devising various methods to support on-line elimination of inferior concepts.
3. Suggesting academic test problems, with many concepts, based on well-known test functions.
4. Comparing the proposed methods using the suggested academic test-problems.
5. Proposing an envisioned interactive procedure to evolve many concepts under multiple-objectives, based on the devised measures and methods.

This thesis is organized as follows. Chapter 2 provides overviews on several thesis-related topics including: evolutionary computation and the SBC approach. Chapter 3 states the problem definition. Next, Chapter 4 describes the suggested search methodology, including: the proposed measures, the concept-ordering techniques, the resource allocation heuristics, the suggested methods of evolution, the superior concepts definition and the proposed evolutionary search framework. Chapter 5 outlines the testing methodology, whereas Chapter 6 provides the testing results and analysis. Chapter 7 briefly describes the envisioned interactive procedure based on the measures and methods that are suggested and investigated in this thesis. Finally, Chapter 8 concludes this thesis and suggests future research topics.

2 BACKGROUND

2.1 Evolutionary Algorithms

There are many techniques for solving Multi-Objective Problems (MOPs). It is beyond the scope of this thesis to review all such methods, and the reader is referred to reviews such as in [12] and [13], for traditional approaches. When solving MOPs using numerical optimization, the use of Evolutionary Algorithms (EAs) has become a leading approach. These algorithms are inspired by theories of evolution. EA employs a population of solutions that are commonly evolved via mutation and cross-over. In order to direct the evolution towards the optimal solutions, the population of solutions is evaluated for their performances, using selection operator, which resembles natural selection. For this purpose, each solution is assigned with its own performance value, which is commonly termed as fitness. The reader is referred to references such as [14]–[20], for background on various types of EAs.

EAs have been applied to many fields of engineering design including mechanical, electrical, aerospace and civil engineering. A survey on the use of EAs in engineering design can be found in [21]. Exploring mechanical design-spaces by EAs has been treated extensively. For example, EAs have been used for structural optimization of trusses including arrangements of bars (e.g. [22]), area and material selection (e.g. [23]), and shapes of bars (e.g. [24]). Others dealt with optimization of beams topology involving finite element analysis (e.g. [25]). Another example is the application of EAs for control of systems [26], which can be broadly classified into two main fields: 1. off-line control design and analysis, and 2. on-line adaptive control. According to [26], EAs were used to find control parameters in almost all control schemes. EAs have been used in attempts to optimize various aspects of intelligent controllers. In Fuzzy control, it has been used to generate the fuzzy rule-base and to tune the membership functions parameters [27]. In neuro-control, EAs have served for weight learning and optimization of the topology of the neural-network [28].

Originally, EAs were developed to solve single-objective optimization problems. Variants of classical EAs were developed for solving MOPs by the early 2000's. Such Multi-Objective Evolutionary Algorithms (MOEAs) commonly search for the Pareto-optimal set. The reader is referred to [29] for an extensive review on MOEAs. Here, the focus is on MOEAs, which have been developed in conjunction with the SBC approach (see section 2.2).

2.2 The Set-Based Concept Approach

2.2.1 Fundamentals

The Set-Based Concept (SBC) approach has been developed to support concept selection rather than solution selection [30]. It involves a comparison of sub-sets of the solution set, where each such subset represents a conceptual design solution (or in short, a concept). The use of a set to represent a concept, reflects the fact that a design concept is not a final solution. As described in Chapter 1 (see Figure 1), different concepts may be associated with different search spaces. In the SBC approach, the solutions are classified into meaningful subsets (concepts). Solutions belonging to a particular subset are characterized by some common features. Nevertheless, in the SBC approach the objective-space is mutual to all concepts. One could think of many ways to divide the search space. An underlining assumption of the SBC approach is that the selected division is done by the designers prior to the search process based on their perception and knowledge. Moreover, it is assumed that the search space and its division to concepts are well-defined prior to the search process.

2.2.2 Comparing SBC methods

Various SBC techniques have been suggested in the literature, differing primarily by the type of computed information that is sought to support concept selection. The simplest technique is that of obtaining a front per concept [5]. The most popular SBC technique is known as the s-Pareto (e.g. [4], [31], [32]). It is based on obtaining the Pareto-front from the entire feasible set of particular solutions, while keeping track to which concept each solution belongs. The s-Pareto technique has been criticized in [33] and [34]. Figure 3, which is adopted from [35] and discussed in [7], shows the individual Pareto-fronts of four concepts. These are designated by •,▫,⅄, and ▪, for the 1st, 2nd, 3rd, and 4th concept respectively. Using the s-Pareto approach would result in a "combined front" of concept 1 and 2 including the front of the 2nd concept and most of the front of the 1st concept. However, as claimed in [33], one should not ignore the front of the 3rd concept, since that its pair-wise comparisons, with the front of the 1st concept and independently with that of the 2nd concept, are not conclusive. One may declare that the 3rd concept is neither dominated by the 1st nor by the 2nd concept. On the other hand, the 4th concept is dominated by any of the others, therefore it may be viewed as of no relevance for selection. One may conclude from the above discussion, that when the search aims at finding concepts for selection, the decision makers should be presented with the fronts of the 1st, 2nd and 3rd concept.

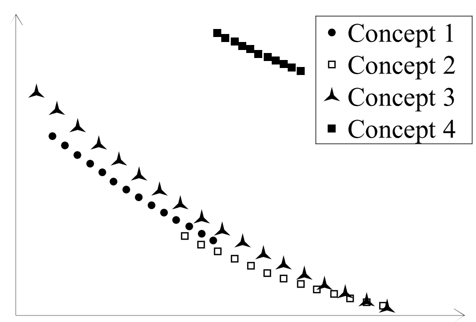

Figure 3: Illustrating s-Pareto vs. Relaxed-Pareto

The optimality versus variability technique of [33] has been suggested as an alternative method to the s-Pareto technique. Yet, it requires finding the fronts of all the considered concepts, which could be computationally prohibitive. To circumvent this problem, a relaxed-Pareto has been suggested [34]. It constitutes a compromise between the s-Pareto and the optimality versus variability technique. With a proper specification of the relaxation, it allows finding concepts 2, 3 and 4 of Figure 3, while filtering-out concept 1. The approach of [34] is the foundation for the development of the concept-based design space exploration in [7] and [11].

2.2.3 *Concept-based search by evolutionary algorithms*

Over more than a decade, various algorithms have been suggested for the SBC approach. These are described in the following.

C_1-NSGA-II and C_2-NSGA-II: In [31] Moshaiov and Avigad discuss simultaneous concept-based search by MOEAs. The algorithms suggested in [31] are C_1-NSGA-II and C_2-NSGA-II, which are based on the well-known NSGA-II [36]. Both C_1-NSGA-II and C_2-NSGA-II aim to find the s-Pareto-front [4]. Yet, they defer by the approach to the computational resource allocation among the evolved concepts.

C-ε-MOEA: In [37] Moshaiov and Snir tailored the well-known ε-MOEA [38] to concept-based search of the s-Pareto front and compared its performance with that of C_1-NSGA-II. The comparison demonstrated that C_1-NSGA-II suffers from premature concept convergence. In contrast, within the limitation of the case studies of [37], C-ε-MOEA did not suffer from that problem.

C_r-NSGA-II: Denenberg and Moshaiov suggested to use the notion of relaxed ε-dominance [35] in order to support concept selection. Their idea is to use a soft evolutionary search approach, which allows concepts with performance close to those of the s-Pareto to survive the evolutionary search process. This approach lets designers to examine the solutions of concepts of the s-Pareto and of concepts that are close to the s-Pareto. The proposed algorithm of [35] constitutes a modification to the well-known NSGA-II.

C_r-ε-MOEA: Following the observation in [37], in his MSc thesis, [39], Denenberg applied the relaxed-Pareto approach to the C-ε-MOEA.

CEC 2015 Algorithm: Based on some of the ideas presented in [35] and [39], an advanced algorithm has been suggested in [7], which is used here as a base for the development of the proposed evolutionary search framework.

3 PROBLEM DEFINITION

The search problem, which is briefly described below, involves two sets including C_{sup}, and C_{can}. The set C_{can} includes the indices of all the considered SBCs (candidate concepts), which are defined by the Decision-Makers (DMs) prior to the search. Given a set C_{can} of n_{can} candidate concepts, the problem is to obtain a set $C_{sup} \subset C_{can}$ with n_{sup} indices of superior concepts, where the number $n_{sup} < n_{can}$ is pre-defined by the DMs.

The superiority of the concepts is determined by a special elimination procedure of inferior concepts, which is a part of the proposed evolutionary search framework. To obtain the set C_{sup}, and the associated Pareto-fronts of the superior concepts, the suggested framework employs an auxiliary set C_{red}. The set C_{red} is intialized to be identical to the set C_{can} . During the evolutionary process this set is updated by a concept elimination process. The elimination process is based on the on-line estimation of inferiority of concepts. Once the cardinality of C_{red}, denoted as n_{red}, is equal to n_{sup}, then C_{red} is declared as C_{sup}.

Let n_o be the dimension of the objective-space $\mathbb{R}^{n_o}$. Let $X_m \subseteq \mathbb{R}^{n_m}$ be the design-space of the m-th concept, and let $\mathbf{f}_m : X_m \to \mathbb{R}^{n_o}$ be the vector of objective-functions that is associated with the m-th concept. Furthermore, let s be any particular design and let m_s and $\mathbf{x}_s$ represent the concept index and the design vector of s, respectively. Also, $x_{s,j}$ be the j-th element of $\mathbf{x}_s$, where j=(1,…,n_m), and $\mathbf{y}_s = \mathbf{f}_{m_s}\left[\mathbf{x}_s\right]$ be the performance vector of s, with $y_{s,i}$ representing the i-th element of $\mathbf{y}_s$, where i=(1,…,n_o).

Without loss of generality, the associated optimization problem, which is based on a Pareto-approach, is hereby defined as finding all the feasible Pareto-optimal solutions and front, for each of the following n_{sup} independent problems:

$$\min \mathbf{f}_m\left[\mathbf{x}\right] \quad \text{for} \quad m \in C_{sup} \tag{1}$$

4 SEARCH METHODOLOGY

This chapter is organized as follows. First, section 4.1 provides several definitions and measures used in the evolutionary search framework. Next, in section 4.2 the proposed concept-ordering techniques are described. Section 4.3 details the suggested resource allocation heuristics. Section 4.4 outlines the proposed ten methods of evolution. Finally, section 4.5 details the proposed evolutionary search framework and its variants.

4.1 Measures

4.1.1 Measure of concept-progress

The on-line evaluation of concept-progress is paramount to the resource allocation and concept elimination during the evolutionary process. The following aims to define a measure of concept-progress, which is based on the characteristics of the performance vector of the evolved individual, which belongs to the considered concept. To formulate the measure, two sets and an index vector are defined as follows.

The population history of the *m*-th concept (P_m) is the set of all the designs of the m-th concepts, which have been evaluated during all past and present iterations.

The non-dominated set of the *m*-th concept (F_m) is the current set of all non-dominated designs of P_m. It is defined as follows:

$$F_m = \left\{ s \in P_m \middle| \nexists\, s' \in P_m : \quad \mathbf{y}_{s'} \succ \mathbf{y}_s \right\} \tag{2}$$

where $\mathbf{y}' \succ \mathbf{y}$ stands for $\mathbf{y}'$ dominates $\mathbf{y}$.

Following [7], each member s of F_m is assigned with an index vector $\mathbf{I}_s \in \mathbb{Z}^{n_o}$ as follows:

$$\mathbf{I}_s = \left(\left\lfloor \frac{y_{s,1}}{\varepsilon_1} \right\rfloor, \ldots, \left\lfloor \frac{y_{s,n_o}}{\varepsilon_{n_o}} \right\rfloor \right) \tag{3}$$

where, ε_i is a pre-defined tolerance (resolution) for the i-th objective. A performance difference is considered insignificant to the DMs if it is lower than the tolerance. Any subset of designs from F_m, with the same index vector $\mathbf{I}$, is termed as an ε-Cell.

The concept-progress measure, which is defined in Eq. 6, is based on an adaptation of the ε-Progress idea of [40]. Here, concept-progress occurs when a new design s (an evolved individual) is accepted to the current concept's non-dominated set (F_m) with a new index vector. Having such a new design with a new index vector, as opposed to one with an already existing index vector, is significant. This is because it means that there is a progress of the concept front, which is larger than the resolution that is significant to the DMs.

Figure 4 illustrates the idea of concept-progress in a bi-objective space with a designated resolution ε in each of the objectives. In the considered problem $|F_m| = 3$. The figure shows the associated three performance vectors, marked by ●, of the three non-dominated solutions of the m-th concept. Now, consider three possible offspring that their performance vectors are marked by ■,✦ and ▲. Only two of these cases, ✦ and ▲, involve the occurrence of a concept-progress. This is due to the fact that in each of these cases the design involves a new index vector. In contrast, the third case, with the performance vector marked by ■, does not involve a new index vector. Hence, the introduction of such a design does not constitute a concept-progress.

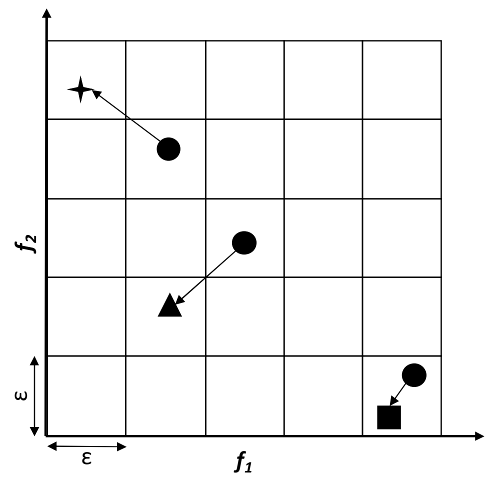

Figure 4: Illustration of concept-progress

During evolution, when a new design s is created for the m-th concept, the following progress-indicator is set for that concept as:

$$e_m = \begin{cases} 1, & \text{if: } s \in F_m \wedge \nexists s' \in F_m : \mathbf{I}_{s'} = \mathbf{I}_s \\ 0, & otherwise \end{cases} \tag{4}$$

In addition, for each concept, a pre-defined number of past records, l_r, of the progress-indicator, e_m, are used to create a sequence of binary values for the m-th concept. To obtain an estimate on the probability of having a concept-progress in the future iterations, a logistic regression is performed on the sequence. The logistic function is [41]:

$$p(z) = \frac{1}{1 + \exp(-z)} \tag{5}$$

where, $p \in [0,1]$ and z is a polynomial function of the independent variable, denoted as t. In the current implementation, z is taken as $z(t) = a + b \cdot t$, where $a, b \in \mathbb{R}$. The parameters for the m-th concept, $a=a_m$ and $b=b_m$, are obtained using the logistic regression process based on the concept's sequence.

The concept-progress measure, of the m-th concept, is defined as:

$$cp_m(i_m) = \frac{1}{1 + \exp(-a_m - b_m \cdot i_m)} \tag{6}$$

where, i_m is the total number of iterations in which the m-th concept was selected during the evolutionary process (see subsection 4.5).

The concept-progress measure is used to estimate the current probability of having progress in the evolution of the m-th concept. Namely, to predict the probability of having an evolved design with a new index vector.

4.1.2 Measures of overall-progress

The overall-progress is calculated as the average concept-progress, of all concepts of C_{red} as follows:

$$op = \frac{1}{\left|C_{red}\right|} \sum_{m \in C_{red}} cp_m \tag{7}$$

To support concept elimination, a significant overall-progress indicator is devised. A significant overall-progress is achieved if the overall-progress value was reduced, by some given ratio, with respect to its last value in which concept elimination has occurred (denoted as op_{el}). The significant overall-progress indicator is defined as follows:

$$op_{sig} = \begin{cases} 1 & \text{if: } op < (1-k) \cdot op_{el} \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

where, $k \in [0,1]$ (in the current implementation $k = 0.1$). The op_{el} is initialized as the overall-progress of the initial population.

4.1.3 Measure of concept-scalar-performance

The scalar-performance of any individual design s is calculated based on a weight vector $\boldsymbol{\omega} \in \mathbb{R}^{n_o}$ that is provided by the DMs prior to the search. The scalar-performance of the m-th concept is defined as the minimal scalar-performance of its non-dominated solutions. Namely:

$$\tilde{y}_m = \min_{s \in F_m} \left\{ \widehat{\boldsymbol{\omega}} \cdot \mathbf{y}_s \right\} \tag{9}$$

where $\widehat{\boldsymbol{\omega}} = \dfrac{\boldsymbol{\omega}}{\|\boldsymbol{\omega}\|}$.

4.1.4 Measures of concept-advance-rate

The advance-rate of the m-th concept is derived from the change of the concept's scalar-performance between the current and the previous value from the last iteration of that concept. It is defined as:

$$ar_m\big|_{i_m} = \tilde{y}_m\big|_{i_m} - \tilde{y}_m\big|_{i_m - 1} \tag{10}$$

In addition, for each concept, the average advance-rate is calculated based on its last Δ iterations.

$$\overline{ar}_m = \frac{1}{\Delta} \sum_{n = i_m - \Delta}^{i_m} ar_m\big|_n \tag{11}$$

It is noted that in the current implementation Δ=500.

4.2 Concept Ordering Techniques

4.2.1 Scalar-performance based orders

The first SP-order approach

This order-of-concepts approach is based on sorting the concepts of C_{red} by their scalar-performance measure value. The sorting is done such that concepts with the lowest concept-scalar-performance are assigned with ordering indexed of 1, and so on. This results with an ordered set of sets, O. For example, consider the following ordered set of sets $O=\{\{1,4\},\{3\},...,\{2,8\}\}$. In the example, the first subset of O includes the indices of concept #1 and #4, these are assigned with order index 1. Next, for the second subset of O, concept #3 is assigned with order index of two. In general, the concepts associated with the i-th subset of O are assigned with order index equal to i.

The second SP-order approach

This order-of-concepts' approach is based on sorting the estimated future scalar-performance of the concepts of C_{red}. The estimation is done, for the m-th concept, as follows:

$$\tilde{y}^{*}_{m} = \tilde{y}_{m} + \Delta i \cdot cp_{m} \cdot \overline{ar}_{m} \tag{12}$$

where, Δi is the number of future iterations for the estimation, $\tilde{y}_m$, cp_m and $\overline{ar}_m$ are the scalar-performance, the concept-progress measure and the average advance-rate of the m-th concept, respectively (see subsection 4.1). It is noted that in the current implementation, Δi=100 iterations. Once the estimation is calculated for all concepts of C_{red}, the ordering is done as in the first SP-order approach.

The third SP-order approach

This order-of-concepts' approach is inspired by the idea of non-domination sorting [36]. Yet, it is considered here as scalar-performance based approach as it uses non-domination sorting based on the scalar-performances of the concepts of C_{red}. The sorting is performed on the concepts of C_{red} using an auxiliary-space. Each concept is evaluated, in the auxiliary-space, by a single performance-vector that consists of the concept's current scalar-performance and its current concept-progress value. The ordering is done using non-domination sorting in the auxiliary-space as follows:

1. Map the concepts of C_{red} to the auxiliary-space, such that each concept's performance vector contains its scalar-performance and the concept-progress measure.
2. Create a temporary set K that includes the auxiliary performance-vectors of the concepts.
3. Set the scalar-performance to be minimized.
4. Set the concept-progress to be maximized.
5. Initialize an ordered set of sets O.
6. While the set K is not empty:
    a. Obtain the non-dominated set of concepts based on the performance-vectors of the set K.

b. Append the obtained non-dominated set of concepts to *O*.

c. Discard the performance-vectors of the obtained non-dominated set from *K*.

4.2.2 Non-domination-based orders

The first ND-order approach

Similar to the third SP-order approach, the first ND-order approach is inspired by the idea of non-domination sorting [36]. Yet, while the former approach employs an auxiliary objective space, here the non-domination sorting is performed in the original objective space. The process steps are:

1. Create a union set *K* of the performance-vectors from all the considered concepts of C_{red}.
2. Initialize an ordered set of sets *O*.
3. While the set *K* is not empty:
   a. Obtain the non-dominated set of concepts based on the performance-vectors of the set *K*.
   b. Append the obtained non-dominated set of concepts to *O*.
   c. Discard the performance-vectors of the obtained non-dominated set from *K*.

The second ND-order approach

This order-of-concepts' approach orders the concepts of C_{red} in an extended-objective-space. This space involves the original objectives, where the additional objective is the concept-progress to be maximized. Using the performance-vectors of the concepts of C_{red} in the extended-objective-space, the ordering procedure is performed as described in the first ND-order approach.

4.3 Resource Allocation Heuristics

During the evolutionary process, the concepts of C_{red} are divided into two types. The first type includes concepts that are to be further evolved and the second type concerns concepts that are candidate to be superior, but are excluded from any future evolution. The latter type includes concepts that are believed to have already been converged (see subsection 4.5). The indices of the concepts of the first type are recorded in the set $C_{evo} \subseteq C_{red}$.

Two substantially different heuristics are suggested to be investigated. The first heuristic is a greedy one, it aims to distribute the computational resources by favoring concepts with higher concept-progress values. The second heuristic is a fair one, namely resources are distributed evenly among the evolving concepts. The proposed heuristics are described below:

The greedy heuristic:

The rationale behind the greedy approach is that concepts of C_{evo} that are most likely to exhibit progress of their current non-dominated set, will be favoured in terms of the resource allocation. Namely, it uses the concept-progress measure (see Eq. 6).

The fair heuristic:

The rationale behind the fair approach is that all the evolving concepts should be treated equally in terms of the allocated resources. Namely, during the evolutionary process the concepts of C_{evo} are chosen in a rotating order. This means that on the first iteration the first concept of C_{evo} is selected and so on until the last concept; then a new cycle starts.

4.4 Methods of Evolution

Ten different methods of evolution are hereby suggested to support on-line concept elimination. Each of these methods is in fact a declaration of a different possible combination of a concept-ordering technique, which is used for concept elimination, and of a resource allocation heuristic. The declaration is served as an input to the evolutionary search framework, which transforms it into a unique evolutionary algorithm based on the selected method. The proposed ten methods of evolution are listed in Table 1, including their identification codes, which are used in the description of the comparison study. It is noted that the process of using the concept-ordering techniques to eliminate concepts is described in the *concept elimination* sub-procedure in subsection 4.5.2.

Table 1: Methods of Evolution

Code	Resource allocation heuristic	Concept ordering technique
GS1	Greedy	1st SP-order
GS2	Greedy	2nd SP-order
GS3	Greedy	3rd SP-order
GN1	Greedy	1st ND-order
GN2	Greedy	2nd ND-order
FS1	Fair	1st SP-order
FS2	Fair	2nd SP-order
FS3	Fair	3rd SP-order
FN1	Fair	1st ND-order
FN2	Fair	2nd ND-order

Each method is coded in an XYZ format, where X represents the resource allocation heuristic of the method, and YZ represents the specific concept ordering technique of the method.

4.5 Evolutionary Search Framework and Variants

The following describes the main and the sub-procedures of the proposed evolutionary search framework and variants. It should be noted that the described evolutionary search framework becomes an evolutionary algorithm variant when one of the methods of the previous section is applied to fully define a search technique.

The suggested evolutionary search framework is based on the concept of a steady-state evolutionary algorithm, similar to that of [38]. In contrast to the generation-based approach, at each iteration of a steady-state evolution, a new individual is created and examined for inclusion in the archive. In the current study, following [7], the idea of a steady-state evolution is used in conjunction with SBCs.

4.5.1 Main procedure of the evolutionary search framework

Figure 5 describes the main procedure of the evolutionary search framework. It starts by performing an *Initialization* sub-procedure (see subsection 4.5.2). During initialization, the set C_{can}, the number n_{sup}, the objective-space resolution vector $\boldsymbol{\varepsilon}$, the design-space resolution vector $\boldsymbol{\varepsilon^*}$ (of each concept), the method of evolution and the number of past records l_r are defined by the DMs. In addition, random population is created for each of the concepts. The individuals of these populations are evaluated and then sorted. For each concept, this results in an initial P_m, an initial F_m and the associated ε-Cells.

As shown in Figure 5, following the initialization, a steady-state concept-based evolution loop takes place. Within the loop, a concept is selected by employing the *Concept and solution selection* sub-procedure (see subsection 4.5.2). From the selected concept a parent individual design, *s*, is selected. The selected individual undergoes *Element mutation* to create an offspring design *s*' (see subsection 4.5.2). Then, in the *Concept status* sub-procedure (see subsection 4.5.2), the sets P_m and F_m are updated as-well-as the concept-progress measure. If the concept-progress is less than a predefined concept-progress threshold, then the concept is removed from the set C_{evo}. Next, the stopping condition of the evolutionary search is checked. If not satisfied, then elimination bypass condition is checked using the associated sub-procedure. If satisfied, then another iteration of the evolutionary loop takes place. Otherwise, based on the concepts' current performances, concepts to be eliminated from the search are to be determined by the *Concept elimination* sub-procedure (see subsection 4.5.2). Once completed, a new iteration of the evolutionary loop starts.

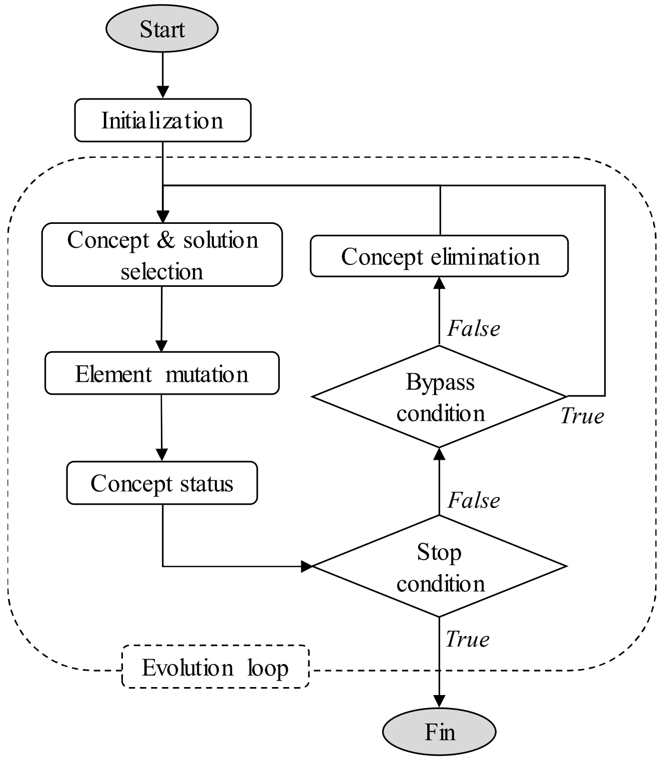

Figure 5: Block diagram of the evolutionary algorithm

4.5.2 Sub-procedures of the evolutionary search framework

Initialization

The initialization sub-procedure contains the following steps:

1. Set the objective-space resolution vector $\boldsymbol{\varepsilon} \in \mathbb{R}_+^{n_o}$.
2. Set the design-space resolution vector $\boldsymbol{\varepsilon}^* \in \mathbb{R}_+^{n_m}$, for each concept, such that for each of the design-parameters the range is divided into N_m resolution intervals, i.e. $\varepsilon_j^* = \dfrac{x_j^{\max} - x_j^{\min}}{N_m}$, $\forall j \in [1, n_m]$.
3. Set the number of past records $l_r \in \mathbb{N}_1$ (to be used by the logistic-regression in the concept-progress measure evaluation).
4. Set the concept-progress-threshold, $cp_{th} \in [0,1]$.
5. Set the method of evolution (see section 4.4).
    a. If the method of evolution involves a scalar-performance based ordering approach, then set an objective-weight vector $\boldsymbol{\omega} \in \mathbb{R}^{n_o}$.
6. Define the set of candidate concepts C_{can}.
7. Set the desired number of superior concepts $n_{sup} \in \mathbb{N}_1$.
8. Initialize the sets C_{red} and C_{evo} to be identical to C_{can}.
9. Initialize, for each concept, the concept-progress to the value of one.
10. Randomly initialize and evaluate a population of designs with equally sized subpopulation for each concept.

11. Initialize, for each concept, the sets P_m and F_m, as defined in subsection 4.1.1.

Concept and solution selection

The concept and solution selection procedure is done as follows. First, a concept is being selected from C_{red}, based on the applied method of evolution (see subsection 4.4). Next, from the selected concept, an ε-Cell is selected. The ε-Cell selection is based on a selection-counter. Each ε-Cell is assigned with a selection-counter that records the number of iterations, in which any individual design *s* of the ε-Cell has been selected for mutation (during the evolutionary process). The selected ε-Cell will be the ε-Cell with the lowest selection-count value. Finally, from the selected ε-Cell, an individual solution is randomly selected. Based on the above, and the applied method of evolution, the concept and solution selection is performed as follows:

For greedy heuristic based methods:

1. Select, from C_{evo}, the concept with the highest concept-progress value. If there is more than one concept with that value, then one of these concepts is randomly selected.
2. Select the ε-Cell, of the selected concept, with the lowest selection-counter value. If there is more than one ε-Cell with the same lowest selection-counter value, then one of these ε-Cells is randomly selected.
3. Randomly select one individual design, *s*, from the selected ε-Cell.

For fair heuristic based methods:

1. Obtain the current concept in the rotation (see subsection 4.4), from C_{evo}.
2. Select the ε-Cell, of the current concept, with the lowest selection-counter value. If there is more than one ε-Cell with the same lowest selection-counter value, then one of these ε-Cells is randomly selected.
3. Randomly select one individual design, *s*, from the selected ε-Cell.

Element mutation

The mutation operator produces a new offspring design *s'* based on a parent design *s*. The design vector of *s'* ($\mathbf{x}_{s'}$) will be identical to the design vector of its parent design *s* ($\mathbf{x}_s$) except of one design parameter, which is randomly selected to be mutated. The assumption here is that, the design-space of each concept is bounded and divided into hyper-boxes, which are defined according to a desired resolution. Let $x_{m,j}^{\min}, x_{m,j}^{\max}$ and $N_{m,j}$ be the lower limit, the upper limit and the number of resolution intervals for the *j*-th component of the design vector of the *m*-th concept, respectively.

The mutation is performed as follows:

1. Initialize $\mathbf{x}_{s'}$ to be identical to $\mathbf{x}_s$.
2. Randomly select one design parameter $j \in (1,...,n_m)$ to be mutated and a direction indicator $t \in \{-1,+1\}$.
3. Calculate the number of resolution intervals available for the mutation step for the selected parameter and direction as follows:

$$\Delta = \begin{cases} \dfrac{x_{s,j} - x_{m,j}^{\min}}{x_{m,j}^{\max} - x_{m,j}^{\min}} \cdot N_{m,j} & \text{if } t = -1 \\[2ex] \dfrac{x_{m,j}^{\max} - x_{s,j}}{x_{m,j}^{\max} - x_{m,j}^{\min}} \cdot N_{m,j} & \text{if } t = +1 \end{cases} \tag{13}$$

4. If $\Delta<1$, then select the other direction and recalculate Δ.
5. Invoke random value drawn from the standard normal distribution $N(\mu, \sigma^2)$, with median value μ and variance σ^2, and calculate a normalized mutation step size, δ, as:

$$\delta = \min\left[\ \Delta, 1 + \left\lfloor \operatorname{abs}\left(\mathrm{N}\left(0, \min\left[\alpha, \beta \cdot \Delta\right]\right)\right) \right\rfloor \right] \tag{14}$$

   It is noted that in the current implementation, $\alpha = 25$ and $\beta = \dfrac{1}{3}$.

6. Set the new value of the selected design parameters of s' according to the mutation step size:

$$x_{s',j} = x_{s,j} + t \cdot \frac{\delta}{N_{m,j}} \cdot \left(x_{m,j}^{\max} - x_{m,j}^{\min}\right) \tag{15}$$

<u>Concept status</u>

The concept status sub-procedure contains the following steps:

1. Update the sets P_m and F_m of the selected concept using the offspring s'.
2. Re-evaluate the concept-progress value for the selected concept.
3. If the concept-progress is less than concept-progress-threshold (i.e. $cp_m<cp_{th}$), then remove the index of the selected concept from C_{evo}.

<u>Elimination bypass condition</u>

Concept elimination sub-procedure is performed if <u>both</u> of the following conditions are satisfied:

1. $n_{red} > n_{sup}$.
2. A significant overall concept-progress is achieved, i.e. $op_{sig} = 1$ (Eq. 8).

<u>Concept elimination</u>

This sub-procedure aims to efficiently (gradually) eliminate non-promising concepts during the evolutionary process. It is based on the applied method of evolution (as detailed in subsection 4.4), where the non-promising concepts, to be eliminated, are chosen from the worst-performing concepts.

The number of concepts to be eliminated, denoted as n_{el}, is determined by the following rule. Let $\widetilde{op}$ be the normalized overall-progress such that $\widetilde{op} \in [0,1]$, and defined as:

$$\widetilde{op} = 1 - \frac{op - cp_{th}}{op_0 - cp_{th}} \tag{16}$$

Also, let n_{red}^{*} be the calculated target cardinality of C_{red} after elimination. It is calculated based on the value of $\widetilde{op}$ as follows:

$$n_{red}^{*} = \max\left[n_{sup},\ \min\left[n_{can},\ n_{can} - \left\lfloor \left(\frac{n_{can} - n_{sup}}{b-a} \right) \cdot \left(\widetilde{op} - a \right) \right\rfloor \right] \right] \tag{17}$$

where, $a, b \in \mathbb{R}$. It is noted that in the current implementation, a=0.10 and b=0.98.

Figure 6 illustrates the gradual concept elimination rule. In the figure, the black curve represents the calculated value of n_{red}^{*} during the evolutionary search.

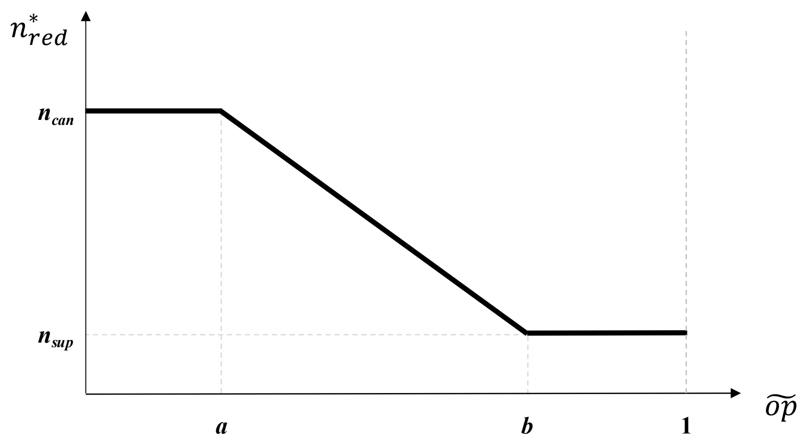

Figure 6: Illustration of the concept elimination rule

Finally, the number of concepts to be eliminated is calculated as:

$$n_{el} = \max\left[0,\ n_{red} - n_{red}^{*} \right] \tag{18}$$

The concept elimination sub-procedure is performed as follows:

1. Calculate the number of concepts to be eliminated, n_{el}, (see Eq 18).
2. Eliminate the worst performing n_{el} concepts from C_{red}, based on the order-of-concepts, starting from the last order index, as follows:
   a. If there is more than one concept in the same order-index, then randomly eliminate a concept of that order-index until n_{el} concepts are eliminated or that the order-index is emptied.
   b. If the last order-index is emptied during the elimination, then proceed with elimination from the next order-index until n_{el} concepts are eliminated.
3. Update op_{el} value as the current overall-progress.

5 TESTING METHODOLOGY

This chapter describes the testing methodology including the test problems that are used in this work. It is organized as follows. Section 5.1 defines the identification accuracy measure of superior-concepts, which is used for evaluating and comparing of the methods of evolution. Section 5.2 details the test problems of this work and Section 5.3 lists the information on the evolutionary runs that were done for the comparison study of the examined methods of evolution.

5.1 Identification Accuracy Measure of Superior-Concepts

The identification accuracy measure of superior-concepts is used for evaluating the performance of a given search technique variant (see subsection 4.5). It measures the identification accuracy of the obtained set of superior-concepts, C_{sup}, at the end of the evolutionary search, with respect to the set of the true-superior-concepts, C^*_{sup}, of a given search problem. The set of the true-superior-concepts is defined according to the procedure described in section 5.5.

The identification accuracy measure is calculated as follows:

$$acc = \frac{\left|C_{sup} \cap C^*_{sup}\right|}{\left|C^*_{sup}\right|} \tag{19}$$

5.2 Description of the Test Problems

The experimental plan involves six test problems. These problems are designed in order to test and compare the performance of the different search techniques (see subsection 4.5) using the identification accuracy measure (see subsection 5.1). The test problems differ by their concept configuration and by their level of difficulty. The difficulty level is expected to influence the identification accuracy, i.e. for a test problem with high difficulty level, the identification accuracy is expected to be very low and vice-versa.

Each test problem involves a unique concept configuration using 100 different virtual concepts. These concepts are defined by modifications to the following well-known bi-objective test-functions: FON, SCH, ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6 (see [36]). In order to obtain the six different concept configurations, these functions are modified by random translations in the objective-space to create the required set of virtual concepts. The Appendix of this thesis provides the details of all the functions and translations that were used to define the six concept configurations. Illustrations of the Pareto fronts of the obtained six configurations are provided in the next chapter. One such example is provided in Figure 7.

In each concept configuration, the true-superior-concepts are based on ZDT4. In general, ZDT4 constitutes a relatively difficult problem as it involves local Pareto fronts. By choosing ZDT4 for the true-superior-concepts it is ensured that the test problems are difficult. To create configurations of different convergence difficulties, additional modifications of the employed functions were used by the following technique.

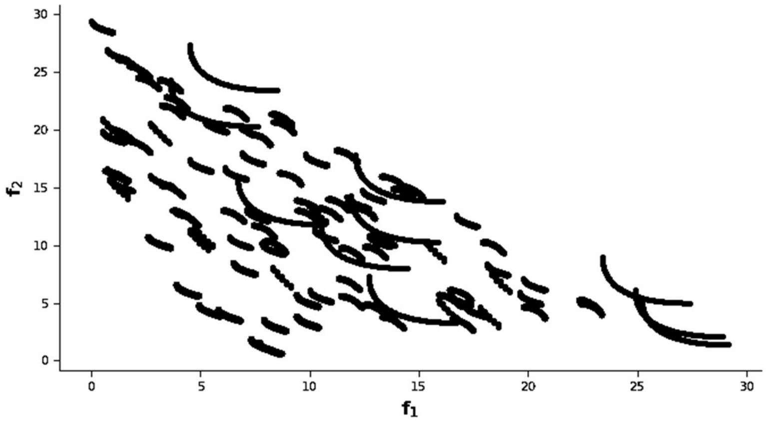

Figure 7: Example of the Pareto-fronts of one of the concept-configurations

The original variables of the aforementioned functions are continuous within given ranges. These variables are re-defined into discrete variables by resolutions that are re determined using a number of divisions in the decision-spaces. The options for the number of divisions, N_m, of the decision-spaces are detailed in Table 2 below. The actual selection of N_m, for each concept of each of the six concept-configuration is defined in the Appendix (see chapter 10).

The decision-space resolution determines the amount of feasible designs in the decision-space, thus affects the convergence rate of the concept. Namely, high resolution in the decision-space (i.e., a vast amount of feasible designs) causes slow convergence of the concept and vice-versa. It is noted that, in the current implementation, for the true-superior-concepts, in each configuration either fine (the highest N_m option) or coarse (the lowest N_m option) decision-space resolution is used.

Table 2: Design-space resolution values for each test-function

Function	N_m Options
FON	100, 200, 300
SCH	20000, 30000, 40000
ZDT1	40, 60, 80, 100
ZDT2	40, 60, 80, 100
ZDT3	80, 100, 120, 140
ZDT4	20, 60, 80, 100, 350
ZDT6	100, 200, 300, 400

5.3 Scalar versus Non-domination based Experiments

The six test problems, which are listed in Table 3, are divided into two groups. Each group consists of three test problems. The first three test problems (1st group) aim to test and compare the scalar-performance based methods of evolution. The last three test-problems (2nd group) aim to test and compare the non-domination-based methods of evolution. Within each group the problems differ by their level of difficulty including hard, medium and easy (see Table 3), and by their concept-configuration.

Table 3: Details of the Experimental Plan

Test problem #	Group #	Difficulty level	Methods of evolution	Section
1	1	Hard	GS1, GS2, GS3 FS1, FS2, FS3	6.1
2	1	Medium	GS1, GS2, GS3 FS1, FS2, FS3	6.2
3	1	Easy	GS1, GS2, GS3 FS1, FS2, FS3	6.3
4	2	Hard	GN1, GN2 FN1, FN2	6.4
5	2	Medium	GN1, GN2 FN1, FN2	6.5
6	2	Easy	GN1, GN2 FN1, FN2	6.6

Given the fundamental difference between the scalar and the non-domination based methods, the approach to find the set of true-superior-concepts is different for each of the groups. The following details how the set of true-superior-concepts is obtained for each group.

For the scalar based methods (1st Group), the set of true-superior-concepts is obtained as follows. First, the Pareto-fronts of all concepts of C_{can} are obtained via evolutionary search without concept elimination (see the next subsections for the run details). Next, the associated concept-scalar-performance are calculated, for each concept. Then, the ordered set of sets O (see subsection 4.2.1) is obtained by performing the first scalar-performance order approach. Finally, the set of true-superior-concepts is defined as the subset of O, which includes the indices of the best n_{sup} concepts. It should be noted that at the end of the evolutionary search the concept-progress is zero for all concepts of C_{red}. This means that all the scalar-performance based orders will yield the same set O. This can be proven analytically by examining the definition of these approaches.

For the non-domination based methods, the set of true-superior-concepts is obtained as follows. First, the Pareto-fronts of all concepts of C_{can} are obtained via evolutionary search without concept elimination. Then, the concepts are ordered by the first non-domination order approach (see subsection 4.2.2). At the end of the evolutionary search the concept-progress is zero, for all concepts of Cred. Therefore, in the second non-domination order approach, the additional objective used to define the extended-objective-space, vanishes. This means that the extended-objective-space is the *original* objective-space. Hence, the first and second non-domination order approaches yields the same set O.

5.4 The Evolutionary Runs

For each method of evolution, on each test problem, a statistical study with 30 independent runs is carried-out. For all of the independent runs the following initial parameters were set:

1. Initial population size, of each concept, was set to 100 individuals.
2. The number of past records, for each concept, was set to l_r=1000.
3. The concept-progress threshold was set to cp_{th}=0.001.
4. The desired number of superior concepts was n_{sup}=10.
5. The objective-space resolution vector was $\boldsymbol{\varepsilon} = [0.05, 0.05]$.
6. If the method of evolution involved a scalar-performance based ordering approach, then the objective-weight vector was $\boldsymbol{\omega} = [1,1]$.

5.5 The Reference Runs and the Evaluation Count

For each test problem, and for each resource allocation heuristic, reference runs are performed without concept elimination. These are coded similarly to the methods of evolution. The runs associated with the greedy heuristic are coded as GR0, while the runs associated with the fair heuristic are coded as FR0. These reference runs are used to compare their computational efforts with those needed by the runs with concept eliminations. This is achieved by averaging the fraction of the number of evaluations required for the test problem runs with respect to the required evaluations in the reference runs. A statistical study with 30 independent runs is carried-out for the reference methods of evolution GR0 and FR0. For all of these independent runs the following initial parameters were set:

1. Initial population size, of each concept, was set to 100 individuals.
2. The number of past records, for each concept, was set to l_r=1000.
3. The concept-progress threshold was set to cp_{th}=0.001.
4. The desired number of superior concepts was n_{sup}=n_{can}.
5. The objective-space resolution vector was $\boldsymbol{\varepsilon} = [0.05, 0.05]$.
6. The reference method of evolution, GR0 or FR0.

6 TESTING RESULTS AND ANALYSIS

This chapter provides the testing results and analysis of the six test problems, which are described in the previous chapter and in the Appendix (see chapter 10). The performances of the evolutionary search variants are compared and analyzed based on the identification accuracy measure (see subsection 5.1), where higher is better. In addition, these variants are compared to the reference runs with respect to the required computational resources. In general, due to the concept elimination, it is expected that the amount of the required computational resources will be reduced. Considering the problem definition (as discussed in Chapter 3), the goal of the evolutionary search is to obtain a set of superior-concepts, as accurate as possible. Therefore, the superior-concepts identification accuracy is considered as the main criterion for comparison. For all the tests, the true-superior-concepts were defined as detailed in section 5.5. In the following sections, for each test and method, the obtained accuracy is illustrated using a colored bar. The height of such a bar indicates the number of independent runs that were obtained with the accuracy value of the bar. It is noted that the shown accuracy values are discrete and ranges from zero to one with 0.1 intervals. An accuracy value of 0.1 means that only one concept, out of the obtained ten superior concepts, is a true-superior-concept. The following sections details the results and analysis of each test problem.

6.1 Test Problem #1

This test problem introduces high difficulty in obtaining the true-superior-concepts, where the true-superior-concepts were defined using ZDT4 with the highest available resolution in the decision-space (see Table 2). Namely, the true-superior-concepts are expected to be eliminated quickly during the evolutionary search. Figure 8 illustrates the Pareto-fronts of the concept configuration of this test problem. The black and grey fronts are associated with the true-superior-concepts and the rest of the concepts, respectively.

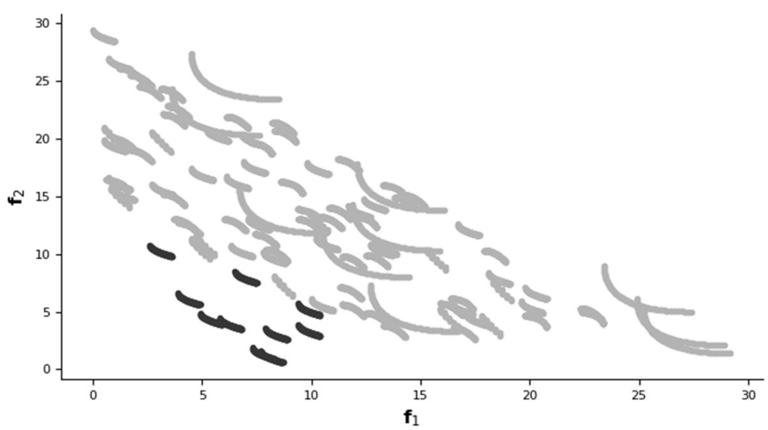

Figure 8: Pareto-fronts of test problem #1

Results:

Figure 9 compares the identification accuracy performances of the following methods of evolution: GS1, GS2 and GS3. The black, grey and white bars are associated with the obtained accuracies of the GS1, GS2 and GS3 methods, respectively.

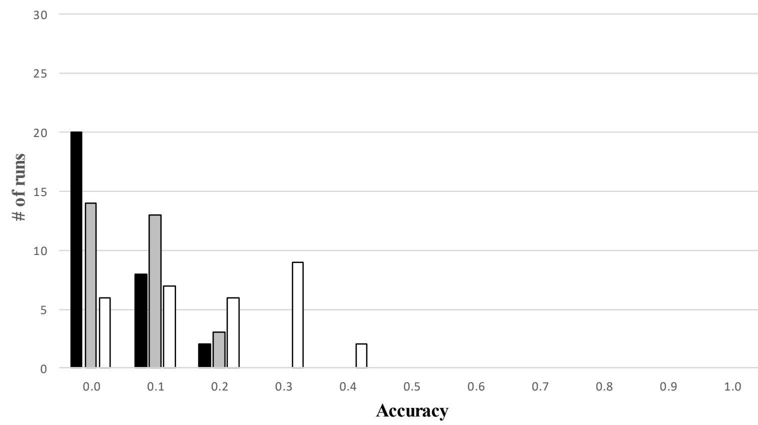

Figure 9: Identification accuracy distribution of GS1, GS2 and GS3

Figure 10 compares the identification accuracy performances of the following methods of evolution: FS1, FS2 and FS3. The black, grey and white bars represent the obtained accuracies of the FS1, FS2 and FS3 methods, respectively.

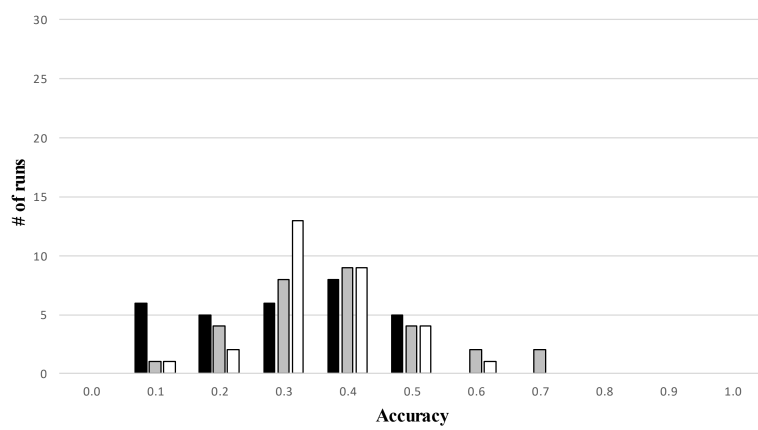

Figure 10: Identification accuracy distribution of FS1, FS2 and FS3

Table 4 summarizes the performances of the various methods of evolution, which are examined in this test problem. It provides the averaged identification accuracy, and the averaged evaluation count as obtained from the repeated runs. As expected, all methods reduced the amount of required evaluations, with respect to the associated reference method (see Section 5.5). In general, the GSx methods achieved better computational resource reduction in comparison with the FSx methods. Yet, all methods achieved substantial computational resource reduction. The best performing method, in terms of computational resource reduction, is the GS2 approach, however, it has poor accuracy. Due to the difficulty level of thi test problem, one should have not expected high accuracy. In general, the FSx methods performed significantly better than the GSx methods on the identification accuracy. The best performing method, on the identification accuracy, is the FS2 method. Based on

Table 4, it appears that, for this test problem, the FS2 method should be preferred when accuracy is the most important criterion.

Table 4: Performance comparison of the examined methods of evolution

Method of evolution	Average evaluation count	Average identification accuracy
GR0	517,402 (100%)	1.000 ± 0.000
GS1	111,048 (21.5%)	0.043 ± 0.068
GS2	104,230 (20.1%)	0.063 ± 0.067
GS3	117,158 (22.6%)	0.180 ± 0.127
FR0	515,617 (100%)	1.000 ± 0.000
FS1	133,751 (25.9%)	0.303 ± 0.140
FS2	132,395 (25.7%)	0.383 ± 0.146
FS3	156,823 (30.4%)	0.353 ± 0.104

6.2 Test Problem #2

This test problem introduces medium difficulty level in obtaining the true-superior-concepts. The true-superior-concepts were defined using ZDT4, where half of them have the highest available resolution in the decision-space, whereas the other half has the lowest available resolution in the decision-space (see Table 2). Figure 11 illustrates the Pareto-fronts of the concept configuration of this test problem. The black and gray fronts are associated with the true-superior-concepts and the rest of the concepts, respectively.

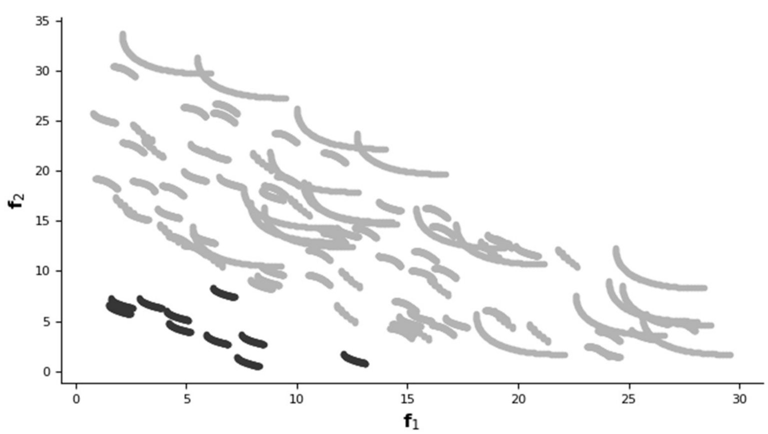

Figure 11: Pareto-fronts of test problem #2

Results:

Figure 12 compares the performance of the following methods of evolution: GS1, GS2 and GS3 on the 2nd test problem. The black, grey and white bars represent the obtained accuracies of the GS1, GS2 and GS3 methods, respectively.

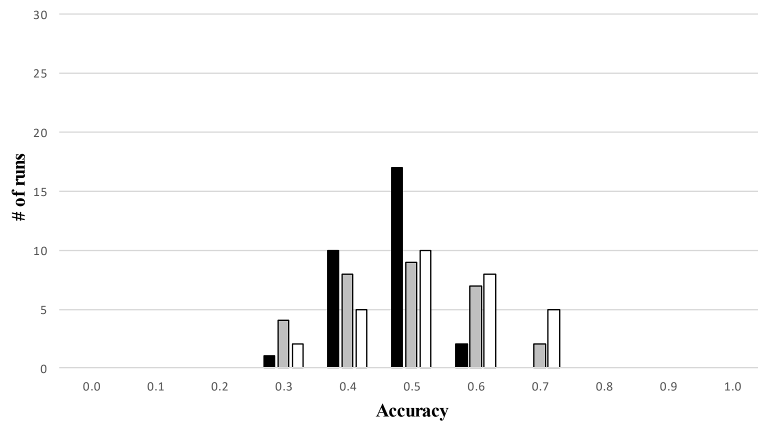

Figure 12: Identification accuracy distribution of GS1, GS2 and GS3

Figure 13 compares the identification accuracy performance of the following methods of evolution: FS1, FS2 and FS3. The black, grey and white bars represent the obtained accuracies of the FS1, FS2 and FS3 methods, respectively.

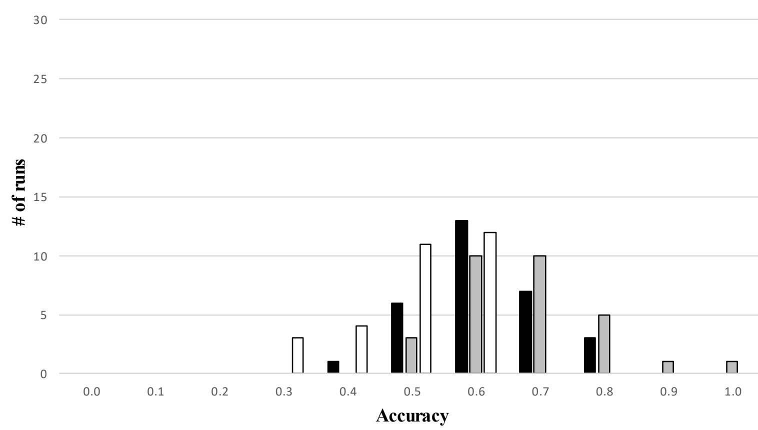

Figure 13: Superior-concepts identification accuracy distribution of FS1, FS2 and FS3

Table 5 compares the averaged performances of the various methods of evolution, which were examined in this test problem. These include the averaged identification accuracy and the averaged evaluation count over the runs.

Table 5: Performance comparison of the examined methods of evolution

Method of evolution	Average evaluation count	Average identification accuracy
GR0	478,237 (100%)	1.000 ± 0.000
GS1	96,884 (20.3%)	0.467 ± 0.066
GS2	86,700 (18.1%)	0.483 ± 0.117
GS3	78,184 (16.3%)	0.530 ± 0.115
FR0	477,595 (100%)	1.000 ± 0.000
FS1	94,334 (19.8%)	0.617 ± 0.099
FS2	92,868 (19.4%)	0.680 ± 0.116
FS3	131,528 (27.5%)	0.507 ± 0.098

As in the previous test problem, all methods achieved substantial computational resource reduction, where the GSx methods performed slightly better than the FSx methods. The best performing method, in terms of computational resource reduction, is the GS3 approach. On the identification accuracy measure, the FSx methods performed better than most of the GSx methods. The best performing method, on the identification accuracy, is the FS2 method. As in the 1st test it appears that for this test problem FS2 method should be preferred when accuracy is the most important criterion.

6.3 Test Problem #3

This test problem introduces low difficulty level in obtaining the true-superior-concepts. The true-superior-concepts were defined using ZDT4 with the lowest available resolution in the decision-space (see Table 2). Namely, the true-superior-concepts are expected to be found easily during the evolutionary search. Figure 14 illustrates the Pareto-fronts of the concept-configuration of this test-problem. The black and grey fronts are associated with the true-superior-concepts and the rest of the concepts, respectively.

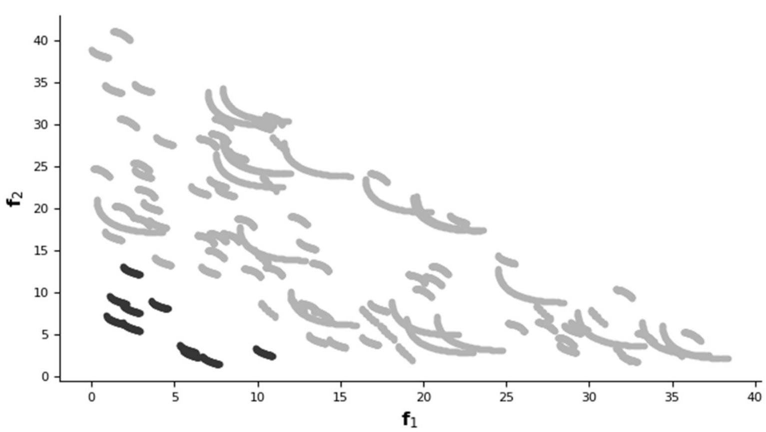

Figure 14: Pareto-fronts of test problem #3

Results:

Figure 15 compares the identification accuracy performance of the following methods of evolution: GS1, GS2 and GS3. The black, grey and white bars represent the obtained accuracies of the GS1, GS2 and GS3 methods, respectively.

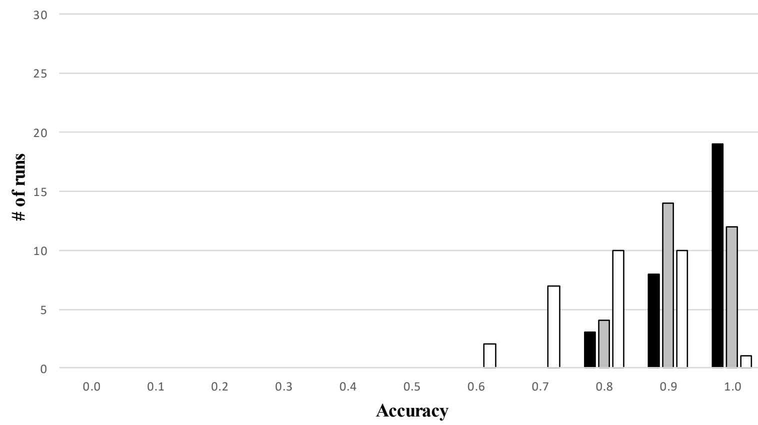

Figure 15: Identification accuracy distribution of GS1, GS2 and GS3

Figure 16 compares the identification accuracy performance of the following methods of evolution: FS1, FS2 and FS3. The black, grey and white bars represent the obtained accuracies of the FS1, FS2 and FS3 methods, respectively.

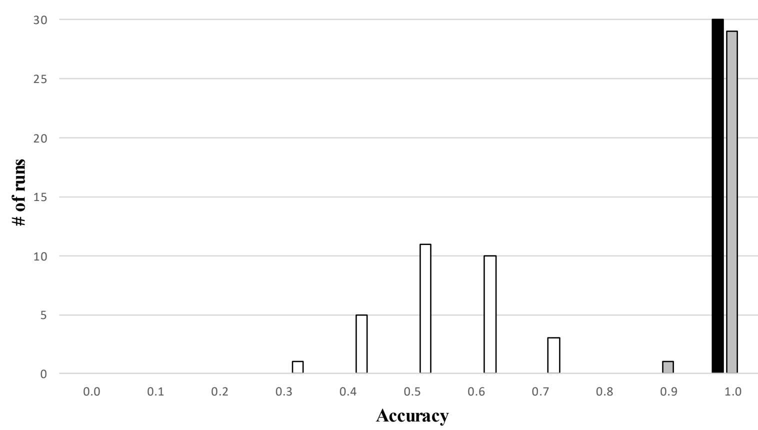

Figure 16: Identification accuracy distribution of FS1, FS2 and FS3

Table 6 compares the performance of the various methods of evolution, which are examined in this test problem. It includes their averaged identification accuracy and the averaged evaluation count over the runs.

Table 6: Performance comparison of the examined methods of evolution

Method of evolution	Average evaluation count	Average identification accuracy
GR0	395,120 (100%)	1.000 ± 0.000
GS1	69,941 (17.7%)	0.953 ± 0.068
GS2	57,129 (14.5%)	0.927 ± 0.069
GS3	51,055 (12.9%)	0.803 ± 0.100
FR0	393,003 (100%)	1.000 ± 0.000
FS1	53,895 (13.7%)	1.000 ± 0.000
FS2	53,352 (13.6%)	0.997 ± 0.018
FS3	104,804 (26.7%)	0.530 ± 0.099

While all methods achieved substantial computational resource reduction, most of the FSx methods performed better than the GSx methods. Yet, the best method, in terms of the computational resource reduction, is the GS3 approach. On the identification accuracy measure, most FSx methods performed better than the GSx methods. The best performing methods, on the identification accuracy, are the FS1, and the FS2 methods, with a minor difference. Based on the results of this test, it appears that eitherFS1 or FS2 when accuracy is the most important criterion.

6.4 Test Problem #4

This test problem introduces high difficulty level in obtaining the true-superior-concepts. The true-superior-concepts were defined using ZDT4 with the highest available resolution in the decision-space (see Table 2). In contrast to previous test problems, here and in the following tests, which belong to the second group of tests, the true-superior-concepts are the concepts belonging to the s-Pareto [4]. Figure 17 illustrates the Pareto-fronts of the concept configuration of this test problem. The black and grey fronts are associated with the true-superior-concepts and the rest of the concepts, respectively.

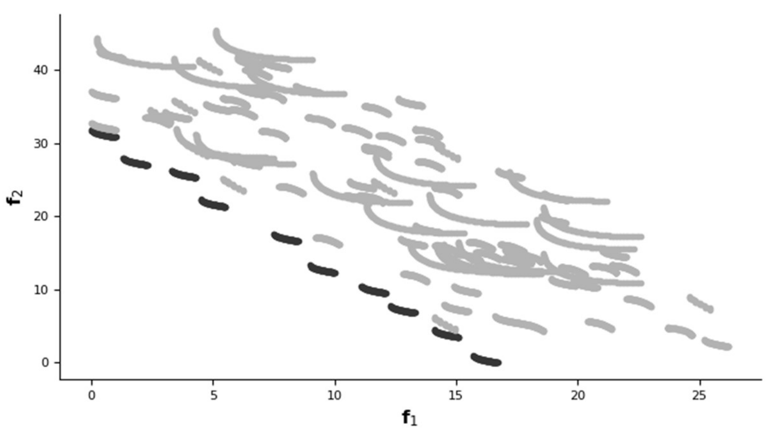

Figure 17: Pareto-fronts of test problem #4

Results:

Figure 18 compares the identification accuracy performance of the following methods of evolution: GN1 and GS2. The black and white bars represent the obtained accuracies of the GN1 and GN2 methods, respectively.

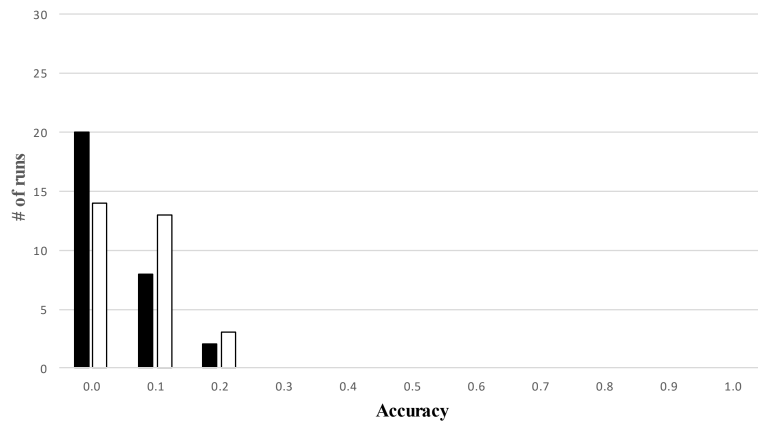

Figure 18: Identification accuracy distribution of GN1 and GN2

Figure 19 compares the identification accuracy performance of the following methods of evolution: FN1 and FS2. The black and white bars represent the obtained accuracies of the FN1 and FN2 methods, respectively.

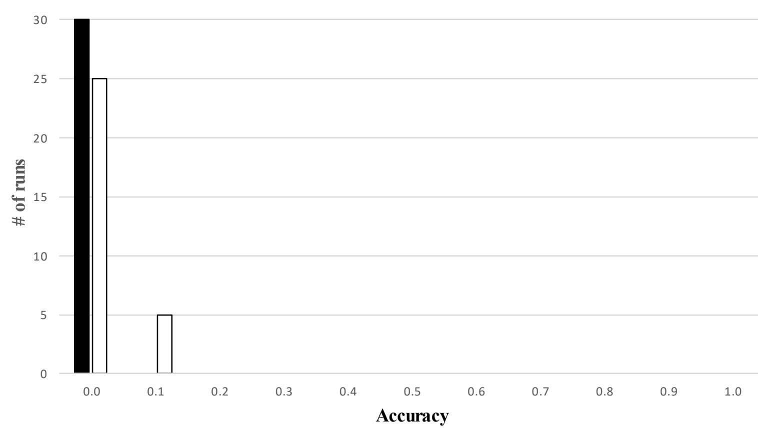

Figure 19: Identification accuracy distribution of FN1 and FN2

Table 7 compares the performance of the various methods of evolution, which are examined in this test problem. It includes the averaged identification accuracy and the averaged evaluation count over the runs.

Table 7: Performance comparison of the examined methods of evolution

Method of evolution	Average evaluation count	Average identification accuracy
GR0	559,335 (100%)	1.000 ± 0.000
GN1	93,794 (16.8%)	0.043 ± 0.068
GN2	98,826 (17.7%)	0.063 ± 0.067
FR0	563,508 (100%)	1.000 ± 0.000
FN1	99,639 (17.7%)	0.000 ± 0.000
FN2	118,621 (21.1%)	0.017 ± 0.038

According to the table, the best performing method, in terms of the computational resource reduction, is the GN1 approach. None of the methods performed well in this hard test in terms of the identification accuracy. Still, from this viewpoint GN2 method is the best.

6.5 Test Problem #5

This test problem introduces medium difficulty level in obtaining the true-superior-concepts. The true-superior-concepts were defined using ZDT4, where half of them have the highest available resolution in the decision-space, while the other half has the lowest available resolution in the decision-space (see Table 2). As in the previous test problem, the true-superior-concepts are the concepts belonging to the s-Pareto. Figure 20 illustrates the Pareto-fronts of the concept configuration of this test problem. The black and grey fronts are associated with the true-superior-concepts and the rest of the concepts, respectively.

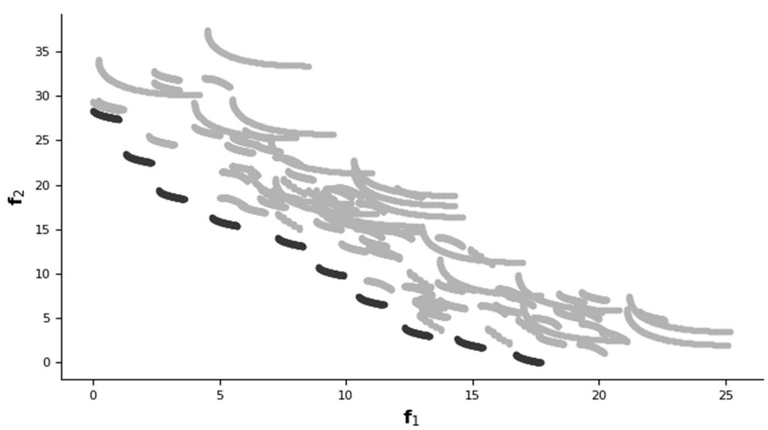

Figure 20: Pareto-fronts of test problem #5

Results:

Figure 21 compares the identification accuracy performance of the following methods of evolution: GN1 and GN2. The black and white bars represent the obtained accuracies of the GN1 and GN2 methods, respectively.

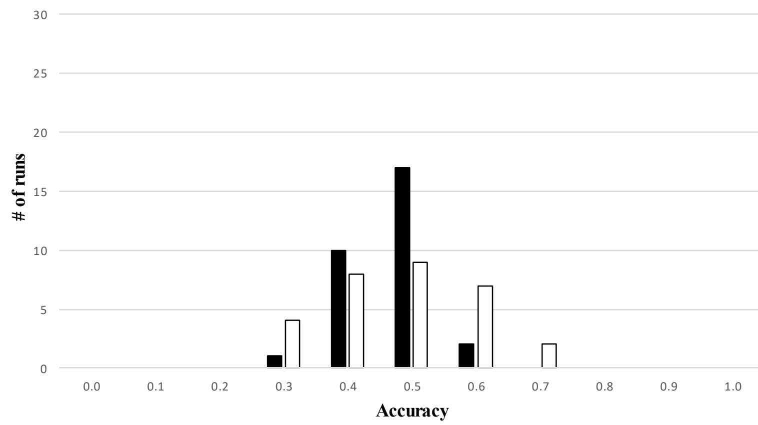

Figure 21: Identification accuracy distribution of GN1 and GN2

Figure 22 compares the identification accuracy performance of the following methods of evolution: FN1 and FN2. The black and white bars represent the obtained accuracies of the FN1 and FN2 methods, respectively.

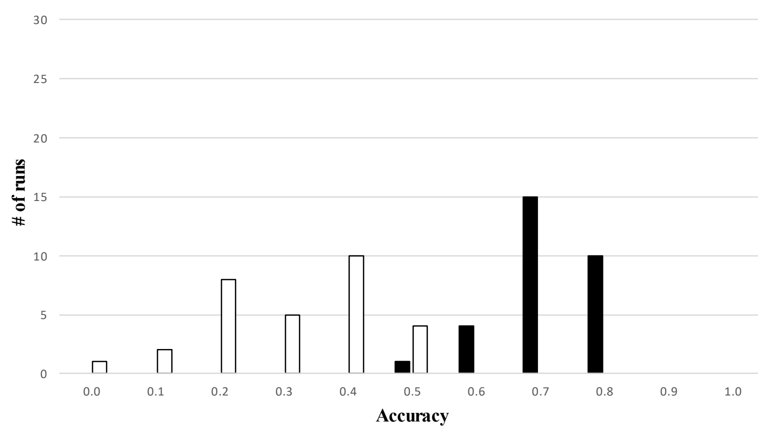

Figure 22: Identification accuracy distribution of FN1 and FN2

Table 8 compares the performance of the various methods of evolution, which are examined in this test problem. These methods are compared based on their averaged identification accuracy and averaged evaluation count over the runs.

Table 8: Performance comparison of the examined methods of evolution

Method of evolution	Average evaluation count	Average identification accuracy
GR0	458,099 (100%)	1.000 ± 0.000
GN1	81,210 (17.7%)	0.467 ± 0.066
GN2	85,080 (18.6%)	0.483 ± 0.115
FR0	479,468 (100%)	1.000 ± 0.000
FN1	98,285 (20.5%)	0.713 ± 0.078
FN2	153,552 (32.0%)	0.310 ± 0.132

According to the table, the best performing method, in terms of computational effort reduction, is the GN1 approach, and the best performing method, on the identification accuracy, is the FN1 method. Based on the results of this test problem FN1 should be preferred when accuracy is the most important criterion.

6.6 Test Problem #6

This test problem introduces low difficulty level in obtaining the true-superior-concepts. The true-superior-concepts were defined using ZDT4 with the lowest available resolution in the decision-space (see Table 2). Figure 23 illustrates the Pareto-fronts of the concept configuration of this test problem. The black and grey fronts are associated with the true-superior-concepts and the rest of the concepts, respectively.

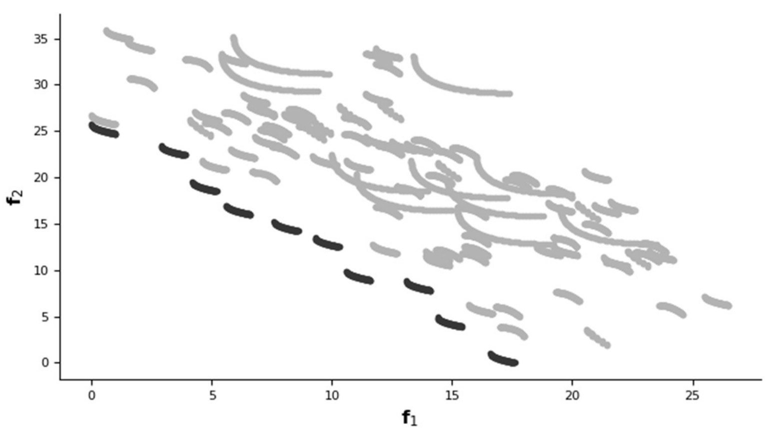

Figure 23: Pareto-fronts of test problem #6

Results:

Figure 24 compares the identification accuracy performance of the following methods of evolution: GN1 and GN2. The black and white bars represent the obtained accuracies of the GN1 and GN2 methods, respectively.

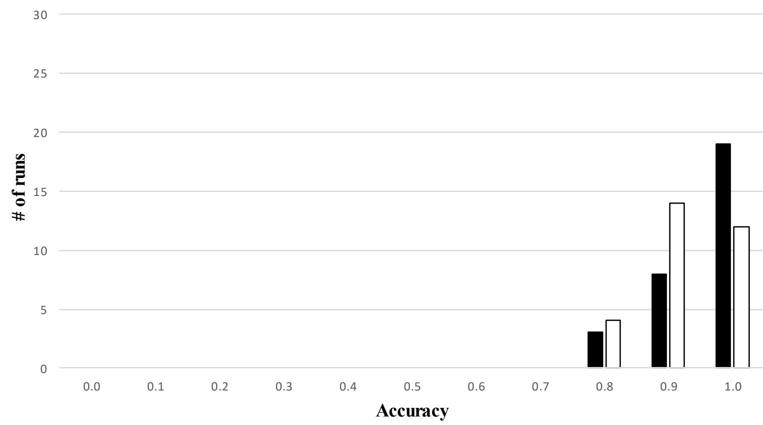

Figure 24: Identification accuracy distribution of GN1 and GN2

Figure 25 compares the identification accuracy performance of the following methods of evolution: FN1 and FN2. The black and white bars represent the obtained accuracies of the FN1 and FN2 methods, respectively.

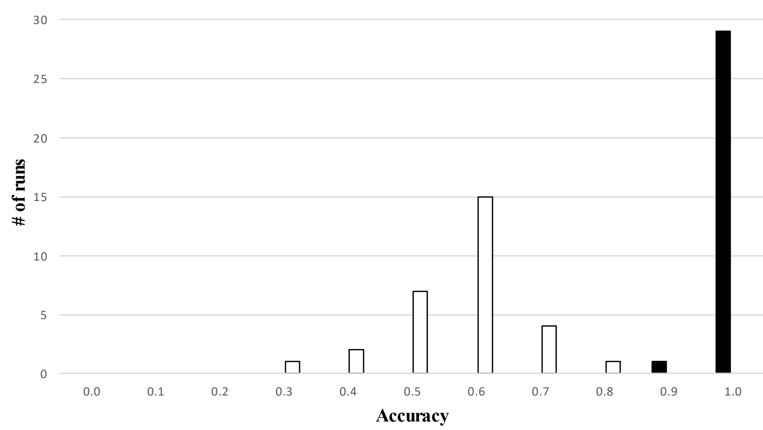

Figure 25: Identification accuracy distribution of FN1 and FN2

Table 9 compares the performance of the various methods of evolution, which are examined in this test problem. These methods are compared based on their average identification accuracy and the averaged evaluation count over the runs.

Table 9: Performance comparison of the examined methods of evolution

Method of evolution	Average evaluation count	Average identification accuracy
GR0	507,512 (100%)	1.000 ± 0.000
GN1	70,124 (13.8%)	0.953 ± 0.068
GN2	81,628 (16.1%)	0.927 ± 0.069
FR0	505,354 (100%)	1.000 ± 0.000
FN1	63,951 (12.7%)	0.997 ± 0.018
FN2	109,461 (21.7%)	0.573 ± 0.101

According to the results, the best performing method, in terms of the computational effort reduction, is the FN1 approach. The best method regarding the identification accuracy, is also the FN1 method. Hence, the best performing method of this test problem is the FN1 method.

7 TOWARDS INTERACTIVE SEARCH

The main goal of this chapter is to briefly describe an envisioned interactive evolutionary procedure for searches that involve many SBCs. The envisioned procedure is expected to be founded on the proposed evolutionary search framework, with a small modification to allow human intervention during the search. It is also expected that the techniques and measure, which have been studied in this thesis, will play a major role in the envisioned interactive procedure. Section 7.1 provides some general background on Interactive Evolutionary Computation (IEC), whereas Section 7.2 outlines the envisioned approach.

7.1 Background on interactive evolutionary computation

Interactive Evolutionary Computation (IEC) is an optimization method that is based on a cooperation (or interaction) between human (s) and computers to perform an evolutionary computation in order to solve an optimization problem [42]. Commonly, IEC involves an evolutionary algorithm whose performance function is replaced by evaluations by a human user. Yet, as suggested in [42], there are also IEC techniques in which other versions of interaction exists. One such example is in fact the case that is considered in the following.

In IEC, the evaluation process is influenced by the user's tacit knowledge, preferences and heuristic understanding. According to [43], the evaluation process relies on user preference information that ranges from implicit to explicit. Implicit preferences may be tacit and difficult for humans to articulate. Since implicit memory is a type of memory in which previous experiences aid the performance of a task without conscious awareness of the previous experience. While, explicit preferences are readily articulated by users and their relevance to evolutionary computation is typically well understood by the users.

IEC has some downsides [42] such as human fatigue. Since the human cooperates with a tireless computer and continuously evaluates individual solutions, the IEC process cannot be continued for many generations, which restrict the practical use of IEC. As a result of human fatigue considerations, often IEC has to search for a goal with a small population size and with few generations.

While considered as belonging to IEC, the following interactive scheme is substantially different than classical IEC studies. This is because, here the DMs are involved not in the evaluation of individual solutions but rather in the evaluation of sub-sets of solutions, namely concepts.

7.2 The envisioned interactive procedure

Figure 26 outlines the envisioned interactive process, based on the measures and techniques that have been studied in this thesis. First, prior to the search, the DMs define the concepts and initialize the required parameters for the evolutionary process. Then, an evolutionary algorithm searches for better solutions in an iterative process.

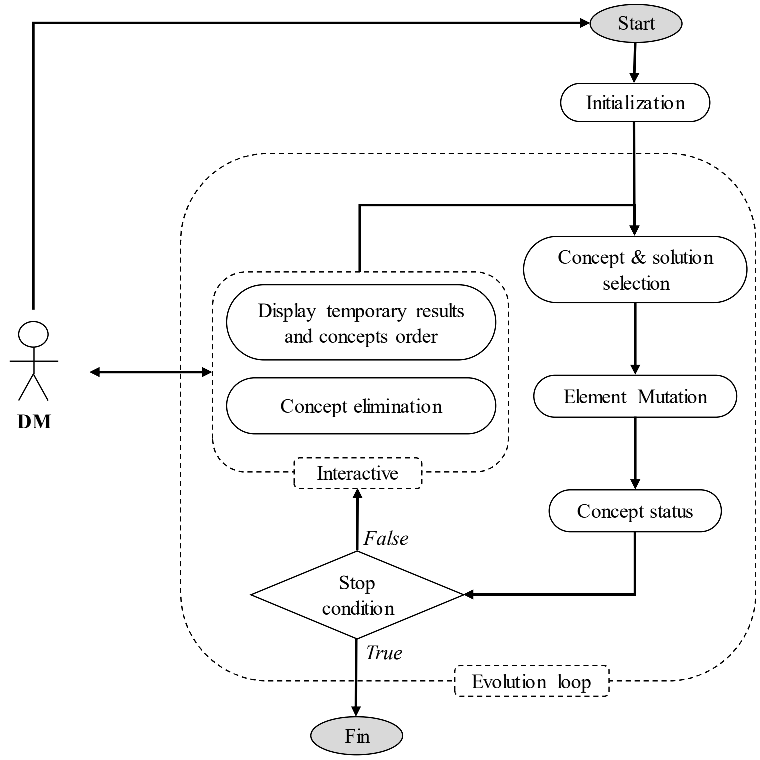

Figure 26: Scheme of the envisioned interactive search

The DMs may interrupt the evolutionary process at any-time to decide which concepts are to be eliminated according to their current and estimated performances. To support decision-making, the DMs are provided with useful information. It consists of the performances of all the non-dominated sets of individual solutions of the concepts of C_{red}, and the current concepts' ordering. Then, the DMs decides on issues such as the number of concepts to be eliminated, artificial reduction or enhancement of performances of concepts, and the current technique to be used till next intervention.

8 SUMMARY & CONCLUSIONS

8.1 Summary

This thesis deals with a preliminary evolutionary search process of SBCs. It is motivated by the need to rationalize the allocation of computational resources during the evolution of SBCs, with many concepts. The current work investigates several methods for on-line intelligent elimination of concepts. Such eliminations aim to produce significant reduction of the required computational resources when trying to find the superior concepts.

Table 10 summarizes the identification accuracies of the best methods of evolution, when accuracy is the most important criterion.

Table 10: Best results summary

Test problem #	Best method of evolution	Average evaluation count	Average Identification accuracy
1	FS2	132,395 (25.7%)	0.383 ± 0.146
2	FS2	92,868 (19.4%)	0.680 ± 0.116
3	FS1, FS2	53,895 (13.7%), 53,352 (13.6%)	1.000 ± 0.000
4	GN2	98,826 (17.7%)	0.063 ± 0.067
5	FN1	98,285 (20.5%)	0.713 ± 0.078
6	FN1	63,951 (12.7%)	0.997 ± 0.018

8.2 Conclusions

As seen from Table 10, some methods performed better than others in different cases. This corresponds with the well-known optimization theorem of no-free lunch [44]. In most test problems, the methods based on the fair resource allocation heuristic are preferred. This is because the fair heuristic does not concern the concept's-progress to select a concept for evaluation. Rather, it allows the concepts to evolve simultaneously, preventing a competition for computational resources among the evolving concepts. In contrast to the fair approach, the greedy heuristic relies on the concept-progress to select a concept for evaluation, by favoring concepts with high concept-progress value. Namely, the evolving concepts are in a state of competition for computational resources. Therefore, it may take a while for concepts with low concept-progress to receive a computational resource. Thus, when using the greedy approach, the ordering and elimination may result in false elimination of superior concepts.

As further seen in Table 10, the FS2 method is the best method among the scalar ones (within the first three test problems). This shows that estimation of the future order of the concepts allows intelligent on-line concept elimination.

In the second group of test problems, the FN1 method was preferred in two out of the three cases. As in the first group, the fair resource allocation heuristic allows the concepts to evolve simultaneously, without competition for computational resources between the evolving concepts. The exception here is in the 4th test problem, in which the GN2 method was preferred. One possible explanation is that, in the ordering procedure, too-many concepts where in the first subset of the ordered set of sets O. Hence, the superior concepts were chosen randomly from the first subset of O. It should be noted that although preferred for the 4^{th} test case, the GN2 produced very poor results, hence it cannot be considered as reliable for such hard problems.

Finally, all investigated methods of evolution have reduced the required computational resources. This is trivial, due to the nature of the elimination process. With fewer concepts to explore, a reduction of computational resources must be expected. In most of the test problems the reduction in the required computational resources was significant (about 80%).

8.3 Future work

Despite the satisfying results, there is still a substantial work to be done towards the envisioned interactive search. Among such additional studies are:

1. Develop and investigate more methods of evolution.
2. Develop the means for on-line estimation of the problem difficulty level.
3. Use an ensemble of methods of evolution, based on the on-line estimation of the difficulty level of the solved problem.

9 REFERENCES

[1] J. Pine, “Mass customizing products and services,” *Strateg. Leadersh.*, vol. 21, pp. 6–55, 1993.

[2] J. J. Cristiano, C. C. White, and J. K. Liker, “Application of multiattribute decision analysis to quality function deployment for target setting,” *IEEE Trans. Syst. Man Cybern. Part C (Applications Rev.*, vol. 31, no. 3, pp. 366–382, 2001.

[3] W. Hsu and B. Liu, “Conceptual design: issues and challenges,” *Comput. Des.*, vol. 14, no. 32, pp. 849–850, 2000.

[4] C. A. Mattson and A. Messac, “Pareto frontier based concept selection under uncertainty, with visualization,” *Optim. Eng.*, vol. 6, no. 1, pp. 85–115, 2005.

[5] J. Andersson, “Multiobjective Optimization in Engineering Design - Applications to Fluid Power Systems,” Linköpings University, 2001.

[6] G. Avigad and A. Moshaiov, “Interactive Evolutionary Multiobjective Search and Optimization of Set-Based Concepts,” *IEEE Trans. Syst. Man, Cybern. Part B Cybern.*, vol. 39, no. 4, pp. 1013–1027, 2009.

[7] A. Moshaiov, A. Snir, and B. Samina, “Concept-based Evolutionary Exploration of Design Spaces by a Resolution-Relaxation-Pareto Approach,” *IEEE Congr. Evol. Comput.*, pp. 1845–1852, 2015.

[8] A. Moshaiov, “The paradox of multimodal optimization: Concepts vs. species in single and multi-objective problems,” in *2016 IEEE Congress on Evolutionary Computation, CEC 2016*, 2016, pp. 1743–1748.

[9] E. Farhi and A. Moshaiov, “Window-of-interest based multi-objective evolutionary search for satisficing concepts,” *2017 IEEE Int. Conf. Syst. Man, Cybern. SMC 2017*, vol. 2017-January, pp. 3705–3710, 2017.

[10] A. Moshaiov and Y. Rizakov, “Signaling and visualization for interactive evolutionary search and selection of conceptual solutions,” *Proceeding fifteenth Annu. Conf. companion Genet. Evol. Comput. Conf. companion - GECCO ’13 Companion*, pp. 1359–1366, 2013.

[11] A. Snir, B. Samina, and A. Moshaiov, “Concept-based Evolutionary Multi-Criteria Exploration of Design Spaces Under Run-time Limitation,” in *Computational Intelligence, 2015 IEEE Symposium Series on*, 2015, pp. 853–860.

[12] J. L. Cohon, “Multicriteria programming: Brief review and application,” *Des. Optim.*, pp. 163–191, 1985.

[13] C.-L. Hwang and A. S. M. Masud, *Multiple objective decision making—methods and applications*. 1979.

[14] F. Hoffmeister and T. Bäck, “Genetic Algorithms and Evolution Strategies: Similarities and Differences,” *Strategies*, vol. 496, pp. 455–469, 1991.

[15] D. B. Fogel and L. J. Fogel, “An Introduction to Evolutionary Programming,” *Eur. Conf. Artif. Evol. (AE 95)*, vol. 1063, no. Goldberg 1996, pp. 21–33, 1995.

[16] J. R. Koza, “Genetic programming as a means for programming computers by natural

selection," *Stat. Comput.*, vol. 4, no. 2, pp. 87–112, 1994.

[17] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. 1989.

[18] J. Horn, N. Nafpliotis, and D. E. Goldberg, "A Niched Pareto Genetic Algorithm for Multiobjective Optimization," *Proc. First IEEE Conf. Evol. Comput.*, pp. 82–87, 1994.

[19] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, vol. 1. 1975.

[20] H.-G. Beyer, H.-G. Beyer, H.-P. Schwefel, and H.-P. Schwefel, "Evolution strategies – A comprehensive introduction," *Nat. Comput.*, vol. 1, no. 1, pp. 3–52, 2002.

[21] G. Renner and A. Ekárt, "Genetic algorithms in computer aided design," *Comput. Des.*, vol. 35, no. 8, pp. 709–726, 2003.

[22] G. Steven, O. Querin, and M. Xie, "Evolutionary structural optimisation (ESO) for combined topology and size optimisation of discrete structures," *Comput. Methods Appl. Mech. Eng.*, vol. 188, no. 4, pp. 743–754, 2000.

[23] C. A. C. Coello and A. D. Christiansen, "Two new GA-based methods for multiobjective optimization," *Civ. Eng. Environ. Syst.*, vol. 15, no. 3, pp. 207–243, 1998.

[24] F. Erbatur, O. Hasançebi, İ. Tütüncü, and H. Kılıç, "Optimal design of planar and space structures with genetic algorithms," *Comput. Struct.*, vol. 75, no. 2, pp. 209–224, 2000.

[25] M. Cerrolaza and W. Annicchiarico, "Genetic algorithms in shape optimization: finite and boundary element applications," *Evol. algorithms Eng. Comput. Sci. Chichester, Engl. John Wiley Sons*, 1999.

[26] P. Fleming and R. Purshouse, "Evolutionary algorithms in control systems engineering: a survey," *Control Eng. Pract.*, vol. 10, no. 11, pp. 1223–1241, 2002.

[27] H. B. Gurocak, "A genetic-algorithm-based method for tuning fuzzy logic controllers," *Fuzzy Sets Syst.*, vol. 108, no. 1, pp. 39–47, 1999.

[28] O. Abramovich and A. Moshaiov, "Multi-objective topology and weight evolution of neuro-controllers," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 670–677.

[29] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, and Q. Zhang, "Multiobjective evolutionary algorithms: A survey of the state of the art," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 32–49, 2011.

[30] A. Moshaiov and G. Avigad, "Concept-based multi-objective problems and their solution by EC," *GECCO '07 Proc. 2007 GECCO Conf. companion Genet. Evol. Comput.*, pp. 2865–2868, 2007.

[31] G. Avigad and A. Moshaiov, "Simultaneous concept-based evolutionary multi-objective optimization," *Appl. Soft Comput.*, vol. 11, pp. 193–207, 2011.

[32] D. Rousis, "A Pareto Frontier Intersection-Based Approach for Efficient Multiobjective Optimization of Competing Concept Alternatives," *Thesis Submitt. to Linkoping Univ. Linkoping*, 2011.

[33] G. Avigad and A. Moshaiov, "Set-based concept selection in multi-objective problems: optimality versus variability approach," *J. Eng. Des.*, vol. 20, no. 3, pp. 217–242, 2009.

[34] A. Moshaiov and G. Avigad, "The Extended Concept-based Multi-objective Path Planning and its A-life implications," in *2007 IEEE Symposium on Artificial Life*, 2007, pp. 259–265.

[35] E. Denenberg and A. Moshaiov, "Evolutionary search of optimal concepts using a relaxed-pareto-optimality approach," in *Evolutionary Computation, 2009. CEC '09. IEEE Congress on*, 2009, pp. 2524–2531.

[36] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast Elitist Multi-Objective Genetic Algorithm: {NSGA-II}," *IEEE Trans.\ Evol. Comput.*, vol. 6, pp. 182–197, 2000.

[37] A. Moshaiov and Y. Snir, "Tailoring \varepsilon-MOEA to concept-based problems," in *Parallel Problem Solving from Nature-PPSN XII*, Springer, 2012, pp. 122–131.

[38] K. Deb, M. Mohan, and S. Mishra, "Evaluating the ε-Domination Based Multi-Objective Evolutionary Algorithm for a Quick Computation of Pareto-Optimal Solutions," *Evol. Comput.*, vol. 13, no. 4, pp. 501–525, 2005.

[39] E. Denenberg, "Evolutionary Search of Near-optimal Concepts Based on Relaxed-Pareto-optimality (Unpublished master's thesis)," Tel-Aviv University, 2012.

[40] D. Hadka and P. Reed, "Borg: An auto-adaptive many-objective evolutionary computing framework," *Evol. Comput.*, vol. 21, no. 2, pp. 231–259, 2013.

[41] S. H. Walker and D. B. Duncan, "Estimation of the Probability of an Event as a Function of Several Independent Variables," *Biometrika*, vol. 54, no. 1/2, p. 167, 1967.

[42] H. Takagi, "Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation," *Proc IEEE*, vol. 89, no. 9, pp. 1275–1296, 2001.

[43] H. Aljawawdeh, C. Simons, and M. Odeh, "Metaheuristic Design Pattern: Preference," *Proc. Companion Publ. 2015 Genet. Evol. Comput. Conf.*, pp. 1257–1260, 2015.

[44] Y.-C. Ho and D. L. Pepyne, "Simple explanation of the no free lunch theorem of optimization," in *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No. 01CH37228)*, 2001, vol. 5, pp. 4409–4414.

10 APPENDIX

10.1 Test problem #1 – concepts configuration

Table 11: Details of test problem #1 concept configuration

Concept #	Fn.	n_m	Obj.-space offset	N_m	Concept #	Fn.	n_m	Obj.-space offset	N_m
1	ZDT4	10	[3.95 , 5.5]	120	51	ZDT1	30	[17 , 3.95]	100
2	ZDT4	10	[9.55 , 2.85]	200	52	ZDT2	30	[13.4 , 2.95]	80
3	ZDT4	10	[8.1 , 2.55]	160	53	ZDT4	10	[19.65 , 6.25]	100
4	ZDT4	10	[7.7 , 0.65]	160	54	ZDT3	30	[4.45 , 10.4]	100
5	ZDT4	10	[4.85 , 3.95]	180	55	ZDT4	10	[12.3 , 13.85]	80
6	ZDT4	10	[2.5 , 9.75]	100	56	SCH	1	[23.55 , 4.8]	20000
7	ZDT4	10	[6 , 3.4]	120	57	ZDT4	10	[18.2 , 7.25]	80
8	ZDT4	10	[6.45 , 7.6]	100	58	ZDT1	30	[0.7 , 19.1]	60
9	ZDT4	10	[9.6 , 4.75]	100	59	ZDT4	10	[0.85 , 26.05]	80
10	ZDT4	10	[7.35 , 0.8]	200	60	ZDT1	30	[16.3 , 4.95]	80
11	ZDT2	30	[3.05 , 23.2]	100	61	ZDT4	10	[19.7 , 4.75]	80
12	ZDT1	30	[11.55 , 13.2]	40	62	ZDT1	30	[16.9 , 11.55]	80
13	SCH	1	[4.45 , 23.5]	40000	63	ZDT3	30	[8.4 , 7.15]	100
14	ZDT2	30	[1.55 , 18.1]	40	64	ZDT6	10	[1.4 , 25]	200
15	ZDT6	10	[13.9 , 13.7]	300	65	FON	3	[16.45 , 5.05]	200
16	SCH	1	[25.3 , 1.5]	40000	66	ZDT2	30	[7.85 , 9.6]	100
17	FON	3	[19.95 , 3.55]	100	67	ZDT6	10	[12.45 , 8.75]	300
18	ZDT1	30	[7.6 , 9.15]	80	68	ZDT4	10	[13 , 9.9]	100
19	ZDT2	30	[0.95 , 19.1]	40	69	ZDT4	10	[5.4 , 19.9]	60
20	ZDT6	10	[10.3 , 12]	100	70	ZDT2	30	[11.4 , 17.05]	40
21	ZDT6	10	[3.45 , 21.7]	200	71	ZDT2	30	[11.2 , 4.4]	60
22	ZDT2	30	[11.2 , 8.8]	100	72	ZDT3	30	[15.4 , 9.55]	100
23	ZDT2	30	[8.4 , 20.55]	40	73	ZDT1	30	[6.9 , 19.55]	60
24	ZDT3	30	[17.8 , 3.5]	120	74	ZDT3	30	[4.8 , 10.5]	120
25	ZDT3	30	[0.85 , 14.65]	120	75	SCH	1	[11.95 , 13.8]	40000
26	ZDT1	30	[2.55 , 15.15]	100	76	ZDT4	10	[12.6 , 10]	100
27	ZDT6	10	[6.15 , 12]	300	77	ZDT1	30	[6.95 , 12.2]	100
28	ZDT3	30	[2.6 , 19.55]	80	78	ZDT1	30	[9.85 , 4.9]	80
29	FON	3	[12 , 12.1]	400	79	SCH	1	[12.65 , 3.35]	30000
30	ZDT2	30	[15.75 , 4.55]	60	80	ZDT6	10	[5.95 , 21]	200
31	FON	3	[9.45 , 12.75]	100	81	FON	3	[7.05 , 18.55]	200
32	ZDT2	30	[3.9 , 11.85]	80	82	ZDT6	10	[16.45 , 2.45]	100
33	ZDT6	10	[3.7 , 11.8]	400	83	FON	3	[0.8 , 15.65]	400
34	ZDT1	30	[9.8 , 16.7]	60	84	FON	3	[8.55 , 15.25]	200
35	SCH	1	[24.75 , 2]	30000	85	SCH	1	[10.35 , 8.15]	40000
36	ZDT1	30	[6.05 , 15.65]	40	86	ZDT6	10	[3.25 , 21.2]	100
37	ZDT4	10	[4.3 , 16.3]	80	87	ZDT1	30	[0.1 , 28.55]	60
38	ZDT1	30	[7.05 , 17.05]	80	88	ZDT6	10	[9.25 , 12.1]	300
39	ZDT1	30	[0.7 , 14.6]	100	89	SCH	1	[3.4 , 20.2]	20000
40	ZDT6	10	[22.35 , 4.15]	200	90	ZDT3	30	[18.15 , 6.85]	100
41	SCH	1	[6.55 , 11.85]	30000	91	FON	3	[13.2 , 15]	400
42	ZDT3	30	[16.1 , 4.5]	140	92	ZDT2	30	[10.75 , 12.95]	60
43	ZDT2	30	[11.35 , 6.15]	60	93	FON	3	[22.2 , 4.2]	100
44	SCH	1	[12 , 10.25]	30000	94	ZDT2	30	[1.7 , 24.65]	80
45	ZDT4	10	[0.55 , 15.55]	80	95	ZDT3	30	[0.7 , 19.95]	120
46	ZDT2	30	[17.7 , 9.2]	80	96	ZDT3	30	[9.75 , 12.55]	100
47	ZDT2	30	[2 , 23.5]	40	97	ZDT6	10	[12.35 , 3.8]	400
48	ZDT1	30	[10.4 , 10.6]	60	98	ZDT6	10	[14.25 , 14.05]	400
49	ZDT2	30	[8.35 , 19.8]	60	99	ZDT1	30	[6.2 , 9.95]	80
50	ZDT6	10	[3 , 14.15]	400	100	ZDT2	30	[7.45 , 10.75]	100

10.2 Test problem #2 – concepts configuration

Table 12: Details of test problem #2 concept configuration

Concept #	Fn.	n_m	Obj.-space offset	N_m	Concept #	Fn.	n_m	Obj.-space offset	N_m
1	ZDT4	10	[12.2 , 0.7]	20	51	SCH	1	[9.85 , 22.35]	20000
2	ZDT4	10	[6 , 7.55]	20	52	ZDT1	30	[23.75 , 1.2]	80
3	ZDT4	10	[7.45 , 0.4]	20	53	ZDT6	10	[8.5 , 17.3]	400
4	ZDT4	10	[5.75 , 2.85]	20	54	SCH	1	[24.55 , 4.4]	30000
5	ZDT4	10	[7.45 , 2.85]	20	55	ZDT2	30	[14.25 , 5.8]	60
6	ZDT4	10	[4.15 , 5]	120	56	SCH	1	[7.7 , 14.5]	20000
7	ZDT4	10	[1.4 , 5.9]	160	57	ZDT3	30	[20.6 , 3.55]	120
8	ZDT4	10	[2.7 , 6.4]	160	58	ZDT3	30	[1.7 , 16.3]	120
9	ZDT4	10	[1.4 , 6.2]	120	59	SCH	1	[15.55 , 12.45]	30000
10	ZDT4	10	[4.35 , 4]	120	60	SCH	1	[18.1 , 1.8]	40000
11	ZDT6	10	[0.75 , 18.35]	100	61	ZDT2	30	[9.3 , 18.6]	60
12	ZDT3	30	[3.85 , 13.8]	140	62	ZDT3	30	[2.75 , 23.9]	100
13	ZDT1	30	[14.65 , 4.7]	80	63	ZDT4	10	[16.7 , 4.25]	40
14	ZDT4	10	[6.55 , 18.6]	40	64	ZDT1	30	[8.4 , 8.6]	60
15	SCH	1	[5.7 , 27.35]	30000	65	SCH	1	[1.9 , 29.65]	30000
16	ZDT2	30	[6.35 , 25.55]	100	66	SCH	1	[17.2 , 10.85]	20000
17	ZDT2	30	[26.85 , 3.9]	40	67	ZDT2	30	[16.3 , 3.8]	100
18	SCH	1	[10.45 , 14.9]	40000	68	ZDT4	10	[8.25 , 17.25]	40
19	ZDT4	10	[20.1 , 11.7]	100	69	SCH	1	[10.25 , 15.1]	30000
20	ZDT3	30	[8.1 , 20.75]	100	70	ZDT3	30	[18.3 , 12]	120
21	ZDT3	30	[18.75 , 5.15]	100	71	ZDT1	30	[5.4 , 13]	60
22	SCH	1	[22.7 , 3.6]	40000	72	ZDT1	30	[3.9 , 15.45]	40
23	SCH	1	[5.35 , 10.7]	30000	73	SCH	1	[12.85 , 19.65]	30000
24	ZDT4	10	[18.6 , 12.55]	60	74	ZDT6	10	[16 , 13.6]	300
25	ZDT3	30	[15.15 , 3.95]	80	75	FON	3	[15.05 , 9.2]	300
26	ZDT2	30	[11.35 , 12.95]	60	76	ZDT4	10	[4.95 , 18.8]	80
27	ZDT1	30	[13.7 , 15.85]	100	77	SCH	1	[8.55 , 12.55]	40000
28	ZDT3	30	[9.65 , 16.1]	100	78	SCH	1	[24.2 , 8.1]	40000
29	ZDT4	10	[25.95 , 4.55]	80	79	ZDT1	30	[15.05 , 5.2]	60
30	SCH	1	[7.75 , 13.05]	30000	80	ZDT1	30	[11.95 , 13.55]	60
31	ZDT6	10	[23.6 , 2.9]	300	81	ZDT6	10	[15.9 , 15.25]	100
32	ZDT2	30	[4.05 , 12.65]	100	82	ZDT6	10	[14.2 , 3.3]	200
33	ZDT1	30	[24.9 , 3.35]	80	83	SCH	1	[23.95 , 4.95]	20000
34	ZDT6	10	[4.05 , 17.35]	200	84	ZDT2	30	[23 , 1.4]	100
35	ZDT2	30	[15.35 , 11.2]	60	85	ZDT6	10	[18.5 , 5.15]	200
36	FON	3	[13.9 , 10.3]	300	86	ZDT2	30	[11.05 , 13.1]	80
37	ZDT2	30	[14.35 , 3.8]	80	87	ZDT2	30	[6.1 , 24.75]	60
38	ZDT1	30	[0.9 , 24.65]	60	88	ZDT2	30	[8.85 , 22.8]	80
39	ZDT3	30	[21.95 , 11.1]	80	89	ZDT2	30	[11 , 20.95]	60
40	ZDT6	10	[4.8 , 11.7]	100	90	ZDT3	30	[15.95 , 8.3]	100
41	ZDT2	30	[16.05 , 9.5]	60	91	FON	3	[2.75 , 18.1]	100
42	ZDT1	30	[6.1 , 21.05]	100	92	ZDT1	30	[7.85 , 8]	100
43	ZDT4	10	[5 , 21.55]	100	93	ZDT2	30	[18.9 , 12.25]	40
44	SCH	1	[8.8 , 17.9]	30000	94	ZDT1	30	[2.35 , 15.2]	100
45	ZDT3	30	[11.6 , 5.5]	100	95	ZDT3	30	[11.85 , 9]	80
46	ZDT6	10	[1.9 , 21.85]	300	96	FON	3	[5.1 , 25.4]	100
47	ZDT6	10	[12.55 , 13.4]	200	97	SCH	1	[25.8 , 1.65]	40000
48	ZDT3	30	[5.85 , 11.2]	120	98	ZDT6	10	[1.6 , 29.55]	400
49	ZDT4	10	[8.5 , 9.4]	80	99	ZDT6	10	[10.7 , 11.3]	300
50	ZDT3	30	[3.1 , 22.1]	120	100	ZDT6	10	[10.45 , 8.7]	100

10.3 Test problem #3 – concepts configuration

Table 13: Details of test problem #3 concept configuration

Concept #	Fn.	n_m	Obj.-space offset	N_m	Concept #	Fn.	n_m	Obj.-space offset	N_m
1	ZDT4	10	[5.5 , 2.9]	20	51	ZDT6	10	[7.3 , 29.5]	100
2	ZDT4	10	[1.85 , 5.35]	20	52	ZDT4	10	[28.9 , 5.5]	60
3	ZDT4	10	[5.4 , 2.3]	20	53	ZDT6	10	[26.9 , 5.4]	100
4	ZDT4	10	[2.05 , 7.8]	20	54	ZDT2	30	[19.5 , 9.55]	40
5	ZDT4	10	[3.55 , 7.95]	20	55	ZDT3	30	[12.15 , 8.25]	80
6	ZDT4	10	[6.8 , 1.6]	20	56	ZDT2	30	[1.65 , 29.65]	80
7	ZDT4	10	[1.1 , 8.6]	20	57	SCH	1	[16.6 , 19.45]	30000
8	ZDT4	10	[0.7 , 6.4]	20	58	SCH	1	[7.2 , 29.95]	20000
9	ZDT4	10	[2.1 , 12.3]	20	59	ZDT4	10	[16.35 , 3.75]	40
10	ZDT4	10	[9.95 , 2.3]	20	60	ZDT2	30	[20.45 , 12.15]	60
11	ZDT2	30	[32.9 , 4]	40	61	ZDT2	30	[7.05 , 16.3]	80
12	ZDT4	10	[5.85 , 21.6]	80	62	ZDT3	30	[18.7 , 2.8]	100
13	ZDT1	30	[9.8 , 29.45]	60	63	FON	3	[13.5 , 12.75]	300
14	ZDT6	10	[16.65 , 23.4]	300	64	SCH	1	[33.15 , 2.5]	40000
15	ZDT4	10	[28.45 , 5.15]	100	65	ZDT1	30	[6.6 , 12.3]	40
16	ZDT2	30	[1.15 , 39.9]	60	66	SCH	1	[24.5 , 8.95]	40000
17	SCH	1	[11.55 , 23.75]	40000	67	ZDT6	10	[20.3 , 11.05]	400
18	ZDT3	30	[30 , 6.9]	80	68	SCH	1	[34.5 , 2.2]	30000
19	SCH	1	[8.7 , 13.9]	20000	69	ZDT4	10	[2.9 , 19.9]	80
20	ZDT1	30	[12.55 , 15.3]	60	70	ZDT6	10	[12.2 , 18.15]	400
21	ZDT1	30	[21.75 , 18.1]	60	71	FON	3	[25.05 , 5.6]	100
22	ZDT1	30	[3.45 , 17.9]	100	72	ZDT2	30	[13.2 , 6.7]	60
23	ZDT1	30	[13.05 , 4.2]	80	73	SCH	1	[19.55 , 17.45]	30000
24	ZDT4	10	[24.35 , 13.3]	80	74	SCH	1	[7.8 , 24]	30000
25	ZDT1	30	[28.35 , 2.9]	80	75	FON	3	[9 , 17.9]	300
26	SCH	1	[19.7 , 17.7]	20000	76	ZDT3	30	[10.35 , 23]	100
27	ZDT1	30	[4 , 27.75]	100	77	ZDT4	10	[14.45 , 3.55]	40
28	FON	3	[2.3 , 17.85]	400	78	FON	3	[6.35 , 15.85]	100
29	SCH	1	[7.7 , 22.45]	40000	79	ZDT6	10	[2.65 , 24.45]	200
30	FON	3	[19.1 , 11.2]	100	80	FON	3	[12.55 , 7.8]	200
31	ZDT1	30	[16.7 , 7.75]	40	81	FON	3	[7.9 , 16.2]	200
32	SCH	1	[21 , 3.2]	30000	82	SCH	1	[19 , 2.9]	30000
33	FON	3	[9.15 , 12.05]	300	83	FON	3	[10.4 , 12.1]	200
34	FON	3	[7.2 , 28.15]	400	84	SCH	1	[0.15 , 17.05]	40000
35	FON	3	[1.55 , 19.6]	300	85	SCH	1	[29.2 , 3.75]	20000
36	ZDT1	30	[2.65 , 23.9]	60	86	ZDT4	10	[2.4 , 34.1]	100
37	ZDT4	10	[1 , 33.95]	40	87	ZDT6	10	[7.2 , 14.1]	300
38	ZDT6	10	[31.55 , 9.6]	200	88	ZDT3	30	[26.8 , 7.6]	120
39	ZDT3	30	[17.55 , 5.05]	80	89	ZDT6	10	[0.2 , 23.75]	200
40	FON	3	[6.7 , 27.55]	400	90	SCH	1	[18 , 4.8]	40000
41	FON	3	[10.6 , 29.95]	300	91	ZDT1	30	[7.3 , 22.4]	80
42	ZDT3	30	[31.65 , 2.75]	100	92	ZDT3	30	[9.6 , 14.1]	100
43	ZDT1	30	[0.95 , 16.2]	80	93	ZDT1	30	[7.7 , 21.3]	100
44	ZDT1	30	[31.95 , 1.8]	100	94	ZDT3	30	[10.25 , 7.9]	100
45	SCH	1	[7.95 , 30.45]	40000	95	FON	3	[2.7 , 21.45]	100
46	SCH	1	[8 , 24.1]	40000	96	ZDT1	30	[8.5 , 25.85]	60
47	SCH	1	[12.2 , 6.3]	20000	97	ZDT6	10	[34.55 , 2.6]	200
48	ZDT4	10	[0.15 , 38.2]	40	98	ZDT3	30	[16.15 , 7.15]	100
49	ZDT2	30	[35.55 , 4.35]	60	99	ZDT2	30	[27.9 , 3.9]	40
50	ZDT3	30	[11 , 27.75]	80	100	ZDT1	30	[3.65 , 13.3]	60

10.4 Test problem #4 – concepts configuration

Table 14: Details of test problem #4 concept configuration

Concept #	Fn.	n_m	Obj.-space offset	N_m	Concept #	Fn.	n_m	Obj.-space offset	N_m
1	ZDT4	10	[0 , 30.8]	500	51	FON	3	[5.5 , 35]	100
2	ZDT4	10	[1.1 , 26.8]	500	52	ZDT6	10	[7.7 , 23.2]	200
3	ZDT4	10	[3.3 , 25.5]	500	53	ZDT2	30	[20.35 , 4.45]	60
4	ZDT4	10	[4.3 , 21.15]	500	54	SCH	1	[3.65 , 28.15]	30000
5	ZDT4	10	[7.5 , 16.75]	500	55	ZDT1	30	[3.15 , 33.4]	60
6	ZDT4	10	[9.05 , 12.25]	500	56	ZDT2	30	[17.8 , 4.25]	100
7	ZDT4	10	[10.9 , 9.55]	500	57	ZDT1	30	[16.6 , 5.45]	60
8	ZDT4	10	[12.5 , 6.7]	500	58	SCH	1	[0 , 40.2]	20000
9	ZDT4	10	[14.15 , 3.45]	500	59	SCH	1	[14.35 , 12.05]	30000
10	ZDT4	10	[15.6 , 0.05]	500	60	ZDT4	10	[13.15 , 17.75]	60
11	ZDT1	30	[0 , 31.8]	60	61	ZDT1	30	[21.05 , 14.4]	80
12	ZDT2	30	[22.05 , 7.9]	40	62	SCH	1	[13.95 , 19]	40000
13	ZDT4	10	[7.05 , 40.3]	80	63	SCH	1	[15.05 , 12.65]	40000
14	ZDT4	10	[16.9 , 25.2]	40	64	SCH	1	[18.5 , 17.3]	40000
15	ZDT6	10	[10.4 , 30.95]	200	65	SCH	1	[5 , 41.5]	30000
16	ZDT4	10	[12.8 , 34.95]	80	66	SCH	1	[11.55 , 24.3]	30000
17	ZDT2	30	[21.3 , 12.4]	40	67	SCH	1	[3.3 , 37.7]	30000
18	ZDT3	30	[14.1 , 5.35]	140	68	ZDT6	10	[11.15 , 27.95]	200
19	ZDT2	30	[10.2 , 21.8]	100	69	SCH	1	[18.8 , 10.75]	40000
20	SCH	1	[18.45 , 15.4]	40000	70	FON	3	[7.1 , 30.85]	100
21	ZDT1	30	[14.8 , 9.35]	100	71	SCH	1	[4.45 , 27]	40000
22	ZDT1	30	[8.6 , 36.7]	80	72	ZDT6	10	[11.75 , 30.05]	400
23	ZDT1	30	[5.85 , 26.7]	60	73	ZDT2	30	[14.2 , 15.15]	40
24	ZDT6	10	[17 , 15.1]	400	74	ZDT1	30	[0.3 , 41.75]	40
25	FON	3	[23.7 , 3.75]	200	75	ZDT6	10	[11.1 , 33.9]	300
26	ZDT1	30	[16.95 , 14.65]	60	76	ZDT6	10	[5.9 , 33.75]	100
27	ZDT6	10	[19.15 , 12.1]	200	77	ZDT1	30	[4.6 , 34.2]	100
28	FON	3	[11 , 28.35]	100	78	FON	3	[17.15 , 13.65]	200
29	FON	3	[20.4 , 12.35]	100	79	ZDT1	30	[19.6 , 10.25]	40
30	ZDT6	10	[16.9 , 12.95]	100	80	ZDT3	30	[5.5 , 24.3]	80
31	FON	3	[17.35 , 13.7]	100	81	ZDT2	30	[13.25 , 26.6]	60
32	ZDT3	30	[2.2 , 33.45]	80	82	ZDT1	30	[5.85 , 40.6]	100
33	ZDT1	30	[18.65 , 22.1]	40	83	FON	3	[11 , 21.9]	200
34	FON	3	[13.2 , 30.85]	300	84	SCH	1	[9.05 , 21.95]	40000
35	ZDT1	30	[10.5 , 24]	80	85	ZDT2	30	[2.1 , 32.6]	60
36	ZDT6	10	[15.95 , 14.15]	400	86	SCH	1	[6.25 , 36.7]	40000
37	ZDT3	30	[4.2 , 40.5]	100	87	ZDT1	30	[25.25 , 2.05]	40
38	SCH	1	[14.45 , 12.15]	20000	88	ZDT4	10	[18.9 , 10.6]	80
39	FON	3	[4.75 , 27.55]	300	89	FON	3	[15 , 13.8]	400
40	FON	3	[6.9 , 35.75]	300	90	ZDT2	30	[9.4 , 16.1]	80
41	ZDT3	30	[3.6 , 34.75]	140	91	ZDT3	30	[11.5 , 23.75]	100
42	FON	3	[8.7 , 32.45]	100	92	SCH	1	[13 , 12.4]	40000
43	ZDT4	10	[-0.05 , 35.9]	60	93	ZDT1	30	[18.5 , 19.25]	60
44	ZDT1	30	[14.65 , 7.2]	100	94	ZDT6	10	[13.55 , 29.55]	300
45	ZDT4	10	[12.85 , 15.8]	40	95	ZDT1	30	[6.05 , 36.75]	100
46	ZDT3	30	[14 , 28.75]	80	96	SCH	1	[17.35 , 21.95]	30000
47	ZDT2	30	[6.1 , 39]	80	97	ZDT3	30	[24.4 , 8.2]	120
48	ZDT6	10	[12.95 , 11.05]	100	98	ZDT2	30	[15.55 , 15.3]	80
49	ZDT4	10	[6.2 , 40.45]	40	99	ZDT3	30	[4.05 , 29.2]	120
50	SCH	1	[11.4 , 17.65]	30000	100	FON	3	[14.3 , 23.05]	100

10.5 Test problem #5 – concepts configuration

Table 15: Details of test problem #5 concept configuration

Concept #	Fn.	n_m	Obj.-space offset	N_m	Concept #	Fn.	n_m	Obj.-space offset	N_m
1	ZDT4	10	[12.1 , 0.8]	20	51	SCH	1	[10.2 , 22.2]	20000
2	ZDT4	10	[6.1 , 7.3]	20	52	ZDT1	30	[23.7 , 1.25]	80
3	ZDT4	10	[7.25 , 0.6]	20	53	ZDT6	10	[8.7 , 17.6]	400
4	ZDT4	10	[6.1 , 2.6]	20	54	SCH	1	[24.55 , 4.7]	30000
5	ZDT4	10	[7.65 , 2.6]	20	55	ZDT2	30	[14.25 , 6.1]	60
6	ZDT4	10	[3.9 , 5]	120	56	SCH	1	[7.65 , 14.35]	20000
7	ZDT4	10	[1.4 , 5.5]	160	57	ZDT3	30	[20.45 , 3.7]	120
8	ZDT4	10	[2.8 , 6.3]	160	58	ZDT3	30	[1.95 , 16.25]	120
9	ZDT4	10	[1.65 , 6.35]	120	59	SCH	1	[15.4 , 12.1]	30000
10	ZDT4	10	[4.05 , 4.05]	120	60	SCH	1	[18.2 , 1.75]	40000
11	ZDT6	10	[0.9 , 18.2]	100	61	ZDT2	30	[9.2 , 18.7]	60
12	ZDT3	30	[3.65 , 13.65]	140	62	ZDT3	30	[2.4 , 23.85]	100
13	ZDT1	30	[14.6 , 4.4]	80	63	ZDT4	10	[16.75 , 4.6]	40
14	ZDT4	10	[6.55 , 18.55]	40	64	ZDT1	30	[8.1 , 8.55]	60
15	SCH	1	[5.45 , 27.15]	30000	65	SCH	1	[2.1 , 29.75]	30000
16	ZDT2	30	[6.45 , 25.7]	100	66	SCH	1	[17.1 , 10.8]	20000
17	ZDT2	30	[26.8 , 4.1]	40	67	ZDT2	30	[16.1 , 3.75]	100
18	SCH	1	[10.35 , 14.65]	40000	68	ZDT4	10	[8.55 , 17.1]	40
19	ZDT4	10	[19.85 , 11.55]	100	69	SCH	1	[10.4 , 14.95]	30000
20	ZDT3	30	[8.05 , 20.75]	100	70	ZDT3	30	[18.45 , 11.8]	120
21	ZDT3	30	[18.85 , 5.2]	100	71	ZDT1	30	[5.25 , 12.75]	60
22	SCH	1	[22.7 , 3.45]	40000	72	ZDT1	30	[3.9 , 15.1]	40
23	SCH	1	[5.25 , 10.35]	30000	73	SCH	1	[12.5 , 19.7]	30000
24	ZDT4	10	[18.65 , 12.5]	60	74	ZDT6	10	[15.9 , 13.35]	300
25	ZDT3	30	[14.9 , 3.8]	80	75	FON	3	[15.1 , 9]	300
26	ZDT2	30	[11 , 12.65]	60	76	ZDT4	10	[5 , 19.1]	80
27	ZDT1	30	[13.8 , 15.9]	100	77	SCH	1	[8.45 , 12.5]	40000
28	ZDT3	30	[9.7 , 16.5]	100	78	SCH	1	[24.25 , 8.1]	40000
29	ZDT4	10	[25.65 , 4.85]	80	79	ZDT1	30	[14.95 , 5.05]	60
30	SCH	1	[7.95 , 12.75]	30000	80	ZDT1	30	[11.95 , 13.35]	60
31	ZDT6	10	[23.8 , 3.15]	300	81	ZDT6	10	[15.6 , 15.45]	100
32	ZDT2	30	[4.3 , 12.55]	100	82	ZDT6	10	[14.2 , 3.2]	200
33	ZDT1	30	[25.1 , 3.1]	80	83	SCH	1	[24.15 , 5.1]	20000
34	ZDT6	10	[3.95 , 17.55]	200	84	ZDT2	30	[23.3 , 1.45]	100
35	ZDT2	30	[15.5 , 10.85]	60	85	ZDT6	10	[18.35 , 5.15]	200
36	FON	3	[13.85 , 10.55]	300	86	ZDT2	30	[11 , 13.4]	80
37	ZDT2	30	[14.35 , 3.85]	80	87	ZDT2	30	[6 , 24.8]	60
38	ZDT1	30	[0.9 , 24.95]	60	88	ZDT2	30	[8.9 , 22.8]	80
39	ZDT3	30	[21.9 , 11.4]	80	89	ZDT2	30	[11.25 , 20.65]	60
40	ZDT6	10	[5.05 , 11.45]	100	90	ZDT3	30	[15.8 , 8.2]	100
41	ZDT2	30	[16.15 , 9.35]	60	91	FON	3	[2.75 , 18]	100
42	ZDT1	30	[6.05 , 21.2]	100	92	ZDT1	30	[7.85 , 8]	100
43	ZDT4	10	[5.35 , 21.75]	100	93	ZDT2	30	[18.6 , 12.3]	40
44	SCH	1	[8.7 , 18.05]	30000	94	ZDT1	30	[2.4 , 14.9]	100
45	ZDT3	30	[11.75 , 5.5]	100	95	ZDT3	30	[12 , 8.9]	80
46	ZDT6	10	[2.1 , 21.85]	300	96	FON	3	[5.05 , 25.55]	100
47	ZDT6	10	[12.7 , 13.45]	200	97	SCH	1	[25.45 , 1.75]	40000
48	ZDT3	30	[5.75 , 11.3]	120	98	ZDT6	10	[1.7 , 29.35]	400
49	ZDT4	10	[6.2 , 40.45]	40	99	ZDT6	10	[10.6 , 10.9]	300
50	SCH	1	[11.4 , 17.65]	30000	100	ZDT6	10	[10.5 , 8.8]	100

10.6 Test problem #6 – concepts configuration

Table 16: Details of test problem #6 concept configuration

Concept #	Fn.	n_m	Obj.-space offset	N_m	Concept #	Fn.	n_m	Obj.-space offset	N_m
1	ZDT4	10	[0 , 24.7]	20	51	ZDT4	10	[13.95 , 11.3]	60
2	ZDT4	10	[2.95 , 22.25]	20	52	ZDT2	30	[13.35 , 23.15]	80
3	ZDT4	10	[4 , 18.65]	20	53	ZDT3	30	[22.35 , 11.3]	140
4	ZDT4	10	[5.5 , 15.8]	20	54	FON	3	[16.85 , 3]	100
5	ZDT4	10	[7.45 , 14.4]	20	55	ZDT2	30	[6.8 , 24.25]	80
6	ZDT4	10	[9.45 , 12.55]	20	56	FON	3	[12.9 , 18.2]	300
7	ZDT4	10	[10.45 , 8.95]	20	57	ZDT1	30	[20.7 , 19.9]	60
8	ZDT4	10	[13.05 , 7.65]	20	58	ZDT1	30	[23.3 , 11.25]	100
9	ZDT4	10	[14.55 , 3.9]	20	59	SCH	1	[13.25 , 29.05]	20000
10	ZDT4	10	[16.75 , -0.15]	20	60	ZDT1	30	[1.65 , 33.6]	100
11	ZDT1	30	[0 , 25.7]	80	61	ZDT6	10	[16.95 , 4.85]	100
12	ZDT2	30	[15.35 , 11.65]	80	62	ZDT3	30	[20.05 , 16.35]	120
13	ZDT2	30	[20.5 , 14.2]	80	63	ZDT2	30	[4.5 , 25.05]	100
14	ZDT2	30	[15.6 , 10.7]	80	64	ZDT1	30	[11.85 , 11.8]	80
15	ZDT2	30	[23.8 , 5]	40	65	FON	3	[3.95 , 31.9]	300
16	ZDT4	10	[13.75 , 10.3]	80	66	ZDT1	30	[21.8 , 16.55]	60
17	ZDT3	30	[9.15 , 25.5]	100	67	ZDT1	30	[5.75 , 22.05]	40
18	ZDT1	30	[6.95 , 23.4]	60	68	ZDT2	30	[13.9 , 19.3]	80
19	ZDT2	30	[5.4 , 26.1]	60	69	SCH	1	[19.65 , 12.8]	20000
20	ZDT1	30	[0.7 , 34.7]	100	70	ZDT4	10	[20.75 , 16.25]	40
21	SCH	1	[16.05 , 18.3]	30000	71	ZDT2	30	[11.4 , 32.25]	60
22	FON	3	[19.4 , 12.35]	200	72	ZDT3	30	[10.15 , 26.75]	140
23	ZDT2	30	[17 , 18.6]	60	73	ZDT6	10	[15.05 , 22.25]	400
24	ZDT6	10	[14.2 , 11.05]	300	74	ZDT1	30	[21.45 , 10.45]	40
25	SCH	1	[13.45 , 18]	40000	75	ZDT6	10	[7.45 , 22.3]	300
26	ZDT1	30	[4.45 , 21]	80	76	ZDT2	30	[8.7 , 24.3]	100
27	ZDT4	10	[15.6 , 5.25]	100	77	ZDT4	10	[18.65 , 11.4]	60
28	ZDT1	30	[4.5 , 26.2]	40	78	ZDT6	10	[14.1 , 21.85]	200
29	ZDT6	10	[22.4 , 11]	300	79	ZDT1	30	[10.7 , 20.9]	100
30	ZDT3	30	[12.05 , 27.2]	140	80	ZDT6	10	[15.35 , 15.7]	300
31	SCH	1	[10.15 , 18.45]	20000	81	ZDT2	30	[15.7 , 12.65]	40
32	ZDT1	30	[6.15 , 27.85]	40	82	SCH	1	[14.8 , 15.65]	20000
33	FON	3	[6.75 , 26.5]	300	83	ZDT1	30	[11.3 , 23.4]	100
34	ZDT6	10	[17.6 , 19.2]	300	84	ZDT1	30	[11.95 , 32.7]	40
35	ZDT3	30	[20.75 , 2.65]	140	85	ZDT6	10	[8.2 , 26.25]	100
36	ZDT3	30	[14.55 , 20.75]	140	86	ZDT1	30	[9.35 , 21.1]	60
37	ZDT4	10	[5.4 , 32.1]	40	87	ZDT6	10	[21.2 , 9.65]	200
38	ZDT4	10	[25.65 , 6.25]	80	88	SCH	1	[10.8 , 16.55]	40000
39	ZDT2	30	[10.7 , 23.9]	100	89	ZDT4	10	[8.2 , 26.15]	80
40	FON	3	[1.7 , 29.5]	200	90	ZDT6	10	[19.15 , 6.55]	300
41	ZDT2	30	[11.75 , 31.05]	40	91	ZDT6	10	[22.8 , 11.9]	200
42	ZDT1	30	[19.05 , 16.4]	80	92	ZDT1	30	[7.85 , 25.75]	40
43	ZDT3	30	[3.95 , 25.15]	120	93	ZDT3	30	[8.85 , 24.75]	120
44	ZDT2	30	[19.15 , 17.75]	80	94	ZDT6	10	[11.6 , 15.7]	200
45	ZDT1	30	[11.25 , 28.25]	100	95	SCH	1	[5.3 , 29.3]	30000
46	ZDT4	10	[19.4 , 11.4]	40	96	SCH	1	[15.05 , 12.85]	30000
47	ZDT1	30	[6.6 , 27]	100	97	ZDT2	30	[7.1 , 24.5]	40
48	FON	3	[10.7 , 25.45]	100	98	FON	3	[6.8 , 19.6]	200
49	ZDT6	10	[12.1 , 22.25]	300	99	ZDT1	30	[12.9 , 22.65]	100
50	SCH	1	[6.05 , 31.2]	40000	100	ZDT1	30	[12.65 , 22.9]	60

<u>תקציר</u>

מחשבים יכולים לחפש ולהשוות הרבה פתרונות קונספטואליים (קונספטים) עבור בעיות מרובות-מטרות. ניתן להשתמש בכך לצורך מציאת קונספטי-על לפני קבלת החלטה על הקונספט הנבחר, ולבסוף על הפתרון הפרטני. אבל, חיפוש כזה בדרך כלל כרוך במגבלה של המחיר החישובי.

מחקר זה נובע מהצורך להצדיק הקצאת משאבי חישוב בתהליך אבולוציה מבוסס פרטו של קונספטים מבוססי קבוצות. אבולוציה כזו כוללת חיפוש סימולטני במרחבי תכן שונים הן ברמה הקונספטואלית והן ברמת הפתרונות הפרטיים, ביחס למטרות האופטימיזציה. במחקר זה מוצעות ונחקרות מספר שיטות להערכה באופן מקוון של נחיתות פתרונות קונספטואליים. שיטות מקוונות כאלו מאפשרות ניפוי חכם של קונספטים נחותים, וכתוצאה מכך מאפשרות הפחתה של המשאבים החישוביים הנדרשים.

על-מנת להעריך את יעילות הטכניקות המוצעות, בתזה זו מוצעים מדדים ייחודיים וגישת אנליזה של הקונספטים המתפתחים באבולוציה. בוצעה חקירה סטטיסטית נרחבת להערכה והשוואה של הטכניקות המוצעות. בהתבסס על התוצאות הסטטיסטיות, מתקבלת המסקנה שחלק מהשיטות המוצעות הן יעילות. יעילות הטכניקות משתנה בהתאם לרמת הקושי של בעיות החיפוש. מסקנות אלו תואמות את המשפט המפורסם שאומר שאין ארוחות חינם באופטימיזציה.

אוניברסיטת תל-אביב

הפקולטה להנדסה ע"ש איבי ואלדר פליישמן
בית הספר לתארים מתקדמים ע"ש זנדמן-סליינר

אופטימיזציה אבולוציונית של קונספטים רבים תחת ריבוי-מטרות

חיבור זה הוגש כעבודת גמר לקראת תואר "מוסמך אוניברסיטה"
בהנדסה מכאנית

על-ידי

ברק סמינה

אדר א' התשע"ט
פברואר 2019

אוניברסיטת תל-אביב

הפקולטה להנדסה ע"ש איבי ואלדר פליישמן
בית הספר לתארים מתקדמים ע"ש זנדמן-סליינר

אופטימיזציה אבולוציונית של קונספטים רבים תחת ריבוי-מטרות

חיבור זה הוגש כעבודת גמר לקראת תואר "מוסמך אוניברסיטה"
בהנדסה מכאנית

על-ידי

ברק סמינה

מחקר זה בוצע בבית הספר להנדסה מכאנית
תחת הנחייתו של ד"ר עמירם מושיוב

אדר א' התשע"ט
פברואר 2019