Preference Integration in Progressive Dynamic Multi-Objective Evolutionary Algorithms

David William Newton

Master of Science
School of Computer Science

McGill University
Montreal,Quebec
2017-12

A thesis submitted to McGill University in partial fulfillment of requirements
for the degree of Master of Science in Computer Science

Acknowledgments

Of the many people who have been enormously helpful in the preparation of this thesis, I am especially thankful to Prof. Paul G. Kry for his help and support in guiding me through to its successful completion. I would also like to express my gratitude to the members of the computer graphics lab at McGill University for their time and valuable feedback during the primary investigation of this project. Special thanks to Youri Tamitegama, Adam Bognat, and Paul-Andre Heneger for their thoughts and work related to the research. Also, special thanks to Marc-Pierre Verge for all his thoughts and time related to the research.

I would also like to warmly acknowledge Emilie Hudon for all her tireless support and encouragement in the project. Her support meant everything. In addition, a special thanks to all my friends for their consideration and motivation. I dedicate this dissertation to my mother and family for their unconditional love and support in every way possible throughout the process of this course, this dissertation, and beyond.

Abstract

Many real-world multi-objective optimization problems found in the disciplines of engineering and design have objectives, decision variables, and constraints that dynamically change during the optimization process. Dynamic multi-objective evolutionary algorithms (DMOEAs) provide an efficient and effective means to address these problems, but the majority of research in this area has focused on theoretical problems in which the user has a limited role and the changes to the problem involve time-dependent objective functions (i.e., where the objective functions have time as an input in addition to other input parameters). In contrast, there has been little research on problems where the number of objectives changes, and no research that considers strategies for problems in which the decision variables and objectives change simultaneously. Further, there has been no research examining how the user might be brought into the DMOEA optimization process in a progressive fashion to guide the process in situations where user interactivity is crucial (e.g., when the number of objectives becomes larger than three). This thesis addresses these issues and proposes a new modification to the popular nondominated sorting genetic algorithm II (NSGA-II), that we call the dynamic progressive NSGA-II (DP-NSGA-II). The algorithm addresses the dual problems of convergence and diversity of solutions when changes occur. It accomplishes this through a combination of memory-based and prediction approaches. When the number of objectives changes, DP-NSGA-II uses a memory-based approach and samples from novelty and Pareto archives to aid diversity and convergence respectively. A user defined global reference direction can also be used to direct the search in circumstances where the number of objectives increases and the objective space goes beyond three dimensions. To deal with changes in the number of decision variables, or time-dependent objective functions, the type and severity of the change is first computed. If the change is small, a prediction-based approach is applied in which already calculated objective values in the Pareto archive are scaled by a computed scale factor, sampled, and used to replace half of the current population. This aids convergence by using values that were known to have performed well previous to the change. The other half of the population is then replaced with random samples from the novelty archive to aid diversity. If the change is big, the Pareto archive is completely emptied, and the current population of solutions is repopulated with samples from the novelty archive. DP-NSGA-II integrates user preferences in a progressive fashion through the application of both a reference direction and solution ranking approach. This allows the user to guide the search at the global and local scales. DP-NSGA-II is applied to a real-world optimization problem and tested against the NSGA-II and dynamic NSGA-II (D-NSGA-II) based on its ability to find optimal and diverse solutions. We show that DP-NSGA-II outperforms NSGA-II in average Pareto rank when the number of objectives change. It also outperforms D-NSGA-II when the number of decision variables change. Further, DP-NSGA-II outperforms both algorithms in novelty for 75% of the test cases.

Abrégé

Dans les disciplines de l'ingénierie et de la conception architecturale, de nombreux problèmes d'optimisation multi-objectifs ont des objectifs, des variables de décision ainsi que des contraintes qui changent de façon dynamique au cours du processus d'optimisation. Pour résoudre ces problèmes, les algorithmes évolutifs multi-objectifs dynamiques (DMOEA) fournissent un moyen efficace et efficient. Par contre, la majorité des recherches dans ce domaine ont porté sur des problèmes théoriques dans lesquels l'utilisateur a un rôle limité et les changements au problème dépendent du temps (c'est-à-dire, lorsque les fonctions de l'objectif ont le temps comme variable, en plus de ces autres paramètres d'entrée). En revanche, il y a eu peu de recherches sur des problèmes où le nombre d'objectifs change, et aucune recherche qui considère des stratégies pour des problèmes dans lesquels les variables de décision et les objectifs changent en nombre. De plus, aucune recherche n'a examiné comment l'utilisateur pourrait être amené à progressivement guider le processus d'optimisation DMOEA lorsque l'interactivité avec l'utilisateur est cruciale (par exemple, lorsque le nombre d'objectifs devient supérieur à trois). Cette thèse aborde ces questions et propose une nouvelle modification de l'algorithme génétique de tri non-populaire II (NSGA-II), que nous appelons le NSGA-II Dynamique Progressif (DP-NSGA-II). DP-NSGA-II intègre progressivement les préférences de l'utilisateur grâce à l'application d'une approche de la direction de référence. La direction de référence guide la recherche à l'échelle globale, tandis que l'approche de classement des solutions guide la recherche à l'échelle locale. L'algorithme aborde le double problème de la convergence et de la diversité des solutions lorsque surviennent des changements dans le nombre de fonctions objectives. Il accomplit ceci à travers une direction de référence globale définie par l'utilisateur et une archive de nouveauté. Lorsque le nombre d'objectifs change, DP-NSGA-II utilise une approche basée sur la mémoire et des échantillons provenant des archives de nouveauté et de Pareto pour favoriser la diversité et la convergence, respectivement. Une direction de référence globale définie par l'utilisateur peut également être utilisée pour diriger la recherche dans des circonstances où le nombre d'objectifs augmente et où l'espace objectif dépasse trois dimensions. Pour gérer les changements dans le nombre de variables de décision ou de fonctions objectives dépendant du temps, le type et la gravité du changement sont d'abord calculés. Si le changement est petit, une approche basée sur la prédiction est appliquée dans laquelle les valeurs objectives déjà calculées dans l'archive de Pareto sont mises à l'échelle par un facteur d'échelle calculé, échantillonné, et utilisé pour remplacer la moitié de la population actuelle. Cela favorise la convergence en utilisant des valeurs qui ont bien fonctionné avant le changement. L'autre moitié de la population est ensuite remplacée par des échantillons aléatoires provenant des archives de la nouveauté pour favoriser la diversité. Si le changement est important, l'archive de Pareto est complètement vidée et la population de solutions actuelle est repeuplée avec des échantillons provenant des archives de nouveauté. DP-NSGA-II intègre les préférences des utilisateurs de manière progressive en appliquant à la fois une direction de référence et une approche de classement des solutions. Cela permet à l'utilisateur de guider la recherche à l'échelle globale et locale. Le DP-NSGA-II est appliqué à un problème d'optimisation du monde réel et testé contre le NSGA-II et le NSGA-II dynamique (D-NSGA-II) en fonction de sa capacité à trouver des solutions optimales et variées. Nous montrons que DP-NSGA-II surpasse NSGA-II dans le rang moyen de Pareto quand le nombre d'objectifs change. Il surpasse également D-NSGA-II lorsque le nombre de variables de décision change. En outre, DP-NSGA-II surpasse les deux algorithmes de nouveauté pour 75% des cas de test.

TABLE OF CONTENTS

Acknowledgments ii
Abstract iii
Abrégé iv
TABLE OF CONTENTS vi
LIST OF TABLES viii
LIST OF FIGURES ix
LIST OF ACRONYMS xi
LIST OF SYMBOLS xii
Chapter 1. Introduction 1
1.1 Contributions 4
Chapter 2. Background 5
2.1 Computational Optimization 5
2.2 Deceptive Problems 6
2.3 Exploration Algorithms 6
2.4 Multi-Objective Evolutionary Algorithms 8
2.4.1 Basic Definitions 8
2.4.2 MOEA Categories 10
2.5 Preference-Integration 12
2.6 Dynamic Multi-Objective Evolutionary Algorithms 15
2.7 Progressive MOEAs in the Design, Engineering, and the Arts 18
2.8 MOEAs in the Conceptual Design Phase 19
Chapter 3. DP-NSGA-II Algorithm and Implementation 21
3.1 DP-NSGA-II Algorithm Overview 21
3.2 Objective and Novelty Evaluation 24
3.3 Selection and Reproduction 24
3.4 The Nondominated, Novelty, and Generational Archives 25
3.5 User-Defined Local Search through Solution Ranking 25
3.6 Change Detection and Response 26
3.6.1 Changing Number of Objectives 28
3.6.2 Time-Dependent Objective Functions 28

3.6.3 Changing Number of Decision Variables ... 31
3.6.4 Simultaneously Changing Objectives and Decision Variables ... 31
3.7 Implementation and User Interface ... 32
Chapter 4. Case Study Application of DP-NSGA-II ... 35
4.1 MOP Description ... 35
4.1.1 Decision Variables ... 36
4.2 Test Scenarios and Evaluation ... 36
4.3 Results and Discussion ... 41
Chapter 5. Conclusions ... 51
5.1 Future Work ... 51
Chapter 6. Appendix A ... 53
REFERENCES ... 62

LIST OF TABLES

Table 2-1 Comparison of preference-based MOEA methodologies. This table has been adapted from the classification proposed by Bechikh, Kessentini et al. (2015). 13

Table 2-2. Comparison of DMOEAs categorized by diversity, memory mechanisms, and prediction approaches. 17

Table 4-1. Summary of facade optimization test #1, with the number of objectives changing over time. 40

Table 4-2. Summary of facade optimization test #2, with the number of decision variables changing over time. 40

Table 4-3. Summary of facade optimization test #3, with time-dependent objective functions. . 41

Table 4-4. Summary of facade optimization test #4, with simultaneous changes. 41

Table 4-5. Performance of DP-NSGA-II against NSGA-II and D-NSGA-II for the four test cases studied here. Shows the percentage of times that DP-NSGA-II outperformed the other algorithms. 49

Table 6-1. Novelty values for test problem 1. Complete data from Figure 4-6............................ 54

Table 6-2. Pareto average rank for test problem 1. Complete data from Figure 4-7. 55

Table 6-3. Novelty values for test problem 2. Complete data from Figure 4-8............................ 56

Table 6-4. Pareto average rank for test problem 2. Complete data from Figure 4-9. 57

Table 6-5. Novelty values for test problem 3. Complete data from Figure 4-10.......................... 58

Table 6-6. Pareto average rank for test problem 3. Complete data from Figure 4-11. 59

Table 6-7. Novelty values for test problem 4. Complete data from Figure 4-12.......................... 60

Table 6-8. Pareto average rank for test problem 4. Complete data from Figure 4-13. 61

LIST OF FIGURES

Figure 3-1. Diagram of a standard Pareto selection-based MOEA. 22

Figure 3-2. Diagram of the proposed DP-NSGA-II algorithm. 22

Figure 3-3. Pseudo code of DP-NSGA-II algorithm. 23

Figure 3-4. Pseudo Code for the change detection and response algorithm for DP-NSGA-II..... 27

Figure 3-5. The left image shows distribution of nondominated solutions for DTLZ2 for two objectives. The right image shows the distribution of solutions after an objective is added. 27

Figure 3-6. The left image shows distribution of nondominated solutions for DTLZ2 for three objectives. The right image shows the distribution of solutions after an objective has been subtracted. 29

Figure 3-7. Pseudo Code for dealing with a changing number of objectives for DP-NSGA-II... 29

Figure 3-8. Pseudo Code for dealing with time-dependent objective functions for DP-NSGA-II. 30

Figure 3-9. Pseudo Code for dealing with changing number of decision variables for DP-NSGA-II 30

Figure 3-10. DP-NSGA-II integrates into the Grasshopper visual programming environment.. 33

Figure 3-11. (1) An image of the Rhinoceros 3D Interface. (2) An image of the visual programing environment in Grasshopper, which is a plugin for Rhinoceros 3D. 33

Figure 3-12. An image of the DP-NSGA-II interface... 34

Figure 4-1. A section and axonometric drawing of the facade system that is optimized. 37

Figure 4-2. Two samples from the optimization process. 37

Figure 4-3. Thermal simulations for 10 different samples in the decision space during the summer. Each column of images represents solutions with the same number of folds, while each row represents solutions with a different spacing for the pipes. These samples are used to create a surrogate model to speed-up the evaluation of designs. 38

Figure 4-4. Image of useful daylight simulation during optimization. 38

Figure 4-5. The decision variables for the optimization involve exploring the depth, the fold pattern, the spacing, and the regional pattern of the CNC bent pipe assembly. 39

Figure 4-6. Average novelty for test #1 in which the number of objectives change. Each symbol represents the average of four runs. 45

Figure 4-7. Pareto rank average for test #1 in which the number of objectives change. Each symbol represents the average of four runs. 45

Figure 4-8 Average novelty for test #2 in which the number of decision variables change. Each symbol represents the average of four runs. 46

Figure 4-9. Pareto rank average for test #2 in which the number of decision variables change. Each symbol represents the average of four runs. 46

Figure 4-10 Average novelty for test #3 in which the objective functions are time-dependent. Each symbol represents the average of four runs. 47

Figure 4-11. Pareto rank average for test #3 in which the objective functions are time-dependent. Each symbol represents the average of four runs. 47

Figure 4-12 Average novelty for test #4 in which the objective functions are time-dependent. Each symbol represents the average of four runs. 48

Figure 4-13. Pareto rank average for test #4 in which the objective functions are time-dependent. Each symbol represents the average of four runs. 48

Figure 4-14. (1) The Rhinoceros 3D interface that the geometry for the facade is created in. (2) An image of the visual programming environment provided by Grasshopper that allows users to interact with the optimization by adding or subtracting design features as the optimization runs. (3) An image of the DP-NSGA-II graphical user interface as the optimization runs... 49

Figure 4-15. Some samples of the non-dominated solutions at various stages during the testing can be seen. 50

LIST OF ACRONYMS

CAD	Computer Aided Design
CFD	Computational Fluid Dynamics
CNC	Computer Numerically Controlled
DMOEA	Dynamic Multi-Objective Evolutionary Algorithm
DMOP	Dynamic Multi-Objective Problem
DP-NSGA-II	Dynamic Progressive NSGA-II
D-NSGA-II	Dynamic NSGA-II
EA	Evolutionary Algorithm
iTDEA	Interactive Territory Defining
MCC	Minimal Criterion Coevolution
MOEA	Multi-objective Evolutionary Algorithms
MOEA/D	Multi-objective Evolutionary Algorithms Decomposition
MOP	Multi-objective Problem
NS	Novelty Search
NSGA	Nondominated Sorting Genetic Algorithm
NSGA-II	Nondominated Sorting Genetic Algorithm II
NSLC	NS with Local Competition
PAES	Pareto Archived Evolutionary Strategy
PF	Pareto Front
POS	Pareto Optimal Set
r-NSGA-II	Reference-direction Nondominated Sorting Genetic Algorithm II
R-NSGA-II	Reference-point Nondominated Sorting Genetic Algorithm II
ROI	Regions of Interest
SPEA	Strength Pareto Evolutionary Algorithm

LIST OF SYMBOLS

ε	Parameter used in Algorithm 4 and 5 in 3-8 and 3-9 to determine when a change to the objective functions of the decision variables is severe.
$\boldsymbol{\Lambda}$	Universe containing all possible objective vectors $\boldsymbol{F(x)}$.
$\rho(\boldsymbol{y})$	Novelty metric calculated as the average Euclidean distance between $\boldsymbol{y}$ and its k nearest neighbors $\boldsymbol{y}_i$ in objective space.
Ω	Universe containing all possible x values.
$\boldsymbol{A}$	Archive of nondominated solutions.
Children	Solutions generated by mixing the decision variables of solutions selected by binary tournament selection through a crossover operation.
$\boldsymbol{D}$	Variable size array of decision variables.
$d(\boldsymbol{y},\boldsymbol{y}_i)$	Euclidean distance between $\boldsymbol{y}$ and $\boldsymbol{y}_i$ in objective space.
$\boldsymbol{F}$	Variable size array of objective functions.
$\boldsymbol{F}(\boldsymbol{x})$	Coordinate space containing the resultant vectors from evaluating MOP's solutions.
$\boldsymbol{F}(\boldsymbol{x},t)$	Set of objective functions dependent on time.
f_1	Normalized value for objective 1.
$f_k(\boldsymbol{x})$	Each of the k objective functions for decision vector $\boldsymbol{x}$.
$f_m(\boldsymbol{x},t)$	Any of the m particular objective functions dependent on time.
$\boldsymbol{G}$	Generational archive.
G_{max}	Maximum generations.
$g_i(\boldsymbol{x})$	Each of the i inequality constraints for decision vector $\boldsymbol{x}$.
gen	Current generation number.
$h_j(\boldsymbol{x})$	Each of the j equality constraints for decision vector $\boldsymbol{x}$.
k	Number of objective functions in the MOP, or parameter for novelty equation.
m	Number of objectives to be minimized or number of inequality constraints.
$\boldsymbol{N}$	Novelty archive.
n	Number of decision vectors or number of non-dominated solutions.
$\boldsymbol{P}$	Array representing the population of solutions.
P^*	Pareto optimal set. Set of all solutions within the decision space whose corresponding objective vector components cannot be all simultaneously improved.
p	Number of equality constraints.
Parents	Vector containing solutions selected through binary tournament selection.
$\boldsymbol{PF}$	Pareto Front. All evaluated objective vectors from P^*.
$\boldsymbol{Pt}$	Vector containing solutions for most recent population.
$\boldsymbol{P}_{\boldsymbol{ranked}}$	Vector containing Pareto ranked solutions.
$\boldsymbol{P}_{\boldsymbol{ideal}}$	Most optimum solution found located along $\boldsymbol{Rg}$.
$\boldsymbol{Rg}$	User-defined global reference direction.
$\boldsymbol{Rl}$	User-defined local reference direction.
S	Population size.
t	Variable representing time.
$\boldsymbol{u}, \boldsymbol{v}$	Any two vectors to be compared.
$\boldsymbol{x}$	Set of vectors for decision variables.
$x_{1\ldots n}$	Each of n decision vectors forming vector set $\boldsymbol{x}$.

$\boldsymbol{x}'$	Feasible vector which would decrease some criterion without causing a simultaneous increase in at least one other criterion (assuming minimization).
$\boldsymbol{y}$	Vector of objective values.

Chapter 1. Introduction

Many fields face challenging optimization problems in which there are multiple and often conflicting objectives that must be simultaneously satisfied. These multi-objective problems (MOPs) are particularly prevalent in real-world design and engineering problems in fields such as architectural, urban, landscape, and industrial design, as well as, civil, industrial, automotive, and aerospace engineering. Research on the process of design suggests that the basic activities and the essential stages of a design process are similar across these disciplines (Archer 1968, Gregory 2013). Research has also shown that the initial phase of a design process, the "conceptual design" phase, has the most impact on the cost of the final design as well as its performance (Duffy, Andreasen et al. 1993, Wang 2001, Wang 2002, Chong, Chen et al. 2009). Therefore, the development of heuristics and tools that can aid in the solution of MOPs for the conceptual phase of design will have a large impact across multiple disciplines.

The conceptual design phase involves the open-ended exploration of a design space (i.e., a space of all possible design solutions). The goal is to find a set of decision variables (i.e., input parameters) that map to a diverse but equivalent set of optimal performing solutions located within the objective space defined by the objectives of the optimization problem. This optimal set, referred to as the Pareto optimal set (POS), is located on an n-dimensional manifold referred to as the Pareto front (PF). MOPs present several challenges for researchers developing algorithms to meet this goal, and these have been well documented by other researchers (Coello, Van Veldhuizen et al. 2002, Deb 2014, Datta and Gupta 2016). Five challenges stand out and are the focus of this thesis:

1. Exploration: A significant stumbling block in developing algorithms to deal with MOPs is ensuring adequate exploration of an objective space as well as diversity in the solutions uncovered (Deb, Pratap et al. 2002). This is especially crucial for the conceptual design phase.
2. User preference-integration: Many algorithms used to solve MOPs require interaction with a decision maker, but how and when the decision maker should be involved in the process is an open problem (Bechikh, Kessentini et al. 2015).
3. Dimensionality: MOPs that have more than three objectives, known as many-objective MOPs can pose major challenges to the computability of the PF due to the curse of dimensionality - once objectives increase past three, the size of the PF can quickly become too large for many algorithms to cover effectively (Bechikh, Elarbi et al. 2017).
4. Dynamics: Many real-world MOPs involved in the conceptual design phase are dynamic and may have objectives, design parameters, and constraints that may change during the optimization process (Azzouz, Bechikh et al. 2017). These problems are known as dynamic MOPs (DMOPs).

5. <u>Deception:</u> A final complication is that MOPs and DMOPs can often be deceptive in nature – meaning they may be prone to leading decision makers down the wrong path in a design space (Goldberg 1987).

There is a broad range of optimization algorithms that have been applied to address these stated problems, for instance enumerative, deterministic, and stochastic algorithms. Of these approaches, multi-objective evolutionary algorithms (MOEAs) have emerged as one of the most widely studied and successful (Coello, Van Veldhuizen et al. 2002). The field of MOEA research can be divided into five categories: Pareto dominance-based methods; indicator-based methods; reference point-based methods; grid-based methods; and decomposition-based methods. Further, because MOEAs require feedback from a decision maker, each of these categories can be divided into three sub groups based on whether a decision maker's preferences are integrated before, during, or after the optimization process. We refer to these variants as a priori, progressive, and a posteriori processes.

Progressive MOEAs demonstrate the best performance when dealing with optimization problems with more than three objectives (i.e., many-objective problems). Li, Deb et al. (2017) propose a progressive decomposition-based algorithm that requires the decision maker to interactively define multiple regions of interest (ROIs) throughout the optimization process. These ROIs then allow the optimization algorithm to focus on exploring a smaller hyper-volume of the objective space, making what would otherwise be a computationally expensive search less costly. Progressive Pareto dominance-based methods have shown significant effectiveness in addressing the challenges posed by many-objective problems. Deb and Kumar (2007) propose R-NSGA-II (Reference-point Nondominated Sorting Genetic Algorithm II), which uses a reference point method to define a single ROI to direct the search. Said, Bechikh et al. (2010) propose r-NSGA-II (Reference-direction Nondominated Sorting Genetic Algorithm II), which uses an aspirational level vector supplied by the decision maker. They demonstrate that progressive MOEAs significantly outperform non-progressive approaches when dealing with many-objective problems. Despite the promise of progressive techniques, there has been relatively little research on progressive MOEAs in comparison to non-progressive variants.

In order to address the challenges posed by dynamic MOPs (DMOPs), dynamic multi-objective evolutionary algorithms (DMOEAs) are currently being studied in the fields of optimization, operations research, and computer science (Azzouz, Bechikh et al. 2017). Research on DMOEAs has mostly looked at cases where the objective function changes with time (i.e., where the objective functions have time as an input in addition to other input parameters) (Azzouz, Bechikh et al. 2017). Some examples include route optimization problems according to real-time traffic (Wahle, Annen et al. 2001); scheduling problems (Deb, Rao N et al. 2007); real-time resource allocation (Palaniappan, Zein-Sabatto et al. 2001); and control problems such as the optimization of indoor heating (Hämäläinen and Mäntysaari 2002). Only one paper has looked at the problem of a changing number of objective functions (Chen, Li et al. 2016) and, to the best of our knowledge, no work addresses the situation where the number of input parameters is changing (dynamic decision space) in addition to a changing number of objectives (dynamic objective space). Further, previous work does not show how user preferences might be integrated in a progressive fashion in such a situation.

This thesis addresses these understudied areas through the development, implementation, and testing of a unique modification to the popular nondominated sorting genetic algorithm II (NSGA-II) (Deb, Pratap et al. 2002). The algorithm developed for this thesis is called the dynamic progressive NSGA-II (DP-NSGA-II). DP-NSGA-II integrates user preferences in a progressive fashion through the application of a hybrid approach. This is done through a reference-direction approach to guide the search at the global scale, and a solution ranking approach to guide the search at the local scale. This hybrid approach works especially well in high dimensional objective spaces where tradeoffs may change drastically at a global scale, but only moderately at a local scale. This interaction engages the decision maker in a manner that requires a level of comparative thought that other approaches involving a reference point, direction, or weighting may not provide. This progressive approach narrows the scope of the search in high-dimensional objective spaces and makes the optimization tractable, while engaging the user's abilities to help search a complex multi-dimensional space.

DP-NSGA-II includes procedures to help maintain diversity when simultaneous changes occur in the objectives and decision variables, something which no previous research has dealt with. The algorithm addresses the dual problems of convergence and diversity of solutions when changes occur through a combination of memory-based and prediction approaches. When the number of objectives changes, DP-NSGA-II uses a memory-based approach and samples from novelty and Pareto archives to aid diversity and convergence respectively. To deal with changes in the number of decision variables, or time-dependent objective functions, the type and severity of the change is first computed. If the change is small, a prediction-based approach is applied in which already calculated objective values in the Pareto archive are scaled by a computed scale factor, sampled, and used to replace half of the current population. This aids convergence by using values that were known to have performed well previous to the change. The other half of the population is then replaced with random samples from the novelty archive to aid diversity. If the change is big, the Pareto archive is completely emptied and the current population of solutions is repopulated with samples from the novelty archive. When simultaneous changes occur (e.g., when the number of decision variables and objective functions change) these methods are used in combination.

DMOEA research has been mostly done using theoretical toy optimization problems. There is little research on how these algorithms should be designed and adapted within the context of real-world problems. Researchers in the field have noted that this lack of real-world testing may be severely biasing our understanding of these optimization algorithms (Coello, Van Veldhuizen et al. 2002, Datta and Gupta 2016). This research attempts to address this deficit by developing and testing a unique progressive DMOEA within the context of a real-world design problem involving geometric optimization. Specifically, DP-NSGA-II is tested against NSGA-II and D-NSGA-II based on its ability to find optimum and diverse solutions. We show that DP-NSGA-II outperforms NSGA-II in average Pareto rank when the number of objectives change. It also outperforms D-NSGA-II when the number of decision variables change. Further, DP-NSGA-II outperforms both algorithms in novelty for 75% of the test cases.

The overwhelming use of toy problems for DMOEA research has meant that most progressive (user-in-the-loop) approaches have tended to be similar in the ways the user interacts with the search process. For example, most progressive processes involve a single monotonous interaction in which the user is asked to supply a weight vector for the objective functions that is used to define a region of interest for the search. This may be sufficient for a toy problem involving generic objective function sets, but for real problems it may not be sufficient. Our approach uses a visual programming interface to enhance user interactivity. Specifically, it allows users to dynamically define and modify both decision variables and objective functions at any time during the optimization process. This allows for an open-ended search process, and reduces user fatigue through interactions that involve comparative thought and active design decisions.

1.1 Contributions

This research is cross-disciplinary in nature, engaging the fields of computer science and those dealing with real-world dynamic MOPs related to the design of physical objects (e.g., civil, industrial, automotive, and aerospace engineering; architectural, urban, landscape, and industrial design), which will be referred to as the fields of design.

This research makes the following contributions to the field of computer science:

- DP-NSGA-II proposes a unique approach to help maintain diversity and provide convergence when the number of objectives and decision variables change simultaneously.
- DP-NSGA-II integrates user preferences in a progressive fashion through a reference-direction approach to guide the search at the global scale, and a solution ranking approach to guide the search at the local scale. Further, it represents the first DMOEA to integrate the user during the optimization process.
- DP-NSGA-II provides a user interface that allows users to interact in unique ways with the optimization process.

Chapter 2. Background

This chapter begins with a review of research related to the field of computational optimization for single and multi-objective problems. The topics of deceptive problems and exploration are then discussed as challenges in the field. Research in Multi-Objective Evolutionary Algorithms (MOEAs) is then categorized and the topic of progressive MOEAs introduced. This leads to a review of research related to the problem of user preference-integration. Then the topic of Dynamic Multi-Objective Evolutionary Algorithms (DMOEAs) is discussed. Lastly, a review of research on the application of MOEAs to the conceptual design phase of engineering and design projects is discussed.

2.1 Computational Optimization

There is a broad spectrum of approaches to the problem of single and multiple objective optimization in the field of computational optimization (Wright and Nocedal 1999, Boyd and Vandenberghe 2004). The central goal of these approaches is to efficiently find the global maximum, or global minimum, of an objective function, or functions, subject to constraints. These approaches can be divided into three categories. Random walk algorithms randomly sample a search space to find optimum solutions, while gradient descent approaches use derivatives. Population-based approaches use a population of solutions to converge on optimal solutions (Coello, Van Veldhuizen et al. 2002).

Random walk approaches have shown some potential with problems that have a relatively small decision space, but fail when that space is large and the probability of finding a global optimum through random selection becomes low (Coello, Van Veldhuizen et al. 2002). Gradient descent methods offer the advantage of speed and guaranteed convergence, but are relegated to design problems that have a continuous and differentiable objective space, or design space. They also tend to prematurely converge on local maxima, or, minima. This approach therefore is very limited, because most practical design problems do not have such mathematically well behaved design spaces.

Population-based approaches, in contrast, generally perform better than random walk approaches and can be applied to a wide array of problems in which gradient descent approaches are not suitable. Population-based approaches have been applied to a wide variety of real-world design problems during various stages of the design process (e.g.,, conceptual design phase; detail development phase) with significant success (Datta and Gupta 2016). Population-based approaches cannot, however, guarantee convergence to a global optimum, and therefore premature convergence is a major problem in the field. It is also worth noting that premature convergence is a problem for all listed techniques.

Evolutionary algorithms (EA) are a popular and widely used population-based approach that uses special operators (e.g., crossover; mutation; replacement) inspired by Darwinian evolutionary theory to successively adapt generations of solutions in a directed fashion with the use of an objective function (Bäck, Fogel et al. 1997). The careful use and modification of these operators has demonstrated success in addressing the problem of premature convergence (Pelikan and Goldberg 2001). Due to their ability to be applied to many different types of problems and the potential of these algorithms to adjust their local and global search capabilities through the adjustment of operator parameters, EAs have been chosen as the principle search approach in this research.

2.2 Deceptive Problems

There has been extensive research on the topic of why EAs sometimes fail and how to improve their failure rates through better design of their components (e.g., the genetic representation; the selection, reproduction, replacement, and mutation operators; the fitness functions) (Goldberg and Richardson 1987, Hu, Goodman et al. 2005, Hornby 2006, Hutter and Legg 2006). There are many reasons why an EA might fail in a particular problem, but the case of so-called deceptive problems is perhaps the most relevant for the fields of design and the task of conceptual design, because research suggests that many real-world MOPs might be deceptive in nature (Whitley 1991, Goldberg 2013). It is therefore necessary to develop search algorithms for the conceptual phase of design that can address deceptive problems.

Deceptive design problems have been defined as problems where the objective function, or functions, involved actively lead the search in a wrong direction, leading to premature convergence on a local maximum or minimum (Deb, Horn et al. 1993). The key principle with deceptive problems is the idea that sometimes to reach a goal you must take a path that has lower fitness initially than other available paths. In this scenario, some solutions with lower fitness become key stepping stones through the objective space to better solutions approaching the global optimum, but the EA's objective function will miss them because it rewards only individuals with high fitness. There has been much research on the topic of deception and approaches that have shown promise in addressing the problem (Goldberg 1987, Whitley 1991, Deb, Horn et al. 1993, De Jong, Watson et al. 2001, Pelikan and Goldberg 2001). Specifically, progressive (i.e., user-in-the-loop) MOEAs show promise in addressing this issue (Branke, Deb et al. 2008, Christman and Woolley 2015). This research therefore uses progressive MOEAs to address deceptive problems for the conceptual design phase.

2.3 Exploration Algorithms

Optimization algorithms are used to find the highest performing solutions in an objective space as quickly as possible. When the objective space is complex and has many local minima, or maxima, these algorithms can converge on solutions that are not optimal. To address the challenges posed by such deceptive problems it is desirable to balance global exploration with local optimization.

One approach to this problem involves the use of algorithms whose focus is on exploring the full extent of an objective space and not just finding a global optimum solution. Such algorithms put exploration as the main objective and have demonstrated an ability to outperform traditional optimization problems with problems that are highly deceptive (Lehman and Stanley 2011, Gomes, Mariano et al. 2015).

Lehman and Stanley (2008) propose Novelty Search (NS) as a means of addressing deceptive problems by doing away with the traditional use of objective functions to reward a solution and instead selecting individual solutions based on their degree of novelty, as expressed in objective space. NS rewards novelty through the calculation of a sparseness metric ρ defined as

$$\rho(\boldsymbol{y}) = 1/k \sum_{\boldsymbol{y}_i \in KNN(\boldsymbol{y})} d(\boldsymbol{y}, \boldsymbol{y}_i) \qquad (1)$$

where ρ is the average Euclidean distance between a vector of objective values $\boldsymbol{y}$ and its k nearest neighbors $\boldsymbol{y}_i$ in objective space. It is important to note here that the objective values are all normalized for this calculation. The nearest neighbors are selected from an archive which stores all the history of the novelty search. Calculating the novelty metric in objective space, instead of decision space, has the advantage of more directly tailoring the search to the unique properties of a problem's objective space, and it can help NS work in very large search spaces in situations where multiple decision vectors map to one objective vector. NS has been demonstrated to outperform EAs significantly in maze navigation problems and has shown effectiveness in evolving novel 2D images in the field of art (Gomes, Mariano et al. 2015, Nguyen, Yosinski et al. 2015). NS, however, has been shown to fail in situations where the design space is large and it also lacks an ability to refine solutions through a local search technique (Mouret 2011). Mouret (2011) and Gomes, Mariano et al. (2015) show that principles from NS can be applied to the design of MOEAs to address the first problem effectively and can even outperform other MOEAs (NS included) in deceptive problem domains.

NS with Local Competition (NSLC) was proposed as a way of improving NS's local optimization ability (Lehman and Stanley 2011) by adding a second objective beyond novelty to help drive a local search. Grid-based approaches such as MAP-Elites (Mouret and Clune 2015), CVT-MAP Elites (Vassiliades, Chatzilygeroudis et al. 2017), and Expansive MAP-Elites dispense with a novelty measure and instead divide the objective space into a discrete set of hypervolumes and then attempts to find the highest performing solution within each hypervolume based on a selected set of objectives. In contrast to these methods, Minimal Criterion Coevolution (MCC) (Brant and Stanley 2017) does not use an archive, but instead depends on two co-evolving populations of solutions along with a minimal criterion objective function to produce an open-ended exploration of the objective space.

The ability to compare and benchmark the exploration capabilities of these exploration-focused algorithms, also known as "illumination algorithms" (Mouret and Clune 2015), is key and currently an open research question. Gomes, Mariano et al. (2015) propose the use of the Jensen-Shannon distance (Endres and Schindelin 2003) to measure the explorative capacity of NS. Wang, Jin et al. (2017) note the difficulty of making such measures in the context of MOPs with more than three objectives and proposed a new metric referred to as pure diversity.

In order to deal with complex objective spaces and also promote exploration, our proposed algorithm calculates the novelty score of each generated solution using Equation 1. The most novel solutions are kept in a novelty archive and used to help promote diversity and exploration when dealing with DMOPs. The injection of novel solutions may negatively affect convergence, and so must be balanced with strategies to promote convergence. To aid convergence, we maintain an archive of high-performing solutions to help generate new solutions.

2.4 Multi-Objective Evolutionary Algorithms

MOEAs are a special class of EAs dedicated to solving optimization problems with multiple objective functions, which are often found in the field of design. MOEAs optimize multiple objective functions and consequently are less susceptible to deceptive problems than single objective EAs (De Jong, Watson et al. 2001, Toffolo and Benini 2003, Mouret 2011). This also means that in most cases there will be multiple solutions to a problem. MOEAs therefore work towards finding a set of optimal solutions, or designs. In certain types of MOEAs, this set is referred to as the Pareto optimal set (POS), and the solutions in this set offer different trade-offs between the objectives. MOEAs therefore require interaction with a decision maker to select which designs in the POS should be examined further, or used as the final solution to the optimization problem.

2.4.1 Basic Definitions

Definition 1. Decision variables are the numerical values which satisfy constraints and are used to optimize a vector function in a MOP. These quantities can be represented as a vector $\boldsymbol{x}$ in a vector space of size n as $\boldsymbol{x} = (x_1, x_2, \dots x_n)$. The decision variables x_i from some universe Ω can be continuous or discrete. In dynamic problems, the size n of $\boldsymbol{x}$ can change during the optimization process.

Definition 2. Constraints are restrictions imposed by specific characteristics of the environment that shape the solution space. Usually, they can be expressed in terms of decision variables (x_i's) and problem parameters in the form of mathematical expressions as follows:

$$g_i(\boldsymbol{x}) \leq 0, \qquad i = 1, \dots, m \tag{2}$$

or equalities

$$h_j(\boldsymbol{x}) = 0, \qquad j = 1, \dots, p \tag{3}$$

where m represents the number of inequality constraints and p represents the number of equality constraints. If $p \geq n$ (i.e., the number of decision variables) and the constraints are independent, the problem is said to be over-constrained, because there would be more unknowns than equations with no degrees of freedom (n - p) left for optimizing (Coello, Van Veldhuizen et al. 2002). In this thesis, constraints are implicitly used to bound decision variables. In dynamic problems, the number and mathematical definition of constraints can change with time.

Definition 3. Objectives are various goals for the MOP expressed in terms of distinct mathematical functions that need to be optimized. The term *objective space* refers to the coordinate space containing the resultant vectors from evaluating a MOP's solutions. The objective values are normalized.

The objective functions can be represented as

$$\boldsymbol{F}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \dots f_k(\boldsymbol{x})) \qquad (4)$$

where k represents the number of objective functions in the MOP. The universe of all possible output vectors is represented by Λ. The function $f_k(\boldsymbol{x})$ can be continuous or discreet. This thesis only considers continuous functions.

Definition 4. A MOP consists of k objectives represented by their objective functions, $m + p$ constraints on the objective functions and n decision variables. A general MOP is defined as minimizing (or maximizing) the objective space $\boldsymbol{F(x)}$ subject to the constraints, $g_i(\boldsymbol{x})$ and $h_j(\boldsymbol{x})$ and x_i from Ω, which contains all possible x values that satisfy an evaluation of $\boldsymbol{F(x)}$.

The k objective functions may be linear or nonlinear and continuous or discrete in nature. The evaluation function,

$$\boldsymbol{F}: \Omega \rightarrow \Lambda \qquad (5)$$

is a mapping from the vector of decision variables $\boldsymbol{x} = (x_1, x_2, \dots x_n)$ to output vectors $\boldsymbol{F(x)}$ (Eq. (4)).

Definition 5. Pareto Dominance A vector $\boldsymbol{u} = (u_l, \dots, u_k)$ is said to dominate another *vector* $\boldsymbol{v} = (v_l, \dots, v_k)$ (denoted by $\boldsymbol{u} \preceq \boldsymbol{v}$) if and only if $\boldsymbol{u}$ is partially less than $\boldsymbol{v}$, i.e.,

$$\forall i \in \{1, \dots, k\}, u_i \leq v_i \wedge \exists i \in \{1, \dots, k\} : u_i < v_i. \qquad (6)$$

Definition 6. Pareto Optimality A solution $\boldsymbol{x} \in \Omega$ is said to be pareto-optimal w. r. t. Ω if and only if there is no $\boldsymbol{x}' \in \Omega$ for which $\boldsymbol{v}$ dominates $\boldsymbol{u}$, where

$$\boldsymbol{v} = \boldsymbol{F}(\boldsymbol{x}') \qquad (7)$$

$$\boldsymbol{u} = \boldsymbol{F}(\boldsymbol{x}) \qquad (8)$$

in the entire decision variable space. In other words, this definition says that the vector $\boldsymbol{x}$ is Pareto optimal if there exists no other feasible vector $\boldsymbol{x}'$ which would decrease some criterion without causing a simultaneous increase in at least one other criterion (assuming minimization).

Definition 7. Pareto Optimal Set For a given MOP, $\boldsymbol{F}(\boldsymbol{x})$, the Pareto Optimal Set, P^*, is defined as:

$$P^* = \{ \boldsymbol{x} \in \boldsymbol{\Omega} \mid \neg\exists\, \boldsymbol{x}' \in \boldsymbol{\Omega}, \boldsymbol{F}(\boldsymbol{x}') \preceq \boldsymbol{F}(\boldsymbol{x})\}. \qquad (9)$$

Pareto optimal set (denoted by P^*) is the set of all solutions within the decision space whose corresponding objective vector components cannot be all simultaneously improved. They form the set of all solutions whose associated vectors are non-dominated and are classified as such based on their evaluated functional values.

Definition 8. Pareto Front The Pareto front $\boldsymbol{PF}$ for a given MOP with objective function $\boldsymbol{F(x)}$, and Pareto Optimal Set, P^*, is defined as:

$$\boldsymbol{PF} = \{\boldsymbol{F}(\boldsymbol{x}) \mid \boldsymbol{x} \in P^*\}. \qquad (10)$$

When plotted in objective space, the non-dominated vectors are collectively known as the Pareto front. ***PF*** is formed from evaluated objective vectors from P^*, of which each is non-dominated with respect to all objective vectors produced by evaluating every possible solution in **Ω**. The normal procedure to generate the Pareto front is to compute a number of points in **Ω** and their corresponding Λ and determine the non-dominated points to produce the Pareto front. MOPs usually have an uncountable set of solutions on a Pareto front. Each solution associated with a point on the Pareto front is a vector whose components represent trade-offs in the decision space.

2.4.2 MOEA Categories

The field of MOEA research can be divided into five categories (Zhang, Tan et al. 2017): Pareto dominance; indicator-based; reference-point; grid-based; and decomposition-based approaches. Pareto dominance based methods comprise the first and most popular category of approaches. These approaches rank solutions through comparing solutions on an objective by objective basis. Any solution that has no objective values lower than any other solution is said to be Pareto-dominant and is a part of the PF. Horn, Nafpliotis et al. (1994) use Pareto domination tournaments for selecting solutions within a genetic algorithm. Srinivas and Deb (1994) propose the Nondominated Sorting Genetic Algorithm (NSGA) which compares all solutions and ranks them into dominated and nondominated categories. Deb, Pratap et al. (2002) propose a more efficient version of the NSGA called NSGA-II which uses an archive of elite solutions to improve diversity. The NSGA-II has become the most popular and explored MOEA to date, although it has been found to perform poorly when the number of objectives goes above three. This is mainly due to the fact that the Pareto ranking operation fails to be selective enough when the number of objectives goes above three.

The second category of MOEAs are known as indicator-based methods. Zadeh (1963) show that a limited set of solutions in the POS could be found with a weighted sum approach, in which each objective is given a weight and summed to create one objective function. Das and Dennis (1997) demonstrate the limitations of weighted sum approaches to find solutions in the POS. Zitzler and Künzli (2004) propose an indicator-based MOEA (IBEA) which uses preference information from a decision maker in the form of a binary function to direct the search towards particular areas of the ROI. Phan and Suzuki (2013) propose an improved version of IBEA called R2-IBEA which eliminates dominance ranking and performs selection by uniformly distributing weight vectors within a desired hyper-volume.

The third category known as reference-point MOEAs focus the search on an ROI of the PF based on a user selected reference point. Deb and Jain (2014) propose an updated version of the NSGA-II called the NSGA-III that uses a user defined reference point to guide the search when dealing with many-objective problems (i.e., MOPs with greater than three objectives). Defining the reference point is not trivial and the NSGA-III does not allow for precise control of the boundary of the ROI.

Grid-based approaches comprise the fourth category. Deb, Mohan et al. (2005) propose ε-MOEA which divides the objective space into a grid of hypervolumes, which are then used to create a grid of well distributed solutions on the PF based on an ε spacing. Yang, Li et al. (2013) propose a grid-based genetic algorithm to solve many objective problems which uses grid dominance and difference to determine the relationship between individual solutions.

The fifth category of MOEAs are decomposition-based approaches. These MOEAs work by optimizing a population of evenly distributed weight vectors in the objective space. This has the effect of changing the multi-objective optimization problem to a population of single objective optimization problems. Zhang and Li (2007) propose a decomposition MOEA called MOEA/D that uses a Tchebycheff approach to optimize a set of evenly distributed weight vectors (i.e., meaning that weight vectors are evenly spaced in objective space). Each weight vector represents a specific trade-off of objectives. MOEA/D has been shown to outperform several competing algorithms (e.g., NSGA-II) and has become the most popular decomposition-based approach. Li, Deb et al. (2017) propose a progressive version of MOEA/D using a non-uniform mapping scheme, which requires the decision maker to interactively define ROIs throughout the optimization process. Decomposition-based MOEAs have been shown to outperform competing algorithms and have demonstrated the ability to address both standard MOPs, as well as, many objective MOPs with similar levels of effectiveness.

MOEAs can further be categorized based on how user preferences are integrated into the algorithm (Cohon and Marks 1975). A priori MOEAs, involve articulating user preferences through surrogate functions before the search begins. They have the drawback of unnecessarily limiting the search space, because designer preferences (based on limited information) are encoded before the search. A posteriori MOEAs integrate user preferences after the search is finished. Because they do not have the burden of encoding designer preferences, as with priori techniques, they are the most widely studied. There are several prominent MOEAs in this category: Multi-Objective Genetic Algorithm (MOGA); NSGA I and NSGA II; Pareto Archived Evolutionary Strategy (PAES); Strength Pareto Evolutionary Algorithm (SPEA).

Progressive MOEAs allow the search and decision making to occur simultaneously. These are the most interactive of the techniques and consequently have the drawback of being more labor intensive for the decision maker. These techniques use the decision maker's preferences throughout the search process to help guide the MOEA through an objective space, therefore, offering the ability of navigating much more complex objective spaces than the other techniques mentioned. Research has also shown that progressive techniques can be more effective at solving deceptive design problems than a priori, or a posteriori, methods (Christman and Woolley 2015).

The distinction between global and local search, or exploration verses exploitation, is important in the study of MOEAs. Global search implies a more explorative and less precise search of the global objective space. Local search implies a narrower and more precise search of a portion of an objective space. Global search techniques are less prone than local search techniques to prematurely converge on suboptimal local maxima, or minima, but such convergence is still a major problem. MOEAs are considered a global search technique and, consequently, premature convergence and maintaining diversity in the solutions contained in the POS are significant problems for the field (Coello, Van Veldhuizen et al. 2002). In addition, there has been much work to improve the local search ability of MOEAs by hybridizing them with local search algorithms (e.g., simulated annealing; hill climbing) to create so-called “mimetic algorithms” (Goh, Ong et al. 2009). Balancing exploration and exploitation in these algorithms, and in MOEAs in general, is an important problem in the field. Too much solution diversity leads to poor convergence, and too little leads to premature convergence.

Progressive MOEA techniques have shown success in addressing both deceptive and many-objective design problems, and are therefore explored in this research. Specifically, the NSGA-II has been demonstrated to be one of the most effective MOEAs for addressing deceptive problems (Deb, Pratap et al. 2002). The development of more robust global and local search capabilities for the NSGA-II are still pressing problems in the field. In addition, the problem of when and how the decision maker should interact with the MOEA is one that is still unresolved and understudied. The development of the NSGA-III attempted to address this issue, but only interacts with the user to define a global search direction. This research proposes a unique MOEA implementation that is a modification of the NSGA-II algorithm. Our algorithm called the dynamic progressive NSGA-II (DP-NSGA-II) is a mimetic algorithm that allows for user-guided global and local searches.

2.5 Preference-Integration

Research into the incorporation of user preferences can be divided into six categories as listed in Table 2-1: weights; ranking solutions; ranking objectives; trade-off; reference point; reference direction approaches. The first category uses a weight-based approach in which the decision maker supplies weight information relative to each objective either a priori or progressively. Deb (2003) proposes the use of weighted objective functions and a modified sharing mechanism for the Nondominated Sorting Genetic Algorithm (NSGA) to bias the distribution of the search towards a preferred area of the Pareto front. Branke and Deb (2005) use a user defined weight vector to modify the crowding distance calculation in NSGA-II. Jin, Okabe, & Sendho (2001) convert linguistic terms via fuzzy logic given by a decision maker into a weighting scheme that redefined the objective functions as a series of single-objective weighted sums, which could guide the search towards a ROI. Wagner & Trautmann (2010) propose the use of desirability functions defined by the decision maker to replace objective functions in the optimization process. An important drawback of weight-based approaches, is that they do not scale effectively when the number of objective becomes large. Specifically, it becomes difficult for the decision maker to supply weights for each objective. It is also difficult to control the spread of the ROI and define multiple ROIs.

Table 2-1 Comparison of preference-based MOEA methodologies. This table has been adapted from the classification proposed by Bechikh, Kessentini et al. (2015).

Preference-Integration Technique	**Modification**	**Preference-Integration**	**Spread Control**	**Scalability**	**Diversity Problems**	**MROI**	**Supports Dynamic Changes**
Weights							
Deb (2003)	Biased Distribution	A Priori	N	N	Y	N	N
Branke and Deb (2005)	Crowding Operator	A Priori	Y	Y	N	N	N
Jin, Okabe et al. (2001)	Objective Aggregation	A Priori	N	N	Y	N	N
Wagner and Trautmann (2010)	Objective Functions	A Priori	N	Y	N	Y	N
Ranking Solutions							
Deb, Sinha et al. (2010)	Dominance	Progressive	N	Y	N	N	N
Koksalan and Karahan (2010)	Dominance	Progressive	N	N	N	N	N
Ranking Objectives							
Jin and Sendhoff (2002)	Objectives Aggregation	A Priori	N	N	Y	N	N
Cvetkovic and Parmee (2002)	Dominance	A Priori	Y	N	N	N	N
Rachmawati and Srinivasan (2010)	Dominance	A Priori	N	Y	N	N	N
Trade-offs							
Branke, Kaußler et al. (2001)	Objective Functions	A Priori	N	N	Y	N	N
Reference Point							
Deb and Sundar (2006)	Crowding Operator	A Priori / Progressive	Y	Y	Y	Y	N
Allmendinger, Li et al. (2008)	Leader Selection Strategy	A Priori / Progressive	Y	N	Y	Y	N
Thiele, Miettinen et al. (2009)	Quality Indicator	A Priori / Progressive	N	Y	N	Y	N
Reference Direction							
Deb and Kumar (2007)	Solution Sorting Mechanism	A Priori / Progressive	Y	Y	Y	Y	N
Deb and Kumar (2007)	Crowding Operator	A Priori / Progressive	Y	Y	Y	Y	Y

The second category of approaches integrates the decision maker's preferences by having them rank given solutions during a progressive process. This ranking is then used to define a weighting vector for the objective functions. Deb, Sinha, Korhonen, & Wallenius (2010) propose a progressively interactive MOEA that presents the decision maker with a sample of solutions every *n* generations, which the decision maker then ranks from best to worst. A value function is then derived from this ranking and used to drive the search. Koksalan & Karahan (2010) use an interactive Territory Defining Algorithm (iTDEA), which progressively elicits and uses a ranking of solutions by the decision maker to adaptively adjust the density of solutions in specific territories along the PF. Solution ranking has major drawbacks when the number of objectives becomes large due to the increased burden placed on the decision maker to provide ranking information. Further, solution ranking can be complicated when decision makers give inconsistent and conflicting rankings.

The third category is objective ranking-based approaches in which the decision maker ranks the objective either a priori, or progressively, and this ranking is used to create a weight vector that will bias the search. Jin and Sendhoff (2002) ask the decision maker to make pair-wise comparisons between a set of objectives by using linguistic statements. These preferences are converted into interval-based weights to drive the search process. Cvetkovic & Parmee (2002) use linguistic terms as well given by the decision maker to rank and compare objectives in a pair-wise manner as more or less important. As the number of objectives increases, objective ranking can run into problems because of the increased burden on the decision maker and the difficulty of pair-wise comparisons when the number of objectives goes over three. In addition, it can be difficult to control the spread of the ROI and to explore multiple ROIs at once.

The fourth category of preference-integration involves the decision maker defining intervals of acceptable and non-acceptable trade-offs between objective functions. For example, for a bi-objective case the decision maker could specify that one unit of improvement in objective one is worth a degradation in objective two by three units. Branke, Kaußler, & Schmeck (2001) follow this process to quantify acceptable tradeoffs which are then used to modify the dominance relation in a NSGA. This approach is limited to a bi-objective case and also may not be able to effectively explore non-linear objective spaces. Controlling spread and the ability to explore multiple ROIs is also limited.

The fifth category involves the use of a decision maker provided reference point in objective space to focus the search on a specific ROI, or multiple ROIs. Deb & Sundar (2006) propose a modified version of the NSGA-II called R-NSGA-II which uses a user defined reference point to modify the crowding distance calculation. Solutions closer to the chosen reference point are ranked higher and assigned a lower crowding distance and given preference during the selection process. Allmendinger, Li, & Branke (2008) use a reference point approach with a particle swarm process in which the decision maker can define multiple reference points and the algorithm can then converge on multiple ROIs. Because reference point approaches ask the decision maker to designate a single point of interest instead of providing multiple weighting, ranking, or trade-off values, they offer an advantage over other approaches when it comes to scalability to more than three objectives. In addition, they also provide for a greater ability to define and search multiple ROIs simultaneously and to control the spread of the ROIs. Their drawbacks include a tendency to lose diversity in the solution set and also defining a proper reference point is not trivial.

The last category of preference-integration requires a decision maker to specify a reference direction for the search. Deb & Kumar (2007) use a reference method with NSGA-II in which the decision maker is asked to set a starting point and also a reference point. A vector is then created between these two points which is then used to drive the search towards a specific ROI. In another paper by Deb & Kumar (2007), the decision maker is asked to define an aspiration point and a reservation point, which is then used as the direction for the search. This direction represents the aspirational objective values for the search as defined by the decision maker. Solutions closer to this vector are rewarded more. Li, Deb et al. (2017) demonstrate the ability to precisely control the spread of the ROI by using a decomposition-based MOEA and a biased distribution to control the extent of multiple ROIs. Reference direction approaches have been shown to scale well to problems with many objectives and they have also demonstrated an ability to provide control of the spread of the ROI. These approaches, however, can suffer from diversity problems and can give poor results when a direction is chosen in a portion of the objective space that is discontinuous.

These preference-integration strategies can further be divided by two criteria: 1. those that promote active learning of the trade-offs possible within a particular objective space by the decision maker and those that are less effective in this respect; 2. those that allow for greater exploration and diversity of solutions and those that tend toward convergence. In terms of the first criteria, solution ranking stands out as an approach that encourages the decision maker to actively look at potential solutions, view the trade-offs between solutions, and then provide a ranking that points towards the best trade-off direction to explore. In effect, the decision maker is defining a reference direction based on the comparison and contrast of real-time information about the objective space provided by the search. This interaction engages the decision maker in a manner that requires a level of comparative thought that other approaches involving a reference point, direction, or weighting may not provide.

In terms of exploration, the following features listed in Table 2-1 have the most impact: ability to maintain diversity; control of the spread of the ROI; scalability; ability to search multiple ROIs. Reference direction approaches score well in almost all these areas except in maintaining diversity. Solution ranking on the other hand performs more poorly in these areas.

In order to address these weak areas and to combine the best of both approaches, a hybrid strategy called enhanced solution ranking is proposed. In this process reference directions are used at the global scale to narrow the search to a few ROIs and solution ranking is then used at the local level in the area around each ROI to allow the decision maker to guide the search locally. This approach works especially well in high dimensional objective spaces where tradeoffs may change drastically and non-linearly at a global scale, but may change less drastically at a local scale.

2.6 Dynamic Multi-Objective Evolutionary Algorithms

Many real-world multi-objective problems are dynamic and have objectives, design parameters, and constraints that change during the optimization process. This is especially true for the conceptual design phase of the design process in which goals, parameters, and constraints may change often and unexpectedly. These problems are known as dynamic multi-objective problems (DMOPs) and can be defined more formally for a minimization problem as follows:

$$\min_{\boldsymbol{x}} \boldsymbol{F}(\boldsymbol{x},t) = \{f_1(\boldsymbol{x},t), f_2(\boldsymbol{x},t), \dots, f_m(\boldsymbol{x},t)\} \backslash \boldsymbol{x} \in \Omega^n \ s.t.\ g(\boldsymbol{x},t) > 0, h(\boldsymbol{x},t) = 0 \qquad (11)$$

where $\boldsymbol{x}$ is a vector of decision variables; *t* represents time and the dynamic nature of the problem; $\boldsymbol{F}$ is the set of objective functions; f_m is a particular objective function; *m* is the number of objectives to be minimized; the functions g and h represent the set of inequality and quality constraints respectively. The min operator is returning the $\boldsymbol{PF}$ for a specified time step *t.* The $\boldsymbol{PF}$ is therefore a function of *t,* and *t* is held constant as the $\boldsymbol{PF}$ is solved for. This produces a sequence of $\boldsymbol{PF}$'s through time. DMOPs may change continuously or in discreet time steps. DMOPs pose different challenges for optimization approaches depending on the type of change (i.e., depending on whether *x, f, g* and *h*, or *m* change).

In order to address these problems, evolutionary algorithms have generally demonstrated the highest performance due to their inherent adaptability (Chen, Li et al. 2016). These algorithms are referred to as dynamic multi-objective evolutionary algorithms (DMOEAs) and following Table 2-2 can be categorized by three types of approach: diversity approaches; memory mechanism; and predictive.

When objectives change, solutions that were converging towards the PF can become irrelevant and can create a lack of diversity needed to find the PF of the new objectives. Diversity approaches attempt to deal with this problem by adding diversity into a solution population when changes to the objective functions occur. Deb, Rao N et al. (2007) propose a modified NSGA-II algorithm called D-NSGA-II that detects changes in the objective function and then randomly replaces a portion of the population to increase diversity. Azzouz, Bechikh et al. (2015) propose the individual diversity multi-objective optimization EA (IDMOEA) that maintains diversity by making diversity an objective and by storing diverse individuals in an archive that is draws from when a change is detected.

Memory-based approaches use an archive to store useful solutions from the history of the search. These solutions can be used when there is a detected change in the objective functions, constraints, or decision variables. Maintaining diversity is therefore a critical issue for these approaches. Goh and Tan (2009) propose competitive-cooperative EA which uses a population dedicated to diversity and several sub-populations focused on a different objective, or sub-problem, in the optimization in order to maintain diversity when an objective is changed. The approach has been shown to be effective, but the computational cost is high. Azzouz, Bechikh et al. (2017) propose a dynamic version of NSGA-II, called Dy-NSGA-II, which detects the severity and frequency of a change and then adaptively applies memory, local search, and random strategies to help convergence. This use of several strategies that adaptively react to the type of change is unique and the algorithm has been demonstrated to outperform many competing algorithms for changes that are both large and small. Chen, Li et al. (2016) deal with the problem of a changing number of objectives and use of a second population dedicated to diversity to add diversity to a current

Table 2-2. Comparison of DMOEAs categorized by diversity, memory mechanisms, and prediction approaches.

DMOEA Approaches	**Modification**	**Scalability**	**MROI**	**Dynamic Objectives**	**Changing # of objectives**	**Dynamic Decision Variables**	**Dynamic Constraints**	**Handle Severe Changes**
Diversity								
Deb, Rao N et al. (2007)	Detection & creation of random solutions	N	N	Y	N	N	N	Y
Azzouz, Bechikh et al. (2015)	Use of diversity objective and novelty archive	N	N	Y	N	N	N	Y
Memory Mechanisms								
Goh and Tan (2009)	Use of multiple sub-populations and a diversity population	N	N	Y	N	N	N	Y
Azzouz, Bechikh et al. (2017)	Adaptively applies memory, LS, and random search	N	N	Y	N	N	N	Y
Chen, Li et al. (2016)	Use of a diversity population	Y	N	Y	Y	N	N	Y
Prediction								
Hatzakis and Wallace (2006)	Use of feedforward prediction to predict PF	N	N	Y	N	N	N	N
Koo, Goh et al. (2010)	Use of predictive vectors to guide search	N	N	Y	N	N	N	N

population when objectives increase. When objectives decrease, they propose a density measure to help reduce duplicate solutions and increase diversity.

Predicative approaches use knowledge about the previous states of the PF to predict the likely state of the new PF. These approaches therefore only work for problems where the objective functions are changing in a predictable and incremental fashion. Hatzakis and Wallace (2006) propose the use of a feed-forward prediction strategy to predict changes in the PF and to create a population of solutions in the forecasted area. If the prediction is correct then convergence to the new PF will be fast and if the prediction is not correct random solutions are added to the population to help converge to the new PF. They combine this approach with an evolutionary algorithm to create Dynamic Queuing Multi-objective Optimizer (D-QMOO). Koo, Goh et al. (2010) propose a dynamic variant of the Multi-Objective Evolutionary Gradient Search (DMO-EGS) in which a set of predictive vectors are created based on the positions of previous PF solutions. These vectors are used to guide a current population of solutions on the PF to the new predicted PF.

The majority of the research on DMOEAs has been toward looking at problems where the objective functions themselves change through time. There has been very little research looking at the case when the number of objectives change through time. In fact, to our knowledge there have only been two papers addressing this issue (Guan, Chen et al. 2005, Chen, Li et al. 2016). There has also been little research looking at problems where the number of decision variables change through the optimization process (Ripon, Tsang et al. 2006, Kleeman and Lamont 2007, Ting, Lee et al. 2009) and no research looking at situations where both the number of objectives and the number of input parameters change simultaneously. To address this unstudied area, this research proposes a unique DMOEA implementation that is derived from the NSGA-II algorithm and following (Azzouz, Bechikh et al. 2017) uses an adaptive approach in which the type of change (e.g., change in objectives; changing number of objectives; change in decision space) and severity of the change is first detected and then a mixture of memory and prediction approaches is used.

There has also been very limited research on how user preferences might be integrated into DMOEAs. Deb, Rao N et al. (2007) propose an adapted a priori-based NSGA-II algorithm which encodes user preferences into a utility function that allows for automatic solution selection within the context of problems that have a constant number of objectives, but the objective function itself changes. Shen and Yao (2015) propose ε-MOEA that uses an a priori user defined weight function to guide the optimization of a DMOP that has a constant number of objectives, but changing objective functions. Research on preference-integration for DMOEAs has been done primarily with a priori approaches and to our knowledge no research has been done looking at how progressive approaches might be applied to DMOEAs. To fill this gap, this research proposes a progressive DMOEA which adaptively integrates user preference information in response to the type and level of severity of change. Depending on the severity of change, users will be prompted to either do nothing (e.g., changes in decision variables or objective functions that have small effects) or define a new global reference direction (e.g., number of objective functions increase; or if severe change in objective functions).

2.7 Progressive MOEAs in the Design, Engineering, and the Arts

In addition to the benefits offered by progressive MOEAs, there are also some important drawbacks that must be considered. According to Takagi (2001), user-fatigue is one major drawback. User fatigue begins to set-in at between 10-20 generations (assuming users are evaluating between 5-10 designs per generation). Another drawback is the lack of interactivity present in most progressive approaches. For example, in most implementations the decision maker is relegated to only interacting with the search during the selection stage of the algorithm. This limited interactivity keeps the decision maker only partially engaged and fails to make full use of their expertise to guide the search.

Progressive processes have been used extensively in problem domains involving MOPs and also those requiring qualitative assessments (e.g., beauty). They have been used in the arts (Sims 1992), music (Marques, Reis et al. 2010), fashion (Kim and Cho 2000), and multiple design fields as shown by Takagi's comprehensive review (Takagi 2001). In the design fields, they have been used for the conceptual design phase mostly. Mueller and Ochsendorf (2015) use a Pareto-based progressive MOEA to evolve truss designs and attempt to address the user-fatigue problem by showing only designs contained in the POS to the decision maker for selection. Mueller attempts to expand the interactivity of the MOEA by allowing the decision maker to interactively change the population size and mutation rate. Von Buelow (2012) follows the same approach to reduce user-fatigue, but also includes an ability for multiple users to evaluate designs at the same time. Von Buelow attempts to address the interactivity problem by allowing users to interactively navigate and select designs for breeding from a database of all created solutions during the search. Ohsaki and Takagi (1998) use user-trained surrogate models to represent user evaluations to reduce user fatigue and increase the size of the search space explored.

This precedent research addresses user fatigue through a spectrum of useful approaches, but still offers a very limited set of possibilities for interaction with a decision maker. In all these examples, the decision space being explored is fixed and the decision maker is relegated to picking designs during the selection phase or changing evolutionary parameters (e.g., population, generation, or mutation) during the search. In contrast, this research proposes the use of a unique progressive DMOEA that allows decision makers to interactively modify the decision space (by adding and subtracting design features) and the objective space (by adding and subtracting objective functions) during the optimization process. This engages the decision maker by allowing them the ability to actively design during the search process and to dynamically adjust the goals of the search as new information is gained. This combination of features allows for an open-ended evolutionary search process to occur, which has the potential to uncover a much greater variety of solutions than traditional EAs. In addition, interacting with the decision maker in this way may help to limit user fatigue by making the process more engaging.

2.8 MOEAs in the Conceptual Design Phase

The field of MOEA research is quite broad with a large variety of algorithms that have been developed and tested for multiple types of design problems and in multiple fields for the conceptual phase of design (e.g., architecture, aerospace, automotive, civil engineering). In the field of architectural design, Von Buelow (2012) uses a MOEA with a parametric Computer Aided Design (CAD) system to evolve building designs with several quantitative and qualitative objective functions relating to daylighting, energy, and aesthetic criteria. Turrin, von Buelow et al. (2012) demonstrate their use in the design of roof structures that are optimized for daylighting, energy, structural load, and aesthetics. Mueller and Ochsendorf (2015) use them in the field of structural engineering to evolve truss designs relative to mass, volume, structural, and aesthetic objective functions. Obayashi, Sasaki et al. (2000) use a MOGA to optimize the shape of aircraft wings. Muyl, Dumas et al. (2004) use a hybrid MOEA to explore the aerodynamic performance of car shapes. Brintrup, Ramsden et al. (2008) use quantitative and qualitative objective functions to evolve ergonomic chair designs.

There are also several commercial CAD packages that feature various MOEA implementations, but all have significant drawbacks. Dassault Systems offers several different parametric CAD packages (e.g., SolidWorks, Catia, and Digital Project) that feature MOEA-based optimization tools for the fields of mechanical, aerospace, automotive, and architectural design. Autodesk currently offers cloud-based optimization tools for their structural, mechanical, and architectural CAD products. Autodesk's parametric CAD plugin Dynamo can be used to optimize designs in Revitt with a plugin called Optimo. McNeel's Rhinoceros 3D includes a single objective EA called Galapagos with its parametric modeling plugin Grasshopper, and there is also a MOEA plugin available called Octopus (Vierlinger and Bollinger 2014). Ansys's suite of engineering design softwares have a MOEA-based optimization platform as well as other optimization plugins available like DesignXplorer. Bentley System's SITEOPS is a tool to explore the configuration possibilities of building sites optimized for geotechnical, hydrological, topographical, legal, and architectural objectives. ESTECO offers an optimization software called modeFrontier which contains implementations of several standard MOEAs for design problems.

The commercial tools and previous research discussed share several drawbacks. The MOEAs implemented in these various works have no design features that specifically help to deal with DMOPs. They also have no features that help to drive a more explorative and open-ended global search. Further, they all have a very limited capacity to interact with the user. The research presented in Chapter 3 directly addresses these issues through the description of a unique progressive DMOEA.

Chapter 3. DP-NSGA-II Algorithm and Implementation

In this chapter, a new modification to NSGA-II is described. The resultant algorithm is called the dynamic progressive NSGA-II (DP-NSGA-II). The DP-NSGA-II is the first DMOEA to deal with the problem of a changing number of objective functions and decision variables simultaneously. The algorithm addresses the dual problems of convergence and diversity of solutions when changes occur through a combination of memory-based and prediction approaches. Further, it represents the first DMOEA to integrate the user during the optimization process. The DP-NSGA-II integrates user preferences in a progressive fashion through a reference-direction approach to guide the search at the global scale, and a solution ranking approach to guide the search at the local scale. This combined global and local approach is also unique.

3.1 DP-NSGA-II Algorithm Overview

Figure 3-1 shows a diagram of a typical Pareto selected based DMOEA, while Figure 3-2 shows in comparison the changes we propose for our algorithm. One important difference is that user preference information is integrated into the search both in an apriori and progressive fashion through the use of a user-defined reference direction at the global scale and a solution ranking approach at the local scale as described in Section 2.5. User preference information is also integrated adaptively in response to the type and level of severity of change in the objectives and decision variables as described Sections 2.6. The other key difference is that the DP-NSGA-II is designed with unique procedures to deal with time-dependent changes in objective functions, changes in the number of objective functions, changes in the number of decision variables, and combinations of these changes that occur simultaneously.

The pseudo code of the DP-NSGA-II is shown in Figure 3-3. DP-NSGA-II uses a Pareto selection approach based on dominance depth (Coello, Van Veldhuizen et al. 2002) and the system only enforces boundary constraints on its decision variables. DP-NSGA-II uses a real number representation of decision variables instead of a binary representation. This approach reduces the degree to which child solutions differ from parents after applying crossover operations, and allows the algorithm to perform a more fine-grained search in high dimensional objective spaces.

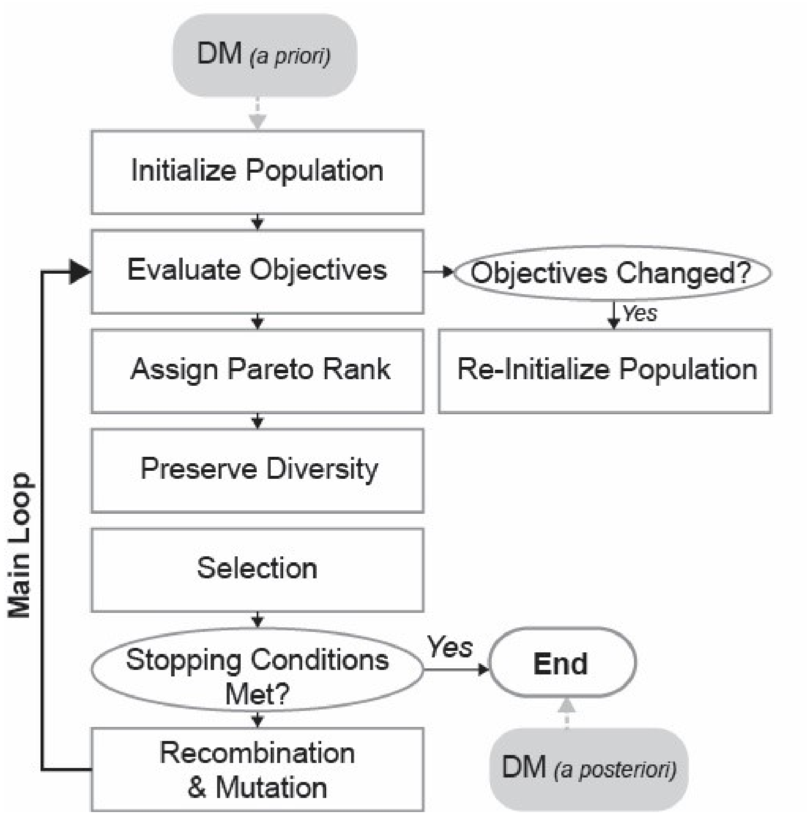


Figure 3-1. Diagram of a standard Pareto selection-based MOEA.

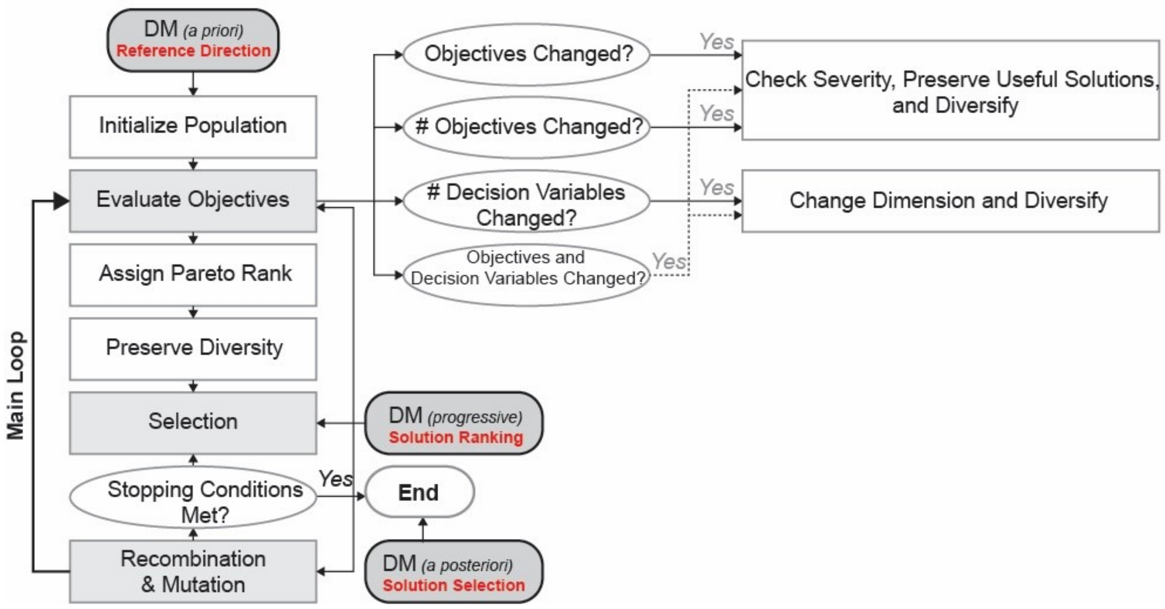


Figure 3-2. Diagram of the proposed DP-NSGA-II algorithm.

Algorithm 1 DP-NSGA-II Algorithm

```
1: INPUT: S (Population Size), $G_{max}$ (Maximum Generations), Rg (User-defined Global Reference Direction), Rl, (User-defined Local Reference Direction), F ( Objective Functions); D (Decision Variables)
2: OUTPUT: A (Archive of Nondominated Solutions), N (Novelty Archive), G (Generational Archive)
3: BEGIN
4: P ← RandomInitialize(S)    ▷ Randomly initialize a population.
5: $gen \leftarrow 1$
6: Rl ← Rg
7: ObjectiveEval(P)
8: P_ranked ← ParetoRank(P,Rg)
9: CalculateNovelty(P_ranked, G)
10: Parents ← BinaryTSelection(P_ranked)
11: Children ← Reproduction(Parents)
12: Children ← Mutation(Children)
13: Pt ← Children
14: while $(gen < G_{max})$ do
15:     ChangeDetectionResponse(A, N, Pt, F, D,Rg, Rl)
16:     ObjectiveEval(Pt)
17:     P_ranked ← ParetoRank(Pt,Rg)
18:     AddToParetoArchive(P_ranked)
19:     Pt← CombineSortByRank(Children, P_ranked)
20:     CalculateNovelty(Pt, G)
21:     AddToGenerationalArchive(Pt)
22:     NS ← SelectMostNovel(Pt, G)
23:     AddToNoveltyArchive(NS)
24:     if CheckConvergence(A) then
25:         Rl ← UserSolutionRanking(PF)
26:     if HasGoalBeenMet(A) then
27:         Break Loop
28:     Parents ← BinaryTSelection(Pt, Rl)
29:     Children ← Reproduction(Parents)
30:     Children ← Mutation(Children)
31:     $gen \leftarrow gen + 1$
end
```

Figure 3-3. Pseudo code of DP-NSGA-II algorithm.

3.2 Objective and Novelty Evaluation

As Figure 3-3 shows, the algorithm starts with the following input parameters: S (population size); G_{max} (maximum number of generations); $\boldsymbol{Rg}$ (user defined global reference direction); $\boldsymbol{Rl}$ (user defined local reference direction); $\boldsymbol{F}$ (array of objective functions); $\boldsymbol{D}$ (array of decision variables). The vector $\boldsymbol{Rg}$ represents a user defined global reference direction to guide the search and is given before the optimization process starts (i.e., a priori). This reference direction is especially useful to guide the search when the number of objectives is greater than three. The optimization process then starts with a population of solutions being randomly initialized by sampling a uniform random distribution. Then the local reference direction used to direct the search at a local scale $\boldsymbol{Rl}$ is initialized with the same direction as the global reference direction. The generated solutions are then evaluated based on the given objective functions.

The population is then ranked into a series of fronts based on the Pareto dominance measure defined in Definition 7 and the global reference direction $\boldsymbol{Rg}$. A well-known problem with Pareto ranking is that it can fail when the number of objectives goes above three (Deb and Sundar 2006, Bechikh, Elarbi et al. 2017). To address this issue, DP-NSGA-II uses a reference direction-based approach as described in Section 2.5. To define $\boldsymbol{Rg}$, the user chooses a weight vector for the objectives. This weight vector is used to define a single objective optimization problem that is iteratively solved with each generation to find an optimum point $\boldsymbol{P}_{ideal}$ in objective space located along the direction defined by $\boldsymbol{Rg}$. This process runs parallel with the MOP optimization. The P_{ideal} point is then used to modify the dominance ranking process. Solution X now dominates solution Y if one of the following criteria are met:

1. X is Pareto dominant to Y
2. X and Y are the same rank, but X is closer to $\boldsymbol{P}_{ideal}$ based on Euclidean distance.

After ranking, a novelty score is then calculated for each solution to promote exploration and encourage diversity. Precedent research has shown that incorporating a novelty score can dramatically improve the explorative capacity of MOEAs (Lehman and Stanley 2011). DP-NSGA-II calculates novelty based on Equation 1 listed in Chapter 2. Solutions are randomly sampled from the generational archive $\boldsymbol{G}$, which holds all past solutions created during the search. For each solution in the current population, the distances in objective space of its k nearest neighbors of these samples of $\boldsymbol{G}$ are calculated and then averaged. This value is used as the novelty score for the solution. This random sample-based approach saves evaluation time, and research by Gomes, Mariano et al. (2015) demonstrates that using such an approach is just as effective as using the entire population of previous solutions for the novelty calculation.

3.3 Selection and Reproduction

Binary tournament selection is then run on the ranked population to select candidates for reproduction. The process works by randomly picking two individuals from the ranked population. The solution with the lowest rank is added to the mating pool. If the solutions have the same rank, the solution with the highest novelty measure is chosen. This is done n times.

For the first generation, this process is run without ***Rl***, but in the proceeding generations if there is a value for ***Rl***, then it is used to bias the selection process. This means that if two solutions of the same rank are chosen, then the solution closer to ***Rl*** will be selected for reproduction. The binary tournament selection process produces a set of solutions that are then used for reproduction.

DP-NSGA-II randomly samples from the selected solutions in the previous step and alternates between the use of midpoint crossover and averaging as its reproduction operators. In midpoint crossover, the decision variables of a parent chosen is split down the middle and half of it is copied to its offspring. The other half of the child's decision variables are then contributed by the second parent through the same operation. In averaging, both genes of parents are averaged together after selection. Alternating between both approaches allows for more diversity in the solutions. Mutation is then randomly applied to the decision variables of newly produced child solutions to promote diversity and exploration based on a probability that is adjustable by the decision maker.

The resultant population of solutions, ***Pt***, are then brought into the main optimization loop where procedures for dealing with dynamic changes to the objectives and decision variables are used to aid convergence and diversity. Solutions are then evaluated, ranked, selected, reproduced and mutated as before.

3.4 The Nondominated, Novelty, and Generational Archives

The nondominated archive ***A*** and the novelty archive ***N*** are used in the optimization process to aid convergence and exploration when objectives or decision variables change. The ***A*** archive stores all nondominated solutions found during the optimization process and the ***N*** archive stores the most novel solutions found. Theses archives are used in a memory mechanism-based approach (defined in Section 2.6) to respond to dynamic changes in objectives and decision variables. The details of their application are discussed in Section 3.6.

The generational archive ***G*** stores all the solutions generated by the search process and is used by the decision maker to interactively explore the history of the search process. The decision maker can choose to move the search back to any point in the history of the process, modify decision variables or objectives, and then continue searching from that modified historical point. The ability to review this search history allows the decision maker to identify trends and areas in the objective space that might be underexplored.

3.5 User-Defined Local Search through Solution Ranking

During each iteration of the main loop of the algorithm a convergence check is made by comparing the values in the nondominated archive ***A*** with previous states of the archive. If the new values added to the ***A*** archive have not changed over the previous three generations, then convergence is assumed around the region of interest defined by $\boldsymbol{Rg}$, and the user is prompted to rank a uniform sampling of nondominated solutions from the ***A*** archive. This ranking is used to create a weight vector ***Rl*** that biases the binary tournament selection process, so that solutions close to this weight vector are preferred over solutions of the same rank but further away. The local reference direction ***Rl***, therefore, works in combination with the global reference direction ***Rg*** to guide the search.

This hybrid global-local approach is unique to the DP-NSGA-II and encourages the decision maker to actively look at potential solutions, view the trade-offs between them, and then steer the search towards the most promising regions. This interaction engages the decision maker in a manner that requires a level of comparative thought that other approaches involving just a single global reference point, reference direction, or weighting function may not provide.

3.6 Change Detection and Response

In order to ensure that a design space is being adequately explored, maintaining a high level of diversity in an evolving population of candidate solutions is essential. A well-known problem with DMOEAs is that they can suffer from a lack of diversity when constraints, objectives, or decision variables change (Datta and Gupta 2016), which can cause the search to prematurely converge. Specifically, as the underlying search space changes, the existing solutions are unable to span the new search space effectively. To address this issue, DP-NSGA-II uses a unique procedure which allows it to adaptively detect and respond to seven different cases in which objectives and decision variables may change individually or simultaneously. Specifically, the algorithm looks to detect and respond to the following: changes in the number of objective functions; time-dependent changes in the objective functions; changes in the number of decision variables; and all possible combinations of these cases. Because each of these types of changes effects the convergence and diversity of the optimization process in unique ways, specific mechanisms are tailored and employed to address each type of change. These mechanisms will be summarized in this section and described in more detail in the following sections.

The pseudo code for the algorithm can be seen in Figure 3-4. The first step in the algorithm is to compute if a change has occurred and to determine what the change is. If a change has occurred it will fit into one of seven possibilities:

- Case 1: The number of objectives has increased or decreased. Algorithm 3 shown in Figure 3-7 and detailed in Section 3.6.1 is called when a change of this type occurs.
- Case 2: Time-dependent changes in objective functions. Algorithm 4 shown in Figure 3-8, and detailed in Section 3.6.2 is called when a change of this type occurs.
- Case 3: The number of decision variables has changed. Algorithm 5, which is detailed in Section 3.6.3 and in Figure 3-9 is called when a change of this type occurs.
- Case 4: Both Case 1 and Case 2. Algorithm 4 and 3 are called in the given order when a change of this type occurs.
- Case 5: Both Case 2 and Case 3. Algorithm 5 is applied.
- Case 6: Both Case 1 and Case 3. Algorithm 5 and 3 are called in the given order when a change of this type occurs.
- Case 7: Cases 1 through 3 are all true. Algorithm 5 and 3 are called in the given order when a change of this type occurs.

Algorithm 2 Change Detection and Response Algorithm

```
1: INPUT: A(archive of non-dominated solutions), N(Novelty archive), Pt(current population of solutions), F(array of objective functions), D(array of decision variables), Rg(user defined global reference direction for search), Rl(user defined local reference direction)
2: OUTPUT: A, N, Pt, F, D, Rg, Rl
3: BEGIN
4: Switch (DetectChangeType(F, D))
5:      case 1: (number of objectives changed)
6:     ChangingNumberOfObjectives(A, N, Pt, F, D, Rg, Rl)
7:      case 2: (objective functions changed)
8:     TimeDependentObjectiveChange(A, N, Pt, F, D, Rg, Rl)
9:      case 3: (decision variables changed)
10:    DecisionVariablesChange(A, N, Pt, F, D, Rg, Rl)
11:     case 4: (objective number and objectives changed)
12:    TimeDependentObjectiveChange(A, N, Pt, F, D, Rg, Rl)
13:    ChangingNumberOfObjectives(A, N, Pt, F, D, Rg, Rl)
14:     case 5: (objectives and dec vars changed)
15:    DecisionVariablesChange(A, N, Pt, F, D, Rg, Rl)
16:     case 6: (objective number and dec vars changed)
17:    DecisionVariablesChange(A, N, Pt, F, D, Rg, Rl)
18:    ChangingNumberOfObjectives(A, N, Pt, F, D, Rg, Rl)
19:     case 7: (everything changed)
20:    DecisionVariablesChange(A, N, Pt, F, D, Rg, Rl
21:    ChangingNumberOfObjectives(A, N, Pt, F, D, Rg, Rl)
22:     case 8: (No change has occurred)
23:    do nothing
24: End
```

Figure 3-4. Pseudo Code for the change detection and response algorithm for DP-NSGA-II.

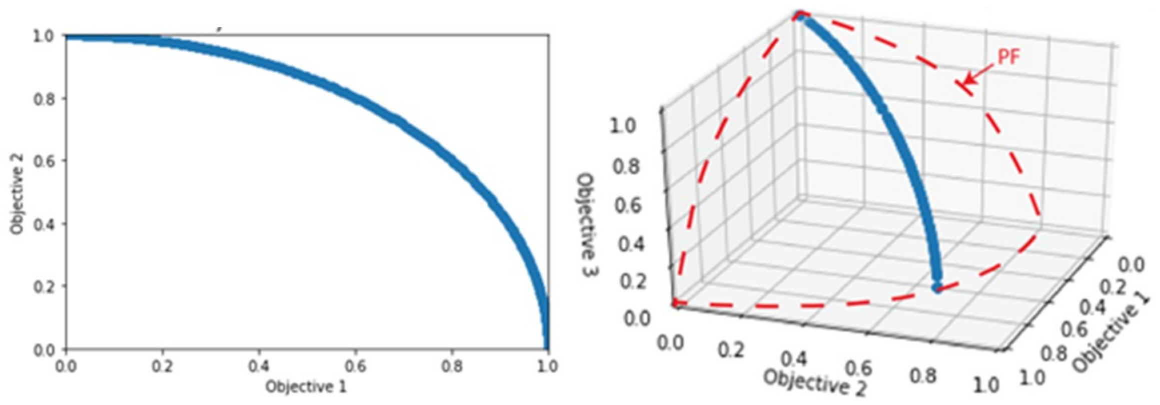


Figure 3-5. The left image shows distribution of nondominated solutions for DTLZ2 for two objectives. The right image shows the distribution of solutions after an objective is added.

3.6.1 Changing Number of Objectives

When the number of objectives increases, the problems posed to convergence and diversity can be clearly seen in Figure 3-5. After the change, the image on the right of the figure shows that the nondominated solutions that spanned the previous objective space are now spanning only a tiny area of the true Pareto front indicated with the red dashed-lines. The result is that both diversity and convergence are negatively affected. To address this issue, the DP-NSGA-II uses the procedure outlined in Figure 3-7. When the objectives increase in number a memory-mechanism approach is taken in which the current population Pt is replaced by 50% random samples from the nondominated archive A and 50% random samples from the novelty archive N. This approach allows some non-dominated solutions to remain a part of the current population to aid convergence. By sampling from the novelty archive, diversity is then promoted. The last step involves prompting the user to redefine a global reference direction Rg to help steer the search and then setting this equal to the local reference direction Rl. This last step is run if user interaction is enabled.

When the number of objectives decreases, diversity becomes a major problem for MOEA optimization processes. Figure 3-6 shows an example of this situation for a distribution of nondominated solutions found after 300 generations on the dynamic benchmarking problem set DTLZ2. The left side of the figure shows the distribution found for three objectives and the right side of the figure shows how the distribution changes when the number of objectives decreases from three to two. After the change, solutions are very close to one another and there are many duplicates. This lack of diversity can cause the optimization process to converge prematurely and to miss nondominated solutions. DP-NSGA-II addresses this problem by using the ***A*** archive of nondominated solutions to ensure convergence, while pruning it for duplicate solutions to help diversity. Diversity is further addressed by the algorithm through sampling form the ***N*** archive. The algorithm shown in Figure 3-7 outlines this procedure.

3.6.2 Time-Dependent Objective Functions

When the objective values themselves change in a time-dependent manner, the severity of the change in the objective functions is first determined, as shown in Figure 3-8. This is done by first taking a random sample of solutions from the ***A*** archive and comparing their objective values before and after a potential change. The average difference between old and new values for each objective is calculated and an average scaling factor computed. A new group of random samples is then taken from the ***A*** archive, and the scaling factor found in the previous step is applied to their values from before the change. This creates a set of estimated objective values that can be compared against a second set of objective values produced after the time-dependent change. If the predicted values are within an error threshold ε of the real objective values after the change, then the change is considered minor. To save computational time, the objective values for the solutions in the ***A*** archive are scaled instead of recomputed and used in the optimization process. The ***A*** archive is then randomly sampled to create the current population ***Pt***. The error threshold ε needs to be adjusted according to the specific optimization problem being addressed.

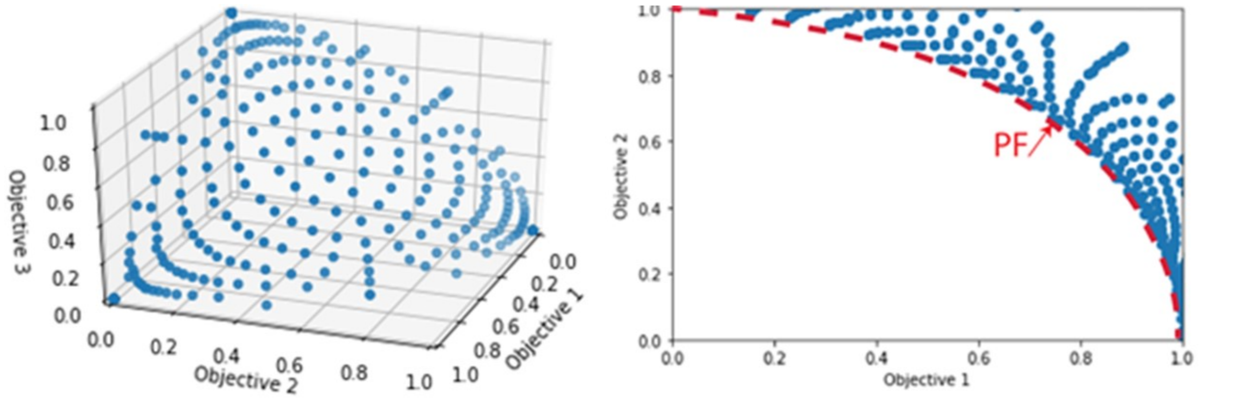


Figure 3-6. The left image shows distribution of nondominated solutions for DTLZ2 for three objectives. The right image shows the distribution of solutions after an objective has been subtracted.

```
Algorithm 3 ChangingNumberOfObjectives
1: INPUT: A(archive of non-dominated solutions), N(Novelty archive), Pt(current population of solutions), F(array of objective functions), D(array of decision variables), Rg(user defined global reference direction for search), Rl(user defined local reference direction)
2: OUTPUT: A, N, Pt, F, D, Rg, Rl
3: if number of objectives increase then
4:     $\mathbf{Pt} \leftarrow \mathit{RandomSample}(\mathbf{A}) + \mathit{RandomSample}(\mathbf{N})$
5:     $\mathbf{Rg} \leftarrow \mathit{UserPrompt}()$
6:     $\mathbf{Rl} \leftarrow \mathbf{Rg}$
7: else if number of objectives decreases
8:     $\mathbf{A} \leftarrow \mathit{RemoveSimiliarSolutions}(\mathbf{A})$
9:     $\mathbf{Pt} \leftarrow \mathit{RandomSample}(\mathbf{A}) + \mathit{RandomSample}(\mathbf{N})$
10: Return
```

Figure 3-7. Pseudo Code for dealing with a changing number of objectives for DP-NSGA-II.

If the changes are larger than ε, then the change is considered severe and ***Pt*** is populated with random samples from the ***N*** archive. The ***N*** archive contains the most diverse solutions found so far in the search, and in a situation where the objective space changes dramatically, these diverse solutions can help ensure a well spread search for new nondominated solutions. In the last step of the procedure, the ***A*** archive is then emptied because its solutions can no longer be guaranteed to be nondominated.

Algorithm 4 TimeDependentObjective

```
1:  INPUT: A(archive of non-dominated solutions), N(Novelty archive), Pt(current population of solutions), F(array of objective functions), D(array of decision variables), Rg(user defined global reference direction for search), Rl(user defined local reference direction)
2:  OUTPUT: A, N, Pt, F, D, Rg, Rl
3:  $\mathbf{A}_{prev} \leftarrow RandomSample(\mathbf{A})$
4:  $\mathbf{A}_{curr} \leftarrow Object\_Eval(\mathbf{A}_{prev})$
5:  $ScalingFactor \leftarrow GetScalingFactor(\mathbf{A}_{prev}, \mathbf{A}_{curr})$
6:  $\mathbf{A}_{prev} \leftarrow ScaleObjectiveValues(\mathbf{A}_{prev}, ScalingFactor)$
7:  $Error \leftarrow FindPredictionError(\mathbf{A}_{prev}, \mathbf{A}_{curr})$
8:  if $Error < \epsilon$ then                          ▷ if change is minor
9:      $\mathbf{A} \leftarrow ScaleObjVectors(\mathbf{A})$
10:     $\mathbf{Pt} \leftarrow RandomSample(\mathbf{A}) + RandomSample(\mathbf{N})$
11: else                                                ▷ if change is large
12:     $\mathbf{Pt} \leftarrow RandomSample(\mathbf{N})$
13:     $\mathbf{A} \leftarrow 0$
14: Return
```

Figure 3-8. Pseudo Code for dealing with time-dependent objective functions for DP-NSGA-II.

Algorithm 5 DecisionVariablesChange

```
1:  INPUT: A(archive of non-dominated solutions), N(Novelty archive), Pt(current population of solutions), F(array of objective functions), D(array of decision variables), Rg(user defined global reference direction for search), Rl(user defined local reference direction)
2:  OUTPUT:P
3:  $\mathbf{N} \leftarrow ChangeDimension(\mathbf{N})$
4:  $\mathbf{A}_{prev} \leftarrow RandomSample(\mathbf{A})$
5:  $\mathbf{A}_{curr} \leftarrow Object\_Eval(\mathbf{A}_{prev})$
6:  $ScalingFactor \leftarrow GetScalingFactor(\mathbf{A}_{prev}, \mathbf{A}_{curr})$
7:  $\mathbf{A}_{prev} \leftarrow ScaleObjectiveValues(\mathbf{A}_{prev}, ScalingFactor)$
8:  $Error \leftarrow FindPredictionError(\mathbf{A}_{prev}, \mathbf{A}_{curr})$
9:  if $Error < \epsilon$ then                          ▷ if change is minor
10:     $\mathbf{A} \leftarrow ChangeDimension(\mathbf{A})$
11:     $\mathbf{A} \leftarrow ScaleObjVectors(A)$
12:     $\mathbf{Pt} \leftarrow RandomSample(\mathbf{A}) + RandomSample(\mathbf{N})$
13: else                                                ▷ if change is large
14:     $\mathbf{Pt} \leftarrow RandomSample(\mathbf{N})$
15:     $\mathbf{A} \leftarrow 0$
16: Return
```

Figure 3-9. Pseudo Code for dealing with changing number of decision variables for DP-NSGA-II.

3.6.3 Changing Number of Decision Variables

Indicated in Figure 3-9 is the procedure that is followed when the decision variables increase or decrease in number. The first step in the process involves applying the *ChangeDimension* routine to change the dimension of the novelty archive ***N***. In the case of an increase in the number of decision variables, the *ChangeDimension* routine adds a new parameter to each solution in ***N*** and randomly initializes it based on a uniform distribution from the domain of the new input parameter. If the number of decision variables has reduced, then the *ChangeDimension* routine simply deletes the selected decision variable from each solution in ***N***. Next, a random sample of solutions from the non-dominated archive ***A*** is taken and their dimension changed with the *ChangeDimension* routine. That sample is then re-evaluated and the new objective values are compared to the old values before the change. Similar to Algorithm 4, if there is a small change, the non-dominated values in archive ***A*** are scaled by a factor consistent with the amount of the change and then those values are randomly sampled to create the current population ***Pt*** for the optimization process. If the change is large, the ***Pt*** is populated from random samples taken from the novelty archive ***N***. The ***A*** archive is then emptied, because none of the non-dominated solutions are viable anymore.

3.6.4 Simultaneously Changing Objectives and Decision Variables

When simultaneous changes occur various combinations of Algorithms 3, 4, and 5 are employed by the DP-NSGA-II to aid in the dual goals of convergence and diversity. Specifically, when the number of objectives change and the objective functions themselves change (e.g., case 4 in Figure 3-4), then Algorithm 4 (i.e., used for time-depended objective functions) is run first to detect the severity of the change in the objective functions. If the change is small, the ***A*** archive will be scaled accordingly and if it is large the current population ***Pt*** and the ***A*** archive will be reset. Then Algorithm 3 (i.e., used for changing number of objective functions) is applied in a successive manner to deal with the change in the number of objective functions.

When the objectives change in a time-dependent manner and the number of decision variables change (e.g., case 5 in Figure 3-4), then only Algorithm 5 (i.e., used for changes in decision variables) is applied. As Figure 3-9 indicates, Algorithm 5 is very similar to Algorithm 4, and so can be used by itself to address this situation by detecting the severity of the change and then scaling the solutions in the ***A*** archive, and reusing them in the case of small changes, or completely reinitializing the current population ***Pt*** and the ***A*** archive if the change is big.

When the number of objectives and decision variables change simultaneously (e.g., case 6 in Figure 3-4), then the change in the number of decision variables is dealt with first through the application of Algorithm 5. This allows the ***A*** archive to be scaled and reused if possible. Algorithm 3 is then applied to increase, or decrease, the dimension of the objective space. Solutions are then sampled from a scaled ***A*** archive to aid in convergence. If the ***A*** archive is empty, randomly sampled solutions from the novelty archive ***N*** are used for the new population.

The final case to be dealt with involves all three types of changes (e.g., changing number of objectives; time-dependent objective functions; changing number of decision variables) occurring at once (e.g., case 7 in Figure 3-4). When this occurs, just as in case 6, the first algorithm to be run is Algorithm 5, because it can handle both time-dependent objective functions and changes in the number of decision variables at once. Further, it can detect the severity of the change and scale the non-dominated solutions in the ***A*** archive for continued use to help convergence if the change is not too severe. After Algorithm 5 is applied, Algorithm 3 is then applied to change the dimension of the objective space to the desired number.

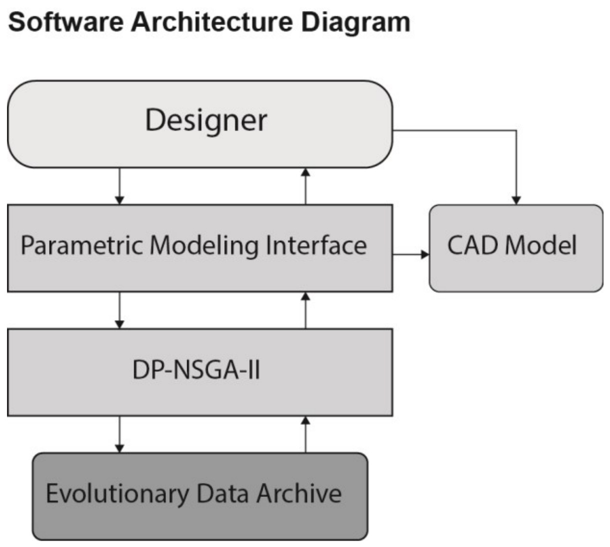
3.7 Implementation and User Interface

DP-NSGA-II was designed and developed as a plug-in for the parametric design environment called Grasshopper 3D, which is a part of McNeel's Rhinoceros 3D CAD environment. This environment was chosen because of its popularity among multiple design and engineering fields for the conceptual design phase and because of its use of a visual programming environment that allows for flexibility in developing interactive graphical user interfaces. As mentioned previously, a key objective for the research was developing unique ways that users could interact with the DP-NSGA-II. In Figure 3-10, the software architecture for the implementation can be seen.

This architecture allows users to interact with the optimization process in unique ways: existing decision variables and objective functions can be modified during the search process; new decision variables and objectives can be added; or existing variables and objectives can be deleted. The decision maker can also add qualitative evaluations during the evaluation phase and intervene in the selection, recombination, and mutation stages at any time during the running of the algorithm. These features allow for an open-ended evolutionary search of a design space and increased designer engagement.

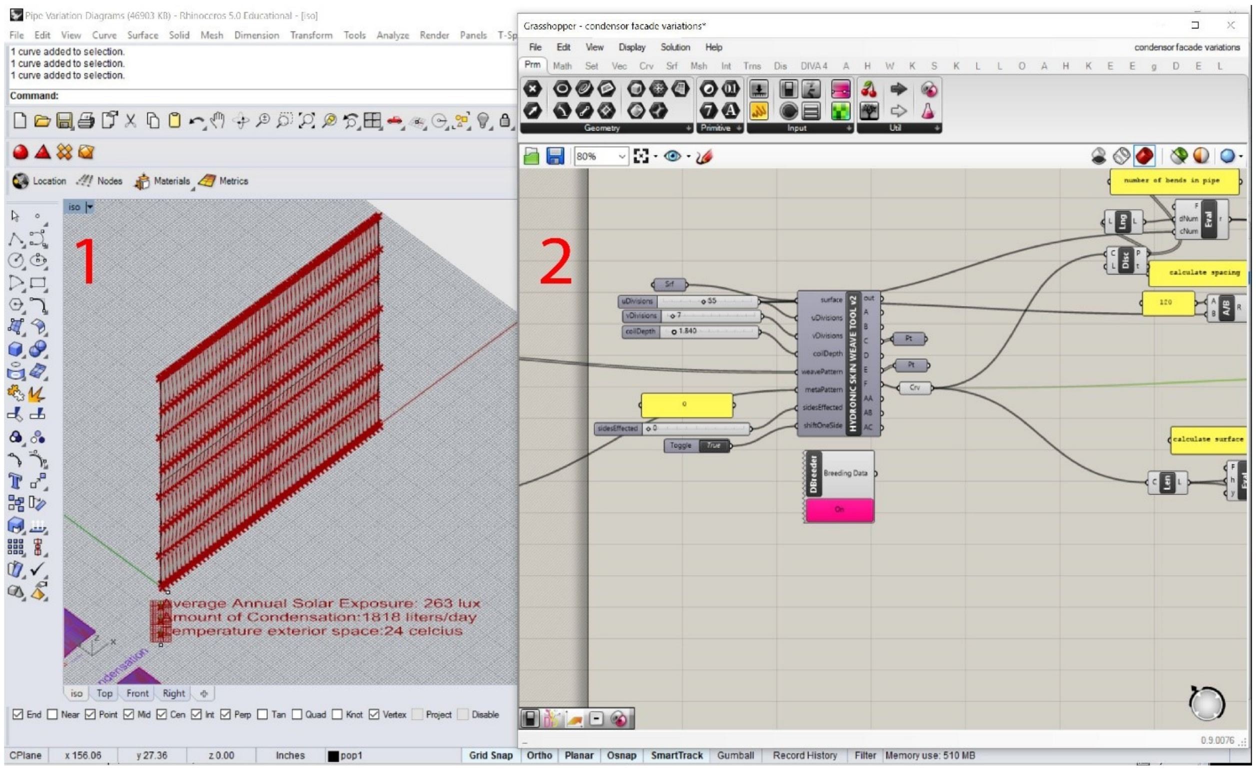
In Figure 3-11 and Figure 3-12 an image of the DP-NSGA-II interface as implemented is shown. The user interacts with the interface in the following ways in order to address the cases described in 3.6 :

- Case 1: To add objectives, the user presses the "Create Function" button in the "Fitness Evaluation" area. This brings up a window in which the function can be named and defined as a quantitative function or a qualitative function. To delete an objective, the user can select the objective given in the list box in the "Fitness Evaluation" area and press the "delete" key.
- Case 2: Time-dependent objective functions can be added using the same process described above.
- Case 3: To add a decision variable, the user must select a parameter in the parametric modeling environment and press the "Create Gene" button in the "Gene List" area. To delete a decision variable, the user selects a decision variable given in the list box located in the "Gene List" area and presses the "delete" key.
- Cases 4-7: The user can institute combinations of changes by following the procedures above.



Software Architecture Diagram
Designer
Parametric Modeling Interface
CAD Model
DP-NSGA-II
Evolutionary Data Archive

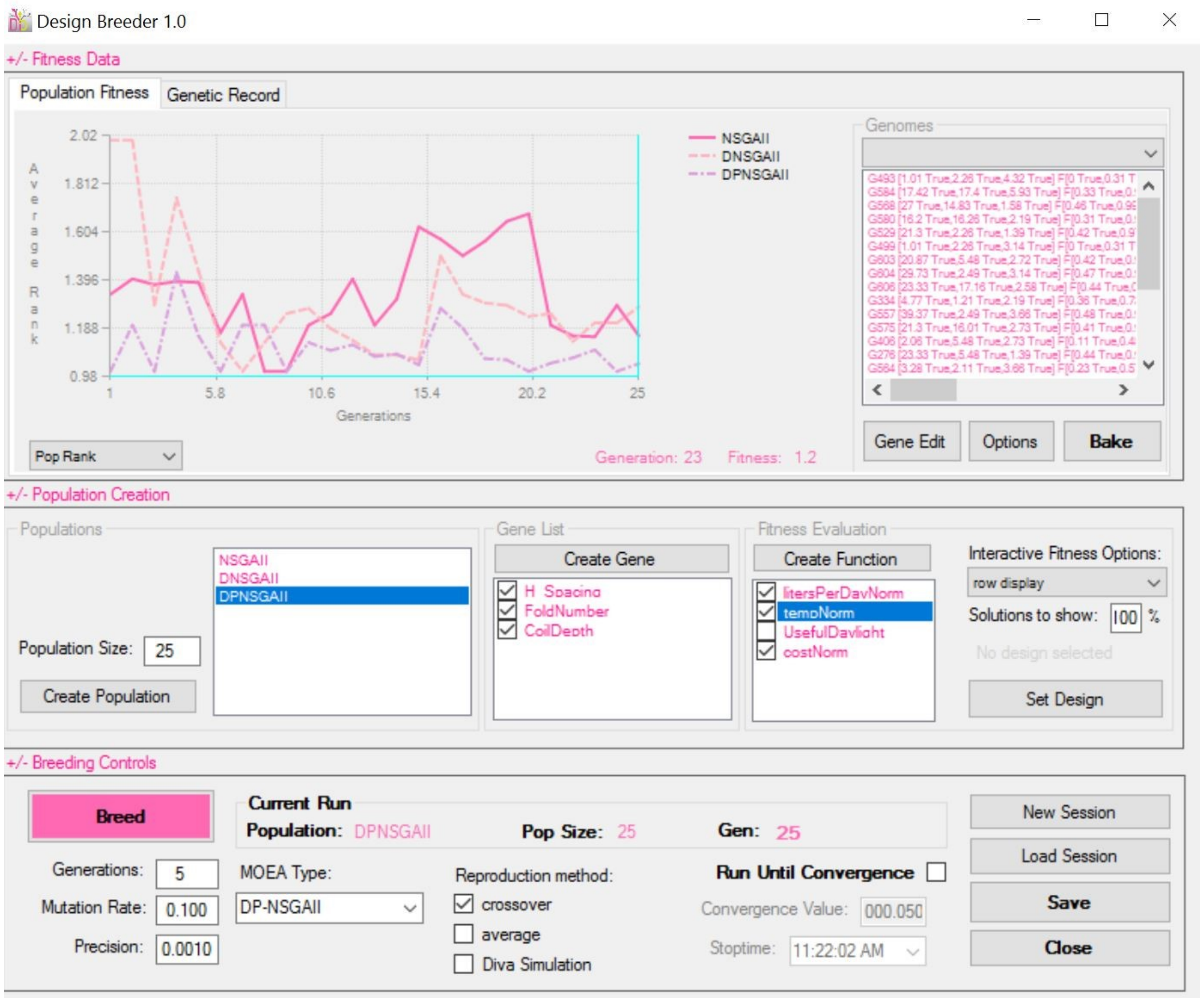
Figure 3-10. DP-NSGA-II integrates into the Grasshopper visual programmig environment.



1 curve added to selection.
1 curve added to selection.
1 curve added to selection.

verage Annual Solar Exposure: 263 lux
mount of Condensation:1818 liters/day
emperature exterior space:24 celcius

Figure 3-11. (1) An image of the Rhinoceros 3D Interface. (2) An image of the visual programing environment in Grasshopper, which is a plugin for Rhinoceros 3D.



```
G493 [1.01 True,2.26 True,4.32 True] F[0 True,0.31 T
G584 [17.42 True,17.4 True,5.93 True] F[0.33 True,0.
G568 [27 True,14.83 True,1.58 True] F[0.46 True,0.9
G580 [16.2 True,16.26 True,2.19 True] F[0.31 True,0.
G529 [21.3 True,2.26 True,1.39 True] F[0.42 True,0.9
G499 [1.01 True,2.26 True,3.14 True] F[0 True,0.31 T
G603 [20.87 True,5.48 True,2.72 True] F[0.42 True,0.
G604 [29.73 True,2.49 True,3.14 True] F[0.47 True,0.
G606 [23.33 True,17.16 True,2.58 True] F[0.44 True,
G334 [4.77 True,1.21 True,2.19 True] F[0.36 True,0.7
G557 [39.37 True,2.49 True,3.66 True] F[0.48 True,0.
G575 [21.3 True,16.01 True,2.73 True] F[0.41 True,0.
G406 [2.06 True,5.48 True,2.73 True] F[0.11 True,0.4
G276 [23.33 True,5.48 True,1.39 True] F[0.44 True,0.
G564 [3.28 True,2.11 True,3.66 True] F[0.23 True,0.5
```

Generation: 23 Fitness: 1.2

Population Size: 25

NSGAII
DNSGAII
DPNSGAII

☑ H_Spacing
☑ FoldNumber
☑ CoilDepth

☑ litersPerDayNorm
☑ tempNorm
☐ UsefulDaylight
☑ costNorm

Interactive Fitness Options: row display
Solutions to show: 100 %
No design selected

Current Run
Population: DPNSGAII Pop Size: 25 Gen: 25

Generations: 5
Mutation Rate: 0.100
Precision: 0.0010
MOEA Type: DP-NSGAII
Reproduction method:
☑ crossover
☐ average
☐ Diva Simulation
Run Until Convergence ☐
Convergence Value: 000.050
Stoptime: 11:22:02 AM

Figure 3-12. An image of the DP-NSGA-II interface.

Chapter 4. Case Study Application of DP-NSGA-II

MOEAs are typically performance tested through the use of artificial problem sets developed by the research community. The two main sets are a) the bi-objective ZDT (Zitzler-Deb-Thiele) (Zitzler, Deb et al. 2000) and b) the scalable DTLZ (Deb-Thiele-Laumans-Ziztler) test suites (Deb, Thiele et al. 2002). These problem sets have analytical solutions and lack the complexity of real-world MOPs. For DIMOEAs, a set of benchmark problems different from the two listed above are used. The FDA problem set is currently the most widely cited (Farina, Deb et al. 2004). Other problem sets often cited include: type I DMOP; type II DMOP; and DCTPs (Goh and Tan 2009). Finally, real-world applications of DMOEAs can be found in the literature for different areas of study, such as route optimization for traffic, mechanical design problems, or scheduling problems (Azzouz, Bechikh et al. 2017).

Progressive MOEAs and DMOEAs are more difficult to test because of the user-interaction involved. Currently, there are no standard problem sets specifically designed to benchmark such algorithms. Instead, problem sets from standard MOEAs and DMOEAs are adapted and used (Li, Deb et al. 2017). These problem sets are generally considered to be useful analogs of some real-world MOPs, but they are still toy problems that can only give limited insights into the performance of these algorithms (Jiang, Ong et al. 2014). The use of real-world problems can therefore be invaluable.

In this thesis, DP-NSGA-II is applied to a real-world MOP in the domain of architectural design. DP-NSGA-II is tested for its capacity to deal with dynamic changes in the number of objectives, the number of decision variables, and time-dependent changes in objective functions. DP-NSGA-II is tested against two state-of-the-art algorithms. Specifically, NSGA-II, as well as, the dynamic version of this algorithm called D-NSGA-II.

The progressive aspects of the algorithm that integrate user preference information are not tested in this setup. This allows the tests to focus exclusively on how the algorithm's procedures for convergence and diversity preservation perform without human interaction. Future tests will look at how human interaction affects the performance of the algorithm.

4.1 MOP Description

The MOP developed to test the DP-NSGA-II algorithm involves the optimization of an exterior solar shading facade system located in Kuwait City, Kuwait. For the optimization, a south facing 10'x10' section of the facade is explored. The facade system is composed of computer numerically controlled (CNC) bent steel pipes connected into an assembly. Cool seawater is circulated in these pipes to collect condensation from the air and also cool exterior spaces. A diagram of the system can be seen in Figure 4-1 and a rendered image of the facade can be seen in Figure 4-2.

The objectives for the optimization are all continuous and each is normalized:

1. Maximizing the amount of condensation harvesting per year.
2. Maximizing useful daylighting levels on the interior of the facade.
3. Minimizing the temperature in the exterior balcony space between the facade and the

building envelope.
4. Minimizing the cost of the fabrication of the CNC folded facade modules.

Diva is a solar analysis plugin for Rhinoceros 3D and it was used for the useful daylight measure, which gives the percentage of time throughout the year where illumination levels are between 100-2000 lux. Autodesk's Computational Fluid Dynamics software was used for air flow and thermal analysis. The amount of condensation harvested was estimated following Bryant and Ahmed (2008).

In order to speed-up the evaluation of designs, a surrogate model for thermal analysis was developed by taking 200 sample simulations throughout the design space. These samples were then used to create a degree three polynomial regression model to predict temperature values based on two parameters: the amount of surface area of the facade and the spacing of the pipes. Images of some of the sample simulations for thermal performance can be seen in Figure 4-3 and images from the solar analysis can be seen in Figure 4-4.

4.1.1 Decision Variables

The decision variables for the optimization process are based on the geometric parameters of the pipe assembly. The *pipe spacing*, *number of folds*, and *depth of system* variables are all continuous and can be seen in Figure 4-5. The *regional fold pattern* variable is discreet and contains options for a limited number of fold patterns that can be applied to the facade. The decision variables are subject to the following constraints:

a) $1'' \leq pipe\ spacing \leq 30''$;
b) $0'' \leq number\ of\ folds \leq 20''$;
c) $1'' \leq depth\ of\ system \leq 12''$.

4.2 Test Scenarios and Evaluation

The performance of the three chosen algorithms (e.g., NSGA-II, D-NSGA-II, DP-NSGA-II) are measured and compared with four different tests:

1. Test one studies the impact of changing the number of objectives, while all other features of the problem stay constant (e.g., decision variables and constraints are constant; there are no time-dependent changes to existing objective functions). The details of the test can be seen in Table 4-1.
2. The second test described in Table 4-2 explores the result of changing the number of decision variables, while keeping all other parameters constant.
3. The third test described in Table 4-3 looks at how time-dependent objective functions might affect each algorithm.
4. The last test studies the result of simultaneous changes in all three areas (i.e., decision variables, objective functions, time-dependent objective functions). The details of the test can be seen in Table 4-4.

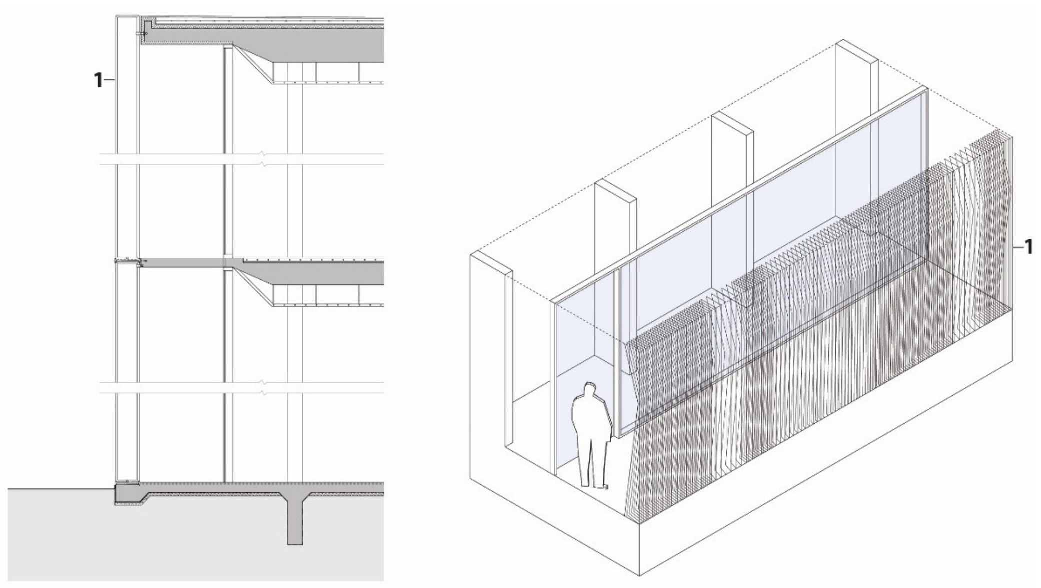


Figure 4-1. A section and axonometric drawing of the facade system that is optimized.

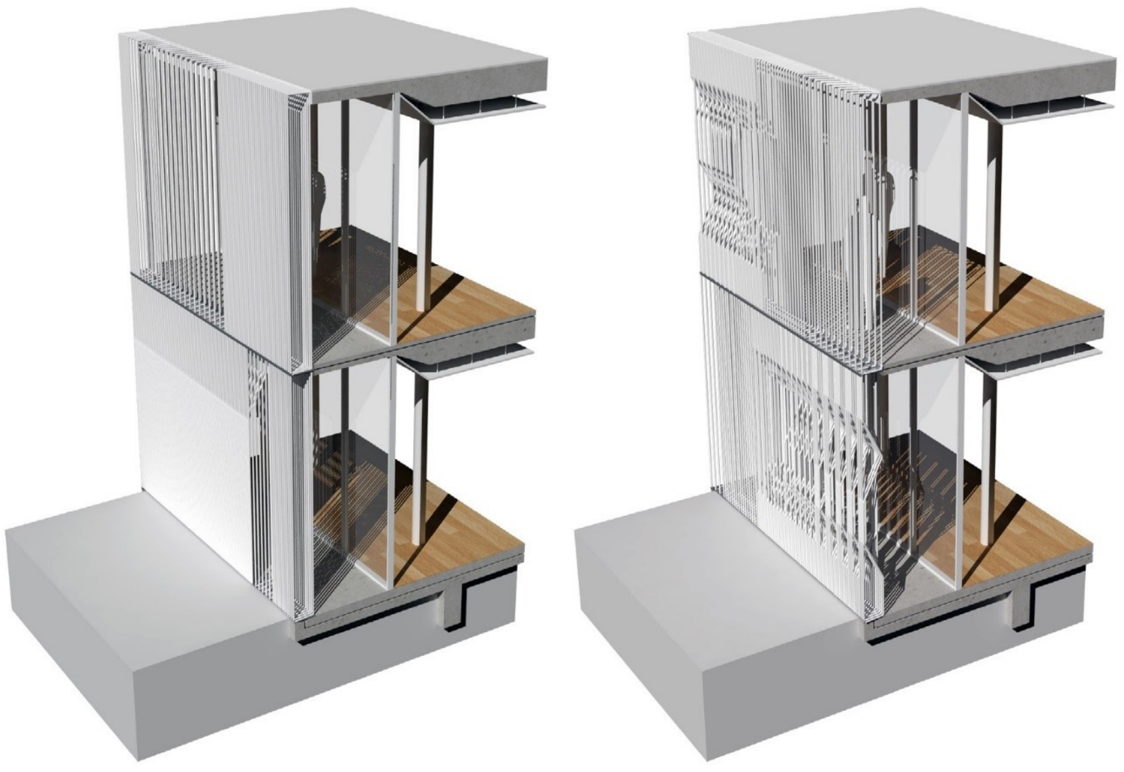


Figure 4-2. Two samples from the optimization process.

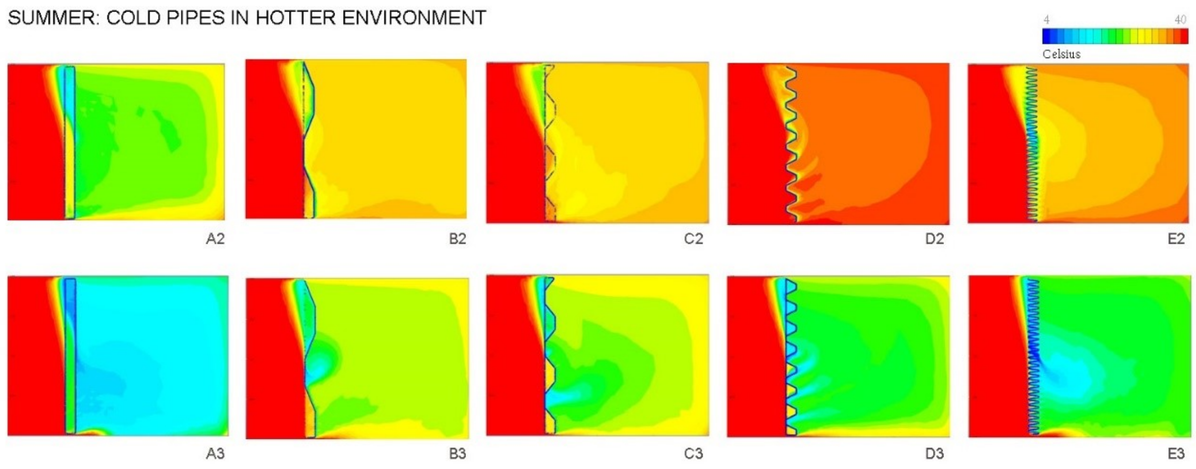


Figure 4-3. Thermal simulations for 10 different samples in the decision space during the summer. Each column of images represents solutions with the same number of folds, while each row represents solutions with a different spacing for the pipes. These samples are used to create a surrogate model to speed-up the evaluation of designs.

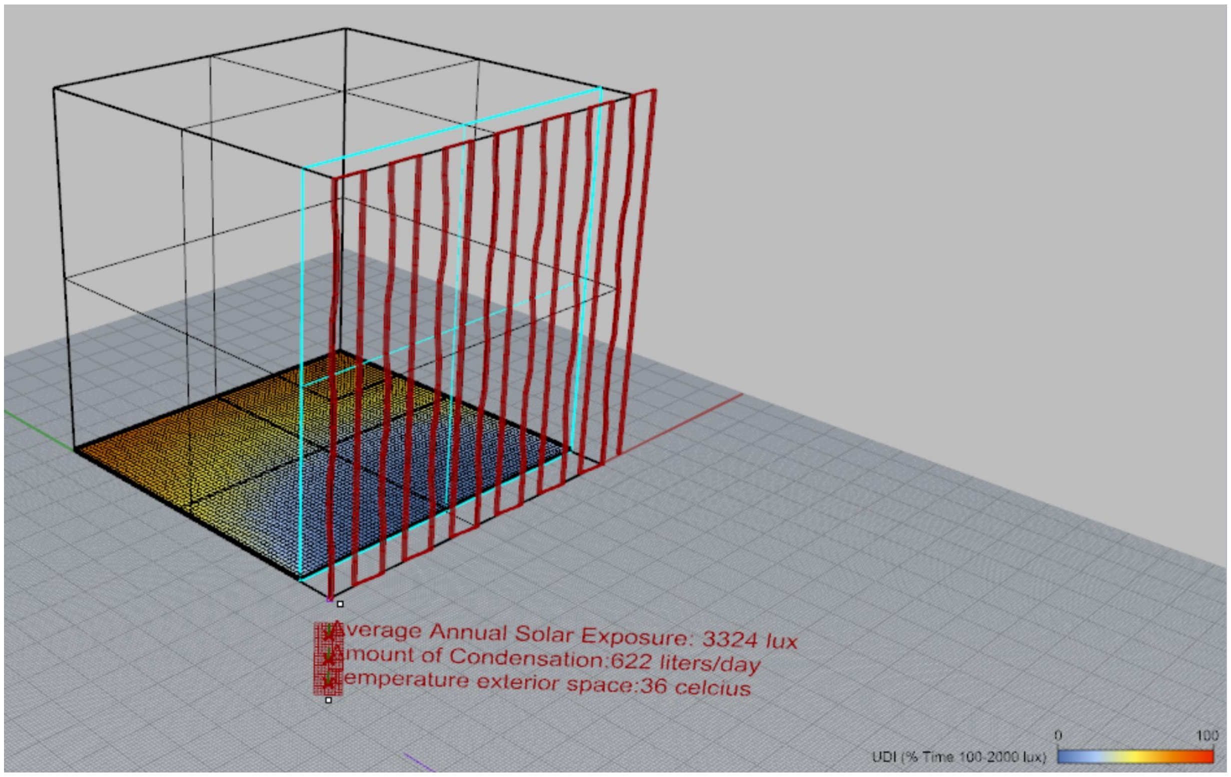

Average Annual Solar Exposure: 3324 lux
Amount of Condensation:622 liters/day
Temperature exterior space:36 celcius
0
100
UDI (% Time 100-2000 lux)

Figure 4-4. Image of useful daylight simulation during optimization.

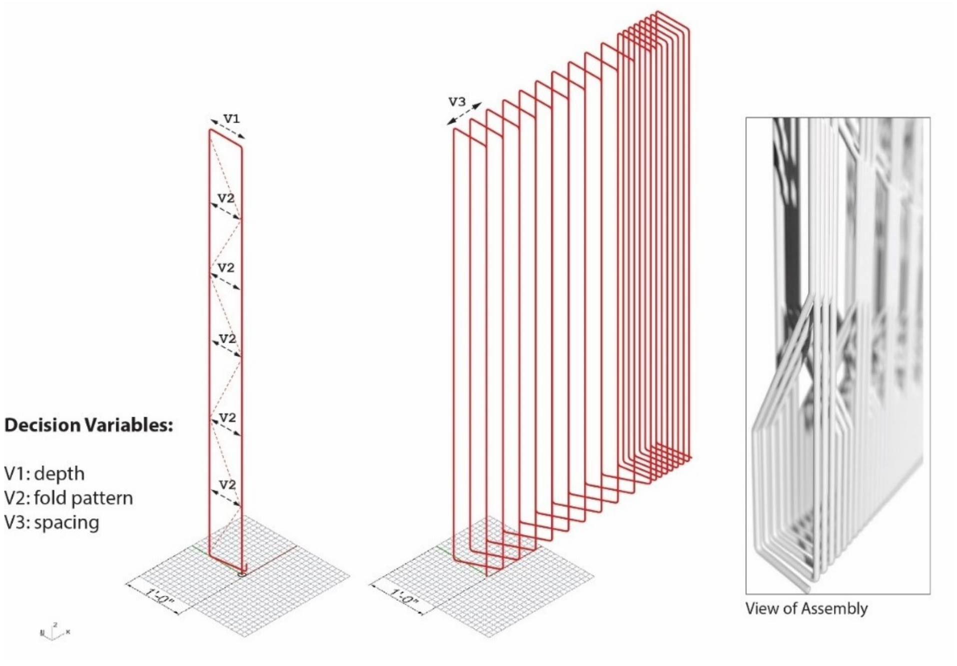


Figure 4-5. The decision variables for the optimization involve exploring the depth, the fold pattern, the spacing, and the regional pattern of the CNC bent pipe assembly.

Each test is run for 25 generations, with changes to the MOP occurring every five generations. Each test is then run four times. A population of 25 solutions is used for each generation. A mutation rate of 0.10 is used by all algorithms. The global $\boldsymbol{Rg}$ and local $\boldsymbol{Rl}$ reference vectors are set to a null value because the user does not provide a reference direction nor solution ranking in these tests. Equation 1 is used for the novelty calculation and the k parameter (i.e., number of nearest neighbors) is set to be 30% of the population size. These numbers are chosen through experimentation, and represent a reasonable trade-off in keeping computation time low, while getting enough results to compare the algorithms.

The evaluation of each algorithm will be done based on two metrics. The first metric involves the calculation of the average novelty per generation of each algorithm. This metric is calculated by averaging the novelty scores for each solution in a generation. This metric is used to give a sense of how diverse the solutions are.

The second metric used to evaluate the algorithms is the average rank per generation. To calculate this metric, the PF of each algorithm is combined into one global set of PF solutions for each generation. This combined global set is then re-ranked to find the nondominated solutions across all three algorithms for each generation. The average rank per generation is then calculated for each algorithm by averaging the new rank values of their respective PFs. This metric indicates which algorithm is producing better solutions per generation.

Table 4-1. Summary of facade optimization test #1, with the number of objectives changing over time.

Generations	Number of Objectives					Time-dependent objective changes	Decision variables
	Total number	Condensation	Temperature	Useful daylight	Cost		
1-5	2	X	X			**No**, for all generations	**3**, for all generations (coil spacing, number of folds, and coil depth)
6-10	3	X	X	X			
11-15	4	X	X	X	X		
16-20	2	X			X		
22-25	3	X	X		X		

Table 4-2. Summary of facade optimization test #2, with the number of decision variables changing over time.

Generations	Number of Objectives	Decision variables					Time-dependent objective functions
		Total number	Coil spacing	Number of folds	Coil depth	Regional fold pattern	
1-5	**3**, for all generations (condensation, temperature, and useful daylight)	1	X				**No**, for all generations
6-10		2	X	X			
11-15		3	X	X	X		
16-20		4	X	X	X	X	
21-25		2	X	X			

Table 4-3. Summary of facade optimization test #3, with time-dependent objective functions.

Generations	**Number of Objectives**	**Time-dependent objective functions (e.g. cost(t))**			**Decision variables**
		Small change	**Big change**	**No change**	
1-5	**3**, for all generations (temperature, cost, and useful daylight)			X	**3**, for all generations (coil spacing, number of folds, and coil depth)
6-10		X			
11-15			X		
16-20				X	
21-25		X			

Table 4-4. Summary of facade optimization test #4, with simultaneous changes.

Generations	**Number of Objectives**					**Decision variables**					**Time-dependence of objective functions**		
	Total number	**Condensation**	**Temperature**	**Useful daylight**	**Cost**	**Total number**	**Coil spacing**	**Number of folds**	**Coil depth**	**Regional fold pattern**	**Small change**	**Big change**	**No change**
1-5	2	X	X			2	X	X					X
6-10	3	X	X		X	3	X	X	X		X		
11-15	4	X	X	X	X	4	X	X	X	X		X	
16-20	3	X	X	X		4	X	X	X	X			X
21-25	3	X	X	X		2	X	X			X		

4.3 Results and Discussion

The tests described in Section 4.2 were run on NSGA-II, D-NSGA-II, and DP-NSGA-II. The full data sets can be found in Appendix A. The results for the first test in which the number of objectives change is shown in Figure 4-6, Figure 4-7, and also Table 4-5. The data shows that DP-NSGA-II had higher average novelty values than NSGA-II for nearly every generation. The algorithm was, however, much closer to D-NSGA-II in terms of this metric. On the average Pareto rank metric, the data shows that DP-NSGA-II outperforms the NSGA-II consistently but is very close to the performance of D-NSGA-II.

The data in Table 4-5 offers another view of the results of the first test. It shows the percentage of time DP-NSGA-II outperforms the other algorithms for all generations run (e.g., 4 runs x 25 generations = 100 generations). It shows that DP-NSGA-II outperforms NSGA-II 80% of the time in relation to novelty and 67% of the time for Pareto rank. Consequently, our algorithm is outperformed by NSGA-II 20% and 33% of the time by these metrics. DP-NSGA-II beats D-NSGA-II 65% of the time in novelty and 49% in Pareto rank. These results indicate that the memory mechanism used by DP-NSGA-II, in which it samples from a novelty archive ***N*** to aid diversity after a change, provides the needed level of diversity and also allows for efficient convergence.

The results for the second test in which the number of decision variables change with time can be seen in Figure 4-8 and Figure 4-9. The plots show that DP-NSGA-II on average outperforms NSGA-II in terms of novelty and is almost equivalent in terms of Pareto rank. DP-NSGA-II performs equivalently to D-NSGA-II in novelty, but narrowly outperforms it in Pareto rank. The more granular view of the data in Table 4-5 indicates that DP-NSGA-II outperforms NSGA-II 86% of the time in novelty and 39% of the time in Pareto rank. While outperforming D-NSGA-II 50% and 70% in terms of these two metrics. This data indicates that DP-NSGA-II converges better than D-NSGA-II by having a better average Pareto rank per generation. This is because DP-NSGA-II can compute the severity of the impact of a change in the number of decision variables and act accordingly. In contrast, D-NSGA-II always performs the same random initialization procedure on 50% of the population if it detects a change. This boosts diversity but throws-out good solutions unnecessarily at times.

The data shows that NSGA-II outperforms DP-NSGA-II in average Pareto rank for test 2. This result is surprising because the addition of added decision variables causes NSGA-II to have a lack of novel solutions, and this should mean the algorithm cannot effectively find the new optimums of the problem. We hypothesize that the reason we are not seeing this response is that some of the added decision variables are redundant in nature. Future implementations of NSGA-II will address the issue of identifying redundant decision variables to improve its performance in this area.

The results for the third test, in which the objective functions are time-dependent, can be seen in Figure 4-10 and Figure 4-11. These plots indicate that DP-NSGA-II narrowly outperforms both baseline algorithms in Pareto ranking just after a small change in the objective functions in generation 5 and a big change in generation 10. At generation 15, there is no change and D-NSGA-II starts to narrowly outperform DP-NSGA-II. In generation 20, when a small change is instituted in the objective functions, both NSGA-II and D-NSGA-II jump ahead of our algorithm in average Pareto ranking. This indicates that the procedures employed by DP-NSGA-II to detect and react to the degree of change in an objective function has success in the early generations. It is only when there is a small change at generation 20 when the influx of diverse solutions causes it to perform below the two other algorithms. We hypothesize that running the test for more generations beyond 25 might show DP-NSGA-II begin to converge back to the high average Pareto ranking it demonstrated in the early generations.

For the third test case, Table 4-5 indicates that DP-NSGA-II outperforms NSGA-II 47% and 49% of the time in novelty and Pareto rank respectively. It outperforms D-NSGA-II 35% and 31% of the time for the same two metrics. DP-NSGA-II shows lower performance overall for this test case against both algorithms as compared to the previous two test cases. We hypothesize that DP-NSGA-II may be too sensitive to change in the objective functions and may be over-correcting. To address this issue, we propose setting a threshold for severity of change in which no corrective action is taken unless that threshold is exceeded. The difficulty with this approach is the setting of the threshold, which may be a unique to each dynamic situation.

The final test focuses on simultaneous changes in the number of objective functions, number of decision variables, and time-dependent changes in objective functions. The results can be seen in Figure 4-12 and Figure 4-13. The plot in Figure 4-12 indicates that DP-NSGA-II maintains high average novelty levels above those of NSGA-II and almost equivalent to those of D-NSGA-II for all generations. In generations 6 and 11 there are large spikes in novelty in response to simultaneous changes. The plot in Figure 4-13 indicates that DP-NSGA-II is equivalent to the other algorithms during the simultaneous changes in the early generations involving increasing the number of objectives functions, the number of decision variables, and making small and large time-dependent changes in objective functions. DP-NSGA-II starts to fall behind in the later generations when the simultaneous changes involve decreasing the number of objectives and decision variables. We hypothesize that this is due to the features of DP-NSGA-II that increase novelty when changes occur. This boost of novelty may temporarily affect convergence, but provides the benefit of more diverse solutions. In the long run, we hypothesize that DP-NSGA-II would converge and outperform the other two algorithms, but further tests need to be done to verify this assertion.

In Table 4-5 a comparison of algorithms on a generation by generation basis reveals that DP-NSGA-II outperforms NSGA-II 96% and 38% percent of the time in novelty and Pareto rank. It outperforms D-NSGA-II 53% and 32% of the time in the same two metrics. These numbers indicate that DP-NSGA-II is outperforming NSGA-II in novelty, but is almost tied with D-NSGA-II in this regard. D-NSGA-II, however, outperforms DP-NSGA-II almost 68% of the time in Pareto rank. We hypothesize that this may be due to the slightly higher level of novelty that DP-NSGA-II demonstrates on average. This allows it to have more diverse solutions and affects its convergence rate. In addition, we hypothesize that the over-correction in dealing with small changes to objective functions, noted earlier, may also be causing the lower Pareto performance.

An image of the interface of DP-NSGA-II can be seen in Figure 4-14. Some samples of the non-dominated solutions at various stages during the testing can be seen in Figure 4-15. The results from all four tests in Table 4-5 indicate that DP-NSGA-II outperforms, or is equivalent to NSGA-II and D-NSGA-II, in novelty for the majority of test cases. The only exception is when the objectives are time-dependent (i.e., test 3). The results show that DP-NSGA-II outperforms NSGA-II in average Pareto rank when the number of objectives change. It also outperforms D-NSGA-II when the number of decision variables change. The test case in which it underperforms is that of test 4 dealing with simultaneous changes. These results indicate that DP-NSGA-II offers advantages over the other algorithms in terms of novelty and that it also offers advantage in Pareto rank for some scenarios of dynamic change.

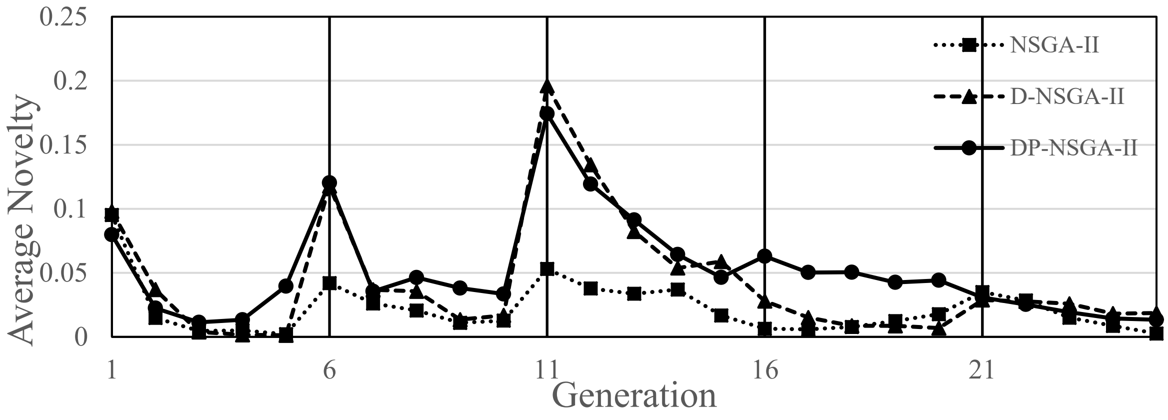


Figure 4-6. Average novelty for test #1 in which the number of objectives change. Each symbol represents the average of four runs.

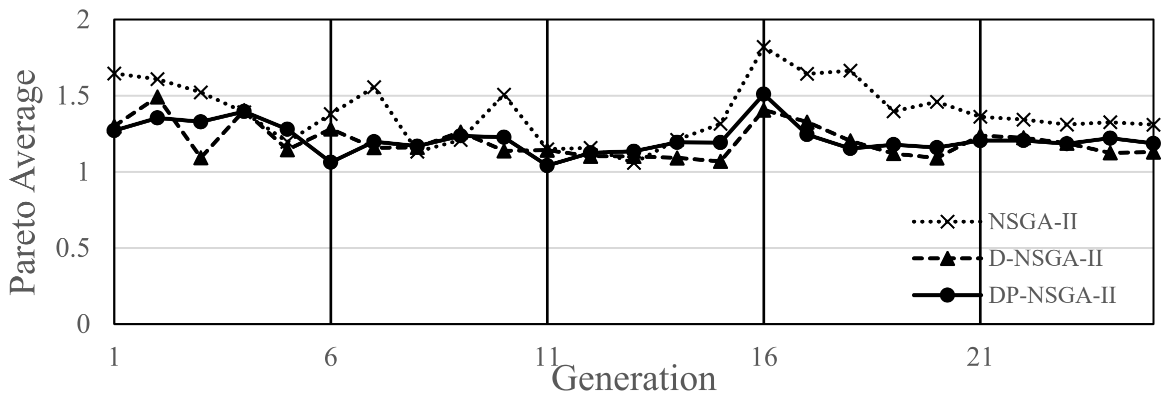


Figure 4-7. Pareto rank average for test #1 in which the number of objectives change. Each symbol represents the average of four runs.

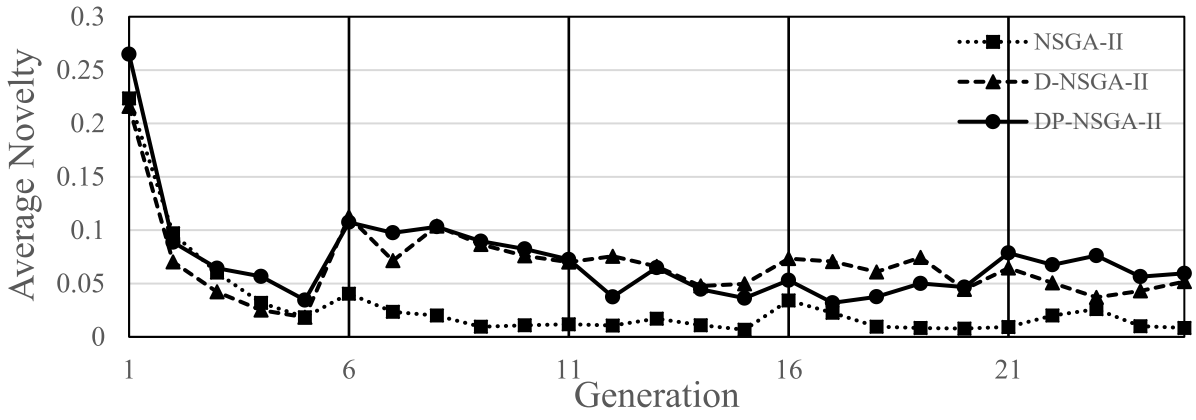


Figure 4-8 Average novelty for test #2 in which the number of decision variables change. Each symbol represents the average of four runs.

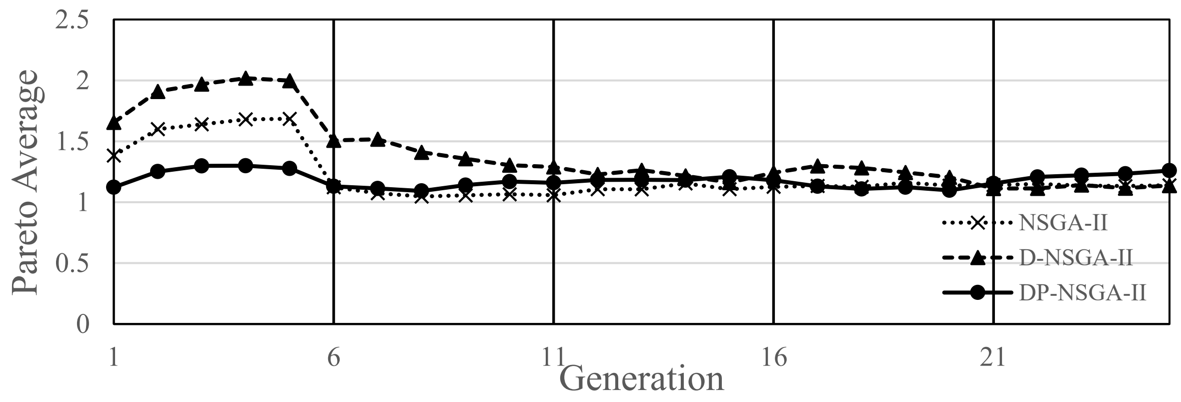


Figure 4-9. Pareto rank average for test #2 in which the number of decision variables change. Each symbol represents the average of four runs.

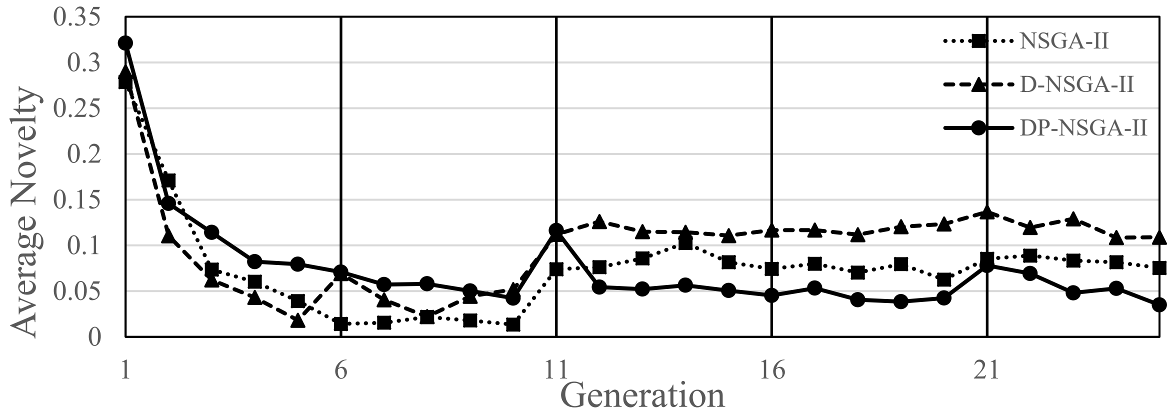


Figure 4-10 Average novelty for test #3 in which the objective functions are time-dependent. Each symbol represents the average of four runs.

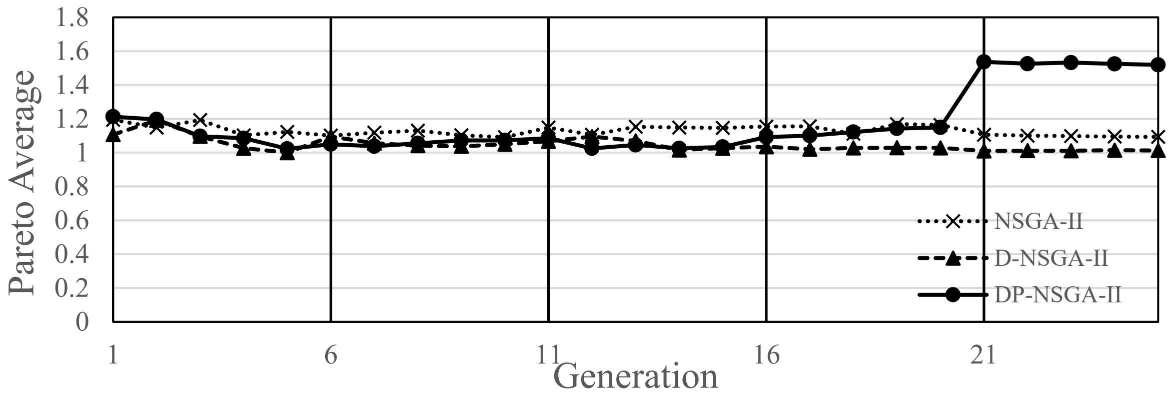


Figure 4-11. Pareto rank average for test #3 in which the objective functions are time-dependent. Each symbol represents the average of four runs.

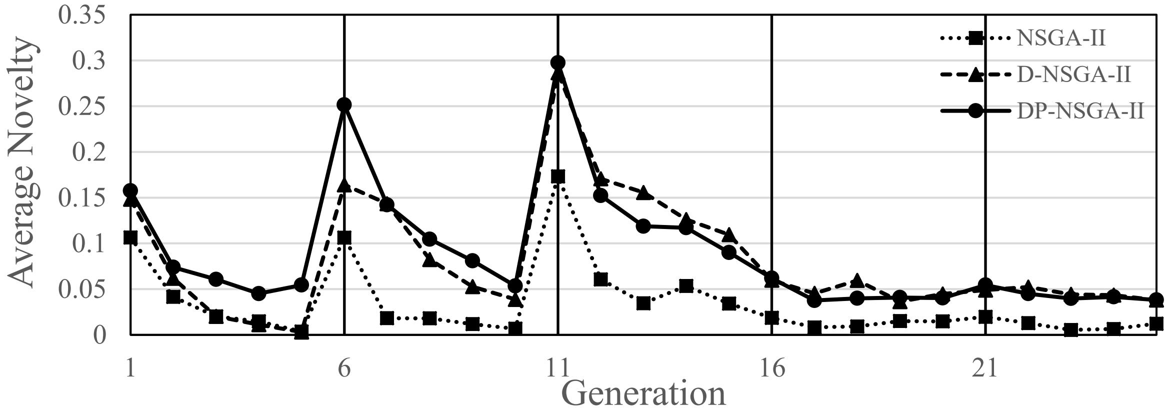


Figure 4-12 Average novelty for test #4 in which the objective functions are time-dependent. Each symbol represents the average of four runs.

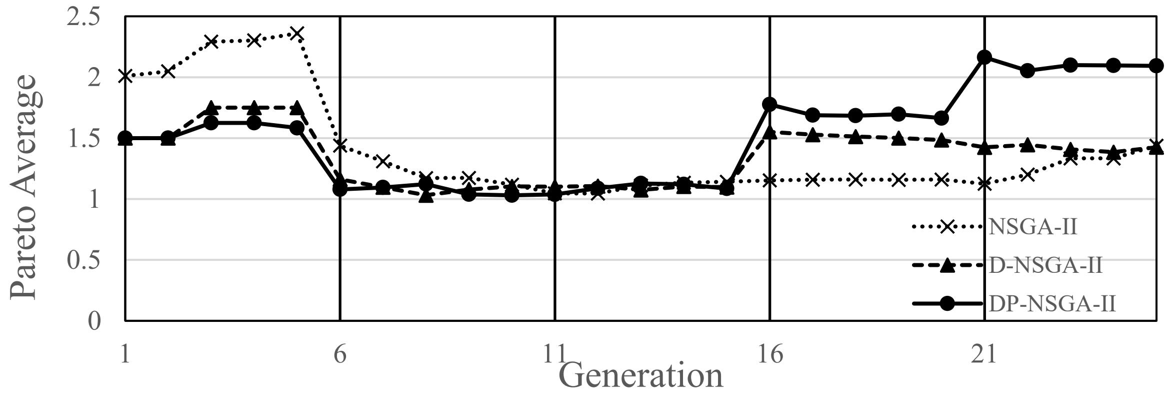


Figure 4-13. Pareto rank average for test #4 in which the objective functions are time-dependent. Each symbol represents the average of four runs.

Table 4-5. Performance of DP-NSGA-II against NSGA-II and D-NSGA-II for the four test cases studied here. Shows the percentage of times that DP-NSGA-II outperformed the other algorithms.

	Against NSGA-II		Against D-NSGA-II	
	Novelty average	Pareto rank average	Novelty average	Pareto rank average
Test 1	80%	67%	65%	49%
Test 2	86%	39%	50%	70%
Test 3	47%	49%	35%	31%
Test 4	96%	38%	53%	32%

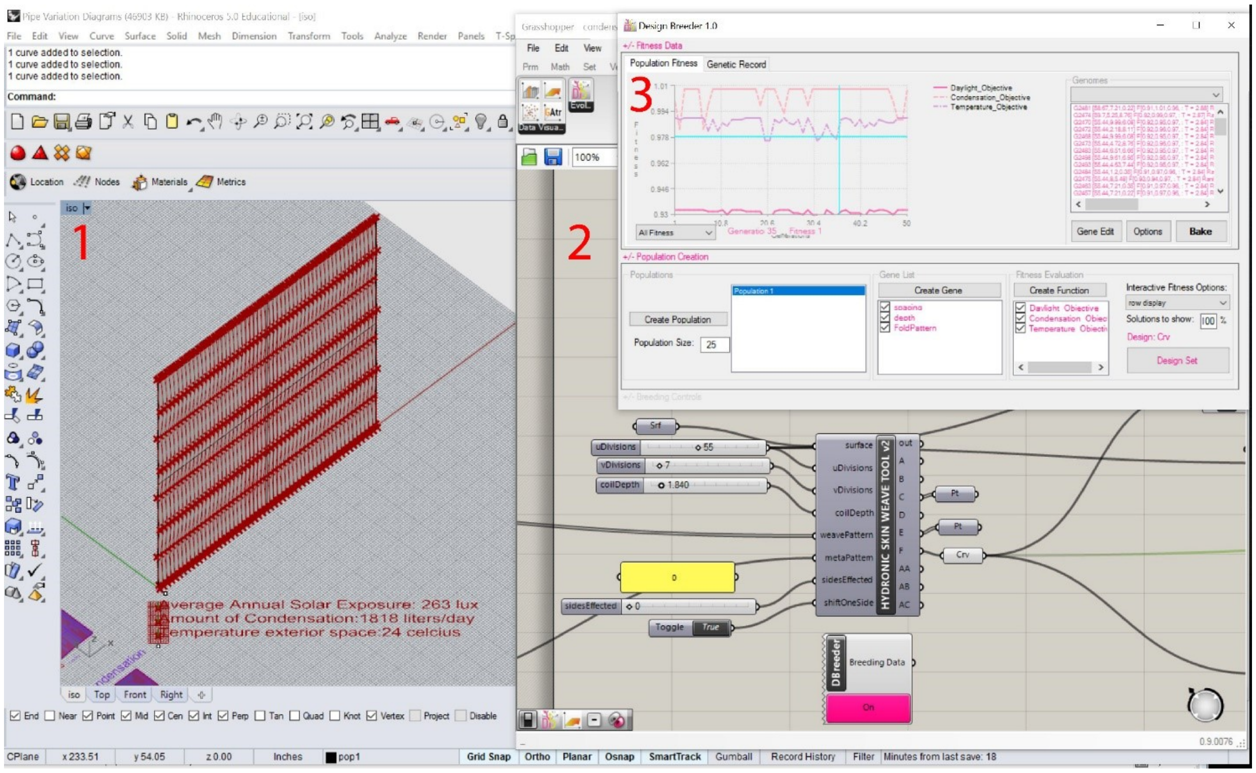


Figure 4-14. (1) The Rhinoceros 3D interface that the geometry for the facade is created in. (2) An image of the visual programming environment provided by Grasshopper that allows users to interact with the optimization by adding or subtracting design features as the optimization runs. (3) An image of the DP-NSGA-II graphical user interface as the optimization runs.

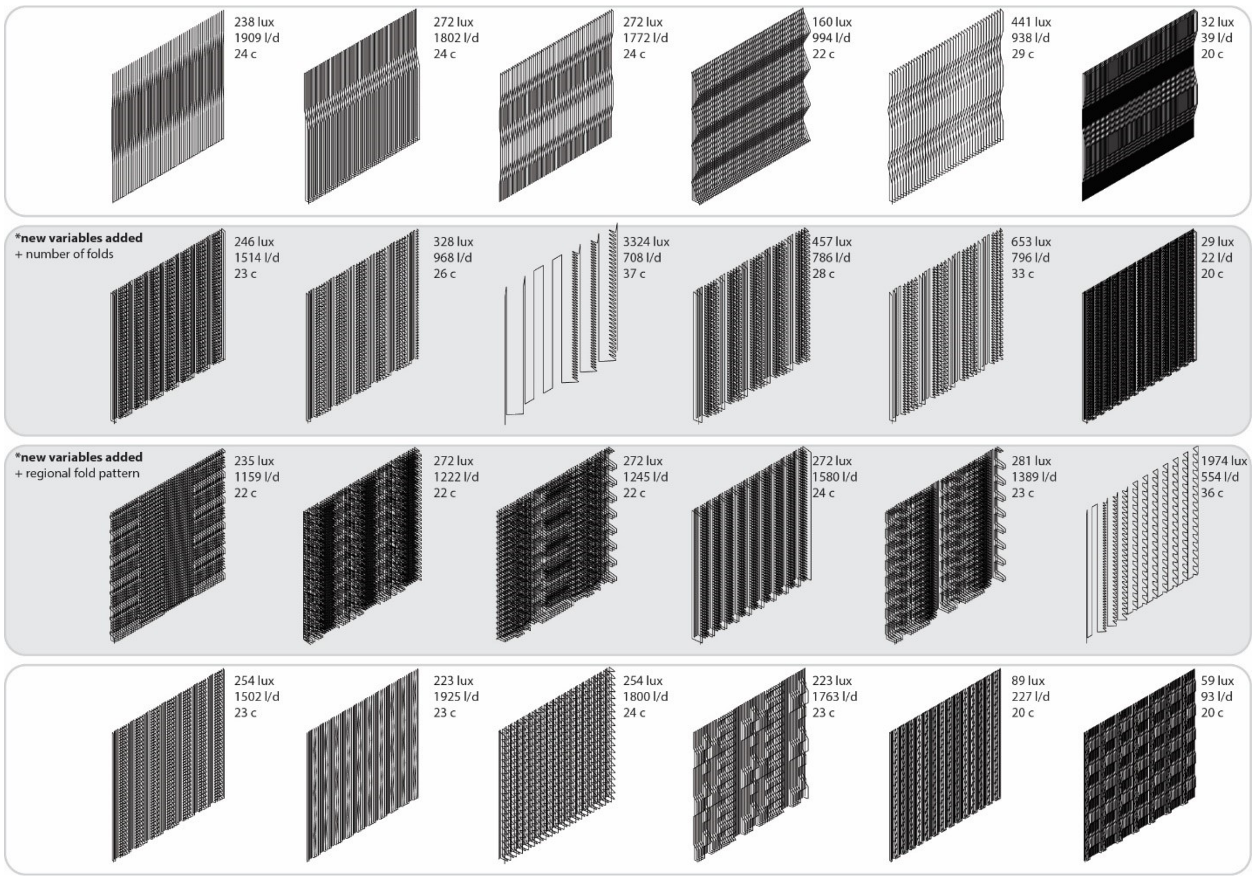


Figure 4-15. Some samples of the non-dominated solutions at various stages during the testing can be seen.

Chapter 5. Conclusions

The research presented in this thesis addressees a gap in the current research literature related to progressive dynamic multi-objective evolutionary algorithms. The thesis proposes a unique modification to the NSGA-II, which we call DP-NSGA-II. DP-NSGA-II is designed to address dynamic MOPs that previous work has not dealt with. Specifically, situations where the number of objectives, the objectives themselves (i.e., time-dependent functions), and the number of decision variables change simultaneously.

DP-NSGA-II uses a memory-based approach, which samples from novelty and Pareto archives, to aid diversity and convergence when the number of objectives change. A user defined global reference direction can also be used to direct the search in circumstances where the number of objectives increases and the objective space goes beyond three dimensions. To deal with changes in the number of decision variables, or time-dependent objective functions, the type and severity of the change is first computed. If the change is small, a prediction-based approach is applied in which already calculated objective values in the Pareto archive are scaled by a computed scale factor, sampled, and used to replace half of the current population. This aids convergence by using values that were known to have performed well previous to the change. The other half of the population is then replaced with random samples from the novelty archive to aid diversity. If the change is big, the Pareto archive is completely emptied, and the current population of solutions is repopulated with samples from the novelty archive.

DP-NSGA-II integrates user preferences in a progressive fashion through the application of a hybrid approach. This is done through a reference-direction approach to guide the search at the global scale, and a solution ranking approach to guide the search at the local scale. This progressive approach narrows the scope of the search in high-dimensional objective spaces and makes the optimization tractable, while engaging the user's abilities to help search a complex multi-dimensional space.

DP-NSGA-II is applied to a real-world MOP in the problem domain of architectural design in Section 4. The tests specifically focus on the ability of the algorithm to deal with dynamic changes to the given MOP without human interaction. The progressive features of the algorithm are left for future testing. DP-NSGA-II is tested against two state-of-the-art evolutionary MOP algorithms: NSGA-II and D-NSGA-II. These algorithms are compared based on two metrics: *average novelty* and *average rank per generation*. The results show that DP-NSGA-II outperforms NSGA-II in average Pareto rank when the number of objectives change. It also outperforms D-NSGA-II when the number of decision variables change. Further, DP-NSGA-II outperforms both algorithms in novelty for 75% of the test cases.

5.1 Future Work

Areas of future development in the research will address the following topics: benchmarking with additional real-world MOPs as well as standard MOEA and DMOEA problem sets; benchmarking against additional DMOEAs; and testing the progressive features of the algorithm. In terms of the first issue, the tests used for this research were limited to one MOP from the problem domain of architecture. Future research on DP-NSGA-II will involve testing it with some of the standard MOEA problem sets. Specifically, dynamic versions of the ZDT and DTLZ problem sets described by Chen, Li et al. (2016), in addition to the FDA problem sets, will be used. It is also necessary to benchmark the user preference-integration features of the algorithm. Specifically, benchmarking those progressive features related to solution ranking and the definition of a global reference direction to direct the search. There are currently no problem sets to benchmark progressive DMOEAs. Future research will therefore develop these problems and then use them to test DP-NSGA-II.

There are additional progressive features of the algorithm that need more exploration and testing as well. DP-NSAG-II allows users to interact with the optimization process in ways previous work has not explored. Specifically, users can dynamically change the decision and objective spaces as the optimization is in progress. This allows for an open-ended evolutionary search to occur, which no previous MOEAs or DMOEAs allow. The hypothesized benefits of this type of interaction is that it addresses the issue of user fatigue, while also allowing the decision maker to be more engaged in the optimization process. Future research will test this hypothesis and explore other opportunities to allow users to interact with the algorithm to improve the capabilities of the optimization process.

Another important direction for the research involves benchmarking against additional DMOEAs beyond NSGA-II and D-NSGA-II. Testing the algorithm against non-Pareto based techniques, such as decomposition-based DMOEAs, would provide a greater understanding of its overall capabilities. In addition, testing the algorithm against progressive decomposition-based approaches, such as those proposed by Li, Deb et al. (2017), would indicate whether the progressive features of the algorithm are providing any benefit.

The problem of developing dynamic multi-objective optimization tools that integrate the user in ways that allow an intractable search to become tractable presents a lot of challenges. The future development discussed in this section is a shortlist of the challenges of highest priority moving forward. It is early days in the study of progressive DMOEAs and there is much for the research community to learn.

Chapter 6. Appendix A

Complete results from test problem sets

The following pages show the detailed results from the four test cases. Each test is run for 25 generations, with changes to the MOP occurring every five generations. Each test is then run four times. A population of 25 solutions is used for each generation. A mutation rate of 0.10 is used by all algorithms. The global $\boldsymbol{Rg}$ and local $\boldsymbol{Rl}$ reference vectors are set to a null value because the user does not provide a reference direction nor solution ranking in these tests. Equation 1 is used for the novelty calculation and the k parameter (i.e., number of nearest neighbors) is set to be 30% of the population size.

Table 6-1. Novelty values for test problem 1. Complete data from Figure 4-6

Algorithm	**NSGA II**					**D-NSGA II**					**DP-NSGA II**				
Run #	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**
Generation															
1	0.092	0.127	0.061	0.101	**0.095**	0.066	0.127	0.054	0.145	**0.098**	0.112	0.071	0.087	0.049	**0.080**
2	0.011	0.014	0.008	0.026	**0.015**	0.018	0.068	0.010	0.051	**0.037**	0.028	0.035	0.014	0.013	**0.022**
3	0.002	0.001	0.002	0.013	**0.005**	0.006	0.004	0.002	0.004	**0.004**	0.008	0.008	0.022	0.008	**0.011**
4	0.009	0.001	0.007	0.003	**0.005**	0.002	0.002	0.001	0.001	**0.002**	0.013	0.020	0.013	0.007	**0.013**
5	0.006	0.001	0.001	0.001	**0.002**	0.003	0.001	0.001	0.000	**0.001**	0.018	0.053	0.036	0.052	**0.040**
6	0.064	0.044	0.024	0.036	**0.042**	0.122	0.122	0.107	0.120	**0.118**	0.096	0.135	0.101	0.150	**0.120**
7	0.043	0.010	0.026	0.025	**0.026**	0.041	0.026	0.020	0.061	**0.037**	0.036	0.034	0.024	0.048	**0.036**
8	0.048	0.005	0.012	0.017	**0.021**	0.034	0.015	0.079	0.015	**0.036**	0.071	0.045	0.037	0.033	**0.046**
9	0.017	0.007	0.005	0.015	**0.011**	0.013	0.011	0.013	0.017	**0.013**	0.039	0.051	0.026	0.037	**0.038**
10	0.017	0.006	0.018	0.009	**0.013**	0.028	0.013	0.012	0.014	**0.017**	0.021	0.050	0.018	0.045	**0.033**
11	0.058	0.058	0.044	0.052	**0.053**	0.204	0.234	0.133	0.213	**0.196**	0.195	0.142	0.174	0.186	**0.174**
12	0.053	0.025	0.031	0.043	**0.038**	0.121	0.120	0.103	0.193	**0.134**	0.167	0.110	0.124	0.076	**0.119**
13	0.069	0.014	0.024	0.028	**0.034**	0.059	0.083	0.054	0.134	**0.082**	0.120	0.049	0.125	0.072	**0.091**
14	0.092	0.013	0.025	0.018	**0.037**	0.083	0.038	0.038	0.056	**0.054**	0.070	0.058	0.090	0.038	**0.064**
15	0.030	0.007	0.016	0.013	**0.017**	0.073	0.072	0.034	0.057	**0.059**	0.053	0.044	0.052	0.038	**0.046**
16	0.013	0.004	0.004	0.004	**0.006**	0.028	0.027	0.024	0.034	**0.028**	0.070	0.071	0.097	0.014	**0.063**
17	0.010	0.002	0.009	0.002	**0.006**	0.029	0.012	0.009	0.010	**0.015**	0.064	0.056	0.069	0.012	**0.050**
18	0.012	0.011	0.007	0.001	**0.008**	0.016	0.003	0.005	0.012	**0.009**	0.054	0.052	0.084	0.011	**0.050**
19	0.014	0.015	0.005	0.015	**0.012**	0.007	0.008	0.005	0.015	**0.009**	0.045	0.050	0.064	0.012	**0.043**
20	0.007	0.021	0.003	0.039	**0.018**	0.009	0.005	0.003	0.011	**0.007**	0.042	0.057	0.064	0.014	**0.044**
21	0.029	0.040	0.020	0.051	**0.035**	0.032	0.032	0.026	0.025	**0.029**	0.014	0.033	0.029	0.047	**0.031**
22	0.025	0.048	0.005	0.035	**0.028**	0.023	0.024	0.036	0.028	**0.028**	0.018	0.021	0.006	0.057	**0.025**
23	0.030	0.000	0.008	0.021	**0.015**	0.019	0.031	0.027	0.027	**0.026**	0.020	0.014	0.008	0.035	**0.019**
24	0.017	0.000	0.008	0.010	**0.009**	0.016	0.008	0.026	0.022	**0.018**	0.016	0.008	0.005	0.028	**0.014**
25	0.008	0.000	0.002	0.000	**0.003**	0.020	0.008	0.028	0.020	**0.019**	0.007	0.009	0.012	0.025	**0.013**

Table 6-2. Pareto average rank for test problem 1. Complete data from Figure 4-7.

Algorithm	NSGA II					D-NSGA II					DP-NSGA II				
Run #	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**
Generation															
1	1.33	1.33	2.25	1.67	**1.65**	2.00	1.00	1.20	1.00	**1.30**	1.00	1.00	1.75	1.33	**1.27**
2	1.40	1.20	1.67	2.17	**1.61**	2.00	1.80	1.17	1.00	**1.49**	1.20	1.00	1.50	1.71	**1.35**
3	1.38	1.17	1.00	2.55	**1.52**	1.29	1.00	1.08	1.00	**1.09**	1.00	1.00	1.50	1.81	**1.33**
4	1.39	1.46	1.00	1.71	**1.39**	1.75	1.81	1.05	1.00	**1.40**	1.43	1.17	1.40	1.59	**1.40**
5	1.38	1.00	1.00	1.40	**1.20**	1.43	1.15	1.00	1.00	**1.14**	1.15	1.11	1.31	1.54	**1.28**
6	1.17	1.25	1.60	1.50	**1.38**	1.13	1.56	1.11	1.33	**1.28**	1.00	1.00	1.00	1.25	**1.06**
7	1.33	1.00	1.89	2.00	**1.56**	1.00	1.33	1.00	1.30	**1.16**	1.20	1.50	1.00	1.09	**1.20**
8	1.00	1.47	1.00	1.06	**1.13**	1.13	1.25	1.07	1.20	**1.16**	1.20	1.20	1.27	1.00	**1.17**
9	1.00	1.80	1.00	1.03	**1.21**	1.25	1.64	1.04	1.13	**1.26**	1.00	1.44	1.50	1.00	**1.24**
10	1.20	2.29	1.00	1.54	**1.51**	1.27	1.17	1.00	1.11	**1.14**	1.13	1.18	1.50	1.10	**1.23**
11	1.25	1.17	1.00	1.18	**1.15**	1.19	1.21	1.00	1.17	**1.14**	1.09	1.07	1.00	1.00	**1.04**
12	1.40	1.00	1.00	1.23	**1.16**	1.14	1.03	1.00	1.24	**1.10**	1.12	1.26	1.04	1.09	**1.12**
13	1.20	1.00	1.00	1.03	**1.06**	1.07	1.12	1.00	1.21	**1.10**	1.06	1.28	1.00	1.19	**1.13**
14	1.31	1.10	1.00	1.43	**1.21**	1.07	1.06	1.00	1.23	**1.09**	1.08	1.49	1.00	1.21	**1.19**
15	1.63	1.00	1.03	1.61	**1.31**	1.05	1.03	1.00	1.20	**1.07**	1.03	1.49	1.00	1.25	**1.19**
16	1.57	1.91	2.80	1.00	**1.82**	1.50	1.38	1.36	1.38	**1.41**	1.27	1.58	1.19	2.00	**1.51**
17	1.50	1.00	2.57	1.50	**1.64**	1.33	1.31	1.47	1.20	**1.33**	1.19	1.37	1.22	1.20	**1.24**
18	1.56	1.02	2.94	1.13	**1.66**	1.30	1.25	1.14	1.13	**1.20**	1.06	1.24	1.32	1.00	**1.15**
19	1.65	1.07	1.78	1.10	**1.40**	1.29	1.00	1.00	1.19	**1.12**	1.05	1.28	1.20	1.18	**1.18**
20	1.68	1.13	1.83	1.20	**1.46**	1.24	1.00	1.05	1.07	**1.09**	1.00	1.27	1.22	1.14	**1.16**
21	1.20	1.00	1.43	1.81	**1.36**	1.25	1.00	1.52	1.19	**1.24**	1.04	1.33	1.22	1.23	**1.20**
22	1.15	1.00	1.51	1.71	**1.34**	1.13	1.09	1.41	1.26	**1.22**	1.06	1.31	1.24	1.21	**1.21**
23	1.15	1.00	1.48	1.61	**1.31**	1.21	1.06	1.16	1.32	**1.19**	1.09	1.38	1.20	1.06	**1.18**
24	1.29	1.00	1.43	1.58	**1.33**	1.21	1.00	1.13	1.16	**1.12**	1.00	1.55	1.17	1.17	**1.22**
25	1.15	1.00	1.01	2.07	**1.31**	1.28	1.00	1.13	1.11	**1.13**	1.03	1.47	1.15	1.09	**1.19**

Table 6-3. Novelty values for test problem 2. Complete data from Figure 4-8

Algorithm	**NSGA II**					**D-NSGA II**					**DP-NSGA II**				
Run #	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**
Generation															
1	0.216	0.222	0.245	0.211	**0.224**	0.153	0.184	0.320	0.207	**0.216**	0.369	0.371	0.150	0.170	**0.265**
2	0.117	0.119	0.109	0.042	**0.097**	0.065	0.086	0.096	0.035	**0.070**	0.096	0.093	0.101	0.065	**0.089**
3	0.086	0.111	0.027	0.017	**0.060**	0.062	0.029	0.049	0.029	**0.042**	0.070	0.050	0.083	0.055	**0.064**
4	0.054	0.039	0.030	0.005	**0.032**	0.057	0.028	0.006	0.009	**0.025**	0.055	0.041	0.056	0.074	**0.057**
5	0.038	0.021	0.011	0.004	**0.018**	0.031	0.029	0.008	0.006	**0.018**	0.032	0.044	0.023	0.039	**0.034**
6	0.052	0.047	0.034	0.029	**0.040**	0.142	0.126	0.101	0.081	**0.113**	0.092	0.144	0.132	0.062	**0.107**
7	0.027	0.011	0.031	0.025	**0.023**	0.097	0.067	0.064	0.058	**0.071**	0.078	0.187	0.087	0.039	**0.098**
8	0.034	0.011	0.016	0.019	**0.020**	0.092	0.123	0.091	0.110	**0.104**	0.079	0.191	0.100	0.042	**0.103**
9	0.013	0.005	0.007	0.013	**0.010**	0.085	0.138	0.028	0.096	**0.087**	0.104	0.118	0.112	0.024	**0.090**
10	0.011	0.010	0.006	0.016	**0.011**	0.091	0.115	0.044	0.054	**0.076**	0.112	0.083	0.100	0.033	**0.082**
11	0.014	0.006	0.010	0.017	**0.012**	0.055	0.076	0.084	0.065	**0.070**	0.107	0.082	0.051	0.051	**0.073**
12	0.011	0.001	0.010	0.020	**0.011**	0.058	0.091	0.093	0.060	**0.076**	0.045	0.029	0.062	0.015	**0.038**
13	0.015	0.005	0.029	0.021	**0.017**	0.055	0.133	0.036	0.040	**0.066**	0.050	0.133	0.056	0.021	**0.065**
14	0.014	0.000	0.011	0.019	**0.011**	0.091	0.060	0.031	0.009	**0.048**	0.058	0.061	0.045	0.014	**0.045**
15	0.002	0.000	0.004	0.021	**0.007**	0.073	0.082	0.034	0.010	**0.050**	0.063	0.016	0.057	0.008	**0.036**
16	0.010	0.094	0.015	0.019	**0.034**	0.085	0.071	0.068	0.069	**0.073**	0.045	0.048	0.043	0.076	**0.053**
17	0.029	0.043	0.005	0.013	**0.023**	0.083	0.048	0.066	0.086	**0.071**	0.028	0.017	0.064	0.019	**0.032**
18	0.010	0.010	0.007	0.012	**0.010**	0.031	0.071	0.069	0.072	**0.061**	0.030	0.039	0.030	0.050	**0.038**
19	0.022	0.004	0.004	0.003	**0.008**	0.038	0.046	0.093	0.120	**0.074**	0.034	0.058	0.063	0.046	**0.050**
20	0.009	0.011	0.005	0.006	**0.008**	0.039	0.048	0.058	0.033	**0.044**	0.026	0.037	0.057	0.066	**0.047**
21	0.014	0.011	0.009	0.004	**0.009**	0.049	0.027	0.073	0.109	**0.065**	0.046	0.112	0.085	0.072	**0.079**
22	0.028	0.025	0.009	0.018	**0.020**	0.039	0.018	0.101	0.045	**0.051**	0.029	0.052	0.112	0.078	**0.068**
23	0.033	0.021	0.012	0.038	**0.026**	0.011	0.017	0.065	0.056	**0.037**	0.051	0.122	0.092	0.040	**0.076**
24	0.002	0.013	0.015	0.011	**0.010**	0.023	0.012	0.061	0.076	**0.043**	0.022	0.061	0.069	0.074	**0.056**
25	0.005	0.002	0.002	0.026	**0.008**	0.046	0.018	0.054	0.090	**0.052**	0.029	0.079	0.096	0.034	**0.060**

Table 6-4. Pareto average rank for test problem 2. Complete data from Figure 4-9.

Algorithm	**NSGA II**					**D-NSGA II**					**DP-NSGA II**				
Run #	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**
Generation															
1	1.00	1.00	1.33	2.20	**1.38**	1.00	1.40	2.67	1.56	**1.66**	1.33	1.16	1.00	1.00	**1.12**
2	1.00	1.00	1.81	2.59	**1.60**	1.00	2.17	2.75	1.72	**1.91**	1.61	1.39	1.00	1.00	**1.25**
3	1.00	1.00	1.86	2.70	**1.64**	1.00	2.35	2.87	1.66	**1.97**	1.54	1.65	1.00	1.00	**1.30**
4	1.00	1.00	1.88	2.84	**1.68**	1.00	2.29	2.90	1.88	**2.02**	1.52	1.68	1.00	1.00	**1.30**
5	1.00	1.00	1.87	2.88	**1.69**	1.00	2.20	2.91	1.88	**2.00**	1.39	1.72	1.00	1.00	**1.28**
6	1.02	1.00	1.30	1.17	**1.12**	1.36	1.77	1.43	1.48	**1.51**	1.19	1.13	1.06	1.14	**1.13**
7	1.04	1.00	1.18	1.08	**1.07**	1.42	1.60	1.71	1.33	**1.52**	1.18	1.10	1.03	1.15	**1.11**
8	1.01	1.00	1.18	1.00	**1.05**	1.35	1.56	1.43	1.30	**1.41**	1.10	1.11	1.06	1.10	**1.09**
9	1.05	1.00	1.17	1.00	**1.06**	1.36	1.71	1.12	1.24	**1.36**	1.15	1.22	1.06	1.13	**1.14**
10	1.06	1.00	1.20	1.00	**1.07**	1.37	1.33	1.33	1.18	**1.30**	1.21	1.25	1.11	1.11	**1.17**
11	1.10	1.00	1.13	1.00	**1.06**	1.40	1.18	1.42	1.16	**1.29**	1.19	1.25	1.08	1.12	**1.16**
12	1.15	1.10	1.18	1.00	**1.11**	1.33	1.15	1.26	1.17	**1.23**	1.25	1.28	1.10	1.11	**1.18**
13	1.20	1.15	1.07	1.00	**1.11**	1.37	1.18	1.35	1.16	**1.26**	1.26	1.35	1.03	1.10	**1.18**
14	1.13	1.19	1.29	1.00	**1.15**	1.32	1.18	1.29	1.08	**1.22**	1.23	1.39	1.03	1.08	**1.18**
15	1.13	1.01	1.29	1.00	**1.11**	1.30	1.13	1.11	1.10	**1.16**	1.21	1.52	1.03	1.08	**1.21**
16	1.06	1.08	1.31	1.05	**1.13**	1.24	1.08	1.48	1.17	**1.24**	1.24	1.26	1.05	1.16	**1.18**
17	1.10	1.11	1.32	1.00	**1.13**	1.24	1.08	1.72	1.15	**1.30**	1.12	1.21	1.07	1.13	**1.13**
18	1.12	1.10	1.29	1.00	**1.13**	1.27	1.06	1.72	1.08	**1.28**	1.10	1.16	1.09	1.09	**1.11**
19	1.23	1.12	1.23	1.06	**1.16**	1.22	1.06	1.59	1.11	**1.24**	1.24	1.12	1.06	1.08	**1.12**
20	1.25	1.03	1.22	1.05	**1.14**	1.21	1.05	1.45	1.10	**1.20**	1.17	1.08	1.08	1.06	**1.10**
21	1.09	1.05	1.31	1.11	**1.14**	1.11	1.09	1.14	1.11	**1.11**	1.33	1.11	1.15	1.02	**1.15**
22	1.10	1.01	1.38	1.11	**1.15**	1.13	1.08	1.16	1.10	**1.11**	1.55	1.11	1.15	1.02	**1.21**
23	1.08	1.01	1.36	1.10	**1.14**	1.11	1.08	1.17	1.20	**1.14**	1.58	1.13	1.14	1.04	**1.22**
24	1.08	1.01	1.36	1.08	**1.13**	1.05	1.10	1.18	1.14	**1.12**	1.62	1.13	1.15	1.04	**1.23**
25	1.13	1.01	1.31	1.10	**1.14**	1.06	1.09	1.24	1.16	**1.14**	1.63	1.12	1.25	1.04	**1.26**

Table 6-5. Novelty values for test problem 3. Complete data from Figure 4-10

Algorithm	NSGA II					D-NSGA II					DP-NSGA II				
Run #	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**
Generation															
1	0.244	0.309	0.301	0.260	**0.279**	0.255	0.319	0.297	0.288	**0.290**	0.335	0.348	0.307	0.294	**0.321**
2	0.153	0.203	0.182	0.146	**0.171**	0.074	0.116	0.147	0.105	**0.110**	0.179	0.166	0.096	0.142	**0.146**
3	0.084	0.058	0.055	0.097	**0.074**	0.057	0.057	0.091	0.044	**0.062**	0.112	0.122	0.093	0.129	**0.114**
4	0.059	0.059	0.042	0.082	**0.060**	0.034	0.033	0.075	0.031	**0.043**	0.093	0.062	0.105	0.068	**0.082**
5	0.019	0.062	0.026	0.051	**0.039**	0.008	0.021	0.014	0.030	**0.018**	0.062	0.083	0.085	0.087	**0.079**
6	0.007	0.020	0.014	0.016	**0.014**	0.116	0.074	0.060	0.027	**0.069**	0.078	0.070	0.083	0.052	**0.071**
7	0.017	0.019	0.017	0.009	**0.016**	0.019	0.018	0.110	0.014	**0.041**	0.036	0.057	0.046	0.089	**0.057**
8	0.023	0.020	0.014	0.030	**0.022**	0.007	0.003	0.071	0.008	**0.022**	0.045	0.058	0.074	0.056	**0.058**
9	0.013	0.027	0.006	0.026	**0.018**	0.121	0.009	0.035	0.013	**0.044**	0.033	0.063	0.045	0.059	**0.050**
10	0.010	0.011	0.004	0.030	**0.013**	0.077	0.103	0.025	0.002	**0.052**	0.019	0.045	0.050	0.055	**0.042**
11	0.063	0.078	0.067	0.087	**0.074**	0.113	0.119	0.118	0.098	**0.112**	0.136	0.097	0.114	0.118	**0.116**
12	0.080	0.055	0.087	0.083	**0.076**	0.154	0.109	0.124	0.118	**0.126**	0.071	0.056	0.037	0.055	**0.054**
13	0.087	0.069	0.094	0.094	**0.086**	0.127	0.098	0.121	0.115	**0.115**	0.036	0.054	0.057	0.062	**0.052**
14	0.095	0.091	0.126	0.097	**0.103**	0.125	0.125	0.098	0.109	**0.114**	0.040	0.088	0.034	0.063	**0.056**
15	0.079	0.070	0.048	0.130	**0.082**	0.141	0.112	0.108	0.081	**0.111**	0.043	0.070	0.038	0.051	**0.051**
16	0.082	0.050	0.071	0.094	**0.074**	0.112	0.096	0.154	0.105	**0.117**	0.087	0.041	0.027	0.027	**0.045**
17	0.092	0.071	0.084	0.073	**0.080**	0.141	0.089	0.134	0.104	**0.117**	0.050	0.051	0.073	0.038	**0.053**
18	0.088	0.069	0.058	0.067	**0.070**	0.095	0.106	0.117	0.130	**0.112**	0.028	0.054	0.036	0.045	**0.041**
19	0.100	0.059	0.066	0.092	**0.079**	0.157	0.116	0.103	0.105	**0.120**	0.018	0.044	0.035	0.058	**0.039**
20	0.062	0.072	0.030	0.086	**0.063**	0.109	0.110	0.162	0.113	**0.123**	0.044	0.026	0.057	0.043	**0.042**
21	0.116	0.053	0.088	0.084	**0.085**	0.137	0.150	0.136	0.123	**0.137**	0.080	0.105	0.063	0.064	**0.078**
22	0.092	0.082	0.086	0.096	**0.089**	0.117	0.116	0.113	0.132	**0.120**	0.070	0.058	0.101	0.047	**0.069**
23	0.070	0.061	0.107	0.096	**0.083**	0.145	0.126	0.131	0.115	**0.129**	0.065	0.058	0.033	0.037	**0.048**
24	0.099	0.046	0.078	0.104	**0.082**	0.107	0.091	0.118	0.118	**0.109**	0.056	0.058	0.045	0.054	**0.053**
25	0.075	0.086	0.049	0.091	**0.075**	0.095	0.108	0.123	0.111	**0.109**	0.020	0.048	0.041	0.031	**0.035**

Table 6-6. Pareto average rank for test problem 3. Complete data from Figure 4-11.

Algorithm	NSGA II					D-NSGA II					DP-NSGA II				
Run #	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**
Generation															
1	1.00	1.33	1.14	1.30	**1.19**	1.00	1.23	1.08	1.11	**1.11**	1.23	1.00	1.29	1.33	**1.21**
2	1.08	1.33	1.00	1.18	**1.15**	1.09	1.31	1.17	1.20	**1.19**	1.29	1.00	1.14	1.36	**1.20**
3	1.07	1.23	1.11	1.36	**1.19**	1.00	1.31	1.07	1.00	**1.10**	1.12	1.00	1.08	1.19	**1.10**
4	1.00	1.31	1.05	1.05	**1.10**	1.00	1.00	1.11	1.00	**1.03**	1.15	1.00	1.00	1.19	**1.09**
5	1.00	1.17	1.04	1.29	**1.12**	1.00	1.00	1.00	1.00	**1.00**	1.05	1.00	1.00	1.04	**1.02**
6	1.00	1.10	1.03	1.28	**1.10**	1.07	1.14	1.17	1.00	**1.10**	1.16	1.00	1.00	1.04	**1.05**
7	1.00	1.45	1.02	1.00	**1.12**	1.04	1.08	1.10	1.00	**1.06**	1.12	1.00	1.00	1.04	**1.04**
8	1.00	1.52	1.00	1.00	**1.13**	1.02	1.07	1.08	1.00	**1.04**	1.17	1.00	1.03	1.03	**1.06**
9	1.00	1.41	1.00	1.00	**1.10**	1.03	1.03	1.09	1.00	**1.04**	1.22	1.03	1.00	1.04	**1.07**
10	1.00	1.36	1.00	1.00	**1.09**	1.04	1.04	1.11	1.00	**1.05**	1.19	1.02	1.03	1.04	**1.07**
11	1.04	1.08	1.28	1.19	**1.15**	1.14	1.02	1.11	1.00	**1.07**	1.13	1.00	1.05	1.16	**1.09**
12	1.12	1.08	1.16	1.07	**1.11**	1.12	1.02	1.09	1.14	**1.09**	1.06	1.00	1.00	1.04	**1.03**
13	1.04	1.14	1.15	1.29	**1.15**	1.00	1.00	1.06	1.22	**1.07**	1.05	1.07	1.03	1.03	**1.05**
14	1.03	1.14	1.14	1.28	**1.15**	1.01	1.00	1.05	1.00	**1.02**	1.05	1.00	1.05	1.00	**1.03**
15	1.03	1.12	1.14	1.29	**1.15**	1.02	1.00	1.03	1.05	**1.03**	1.04	1.00	1.07	1.02	**1.03**
16	1.03	1.11	1.19	1.29	**1.15**	1.01	1.00	1.05	1.08	**1.03**	1.00	1.27	1.08	1.02	**1.09**
17	1.02	1.10	1.19	1.31	**1.16**	1.02	1.02	1.04	1.00	**1.02**	1.00	1.30	1.10	1.00	**1.10**
18	1.02	1.16	1.18	1.09	**1.11**	1.04	1.03	1.04	1.00	**1.03**	1.00	1.37	1.12	1.00	**1.12**
19	1.02	1.15	1.17	1.33	**1.17**	1.06	1.02	1.04	1.00	**1.03**	1.00	1.44	1.13	1.00	**1.14**
20	1.02	1.14	1.17	1.33	**1.16**	1.07	1.01	1.03	1.00	**1.03**	1.00	1.45	1.13	1.01	**1.15**
21	1.02	1.08	1.17	1.16	**1.11**	1.02	1.01	1.01	1.00	**1.01**	1.14	1.78	1.96	1.26	**1.54**
22	1.02	1.08	1.16	1.15	**1.10**	1.02	1.01	1.01	1.00	**1.01**	1.10	1.79	1.94	1.26	**1.53**
23	1.02	1.07	1.16	1.14	**1.10**	1.02	1.01	1.01	1.00	**1.01**	1.12	1.81	1.94	1.27	**1.53**
24	1.01	1.07	1.16	1.13	**1.10**	1.02	1.03	1.01	1.00	**1.01**	1.11	1.82	1.91	1.26	**1.52**
25	1.01	1.07	1.17	1.13	**1.09**	1.01	1.03	1.01	1.00	**1.01**	1.10	1.80	1.90	1.27	**1.52**

Table 6-7. Novelty values for test problem 4. Complete data from Figure 4-12

Algorithm	**NSGA II**					**D-NSGA II**					**DP-NSGA II**				
Run #	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**
Generation															
1	0.193	0.046	0.126	0.061	**0.106**	0.140	0.143	0.165	0.145	**0.148**	0.231	0.168	0.098	0.133	**0.158**
2	0.054	0.023	0.055	0.033	**0.042**	0.059	0.071	0.069	0.049	**0.062**	0.103	0.072	0.053	0.068	**0.074**
3	0.028	0.011	0.024	0.018	**0.020**	0.026	0.034	0.011	0.010	**0.020**	0.076	0.061	0.069	0.036	**0.061**
4	0.026	0.018	0.001	0.014	**0.015**	0.022	0.012	0.004	0.006	**0.011**	0.053	0.042	0.042	0.043	**0.045**
5	0.001	0.004	0.000	0.010	**0.004**	0.005	0.002	0.004	0.001	**0.003**	0.075	0.066	0.024	0.054	**0.055**
6	0.004	0.195	0.002	0.224	**0.106**	0.226	0.163	0.108	0.158	**0.164**	0.300	0.224	0.206	0.275	**0.251**
7	0.002	0.019	0.001	0.052	**0.018**	0.169	0.075	0.141	0.188	**0.143**	0.138	0.137	0.118	0.175	**0.142**
8	0.004	0.029	0.000	0.039	**0.018**	0.065	0.119	0.098	0.048	**0.082**	0.125	0.104	0.089	0.100	**0.105**
9	0.005	0.006	0.008	0.028	**0.012**	0.021	0.028	0.115	0.047	**0.053**	0.092	0.103	0.043	0.086	**0.081**
10	0.000	0.002	0.000	0.025	**0.007**	0.022	0.011	0.102	0.021	**0.039**	0.071	0.062	0.033	0.048	**0.053**
11	0.034	0.322	0.023	0.313	**0.173**	0.233	0.293	0.363	0.258	**0.287**	0.284	0.238	0.329	0.339	**0.297**
12	0.000	0.092	0.025	0.125	**0.061**	0.186	0.169	0.146	0.181	**0.171**	0.176	0.152	0.149	0.132	**0.152**
13	0.000	0.050	0.054	0.035	**0.035**	0.152	0.142	0.158	0.171	**0.156**	0.108	0.082	0.175	0.110	**0.119**
14	0.000	0.054	0.060	0.099	**0.053**	0.160	0.100	0.132	0.113	**0.126**	0.086	0.131	0.125	0.127	**0.117**
15	0.014	0.019	0.070	0.034	**0.034**	0.112	0.102	0.110	0.115	**0.110**	0.080	0.082	0.118	0.080	**0.090**
16	0.000	0.010	0.051	0.013	**0.019**	0.056	0.060	0.058	0.066	**0.060**	0.062	0.061	0.047	0.078	**0.062**
17	0.000	0.011	0.018	0.004	**0.008**	0.040	0.038	0.053	0.050	**0.045**	0.034	0.043	0.031	0.042	**0.038**
18	0.000	0.002	0.031	0.004	**0.009**	0.033	0.082	0.059	0.064	**0.060**	0.045	0.030	0.042	0.042	**0.040**
19	0.001	0.001	0.025	0.033	**0.015**	0.040	0.042	0.028	0.036	**0.036**	0.042	0.028	0.059	0.035	**0.041**
20	0.020	0.006	0.026	0.007	**0.015**	0.047	0.052	0.028	0.053	**0.045**	0.038	0.037	0.048	0.038	**0.040**
21	0.020	0.013	0.030	0.017	**0.020**	0.053	0.043	0.051	0.049	**0.049**	0.065	0.051	0.061	0.040	**0.054**
22	0.002	0.008	0.034	0.007	**0.013**	0.072	0.029	0.059	0.049	**0.052**	0.060	0.046	0.042	0.033	**0.045**
23	0.004	0.001	0.016	0.000	**0.005**	0.044	0.032	0.057	0.045	**0.044**	0.046	0.045	0.035	0.034	**0.040**
24	0.000	0.003	0.023	0.000	**0.006**	0.054	0.019	0.057	0.043	**0.043**	0.035	0.053	0.050	0.029	**0.042**
25	0.027	0.001	0.021	0.000	**0.012**	0.043	0.018	0.050	0.042	**0.038**	0.035	0.047	0.040	0.032	**0.038**

Table 6-8. Pareto average rank for test problem 4. Complete data from Figure 4-13.

Algorithm	NSGA II					D-NSGA II					DP-NSGA II				
Run #	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**	**1**	**2**	**3**	**4**	**Average**
Generation															
1	1.00	2.42	2.00	2.63	**2.01**	2.00	2.00	1.00	1.00	**1.50**	1.00	1.00	2.00	2.00	**1.50**
2	1.00	2.69	2.00	2.50	**2.05**	2.00	2.00	1.00	1.00	**1.50**	1.00	1.00	2.00	2.00	**1.50**
3	2.00	2.67	2.00	2.50	**2.29**	3.00	2.00	1.00	1.00	**1.75**	1.00	1.00	2.50	2.00	**1.63**
4	2.00	2.58	2.00	2.63	**2.30**	3.00	2.00	1.00	1.00	**1.75**	1.00	1.00	2.50	2.00	**1.63**
5	2.00	2.58	2.00	2.86	**2.36**	3.00	2.00	1.00	1.00	**1.75**	1.00	1.00	2.33	2.00	**1.58**
6	1.00	1.53	1.50	1.73	**1.44**	1.29	1.00	1.25	1.11	**1.16**	1.00	1.13	1.09	1.10	**1.08**
7	1.00	1.35	1.33	1.56	**1.31**	1.00	1.00	1.23	1.15	**1.10**	1.07	1.10	1.06	1.15	**1.09**
8	1.00	1.29	1.14	1.26	**1.17**	1.00	1.00	1.06	1.07	**1.03**	1.04	1.18	1.00	1.27	**1.12**
9	1.00	1.33	1.06	1.30	**1.17**	1.06	1.00	1.10	1.15	**1.08**	1.00	1.15	1.00	1.00	**1.04**
10	1.00	1.20	1.00	1.27	**1.12**	1.16	1.00	1.05	1.21	**1.10**	1.06	1.06	1.00	1.00	**1.03**
11	1.04	1.15	1.00	1.00	**1.05**	1.00	1.17	1.04	1.20	**1.10**	1.05	1.07	1.03	1.00	**1.04**
12	1.15	1.03	1.00	1.00	**1.05**	1.04	1.24	1.03	1.11	**1.11**	1.04	1.11	1.06	1.15	**1.09**
13	1.41	1.03	1.00	1.00	**1.11**	1.04	1.19	1.03	1.04	**1.07**	1.19	1.18	1.06	1.08	**1.13**
14	1.51	1.02	1.00	1.00	**1.13**	1.04	1.25	1.00	1.13	**1.10**	1.18	1.16	1.00	1.14	**1.12**
15	1.57	1.00	1.00	1.00	**1.14**	1.05	1.28	1.00	1.06	**1.10**	1.03	1.06	1.03	1.23	**1.09**
16	1.61	1.00	1.00	1.00	**1.15**	1.31	1.76	1.14	2.00	**1.55**	1.67	1.95	1.20	2.29	**1.78**
17	1.63	1.00	1.00	1.00	**1.16**	1.28	1.78	1.09	1.96	**1.53**	1.41	2.00	1.29	2.06	**1.69**
18	1.64	1.00	1.00	1.00	**1.16**	1.23	1.75	1.07	2.00	**1.51**	1.37	2.19	1.24	1.94	**1.68**
19	1.63	1.00	1.00	1.00	**1.16**	1.24	1.70	1.06	2.00	**1.50**	1.41	2.17	1.28	1.93	**1.70**
20	1.64	1.00	1.00	1.00	**1.16**	1.19	1.69	1.06	2.00	**1.48**	1.38	1.95	1.24	2.08	**1.66**
21	1.00	1.00	1.50	1.00	**1.13**	1.03	1.77	1.11	1.80	**1.42**	1.88	2.47	1.96	2.35	**2.16**
22	1.00	1.00	1.80	1.00	**1.20**	1.02	1.84	1.11	1.80	**1.44**	1.84	2.33	1.85	2.19	**2.05**
23	1.33	1.00	2.00	1.00	**1.33**	1.02	1.85	1.00	1.76	**1.41**	1.88	2.46	1.88	2.17	**2.10**
24	1.33	1.00	2.00	1.00	**1.33**	1.02	1.79	1.00	1.72	**1.38**	1.93	2.44	1.82	2.19	**2.10**
25	1.75	1.00	2.00	1.00	**1.44**	1.02	1.85	1.00	1.83	**1.43**	1.92	2.52	1.76	2.17	**2.09**

REFERENCES

Allmendinger, R., X. Li and J. Branke (2008). "Reference point-based particle swarm optimization using a steady-state approach." Simulated Evolution and Learning: 200-209.

Archer, B. L. (1968). The structure of design processes, Royal College of Art.

Azzouz, R., S. Bechikh and L. Ben Said (2015). Multi-objective optimization with dynamic constraints and objectives: new challenges for evolutionary algorithms. Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, ACM.

Azzouz, R., S. Bechikh and L. B. Said (2017). "A dynamic multi-objective evolutionary algorithm using a change severity-based adaptive population management strategy." Soft Computing **21**(4): 885-906.

Azzouz, R., S. Bechikh and L. B. Said (2017). Dynamic multi-objective optimization using evolutionary algorithms: a survey. Recent Advances in Evolutionary Multi-objective Optimization, Springer**:** 31-70.

Bäck, T., D. Fogel and Z. Michalewicz (1997). "Handbook of evolutionary computation." Release **97**(1): B1.

Bechikh, S., M. Elarbi and L. B. Said (2017). Many-objective Optimization Using Evolutionary Algorithms: A Survey. Recent Advances in Evolutionary Multi-objective Optimization, Springer**:** 105-137.

Bechikh, S., M. Kessentini, L. B. Said and K. Ghédira (2015). "Chapter four-preference incorporation in evolutionary multiobjective optimization: A survey of the state-of-the-art." Advances in Computers **98**: 141-207.

Boyd, S. and L. Vandenberghe (2004). Convex optimization, Cambridge university press.

Branke, J. and K. Deb (2005). Integrating user preferences into evolutionary multi-objective optimization. Knowledge incorporation in evolutionary computation, Springer**:** 461-477.

Branke, J., K. Deb, K. Miettinen and R. Słowiński (2008). Multiobjective optimization: Interactive and evolutionary approaches, Springer.

Branke, J., T. Kaußler and H. Schmeck (2001). "Guidance in evolutionary multi-objective optimization." Advances in Engineering Software **32**(6): 499-507.

Brant, J. C. and K. O. Stanley (2017). "Minimal Criterion Coevolution: A New Approach to Open-Ended Search."

Brintrup, A. M., J. Ramsden, H. Takagi and A. Tiwari (2008). "Ergonomic chair design by fusing qualitative and quantitative criteria using interactive genetic algorithms." IEEE Transactions on Evolutionary Computation **12**(3): 343-354.

Bryant, J. A. and T. Ahmed (2008). "Condensate water collection for an institutional building in Doha, Qatar: an opportunity for water sustainability."

Chen, R., K. Li and X. Yao (2016). "Dynamic Multi-Objectives Optimization with a Changing Number of Objectives." arXiv preprint arXiv:1608.06514.

Chong, Y. T., C.-H. Chen and K. F. Leong (2009). "A heuristic-based approach to conceptual design." Research in Engineering Design **20**(2): 97-116.

Christman, J. R. and B. G. Woolley (2015). Augmenting Interactive Evolution with Multi-objective Optimization. Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on, IEEE.

Christman, J. R. and B. G. Woolley (2015). "Augmenting Interactive Evolution with Multi-objective Optimization." 973-980.

Coello, C. A. C., D. A. Van Veldhuizen and G. B. Lamont (2002). Evolutionary algorithms for solving multi-objective problems, Springer.

Cohon, J. L. and D. H. Marks (1975). "A review and evaluation of multiobjective programing techniques." Water Resources Research **11**(2): 208-220.

Cvetkovic, D. and I. C. Parmee (2002). "Preferences and their application in evolutionary multiobjective optimization." IEEE Transactions on evolutionary computation **6**(1): 42-57.

Das, I. and J. E. Dennis (1997). "A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems." Structural and multidisciplinary optimization **14**(1): 63-69.

Datta, R. and A. Gupta (2016). "Recent Advances in Evolutionary Multi-objective Optimization."

De Jong, E., R. Watson and J. Pollack (2001). "Reducing bloat and promoting diversity using multi-objective methods."

Deb, K. (2003). "Multi-objective evolutionary algorithms: Introducing bias among Pareto-optimal solutions." Advances in evolutionary computing: 263-292.

Deb, K. (2014). Multi-objective optimization. Search methodologies, Springer**:** 403-449.

Deb, K., J. Horn and D. E. Goldberg (1993). "Multimodal deceptive functions." Complex Systems **7**(2): 131-154.

Deb, K. and H. Jain (2014). "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints." IEEE Trans. Evolutionary Computation **18**(4): 577-601.

Deb, K. and A. Kumar (2007). Interactive evolutionary multi-objective optimization and decision-making using reference direction method. Proceedings of the 9th annual conference on Genetic and evolutionary computation, ACM.

Deb, K., M. Mohan and S. Mishra (2005). "Evaluating the ε-domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solutions." Evolutionary computation **13**(4): 501-525.

Deb, K., A. Pratap, S. Agarwal and T. Meyarivan (2002). "A fast and elitist multiobjective genetic algorithm: NSGA-II." IEEE transactions on evolutionary computation **6**(2): 182-197.

Deb, K., U. Rao N and S. Karthik (2007). Dynamic multi-objective optimization and decision-making using modified NSGA-II: a case study on hydro-thermal power scheduling. Evolutionary Multi-Criterion Optimization, Springer.

Deb, K., A. Sinha, P. J. Korhonen and J. Wallenius (2010). "An interactive evolutionary multiobjective optimization method based on progressively approximated value functions." IEEE Transactions on Evolutionary Computation **14**(5): 723-739.

Deb, K. and J. Sundar (2006). Reference point based multi-objective optimization using evolutionary algorithms. Proceedings of the 8th annual conference on Genetic and evolutionary computation, ACM.

Deb, K., L. Thiele, M. Laumanns and E. Zitzler (2002). Scalable multi-objective optimization test problems. Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on, IEEE.

Duffy, A., M. Andreasen, K. MacCallum and L. Reijers (1993). "Design coordination for concurrent engineering." Journal of Engineering Design **4**(4): 251-265.

Endres, D. M. and J. E. Schindelin (2003). "A new metric for probability distributions." IEEE Transactions on Information theory **49**(7): 1858-1860.

Farina, M., K. Deb and P. Amato (2004). "Dynamic multiobjective optimization problems: test cases, approximations, and applications." IEEE Transactions on evolutionary computation **8**(5): 425-442.

Goh, C.-K., Y. S. Ong and K. C. Tan (2009). "Multi-objective memetic algorithms."

Goh, C.-K. and K. C. Tan (2009). "A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization." IEEE Transactions on Evolutionary Computation **13**(1): 103-127.

Goldberg, D. E. (1987). "Simple genetic algorithms and the minimal, deceptive problem." Genetic algorithms and simulated annealing **74**: 88.

Goldberg, D. E. (2013). The design of innovation: Lessons from and for competent genetic algorithms, Springer Science & Business Media.

Goldberg, D. E. and J. Richardson (1987). Genetic algorithms with sharing for multimodal function optimization. Genetic algorithms and their applications: Proceedings of the Second International Conference on Genetic Algorithms, Hillsdale, NJ: Lawrence Erlbaum.

Gomes, J., P. Mariano and A. L. Christensen (2015). "Devising Effective Novelty Search Algorithms." 943-950.

Gregory, S. A. (2013). The design method, Springer.

Guan, S.-U., Q. Chen and W. Mo (2005). "Evolving dynamic multi-objective optimization problems with objective replacement." Artificial Intelligence Review **23**(3): 267-293.

Hämäläinen, R. P. and J. Mäntysaari (2002). "Dynamic multi-objective heating optimization." European Journal of Operational Research **142**(1): 1-15.

Hatzakis, I. and D. Wallace (2006). Dynamic multi-objective optimization with evolutionary algorithms: a forward-looking approach. Proceedings of the 8th annual conference on Genetic and evolutionary computation, ACM.

Horn, J., N. Nafpliotis and D. E. Goldberg (1994). A niched Pareto genetic algorithm for multiobjective optimization. Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on, Ieee.

Hornby, G. S. (2006). ALPS: the age-layered population structure for reducing the problem of premature convergence. Proceedings of the 8th annual conference on Genetic and evolutionary computation, ACM.

Hu, J., E. Goodman, K. Seo, Z. Fan and R. Rosenberg (2005). "The hierarchical fair competition (hfc) framework for sustainable evolutionary algorithms." Evolutionary Computation **13**(2): 241-277.

Hutter, M. and S. Legg (2006). "Fitness uniform optimization." IEEE Transactions on Evolutionary Computation **10**(5): 568-589.

Jiang, S., Y.-S. Ong, J. Zhang and L. Feng (2014). "Consistencies and contradictions of performance metrics in multiobjective optimization." IEEE Transactions on Cybernetics **44**(12): 2391-2404.

Jin, Y., T. Okabe and B. Sendho (2001). Adapting weighted aggregation for multiobjective evolution strategies. Evolutionary multi-criterion optimization, Springer.

Jin, Y. and B. Sendhoff (2002). Incorporation Of Fuzzy Preferences Into Evolutionary Multiobjective Optimization. GECCO.

Kim, H.-S. and S.-B. Cho (2000). "Application of interactive genetic algorithm to fashion design." Engineering applications of artificial intelligence **13**(6): 635-644.

Kleeman, M. P. and G. B. Lamont (2007). Scheduling of flow-shop, job-shop, and combined scheduling problems using MOEAs with fixed and variable length chromosomes. Evolutionary Scheduling, Springer**:** 49-99.

Koksalan, M. and I. Karahan (2010). "An interactive territory defining evolutionary algorithm: iTDEA." IEEE Transactions on Evolutionary Computation **14**(5): 702-722.

Koo, W. T., C. K. Goh and K. C. Tan (2010). "A predictive gradient strategy for multiobjective evolutionary algorithms in a fast changing environment." Memetic Computing **2**(2): 87-110.

Lehman, J. and K. O. Stanley (2008). Exploiting Open-Endedness to Solve Problems Through the Search for Novelty. ALIFE.

Lehman, J. and K. O. Stanley (2011). "Abandoning objectives: Evolution through the search for novelty alone." Evolutionary computation **19**(2): 189-223.

Lehman, J. and K. O. Stanley (2011). Evolving a diversity of virtual creatures through novelty search and local competition. Proceedings of the 13th annual conference on Genetic and evolutionary computation, ACM.

Li, K., K. Deb and X. Yao (2017). "Integration of Preferences in Decomposition Multi-Objective Optimization." arXiv preprint arXiv:1701.05935.

Marques, V. M., C. Reis and J. T. Machado (2010). Interactive evolutionary computation in music. Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on, IEEE.

Mouret, J.-B. (2011). Novelty-based multiobjectivization. New horizons in evolutionary robotics, Springer**:** 139-154.

Mouret, J.-B. and J. Clune (2015). "Illuminating search spaces by mapping elites." arXiv preprint arXiv:1504.04909.

Mueller, C. T. and J. A. Ochsendorf (2015). "Combining structural performance and designer preferences in evolutionary design space exploration." Automation in Construction **52**: 70-82.

Muyl, F., L. Dumas and V. Herbert (2004). "Hybrid method for aerodynamic shape optimization in automotive industry." Computers & Fluids **33**(5): 849-858.

Nguyen, A. M., J. Yosinski and J. Clune (2015). "Innovation Engines." 959-966.

Obayashi, S., D. Sasaki, Y. Takeguchi and N. Hirose (2000). "Multiobjective evolutionary computation for supersonic wing-shape optimization." IEEE transactions on evolutionary computation **4**(2): 182-187.

Ohsaki, M. and H. Takagi (1998). "Reduction of the fatigue of human interactive EC operators-improvement of present interface by prediction of evaluation order." JOURNAL-JAPANESE SOCIETY FOR ARTIFICIAL INTELLIGENCE **13**: 712-719.

Palaniappan, S., S. Zein-Sabatto and A. Sekmen (2001). Dynamic multiobjective optimization of war resource allocation using adaptive genetic algorithms. SoutheastCon 2001. Proceedings. IEEE, IEEE.

Pelikan, M. and D. E. Goldberg (2001). Escaping hierarchical traps with competent genetic algorithms. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001).

Phan, D. H. and J. Suzuki (2013). R2-IBEA: R2 indicator based evolutionary algorithm for multiobjective optimization. Evolutionary Computation (CEC), 2013 IEEE Congress on, IEEE.

Rachmawati, L. and D. Srinivasan (2010). "Incorporating the notion of relative importance of objectives in evolutionary multiobjective optimization." IEEE transactions on evolutionary computation **14**(4): 530-546.

Ripon, K. N., C.-H. Tsang, S. Kwong and M.-K. Ip (2006). Multi-objective evolutionary clustering using variable-length real jumping genes genetic algorithm. 18th International Conference on Pattern Recognition (ICPR'06), IEEE.

Said, L. B., S. Bechikh and K. Ghédira (2010). "The r-dominance: a new dominance relation for interactive evolutionary multicriteria decision making." IEEE Transactions on Evolutionary Computation **14**(5): 801-818.

Shen, X.-N. and X. Yao (2015). "Mathematical modeling and multi-objective evolutionary algorithms applied to dynamic flexible job shop scheduling problems." Information Sciences **298**: 198-224.

Sims, K. (1992). Interactive evolution of dynamical systems. Toward a practice of autonomous systems: Proceedings of the first European conference on artificial life.

Srinivas, N. and K. Deb (1994). "Muiltiobjective optimization using nondominated sorting in genetic algorithms." Evolutionary computation **2**(3): 221-248.

Takagi, H. (2001). "Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation." Proceedings of the IEEE **89**(9): 1275-1296.

Thiele, L., K. Miettinen, P. J. Korhonen and J. Molina (2009). "A preference-based evolutionary algorithm for multi-objective optimization." Evolutionary computation **17**(3): 411-436.

Ting, C.-K., C.-N. Lee, H.-C. Chang and J.-S. Wu (2009). "Wireless heterogeneous transmitter placement using multiobjective variable-length genetic algorithm." IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **39**(4): 945-958.

Toffolo, A. and E. Benini (2003). "Genetic diversity as an objective in multi-objective evolutionary algorithms." Evolutionary Computation **11**(2): 151-167.

Turrin, M., P. von Buelow, A. Kilian and R. Stouffs (2012). "Performative skins for passive climatic comfort." Automation in Construction **22**: 36-50.

Vassiliades, V., K. Chatzilygeroudis and J.-B. Mouret (2017). A comparison of illumination algorithms in unbounded spaces. Workshop" Measuring and Promoting Diversity in Evolutionary Algorithms", Genetic and Evolutionary Computation Conference.

Vierlinger, R. and K. Bollinger (2014). Accommodating change in parametric design. Proceedings of ACADIA.

Von Buelow, P. (2012). "ParaGen: Performative Exploration of generative systems." Journal of the International Association for Shell and Spatial Structures **53**(4): 271-284.

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

Wahle, J., O. Annen, C. Schuster, L. Neubert and M. Schreckenberg (2001). "A dynamic route guidance system based on real traffic data." European Journal of Operational Research **131**(2): 302-308.

Wang, H., Y. Jin and X. Yao (2017). "Diversity assessment in many-objective optimization." IEEE transactions on cybernetics **47**(6): 1510-1522.

Wang, J. (2001). "Ranking engineering design concepts using a fuzzy outranking preference model." Fuzzy sets and systems **119**(1): 161-170.

Wang, J. (2002). "Improved engineering design concept selection using fuzzy sets." International Journal of Computer Integrated Manufacturing **15**(1): 18-27.

Whitley, D. (1991). "Fundamental principles of deception in genetic algorithms." F oundations of Genetic Algorithms: 221-241.

Wright, S. J. and J. Nocedal (1999). "Numerical optimization." Springer Science **35**(67-68): 7.

Yang, S., M. Li, X. Liu and J. Zheng (2013). "A grid-based evolutionary algorithm for many-objective optimization." IEEE Transactions on Evolutionary Computation **17**(5): 721-736.

Zadeh, L. (1963). "Optimality and non-scalar-valued performance criteria." IEEE transactions on Automatic Control **8**(1): 59-60.

Zhang, C., K. C. Tan, L. H. Lee and L. Gao (2017). "Adjust weight vectors in MOEA/D for bi-objective optimization problems with discontinuous Pareto fronts." Soft Computing: 1-16.

Zhang, Q. and H. Li (2007). "MOEA/D: A multiobjective evolutionary algorithm based on decomposition." IEEE Transactions on evolutionary computation **11**(6): 712-731.

Zitzler, E., K. Deb and L. Thiele (2000). "Comparison of multiobjective evolutionary algorithms: Empirical results." Evolutionary computation **8**(2): 173-195.

Zitzler, E. and S. Künzli (2004). Indicator-based selection in multiobjective search. International Conference on Parallel Problem Solving from Nature, Springer.