

YAŞAR UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

MASTER THESIS

**MULTI-OBJECTIVE COMPUTATIONAL DESIGN
IN ARCHITECTURE**

Cemre UĞURLU

Thesis Advisor: Prof. Dr. M. Fatih Taşgetiren

Department of Industrial Engineering

Presentation Date: 13.07.2015

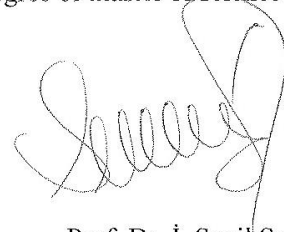
**Bornova-İZMİR
2015**

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of master of science.



Prof. Dr. M. Fatih Taşgetiren (Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of master of science.

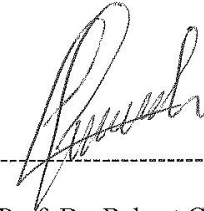


Prof. Dr. İ. Sevil Sarıyıldız

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of master of science.



Yard. Doç. Dr. Ömer Öztürkoğlu



Prof. Dr. Behzat GÜRKAN
Director of the Graduate School

ABSTRACT

MULTI-OBJECTIVE COMPUTATIONAL DESIGN IN ARCHITECTURE

UGURLU, Cemre

M.Sc. in Industrial Engineering

Supervisor: Prof. Dr. M. Fatih TAŞGETİREN

June 2015, 95 pages

This thesis examines the real-parameter optimization methods through constrained single objective and multi-objective test functions. Based on this experience, it presents applicability of these methods in architectural manner using computational design techniques.

In the first phase of the study, benchmark functions presented in CEC 2006 have been considered. Proposed algorithm (EDE algorithm) has been applied to test the functions and competitive results have been gathered. Second phase of the study was regarded to the multi-objective constrained real parameter optimization. This part also required testing EDE algorithm through multi-objective constrained test functions. On the other hand, since NSGA-II and JDE algorithms have already yielded good performance on previous architectural problems; they have been implemented again to the most famous constrained multi-objective test functions in the literature. After the deep revision of the selected algorithms and presentation of the constraint handling methods that have been used, novel architectural applications have been sought. Three multi-objective constrained architectural case studies have been defined.

The first application presents the results obtained by NSGA-II, JDE and EDE algorithms on a restaurant layout optimization problem that tries to maximize total profit while minimizing investment. The algorithms were implemented in a Parametric Design Environment that is familiar in the architectural practice. It is demonstrated that the JDE algorithm achieved slightly better performance than NSGA-II and EDE algorithms in terms of hypervolume calculation, and achieve promising results when the Pareto front approximation is examined. To the best of knowledge about literature, this is the first application of multi-objective approach for restaurant design.

Application 2 focuses on the conceptual design and the development of a floating neighborhood by taking advantage of computational methods. An application to a concept design of a floating neighborhood in the region of Urla – a coastal town close to İzmir in Turkey, has been studied. The scenario that has been addressed concerns the development of an efficient floating settlement between four islands that are local to the study region. This study is revolved around two issues. The first one is about configuration of the functions (accommodation, marine, yacht club, public area) in order to maximize accessibility, wind protection and visibility subject to both technical and nontechnical constraints. The second issue is to find a suitable form, generated by shortest walk algorithm that decides how to create roads between functions where their places are gathered from optimization solutions. For the configuration of these functions, since wind protection and visibility objectives, as well as accessibility and visibility are conflicting with each other, multi-objective evolutionary algorithms have been used. With respect to the algorithm comparison, it is found that NSGA-II performs better than DE algorithm while EDE has better formance than JDE.

In application 3, Pareto optimal design solutions have been tried to find for a specific case study which is floating underwater hotel room design. The problem is shaping both underwater part and superstructure part of the hotel room. A multi-objective problem was formulated by considering the minimization of total cost and the maximization of shading performance. It is demonstrated that the solutions are all feasible and have sensible shapes.

Due to the perfection requirement in architecture for both technical and aesthetic scopes, as well as its complexity, better performing algorithms or constraint handling methods were tried to be obtained. This can be also justified by the fact that one algorithm that has good performance in one problem may not have a good performance on other problems. It can be defined by “No Free Lunch”.

This thesis is consisted of 5 chapters which include all of these subjects.

Keywords: Multi-Objective Architectural Design, Design Optimization, Evolutionary Algorithms, Differential Evolution, Constraint Handling, Restaurant Layout Optimization, Floating Settlements, Hotel Room Design and Computational Design.

ÖZET

MİMARİDE ÇOK AMAÇLI BİLİŞİMSEL TASARIM

Cemre UĞURLU

Yüksek Lisans Tezi, Endüstri Mühendisliği Bölümü

Tez Danışmanı: Prof. Dr. M. Fatih TAŞGETİREN

Haziran 2015, 95 sayfa

Bu tezde kısıtlı tek amaçlı ve kısıtlı çok amaçlı reel sayı optimizasyon yöntemleri incelenmiştir. Bu incelemeden elde edilen tecrübe dahilinde bu yöntemler bilişimsel tasarım teknikleri kullanılarak mimariye uygulanmıştır.

Çalışmanın ilk aşamasında, CEC 2006’da sunulan kıyaslama fonksiyonları dikkate alınmıştır. Önerilen algoritma (EDE algoritması) bu test fonksiyonlarına uygulanmıştır ve rekabet edebilen sonuçlar elde edilmiştir. Çalışmanın ikinci aşamasında çok amaçlı reel sayı optimizasyonu ile ilgilenilmiştir. Bu kısımda da EDE algoritması çok amaçlı fonksiyonlarda test edilmiştir. Aynı zamanda, NSGA-II ve DE algoritmaları daha önceki mimari tasarım problemlerinde iyi bir performans gösterdiği için bu algoritmalar literatürde en ünlü olan çok amaçlı test fonksiyonlarına uygulanmıştır. Algoritmaların kapsamlı literatür taramalarından ve kısıtları ele alma yöntemlerinin sunumundan sonra özgün mimari uygulamalar aranmıştır. Çok amaçlı kısıtlı üç örnek mimari çalışma tanımlanmıştır.

İlk uygulamada, toplam karı artırmayı ve yatırım maliyetini azaltmayı amaçlayan restoran tasarım probleminin NSGA-II, JDE ve EDE algoritmalarından elde edilen sonuçları sunulmaktadır. Algoritmalar mimarinin aşına olduğu Parametrik Tasarım Ortamına uyarlanmıştır. JDE algoritması NSGA-II ve EDE algoritmalarından daha iyi performans göstermiştir ve Pareto’da çıkan sonuçlar tatminkardır. Elde edilen bilgilere göre, çok amaçlı bakışın restoran tasarımına uygulandığı ilk örnektir.

İkinci uygulamada, yüzen mahalle kavramsal tasarımının bilişimsel yöntemlerden faydalanılarak geliştirilmesine odaklanılmıştır. Kavramsal tasarım için uygulama bölgesi İzmir’de bir sahil kasabası olan Urla olarak seçilmiştir. Tasarımın seçilen bölgedeki dört adadan faydalanılarak geliştirilmesi hedeflenmiştir. Bu çalışma iki temel konuyu içermektedir. Birincisi mahallenin içerisindeki her bir fonksiyonun ulaşılabilirlik, rüzgardan korunma ve görünürlük

amaçlarına göre ve teknik ya da teknik olmayan kısıtlar doğrultusunda dört ada arasına veya çevresine uygun yerleşim bulmaktır. Diğer konu ise birinci optimizasyon modelinden elde edilen koordinatların üzerine en kısa yürüme algoritmasını kullanarak fonksiyonlar arası yürüme yollarını yaratmak ve uygun bir form oluşturmaktır. Fonksiyonların yerleşimi için rüzgardan korunma ve görünürlük amaçları çatıştığı ve ulaşılabilirlik ile görünürlük amaçları çatıştığı için çok amaçlı evrimsel algoritmalarından faydalanılmıştır. Algoritma karşılaştırmasına göre, NSGA-II, DE algoritmasından daha iyi, EDE algoritması da JDE algoritmasından daha iyi sonuçlar vermiştir.

Üçüncü uygulamada sualtı otel odası tasarımı için elde edilen Pareto optimal tasarım sonuçları bulunmaya çalışılmıştır. Problemden, otel odasının sualtı ve suüstü katlarının nasıl bir şekilde olacağına odaklanılmıştır. Çok amaçlı problemin formülleri, maliyet enazlaması ve gölge performansı ençoklaması hedeflerine bağlı olarak tasarlanmıştır. Uygulanabilir sonuçlar ve ilgi çekici tasarımlar elde edilmiştir.

Mimarinin mükemmelliğe ulaşma ihtiyacı ve karmaşık yapısından dolayı, daha iyi performans gösteren algoritmalar aranmaktadır. Bu durum, bir algoritma bir problemde çok iyi performans gösterirken başka bir problemde güzel sonuçlar vermeyebileceği ile de açıklanabilir.

Bu tez, yukarıda bahsedilen konuları içeren 5 üniteden oluşmaktadır.

Anahtar sözcükler: Çok Amaçlı Mimari Tasarım, Tasarım Optimizasyonu, Evrimsel Algoritmalar, Restoran Tasarım Optimizasyonu, Yüzen Yapılar, Otel Odası Tasarımı ve Bilişimsel Tasarım.

To my mother, Özlem

ACKNOWLEDGEMENTS

First of all, I am grateful to my supervisor, **Prof. Dr. M. Fatih Taşgetiren**, for encouraging me to complete this thesis. He was more than a supervisor and supported me at each stage of the study although sometimes facing challenges in this kind of multidisciplinary work. I owe my deepest gratitude to him.

I would like to show my gratitude to my Dean, **Prof. Dr. Sevil Sarıyıldız** for providing me an opportunity to take place in architectural field and for sharing her pearls of wisdom with me during this study.

I would like to express the deepest appreciation to my teacher **Ioannis Chatzikonstantinou** to support and encourage me as well as to help me in doing research and i came to know about so many new things.

I gratefully acknowledge the contributions of my friend, **Berk Ekici**. I would like to thank my friend and colleague **Ayça Kırımtat** for sincerely collaboration and I am really thankful to **Eda Paykoç** for encouraging me while doing this research. I also would like to thank **Ömer Öztürkoğlu** for his support.

I wish to express my sincerely thank to my head of department, **Zeynep Tuna Ultav** for encouragement in carrying out this project and for providing me opportunities to allow time for preparation of this project.

I am eternally grateful to my parents. I would first like to thank my mother, **Özlem Oğul**, without her continuous support and encouragement i never would have been able to achieve my goals. I am very thankful to my father, **Kemal Oğul** for always protecting me. I want to express gratitudes to my beloved sister, **Can Tığa** and her husband rather my beloved brother **Gökhan Tığa** and my sweety niece **Ladin Tığa**. I also thank to my grandmother, **Nezihe Kazaz**.

A special thank you to my fiancée, **Cahit Çubukçuoğlu** for always encouraging me.

Cemre UĞURLU
İzmir, 2015

TEXT OF OATH

I declare and honestly confirm that my study, titled “Multi-Objective Computational Design in Architecture” and presented as a Master’s, has been written without applying to any assistance inconsistent with scientific ethics and traditions, that all sources from which I have benefited are listed in the bibliography, and that I have benefited from these sources by means of making references.

TABLE OF CONTENTS

	Page
ABSTRACT	iii
ÖZET	v
ACKNOWLEDGEMENTS	viii
TEXT OF OATH	ix
TABLE OF CONTENTS	x
INDEX OF FIGURES	xiv
INDEX OF TABLES	xvii
INDEX OF SYMBOLS AND ABBREVIATIONS	xviii
1 INTRODUCTION	1
1.1 Research Goal	2
1.2 Methodology	2
2 PROPOSED ALGORITHMS	4
2.1 Non-Dominated Sorting Genetic Algorithm-II	4
2.2 Differential Evolution Algorithm	5
2.2.1 Steps of DE Algorithm	6
2.2.1.1 Initialization of Target Population	7
2.2.1.2 Difference -Vector Based Mutation	7

2.2.1.3	Crossover	8
2.2.1.4	Selection	9
2.2.2	Self- Adaptive Differential Evolution	9
2.2.3	Ensemble Differential Evolution	10
2.2.4	Multi-Objective Approach for DE Algorithm	11
3	CONSTRAINED REAL PARAMETER OPTIMIZATION	14
3.1	Constraint Handling Methods	15
3.1.1	Superiority of Feasible Solutions	15
3.1.2	ϵ - Constraint (EC)	15
3.2	Computational Results of Constrained RPO	16
4	MULTI-OBJECTIVE CONSTRAINED REAL PARAMETER OPTIMIZATION	20
4.1	Multi-Objective Test Functions	20
4.2	Optimization of MO Test Functions	20
4.3	Performance Measurement	21
5	APPLICATIONS ON ARCHITECTURAL OPTIMIZATION PROBLEMS	23
5.1	Application I: Architectural Design of Restaurant Layouts	23
5.1.1	Problem Definition of Application I	24
5.1.2	Objective Functions of Application I	24

5.1.2.1	Revenue Calculation	26
5.1.2.2	Running Cost Calculation	26
5.1.2.3	Investment Calculation	28
5.1.3	Parametric Model of Application I	28
5.1.4	Computational Results of Application I	30
5.2	Application II: Sustainable Designs for Floating Settlements	34
5.2.1	Problem Definition of Application II	36
5.2.2	Optimization Model of Application II	36
5.2.2.1	Objective Function I: Accessibility	37
5.2.2.2	Objective Function II: Wind Protection	38
5.2.2.3	Objective Function III: Visibility	38
5.2.2.4	Constraints	39
5.2.3	Form Finding	41
5.2.4	Parametric Model of Application II	42
5.2.5	Computational Results of Application II	42
5.3	Application III: Floating Underwater Hotel Room Design	49
5.3.1	Problem Definition of Application III	51
5.3.1.1	Decision Variables	51
5.3.1.2	Objectives	51

5.3.1.3	Constraints	53
5.3.2	Generative Model of Application III	55
5.3.3	Computational Results of Application III	56
6	CONCLUSIONS & FUTURE WORK	61
	REFERENCES	63
	CURRICULUM VITEA	77
	APPENDIX 1 CEC 2006 BENCHMARKS	78
	APPENDIX 2 MULTI-OBJECTIVE TEST FUNCTIONS	93
	APPENDIX 3 ASSUMPTIONS FOR APPLICATION I	95

INDEX OF FIGURES

Figure 1.1 Method of the Study	3
Figure 2.1 Steps of DE algorithm	7
Figure 2.2 Outline of EDE algorithm	11
Figure 5.1 Grasshopper model of restaurant design	29
Figure 5.2 Correspondance of decision variables to design dimensions	29
Figure 5.3 Complete model in Grasshopper	30
Figure 5.4 Pareto Front Approximation for NSGA-II	32
Figure 5.5 Pareto Front Approximation for DE Algorithm	33
Figure 5.6 Pareto Front Approximation for EDE Algorithm	33
Figure 5.7 Pareto front solutions (highest crowding distances & visual inspection) obtained with the NSGA-II	33
Figure 5.8 Pareto front solutions (highest crowding distance & visual inspection) obtained with the DE algorithm	34
Figure 5.9 Pareto front solutions (highest crowding distance & visual inspection) obtained with the EDE algorithm	34
Figure 5.10 A top view from the islands of Urla	36
Figure 5.11 Top view of the project region to illustrate protected regions	39
Figure 5.12 A view of project region's cruise-ship passing line	39
Figure 5.13 Example of generating the intersected lines for a housing unit	40

Figure 5.14 Generating the roads between functions with using shortest walk algorithm	42
Figure 5.15. Overview of the Grasshopper model	42
Figure 5.16 Pareto Front approximation for NSGA-II	43
Figure 5.17 Pareto Front approximation for DE Algorithm	44
Figure 5.18 Pareto Front approximation for EDE Algorithm	44
Figure 5.19 Alternative solutions for NSGA-II	46
Figure 5.20 Alternative solutions for DE Algorithm	46
Figure 5.21 Alternative solutions for EDE Algorithm	46
Figure 5.22 Final whole design of floating neighbourhood	47
Figure 5.23 Other perspective of whole design of floating neighbourhood	47
Figure 5.24 Computer render of one of the housing places	47
Figure 5.25 One of the housing places, coral bay	48
Figure 5.26 One of the housing places, mimosa bay	48
Figure 5.27 Marine part of the floating neighbourhood	48
Figure 5.28 All housing places	49
Figure 5.29 Yatch Club part of the floating neighbourhood	49
Figure 5.30 A top view of Marmaris/ Bozburun coast	50
Figure 5.31 Representation for center of gravity and center of buoyancy (shipinspection web site)	54

Figure 5.32 Grasshopper model of underwater floating hotel room	55
Figure 5.33 Superstructure shading device form	56
Figure 5.34 Pareto Front Approximation for NSGA-II	56
Figure 5.35 Pareto Front Approximation for DE	57
Figure 5.36 Pareto Front Approximation for EDE	57
Figure 5.37 Elevation of the proposed hotel room	59
Figure 5.38 Multiple rooms	59
Figure 5.39 Superstructure part illustration	59
Figure 5.40 Superstructure part illustration	60
Figure 5.41 Underwater part illustration	60

INDEX OF TABLES

Table 1 Computational Results of EDE, DE-VNS, SAMO-DE, MDE, ECHT-EP2 FOR CEC 2006 Test Problems.....	19
Table 2 Hypervolume Results of MO functions.....	22
Table 3 Hypervolume Results of Application I.....	32
Table 4 Hypervolume Results of Application II.....	45
Table 5 Hypervolume Results of Application III.....	58

INDEX OF SYMBOLS AND ABBREVIATIONS

<u>Symbols</u>	<u>Explanations</u>
$\nu(\bar{x})$	Average violation of number of constraints
δ	Tolerance value for equality constraints
t_c	Control generation
I_h	Hypervolume indicator
x_{ij}^g	i^{th} target individual at generation g
v_{ij}^g	i^{th} mutant individual at generation g
u_{ij}^g	i^{th} trial individual at generation g
$f(x)$	Fitness value of solution (x)
F_i	Mutation scale factor
CR_i	Crossover probability
KF	Value randomly chosen within the range [0,1]
x_k	Coordinate of kitchen x-axis
y_k	Coordinate of kitchen y-axis
x_s	Coordinate of service point at x-axis
y_s	Coordinate of service point at y-axis
x_{t_i}	Coordinate of each table (i) at x-axis

y_{t_i}	Coordinate of each table (i) at y-axis
WR	Width of the restaurant
LR	Length of the restaurant
WK	Width of the kitchen
LK	Length of the kitchen
RK	Rotation degree of kitchen
H	Height of the restaurant & kitchen
NT	Number of tables
NTA	Number of tables which are further from windows
NTN	Number of tables which are close to windows
ORA	Occupancy rate for tables which are further from windows
ORN	Occupancy rate for tables which are close to windows
CPT	Customer per table
CST	Customer staying time
A	Average number of customers
WHM	Working hours monthly
PR	Price per customer
RV	Revenue
TP	Total profit

RC	Running cost
SC	Staff cost
EC	Energy cost
LC	Lighting cost
cpk	Cost per Kwh
ecl	Energy consumption for a unit light bulb
noph	Number of orders per hour
wdp	Amount of how many meters each person can walk per hour
ucs	Unit staff cost per month
d_i	Distance between service point and table (i), $i=1, \dots, NT$
RNS	Required number of staff
HC	Total cost of heating
HL	Total heat loss rate
HT	The heat loss rate through walls & windows
LRW	The heat loss rate through walls
LRG	The heat loss rate through windows
HV	The heat loss rate through ventilation
V	Total volume of space
d_a	Density of air

c_a	Specific heat of air
n	Air exchange rate per hour (volumes)
U_g	Heat transfer coefficient of glass
U_F	Heat transfer coefficient of floor
U_w	Heat transfer coefficient of wall
ΔT_{out}	Temperature difference with outside
ΔT_F	Temperature difference with basement
INV	Investment
CC	Construction cost
KC	Kitchen cost
DC	Diner cost
BC	Building cost
TC	Tables cost
GC	Glass cost
WC	Wall cost
L_g	Total length of glass
L_w	Total length of wall
p_j	Horizontal position of window on wall, $j=1, \dots, 4$
w_j	Width of window

y_j	Percentage of the window within the wall
hx	x coordinate of houses
hy	y coordinate of houses
mx	x coordinate of marine
my	y coordinate of marine
yx	x coordinate of yacht club
yy	y coordinate of yacht club
px	x coordinate of public
py	y coordinate of public
pix	x coordinate of protected region
piy	y coordinate of protected region
$db_{h,p}$	Distance between functions houses and public
$db_{y,p}$	Distance between functions yacht club and public
$db_{m,p}$	Distance between functions marine and public
$db_{m,y}$	Distance between functions marine and yacht club
db_{h,p_i}	Distance between houses and protected area(i),i=1,2
db_{m,p_i}	Distance between marine and protected area(i),i=1,2
$noil_h$	Number of intersected lines for houses
$noil_y$	Number of intersected lines for yacht club

$noil_m$	Number of intersected lines for marine
Wdy	Water depth for yacht club
z_1	Accessibility
z_2	Wind protection
z_3	Visibility
ur_x	Length of underwater roof edge
ur_y	Width of underwater roof edge
uf_x	Length of underwater floor edge
uf_y	Length of underwater floor edge
h_u	Height of underwater
h_s	Height of superstructure
a_1	z coordinate of 1st control point of superstructure roof
a_2	z coordinate of 2nd control point of superstructure roof
a_3	z coordinate of 3rd control point of superstructure roof
b_1	z coordinate of 4th control point of superstructure roof
b_2	z coordinate of 5th control point of superstructure roof
b_3	z coordinate of 6th control point of superstructure roof
s	Sizes of shading device
$r_{win(1,..,3)}$	Radius of underwater windows (1,..,3)

ucc	Unit concrete cost
ugc	Unit plexi glass cost
utc	Unit textile cost
UFC	Underwater floor cost
URC	Underwater roof cost
UWC	Underwater windows cost
UWAC	Underwater walls cost
SBG	Superstructure barrier glass cost
SBW	Superstructure barrier wall cost
TLA	Total lateral area of underwater
TSA	Total surface area of barriers
SRA	Superstructure roof surface area
SRC	Superstructure roof cost
TC	Total cost
SP	Shading performance

Abbreviations

GA	Genetic Algorithm
ES	Evolution Strategy
DE	Differential Evolution

PSO	Particle Swarm Optimization
EP	Evolutionary Programming
QN	Quasi Newton
SA	Simulated Annealing
TS	Tabu Search
CEC	Congress on Evolutionary Computation
JDE	Self-Adaptive Differential Evolution
EDE	Ensemble Differential Evolution
NSGA-II	Non-Dominated Sorting Genetic Algorithm-II
SF	Superiority of Feasible Solutions
RPO	Real Parameter Optimization
MDE	Modified Differential Evolution
ECHT-EP2	Ensemble of Constraint Handling Techniques
SAMO-DE	Self Adaptive Multi Operator Differential Evolution
VNS	Variable Neighborhood Search
MO	Multi-Objective
MOEA	Multi-Objective Evolutionary Algorithm
ICEO	International Contest on Evolutionary Optimization
3D	Three Dimensional

1 INTRODUCTION

Optimization problem is used to maximize or minimize some function of decision variables subject to hard or soft constraints. If the decision variables contain real values, the problem is called real parameter optimization. Real parameter optimization is one of the earliest applications of evolutionary computation. Real-parameter GAs, evolution strategies (ES), differential evolution (DE), particle swarm optimization (PSO), evolutionary programming (EP), classical methods such as quasi-Newton method (QN), hybrid evolutionary-classical methods, other non-evolutionary methods such as simulated annealing (SA), tabu search (TS) are some of the popular approaches and each of them has their improved versions. In the recent years, many types of optimization algorithms have been exploring to solve real-parameter optimization problems among the evolutionary computation committees or conferences as well as journals. Various algorithms, developed for CEC 2006 Special Session on Constrained Real-Parameter Optimization presented plausible solutions, including ε Constrained Differential Evolution with Gradient-Based Mutation and Feasible Elites (Takamaha T. & Sakai S., 2006), Dynamic Multi-Swarm Particle Swarm Optimizer with a Novel Constraint-Handling Mechanism (Liang J.J., Suganthan P. N., 2006), Self-adaptive Differential Evolution Algorithm (Huang V. L., 2006), A Multi-Populated Differential Evolution Algorithm (Tasgetiren M. F., Suganthan P. N., 2006). In these kind of studies, optimization techniques have been applied in many field of science including standard test problems (Sphere function, Rosenbrock function or Schwefel function) or various engineering cases are considered so far.

On the other hand, the main focus of the study revolves around application of optimization techniques to architectural design problems. Architectural design is a design field that is characterized by complexity. If we refer to the design of buildings, building complexes, urban areas or even interiors, there are several factors that contribute to the complexity of the problem. As few significant ones, we may mention the existence of an excessive number of possible solutions, the presence of conflicting goals that entail hard and soft aspects of the architectural design, as well as the intricate, non-linear relationships between design parameters (decision variables) and design criteria (objective functions). A design cannot be evaluated only based on aesthetics; nor can it aim just for technical or economical perfection. Both hard and soft aspects should be considered. This condition calls for systematic approaches to the identification of promising solutions, and as such

the use of computational optimization methods for architectural design is highly desirable. Due to acceptance that the design variables of architectural problems are real parameters, it was dealt with Real Parameter Optimization; so that evolutionary algorithms may be beneficial to get feasible design solutions according to the design objectives.

1.1 Research Goal

In accordance with the previous statements, the main goal of this thesis is to experience the power of real-parameter optimization methods through constrained single objective and multi-objective test functions. Based on this experience, further goal is to present its applicability in architectural problems by using computational design techniques.

1.2 Methodology

This thesis firstly focuses on problem definitions of CEC'2006 competition for constrained single objective real-parameter numerical optimization. In "CEC'2006 Constrained Single Objective Real-Parameter Optimization Special Session", algorithms that are competing through test functions presented beforehand. Various researchers have tried to beat the best solutions so far. To tackle these problem definitions, the results gathered from Self-Adaptive Differential Evolution Algorithm (JDE) and Ensemble Differential Evolution Algorithm (EDE) were compared with the best performing algorithms in the literature. Then, multi-objective constraint test functions were evaluated through using the same algorithms. After algorithm comparison for all benchmarks, these algorithms were applied to novel architectural cases, namely; restaurant design, floating settlement design and underwater hotel room design. They were formulated by considering user preferences, satisfactions as well as energy efficiency, site or function requirements and others. Finally, the results of the models gathering from optimization process were presented. The methodology of this study can be found in Figure 1.1.

Constrained Real-Parameter Optimization part of this study aims to test the performance of EDE Algorithm through CEC 2006 benchmarks with making comparison with the best performing algorithms in the literature. According to this parts' results, EDE Algorithm is competitive with the best performing algorithms. Multi-Objective Real Parameter Optimization part was handled due to the fact that

it was dealt with more than one objective function within several architectural problems. This part also requires testing EDE Algorithm through multi-objective constrained test functions. NSGA-II and JDE were also tested because they have already shown their strong performance in kind of architectural problems as mentioned in (Chatzikonstantinou, 2011) and (Ekici, 2014). After this part, NSGA-II, JDE and EDE algorithms were applied to architectural problems considered in this thesis. They are restaurant design, floating settlement design and underwater hotel room design. In such problem, one of the algorithms gave better design solutions whereas better design alternatives were gathered from other algorithm in another problem.

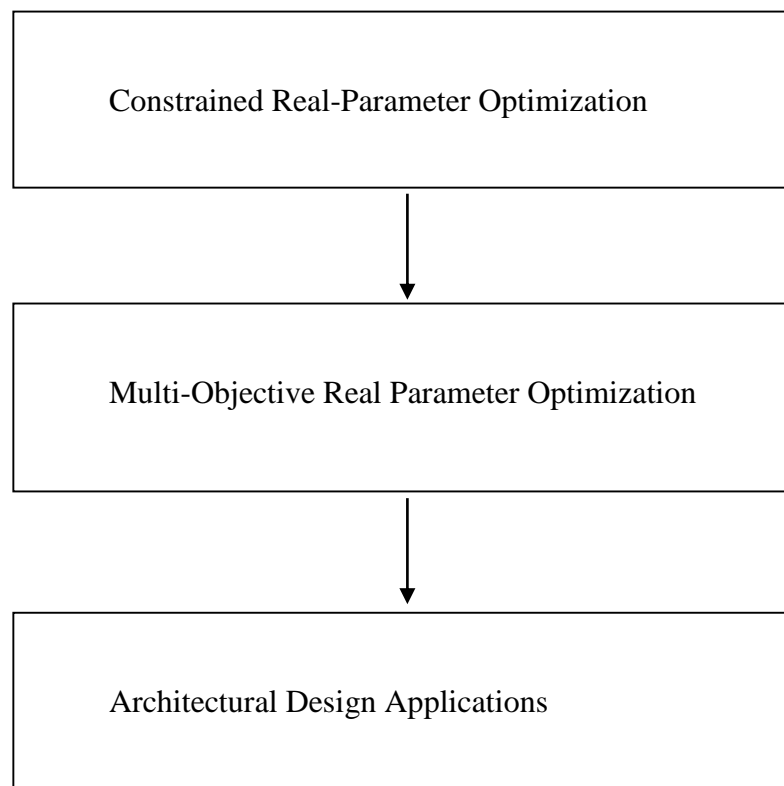


Figure 1.1. Method of the Study

2 PROPOSED ALGORITHMS

2.1. Non-Dominated Sorting Genetic Algorithm-II

The NSGA II (Non-Dominated Sorting Genetic Algorithm - II) is a Multi-Objective Evolutionary Algorithm (MOEA) developed by prof. Deb at the Kanpur Genetic Algorithms Laboratory (KanGAL), which is widely referred to and known for its speed and robustness. NSGA-II is an elitist, multi-objective GA that can also handle constraints. The algorithm is an improved version with the controlled elitism and dynamic crowding distance of the earlier NSGA algorithm, also by K. Deb, A. Pratap, S. Agarwal, T. Meyarivan (2002). It is acknowledged that NSGA-II is able to address difficult real-world multi-objective problems, and for which it may achieve a homogenously distributed set of Pareto-optimal solutions. The key features of the algorithm are as follows:

- Nondominated sorting of solutions, using a fast $O(MN^2)$ sorting algorithm
- Parameterless diversity calculation mechanism based on the cuboid volume between neighboring elements of the same rank, a measured termed “Crowding Distance”
- Diversity-preserving binary tournament selection, based on comparison of constraint violation, ranking and crowding distance
- Diversity preserving elitist approach that combines elite parent and offspring members taking into account constraint violation, rank and uniqueness
- Simulated Binary Crossover genetic operator (K. Deb et. al., 2002)
- Polynomial mutation operator

The sequence of steps the algorithm performs in each generation is as follows:

- The parent population is sorted according to each of its member’s non-domination. Through this process, each solution (population member) is assigned a value that represents its rank within the population. NSGA-II achieves a fast sorting of population members, through extensive book-keeping, namely through associating the number of solutions dominating, and references to solutions that are dominated by any population member.

This procedure places the population members to discrete sets according to their Pareto ranking.

- For each of the population members, the crowding distance is calculated. The crowding distance is calculated for members of each rank separately, and, for each of the members, comprises of the distances to their nearest neighbours, summed overall objective function dimensions.
- In this step the mating pool is formed; individuals are selected from the parent population following a binary tournament, until the mating pool is filled. During the tournament selection, the constraint violation of the solutions is firstly compared. In case both solutions are violating, the one with the least violation is selected. In case one is violating, the other is selected. In case none are violating, non-dominating ranks are compared. The solution with the lower rank among the two is picked. In case they are non-dominating, crowding distance is finally compared, and the one with the highest crowding distance is selected.
- The mating pool individuals are subjected to the genetic operators, crossover through SBX and polynomial mutation, in order to form the next generation.
- The elitism step is performed. In this step, the current and previous populations' members are merged in different pools according to their non-dominance, and each of those is sorted with members having the highest crowding distance coming first. The elitist population is formed by including as many of the first rank individuals as possible for the population size; if there are spaces left, the second rank follows and so on.

2.2. Differential Evolution Algorithm

DE algorithm was firstly presented by Storn & Price (1995) as a new heuristic approach to minimize objective function that has possibly nonlinear and non-differentiable continuous space.

Recent survey study written by Das & Suganthan (2011) clearly explained the history of DE and its success. DE is potentially one of the most powerful stochastic real-parameter optimization algorithms in current use. DE eventuated to be the best evolutionary algorithm for solving the real-valued test function suite of

the 1st ICEO (International Contest on Evolutionary Optimization) and as one of the best among the competing algorithms at 2nd ICEO in 1997. In two journal articles, Price (1997) and Storn and Price (1997) describe the algorithm in ample details followed immediately in quick succession. In 2005, CEC competition on real parameter optimization, on 10-D problems classical DE secured 2nd rank and a self-adaptive DE variant called SaDE (Qin, Suganthan, 2005) secured third rank although they performed poorly over 30-D problems. A recent study by Neri and Tirronen (2010) reviewed the variants of DE for single-objective optimization problems, as well as compared them on a set of benchmark problems. According to review studies, it is pointed out that DE-variants are as powerful as original DE on solving the complex problems.

DE algorithm has variety of advantages. The algorithm has a simple code structure and it provides users for simple implementation of this algorithm to practically solve problems. Other advantage is that the number of control parameters in DE is few (Cr, F, and NP in classical DE). If just a simple rule of F and Cr is changed, the performance of the algorithm is significantly improved without imposing any serious computational burden as presented in Brest et. al. (2006), Qin et. al. (2009), Zhang and Sanderson (2009). Moreover, DE is better to tackle the large scale and expensive optimization problems because of its feature that the space complexity of DE is lower than the other competitive real parameter optimizers as seen in Hansen and Ostermeier (2001).

2.2.1. Steps of DE Algorithm

DE is a simple real parameter optimization algorithm. For real parameter optimization, each decision variable is a real number. In order to describe it, the *DE/rand/1/bin* scheme of Storn and Price (1995) was chosen. It was applied in a variety of applications that can be found in Corne et al. (1999), Lampinen (2001), Babu and Onwubolu (2004), Price et al. (2005) and Chakraborty (2008) and Das and Suganthan (2011). DE works through a simple cycle of stages, presented in Figure 2.1.

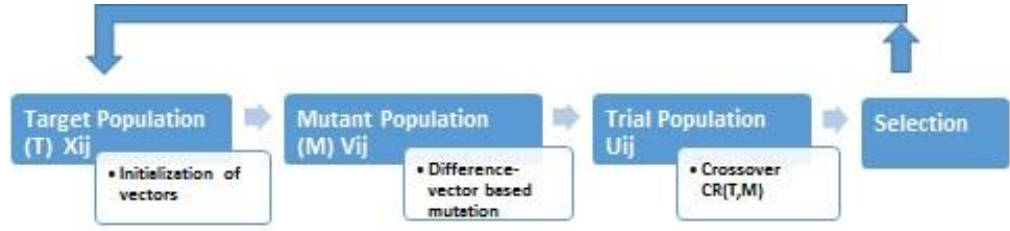


Figure 2.1. Steps of DE algorithm

2.2.1.1. Initialization of Target Population

In the traditional *DE* algorithm, initial target population has NP (Number of Parents) individuals having a D -dimensional real-valued parameter vectors. Each vector, also known as chromosome, keeps an alternative solution to the multidimensional optimization problem.

Each vector is obtained randomly and uniformly within the search space constrained by the prescribed minimum and maximum bounds: $[x_{ij}^{min}, x_{ij}^{max}]$. Thus, the initialization of j^{th} component of i^{th} vector can be defined as:

$$x_{ij}^0 = x_{ij}^{min} + (x_{ij}^{max} - x_{ij}^{min}) \times r \quad (1)$$

where x_{ij}^0 is the i^{th} target individual at generation $g = 0$; and r is a uniform random number in the range $[0,1]$.

2.2.1.2. Difference -Vector Based Mutation

Mutation is a way to get new solutions. Nevertheless, it consists in random changing the value of parameters in the context of GAs and EAs. In DE-literature, it mutates the base vectors (secondary parents) with scaled population-derived difference vectors. The difference vector based mutation is believed to be one of the main strength of DE (Storn & Price, 1997), (Price & Storn, 1997). These differences tend to adapt to the natural scaling of the problem over generations. Hence, DE differs from the other EAs to need only the specification of a single relative scale factor F for all variables.

As definition of Das and Suganthan (2011), a parent vector from the current generation is called target vector, a mutant vector obtained through the differential

mutation operation is known as donor vector and finally an offspring formed by recombining the donor with the target vector is called trial vector.

In order to obtain mutant individuals, the weighted difference of two individuals from target population is added to a third individual randomly selected from population.

$$v_{ij}^g = x_{aj}^{g-1} + F \times (x_{bj}^{g-1} - x_{cj}^{g-1}) \quad (2)$$

where a, b, c are three randomly chosen individuals from the target population such that $(a \neq b \neq c \neq i \in (1, \dots, NP))$ and $j = 1, \dots, D$. $F > 0$ is a mutation scale factor influencing the differential variation between two individuals.

2.2.1.3. Crossover

Genetic Algorithms generally recombine two vectors to create two separate trial vectors with one-point crossover, but DE algorithm is managed crossover to produce one single trial vector. N-point crossover is one of the most popular crossover techniques for real coded GAs. According to this technique, the offspring vector is randomly divided into $(n + 1)$ blocks such that parameters in adjacent partitions are acquired from different parent vectors.

In this step, binomial crossover is implemented to each variables if a randomly generated number between 0 and 1 is less than or equal to the CR value. In this case, the number of parameters acquired from the donor has a binomial distribution. Trial individuals are obtained by recombining mutant individuals with its corresponding target individuals. The scheme may be outlined as

$$u_{ij}^g = \begin{cases} v_{ij}^g & \text{if } r_{ij}^g \leq CR \text{ or } j = D_j \\ x_{ij}^{g-1} & \text{otherwise} \end{cases} \quad (3)$$

where the index D_j is a randomly chosen dimension ($j = 1, \dots, D$). It makes sure that at least one parameter of the trial individual u_{ij}^g will be different from the target individual x_{ij}^{g-1} .

“CR” is called the crossover rate and appears as a control parameter of DE just like F . CR is a user-defined crossover constant in the range $[0,1]$, and r_{ij}^g is a uniform random number in $[0,1]$.

When trial individuals are generated, parameter values might violate search ranges. For this reason, parameter values violating the search range are randomly and uniformly re-generated as follows:

$$x_{ij}^g = x_{ij}^{min} + (x_{ij}^{max} - x_{ij}^{min}) \times r \quad (4)$$

2.2.1.4. Selection

To hold the population size constant as generations pass, the next step of the algorithm is selection. For the next generation, selection is normally based on the survival of the fittest among the trial and target individuals such that:

$$x_i^g = \begin{cases} u_i^g & \text{if } f(u_i^g) \leq f(x_i^{g-1}) \\ x_i^{g-1} & \text{otherwise} \end{cases} \quad (5)$$

According to this equation, if the fitness value of new trial vector yields an equal or lower value, it replaces the corresponding target individual in the next generation; otherwise the target is kept in the population. Thus, the population is never gets worse.

2.2.2. Self- Adaptive Differential Evolution

Selecting the control parameter of DE is serious task due to the possibility of getting different conclusions for just one change. It was already mentioned about that there are several variants of DE. In this study, the DE scheme presented by Storn et al. (1995) and Das et al. (2005) was applied which can be classified using notation as DE/rand/1/bin strategy.

In the Janez's article, the version of a self-adaptive DE is compared with the classical DE algorithm and with the FADE algorithm (Liu and Lampinen, 2005) by testing on benchmark optimization problems taken from literature. They concluded that "DE algorithm using the self-adaptive control parameter settings is better or at least comparable to the standard DE algorithm and evolutionary algorithms from literature considering the quality of the solutions found with". Their proposed algorithm gives better results in comparison with the FADE algorithm.

In the *DE* algorithm above, a novel self-adapting parameter scheme developed by Brest et al. (2006) was employed, so called *jDE*. It is very simple,

effective and converges much faster than the traditional DE, particularly when the dimensionality of the problem is high or the problem concerned is complicated. In *jDE*, each individual has its own F_i and CR_i values. Initially, they are assigned to $CR_i = 0.5$ and $F_i = 0.9$ and they are updated as follows:

$$F_i^g = \begin{cases} F_l + r_1 \cdot F_u & \text{if } r_2 < t_1 \\ F_i^{g-1} & \text{otherwise} \end{cases} \quad (6)$$

$$CR_i^g = \begin{cases} r_3 & \text{if } r_4 < t_2 \\ CR_i^{g-1} & \text{otherwise} \end{cases} \quad (7)$$

where $r_j \in \{1,2,3,4\}$ are uniform random numbers in the range $[0,1]$. t_1 and t_2 denote the probabilities to adjust the F and CR . They are taken as $t_1 = t_2 = 0.1$ and $F_l = 0.1$ and $F_u = 0.9$.

2.2.3. Ensemble Differential Evolution

In this study, an ensemble approach for DE algorithm was applied. The ensemble is achieved in such a way that each individual is assigned to one of the four distinct differential mutation strategies. These mutation strategies employed are applied to each individual to generate the mutant individual. Then we recombine the mutant individual with the target individual by means of binomial crossover operator to generate the trial individual. In this approach, each decision variable has values pool for competition of producing better future offspring according to their success in the past generations.

In the mutation part, kind of mutation strategies are randomly operated. The ensemble idea was presented in Tasgetiren et al. (2010) and Mallipeddi et. al. (2011). In those studies, ensemble of mutation strategies was considered to develop EDE algorithm. Inspiring from those studies, following mutation strategies (M_i) has been considered in this thesis. The outline of EDE is given in Figure 2.2.

M_1 :

$$v_{ij}^g = x_{aj}^{g-1} + F \times (x_{bj}^{g-1} - x_{cj}^{g-1}) \quad (8)$$

M_2 :

$$v_{ij}^g = x_{aj}^{g-1} + F \times (x_{bj}^{g-1} - x_{cj}^{g-1}) + F \times (x_{dj}^{g-1} - x_{ej}^{g-1}) \quad (9)$$

M_3 :

$$v_{ij}^g = x_{ij}^{g-1} + KF \times (x_{aj}^{g-1} - x_{ij}^{g-1}) + KF \times (x_{bj}^{g-1} - x_{cj}^{g-1}) \quad (10)$$

M_4 :

$$v_{ij}^g = x_{ij}^{g-1} + F \times (x_{aj}^{g-1} - x_{ij}^{g-1}) + F \times (x_{bj}^{g-1} - x_{cj}^{g-1}) \quad (11)$$

where a, b, c, d, e are five randomly chosen individuals from the target population such that $(a \neq b \neq c \neq d \neq e \neq i \in (1, \dots, NP))$ and $j = 1, \dots, D$. $F > 0$ is a mutation scale factor influencing the differential variation between two individuals and $KF = r * F$ where r is randomly chosen within the range $[0,1]$.

Procedure EDE()

- Step 1.* Set parameters $g = 0, NP = 100, M_{max} = 4$
- Step 2.* Establish initial population randomly
 $P^g = \{x_1^g, \dots, x_{NP}^g\}$ with $x_i^g = \{x_{i1}^g, \dots, x_{iD}^g\}$
- Step 3.* Assign a mutation strategy to each individual randomly
 $M_i = \text{rand}() \% M_{max}$ for $i = 1, \dots, NP$
- Step 4.* Evaluate population and find x_{best}^g
 $f(P^g) = \{f(x_1^g), \dots, f(x_{NP}^g)\}$
- Step 5.* Assing $CR[i] = 0.5$ and $F[i] = 0.9$ to each individual
- Step 6.* Repeat the following for each individual x_i^g
- obtain $v_i^g = M_i(x_i^g)$
 - obtain $u_i^g = CR_i(x_i^g, v_i^g)$
 - obtain $x_i^g = \begin{cases} u_i^g & \text{if } f(u_i^g) \leq f(x_i^{g-1}) \\ x_i^{g-1} & \text{otherwise} \end{cases}$
 - if $f(u_i^g) > f(x_i^{g-1})$, $M_i = \text{rand}() \% M_{max}$
 - if $(f(x_i^g) \leq f(x_{best}^g))$, $x_{best}^g = x_i^g$
 - Update F_i^g and CR_i^g
- Step 7.* If the stopping criterion is not met, go to Step 6, else stop and return π_{best}

Figure 2.2. Outline of EDE algorithm

2.2.4. Multi-Objective Approach for DE Algorithm

In this study, since the selected architectural case problems involve multiple objectives that are mostly conflict each other, as the name implies, multi-objective algorithms are called for solving kind of complex problems. In this study, the DE algorithm was implemented in multi-objective manner. Hence, review of multi-objective DE studies was needed.

According to Das & Suganthan's explanation, Chang et al. (1999) firstly presented multi-objective DE algorithm with the Pareto-optimal set based multi-objective tuning of fuzzy automatic train operation for mass transit system. DE/rand/1/bin scheme with a Pareto optimal set that stores the non-dominated solutions gathered during the search. Memetic Pareto artificial neural networks are another approach used in Abbass, (2001)' paper. Generalized differential evolution (GDE) was presented for MO optimization problems with an extension of DE/rand/1/bin variant. Then, other versions of GDE has been derived. For example, GDE2 (Kukkonen et. al., 2004) has a crowding distance measurement to choose the best solution from candidates. Later, a combination of GDE and GDE2 is improved by Kukkonen & Lampinen namely GDE3. Distribution of the solutions in final Pareto front is improved by increasing population size and using non-dominated sorting strategy of NSGA-II (Deb et. al., 2002)

Robic and Filipic (2005) proposed DEMO (DE for Multi-Objective Optimization). Pareto-based ranking and crowding distance sorting approaches are added to original DE algorithm.

In Xue et al. (2003), a Pareto-based approach was introduced to implement the selection of the best individual to deal with the MO DE. If a solution is dominated, a set of non-dominated individuals can be identified and the "best" turns out to be any individual (randomly picked) from this set. Also, the authors adopt $(\mu+\lambda)$ selection, Pareto ranking and crowding distance in order to produce and maintain well-distributed solutions.

Iorio and Li (2004) presented the Differential Evolution variant of the NSGA-II. It has demonstrated rotational invariance and superior performance over the NSGA-II on this problem. The real-coded crossover and mutation rates within the NSGA-II have been replaced with a simple Differential Evolution scheme, and results were reported on a rotated problem which has presented difficulties using existing Multi-objective Genetic Algorithms.

The cases of architectural design problems in this thesis defined as multi-objective real-parameter constrained optimization problems. Due to have multi-objective optimization problems, one-to-one selection as in equation (5) was not used. Inspired by previous studies mentioned above, the non-dominated sorting procedure developed by Deb, et al. is applied to DE algorithm. For the sake of clarity about the implementation, it should be pointed out that the target population

was combined with the trial population at each generation. Then, the non-dominated sorting procedure developed for NSGA-II was applied in order to define the target population for next generation.

3 CONSTRAINED REAL PARAMETER OPTIMIZATION

Constrained Real Parameter Optimization Algorithms are the first step of complex optimization problems because multi-objective optimization algorithms were developed by inspiring from the research on single objective optimization algorithms. Another reason of considering Constrained Optimization is that almost all optimization problems have a number of diverse constraints that modify the shape of the search space. According to J.J. Liang et al. (2006), since evolutionary algorithms and other meta-heuristics behave like unconstrained search technique because of their nature during optimization process, additional mechanism is needed. Previously, the most frequently used method is the penalty functions to incorporate constraints. However for solving a problem the optimum lies in the boundary between the feasible and the infeasible regions or when the feasible region is disjoint. In addition to this, penalty functions need a careful fine-tuning to decide the most suitable penalty factors to be used with meta-heuristics. For these reasons, same ranking methods are available in the literature to handle the constraints.

In this thesis, 22 benchmark problems which were presented at CEC'2006 are considered to solve by using JDE and EDE algorithms. The list of benchmark functions are in Appendix 1. (<http://www.ntu.edu.sg/home/epsugan/>)

All of the 22 test functions were defined with minimization problem as following:

$$\text{Minimize } f(x), x = [x_1, x_2, \dots, x_n] \quad (12)$$

subject to that constraints:

$$g_i(x) \leq 0, i = 1, \dots, q \quad (13)$$

$$h_j(x) = 0, j = q + 1, \dots, m \quad (14)$$

For the converting of equality constraints into inequality form, following strategy was used:

$$|h_j(x)| - \varepsilon \leq 0, \text{ for } j = q + 1, \dots, m \quad (15)$$

If equations (5) and (6) are satisfied, a solution x is defined as feasible. In accordance with the value of ε taken as $\varepsilon = 0.0001$ in the special session, so it was taken the same value in this study.

3.1 Constraint Handling Methods

The researchers also proposed a variety of approaches for constraint handling. In this thesis, the methods used for constraint handling are described the following parts:

3.1.1 Superiority of Feasible Solutions

SF (Superiority of Feasible Solutions) is one of the constraint handling methods proposed by Deb (2000). It is developed for constrained optimization. Based on Karabulut and Tasgetiren (2014); if constraint violation precedes the objective function value, both constraint violation and objective function value would be optimized based on lexicographic ordering. In this method, for instance, there are two solutions (a and b) to find out which one is better. In the condition of a is better than b, a can be feasible and solution b is not, or both solutions should be feasible, but a has the smaller objective function value, or none are feasible, but a has a smaller overall constraint violation amount calculated as follows:

$$G_i(x) = \max\{g_i(x), 0\} \quad i = 1, \dots, p$$

$$H_i(x) = \max\{|h_i(x)| - \delta, 0\} \quad i = p + 1, \dots, m$$

$$v(x) = \frac{\sum_{i=1}^p G_i(x) + \sum_{i=p+1}^m H_i(x)}{m} \quad (16)$$

where $v(x)$ is the average violation of m number of constraints. Additionally, δ is the tolerance value for equality constraints and it's generally taken as 0.0001 in the literature.

3.1.2 ε -Constraint (EC)

Another constraint handling method is ε -constraint as proposed by Takahama and Sakai (2006). From the point of their research, an appropriate control for the epsilon parameter is needed when the good feasible solutions for problems with equality constraints is obtained. According to control generation notated by t_c , the

ε level is updated. After generation counter t becomes higher than control generation t_c , the ε level is set to zero to finalize with feasible solutions. The solutions which have violations less than $\varepsilon(t)$ are taken to become feasible solutions into account for selection process of the next generation. The main notion can be explained with equations:

$$\varepsilon(0) = v(x_\theta) \quad (17)$$

$$\varepsilon(t) = \begin{cases} \varepsilon(0) \left(1 - \frac{t}{t_c}\right)^{cp}, & 0 < t < t_c. \\ 0, & t \geq t_c \end{cases} \quad (18)$$

where x_θ is the top θ -th individual.

Starting from ‘‘CEC’06 Special Session on Real Parameter Optimization’’, various types of algorithms were developed to solve benchmark test functions. Over the years, standard test functions were improved and researchers called for better performing algorithms. This part of this study was inspired from CEC competition proposed by Liang, Runarsson, Mezura-Montes, Clerc, Suganthan, Coello Coello, Deb (2006) as well as by Tasgetiren and Suganthan (2006).

3.2 Computational Results of Constrained RPO

The EDE Algorithm was coded in C++ and run on an Intel P4 1.33 GHz Laptop PC with 256 MB memory. The population size is taken as $NP=100$. Injection probability is taken as 0.005 whereas the diversification probability is taken as 0.005. For the EC constraint handling method, following parameters are used as $\theta = 0.25 \times NP$, $t_c = 0.4 \times MaxGen$ and $cp = 2$. It was carried out 25 replications for each benchmark problem and average, minimum and standard deviation of 25 replications are provided. To be noted that real numbers are rounded to zero after 10 digits in the standard deviation calculations. DE Algorithm with ensemble strategies (EDE) was compared to the best performing algorithms from the literature such as MDE (Mezura-Montes et al., 2006), ECHT-EP2 (Mallipedi et al., 2010) and SAMO-DE (Saber et al., 2011) and DE-VNS (Tasgetiren et al., 2015). The presentation of overall analysis and comparison based on the results are given in Table 1. Since EDE algorithm is able to find 10 of 19 optimal solutions with zero standard deviation, it can be said that this algorithm performs as well as SAMO-DE

(12 optimal solutions with 0 std), DE-VNS (13 optimal solutions with 0 std), ECHT-EP2 (14 optimal solutions with 0 std). According to the results of Tasgetiren et al.'s study (2015), MDE was the clear winner due to the fact that there are 19 optimal solutions while standard deviations are all zero. MDE algorithm was run for 240,000 functions evaluations, that's why the same number of functions evaluations was ran for EDE algorithm. In 2 benchmarks, the standard deviation of the EDE algorithm was smaller than DE-VNS and SAMO-DE, respectively.

Problem		EDE	DE-VNS	SAMO-DE	MDE	ECHT-EP2
	FEs	240,000	240,000	240,000	500,000	240,000
g01	Best	-15.0000	-15.0000	-15.0000	-15.0000	-15.0000
	Avg	-15.0000	-15.0000	-15.0000	-15.0000	-15.0000
	Std	0.00E-00	0.00E-00	0.00E-00	0.00E-00	0.00E-00
g02	Best	-0.8036191	-0.8036191	-0.8036191	-0.8036191	-0.8036191
	Avg	-0.7861684	-0.789822	-0.79873521	-0.78616	-0.7998220
	Std	1.65E-02	1.87E-02	8.80050E-03	1.26E-02	6.29E-03
g03	Best	-1.0005	-1.0005	-1.0005	-1.0005	-1.0005
	Avg	-1.0005	-1.0005	-1.0005	-1.0005	-1.0005
	Std	0.00E-00	0.00E-00	0.00E-00	0.00E-00	0.00E-00
g04	Best	-30665.54	-30665.5386	-30665.5386	-30665.539	-30665.539
	Avg	-30665.54	-30665.5386	-30665.5386	-30665.539	-30665.539
	Std	0.00E-00	0.00E-00	0.00E-00	0.00E-00	0.00E-00
g05	Best	5126.497	5126.497	5126.497	5126.497	5126.497
	Avg	5153.561	5126.497	5126.497	5126.497	5126.497
	Std	8.971867E+001	0.00E-00	0.00E-00	0.00E-00	0.00E-00
g06	Best	-6961.814	-6961.813875	-6961.813875	-6961.814	-6961.814
	Avg	-6961.814	-6961.813875	-6961.813875	-6961.814	-6961.814
	Std	0.00E-00	0.00E-00	0.00E-00	0.00E-00	0.00E-00
g07	Best	24.3062	24.3062	24.3062	24.3062	24.3062
	Avg	24.55890	24.306209	24.3096	24.3062	24.3063
	Std	3.410007E-001	2.17E-07	1.58880E-03	0.00E-00	3.19E-05

g08	Best	0.095825	-0.095825	-0.095825	-0.095825	-0.095825
	Avg	0.095825	-0.095825	-0.095825	-0.095825	-0.095825
	Std	0.00E-00	0.00E-00	0.00E-00	0.00E-00	0.0E-00
g09	Best	680.630	680.630	680.630	680.630	680.630
	Avg	680.630	680.630	680.630	680.630	680.630
	Std	0.00E-00	0.00E-00	1.15670E-05	0.00E-00	0.00E-00
g10	Best	7049.252	7049.24802	7049.24810	7049.24802	7049.2483
	Avg	7058.172	7049.24803	7059.81345	7049.24802	7049.2490
	Std	1.798828E+001	4.02E-05	7.856E-00	0.00E-00	6.60E -04
g11	Best	0.7499	0.7499	0.7499	0.7499	0.7499
	Avg	0.7499	0.7499	0.7499	0.7499	0.7499
	Std	0.00E-00	0.00E-00	0.00E-00	0.00E-00	0.00E-00
g12	Best	-1.0000	-1.0000	-1.0000	-1.0000	-1.0000
	Avg	-1.0000	-1.0000	-1.0000	-1.0000	-1.0000
	Std	0.00E-00	0.00E-00	0.00E-00	0.00E-00	0.00E-00
g13	Best	0.05394151	0.053942	0.053942	0.053942	0.053942
	Avg	0.05671902	0.053942	0.053942	0.053942	0.053942
	Std	0.01388753	0.00E-00	0.00E-00	0.00E-00	0.00E-00
g14	Best	-47.76487	-47.76489	-47.76489	-47.764887	-47.7649
	Avg	-47.67010	-47.76489	-47.68115	-47.764874	-47.7648
	Std	0.1545972	4.64E-06	4.04300E-02	1.400E-05	2.72E-05
g15	Best	961.7150	961.71502	961.71502	961.71502	961.71502
	Avg	961.7150	961.71502	961.71502	961.71502	961.71502
	Std	0.00E-00	0.00E-00	0.00E-00	0.00E-00	0.00E-00
g16	Best	-1.905155	-1.905155	-1.905155	-1.905155	-1.905155
	Avg	-1.905155	-1.905155	-1.905155	-1.905155	-1.905155
	Std	0.00E-00	0.00E-00	0.00E-00	0.00E-00	0.00E-00
g17	Best	8853.542	8853.5397	8853.5397	8853.5397	8853.5397
	Avg	8952.354	8877.3107	8853.5397	8853.5397	8853.5397

	Std	84.08253	3.94E+01	1.15E-05	0.00E-00	2.13E -08
g18	Best	-0.866025	-0.866025	-0.866025	-0.866025	-0.866025
	Avg	-0.8583828	-0.834185	-0.866024	-0.866025	-0.866025
	Std	3.821291E-002	7.12E-02	7.04367E-07	0.00E-00	0.00E-00
g19	Best	32.66681	32.656077	32.655593	32.655693	32.6591
	Avg	33.00152	32.685099	32.757340	33.34125	32.6623
	Std	2.874459E-001	3.73E-02	6.145E-02	8.475E-01	3.4E -03
g21	Best	193.7245	193.72451	193.72451	193.72451	193.7246
	Avg	266.8103	193.72456	193.771375	193.72451	193.7438
	Std	5.024687E+001	2.84E-04	1.9643E-02	0.00E-00	1.65E-02
g23	Best	-397.2818	-400.0527	-396.165732	-400.0551	-398.9731
	Avg	-173.4233	-372.9920	-360.817656	-400.0551	-373.2178
	Std	1.246541E+002	5.75E+01	1.9623E+01	0.00E-00	3.37E+01
g24	Best	-5.508013	-5.508013	-5.508013	-5.508013	-5.508013
	Avg	-5.508013	-5.508013	-5.508013	-5.508013	-5.508013
	Std	0.00E-00	0.00E-00	0.00E-00	0.00E-00	0.00E-00

Table 1 Computational Results of EDE, DE-VNS, SAMO-DE, MDE, ECHT-EP2 FOR CEC 2006 Test Problems

4 MULTI-OBJECTIVE CONSTRAINED REAL PARAMETER OPTIMIZATION

Until this part, constrained single objective optimization techniques were considered since they are the basis of multi-objective optimization. In real life, more objectives are required and it is constantly faced with constraints beyond standard test functions. Optimization for multiple conflicting goals is a difficult task. The choice of a placement of windows in a room can be done by taking daylight into account but the design may not be a good choice in order to minimize heat losses. Therefore, evolutionary algorithms were made of use for multi-objective optimization, as well.

4.1 Multi-Objective Test Functions

This part deals with making a comparison between constrained multi-objective algorithms in order to check their behaviors and results in multi-objective problems. However, in this kind of architectural studies, ZDT-1 test problem developed by Zitzler Et Al. (2000) is generally applied to Grasshopper Parametric Modelling environment before passing to design problems, such that Chatzikonstantinou (2011) has been used ZDT-1 problem to test NSGA-II version of Lotus component and Ekici (2014) has been applied to test both NSGA-II and DE versions of Lotus component. In this study, some of the constrained multi-objective test problems were applied to Grasshopper environment to tackle constrained multi-objective problems. CONSTR (Deb et. al. 2001), SRN (Jiménez F. et. al, 2002) and OSY (Osyezka & Kundu, 1995) were selected among the functions, where their formulizations are given in Appendix 2. NSGA-II, JDE and EDE algorithms are implemented to Grasshopper environment for testing the algorithms in it.

4.2 Optimization of MO Test Functions

Optimization of the test problems were done by a component, namely Lotus, originally developed by Ioannis Chatzikonstantinou and Michael S. Bittermann in TU Delft to use in Grasshopper Parametric Modelling Environment. It was firstly based on NSGA-II (Deb et al, 2002) and Lotus-NSGA-II gave plausible design solutions into solve complex architectural problems (Chatzikonstantinou, 2011). Later on, Lotus component has been updated by Ioannis Chatzikonstantinou, Berk Ekici, Cemre Uğurlu, and under the supervision of Fatih Taşgetiren and Sevil

Sarıyıldız, in Yaşar University. JDE (Brest et al., 2006) and EDE (Mallipedi et al., 2010) algorithms were implemented to this component. It is already experienced that simple DE gave satisfactory results into a complex high-rise building design (Ekici, 2014).

For the optimization of the MO test functions, Lotus was made use of for testing problems with its NSGA-II, JDE and EDE versions. Algorithms were terminated after 500 generations with 100 population size for all test functions.

4.3 Performance Measurement

Due to the calculation of algorithm performances, hypervolume indicator that was proposed by Zitzler and Thiele (1999) was applied. Minella Et Al (2011) explains the procedure of hypervolume indicator (I_h) which measures the objective space domination by a given Pareto set of points as follows: After the objective solutions gathered from both algorithms were taken, I_H values were calculated for each algorithm. Given a set S of solutions and being $s \in S$ one solution, Obj is the number of objectives, h is the number of solutions with each solution (i) in set S , $\min S_o$ is the best value and $\max S_o$ is the worst value for each objective (o) in set S . I_H is calculated as follows:

$$I_H(s) = \sum_{o=1}^{Obj} \sum_{i=1}^h \frac{(s_{i,o} - \min S_o)}{(\max S_o - \min S_o)} \quad (19)$$

Based on this equation, hypervolume indicator of NSGA-II for CONSTR test function was calculated as 0.3675 while the result of JDE is 0.4789 and the result of EDE was 0.4802. Hypervolume indicator of NSGA-II for SRN test function was calculated as 0.4654 while the result of JDE was 0.5001 and the result of EDE was 0.5000. Lastly, hypervolume indicator of NSGA-II for OSY test function was calculated as 0.4992 while the result of JDE was 0.6595 and the result of EDE was 0.5883. The bigger solution means better according to this performance measurement. On the other hand, it is obvious to claim that each algorithm may give better results for different problems. Accordingly, EDE algorithm was the winner for CONSTR function while DE results of SRN and OSY functions are bigger than the results gathered by EDE algorithm.

As a conclusion of this test bed, it can be said that, all algorithms are capable to deal with multi-objective problems.

Functions/Algorithms	NSGA-II	JDE	EDE
CONSTR	0.3675	0.4789	0.4802
SRN	0.4654	0.5001	0,5000
OSY	0.4992	0.6595	0.5883

Table 2 Hypervolume results of MO test functions

5 APPLICATIONS ON ARCHITECTURAL OPTIMIZATION PROBLEMS

5.1 Application I: Architectural Design of Restaurant Layouts

This thesis attempts first and foremost to investigate how computational optimization may be applied to design. In particular, this part attempts to demonstrate a possible application of evolutionary computation in the design of a detached or semi-detached restaurant space. The main problem entails the configuration of restaurant functions, the decisions regarding the restaurant shell composition (fraction and position of windows, dimensions), and how to shape and place the kitchen and service areas. A two-objective problem was then formulated by considering the minimization of investment and the maximization of profit, which are clearly two conflicting objectives. It is important to point out that in the calculation of the objective functions, not only technical aspects were taken into account, but also ones pertaining to typical customer preferences, which give an idea of how “attractive” a restaurant arrangement would be. For example, as such, the objectives that formulated, while centered on economics and efficient function of the restaurant, build upon concepts relating to customer satisfaction and quality of service. The results of the application of the NSGA-II, JDE and EDE algorithms were formulated to identify suitable solutions to the above mentioned design problem.

There are several different approaches to multi-objective optimization. The simplest method is to take the weighted combination of different objectives into one single function, thus converting the problem into a single-objective one. This approach has some drawbacks. Since each objective must be associated with a weight, determining these weights is a difficult task. Instead of this approach, the concept of Pareto-optimality is employed to generate non-dominated solutions called Pareto frontier. In this approach, solutions are evaluated according to an objective function vector $\vec{f} = (f_1, \dots, f_q)$ with q objectives. It is said that \vec{u} dominates \vec{v} if and only if (1) $f_i(u) \leq f_i(v)$ for $i = 1, \dots, q$ and (2) $f_i(u) < f_i(v)$ for at least one index $i = 1, \dots, q$. By using Pareto frontier, the decision support system provides a set of solutions to the decision maker, who is free to choose a desired solution from this Pareto frontier.

Applications of evolutionary computation in the field of architectural design are relatively scarce. In what follow this section provides a brief review. There have

been some attempts to apply computational intelligence to architectural design. For example, genetic algorithms were applied as a generative and search procedure to look for optimized design solutions in terms of thermal and lighting performance in a building (Caldas and Norford, 2002). The objectives of another study with a genetic algorithm are illuminance and glare that are conflicted each other while making facade design in (Gagne, Andersen, 2010). Other than the above, there have been studies on the spatial layout optimization. For example, two separate optimization models have been developed to model different part of the building layout design problem in (Baldock, Shea, 2006) where Genetic Algorithms and Simulated Annealing optimization algorithms are used to solve the problems. An evolutionary algorithm for architectural layout design was proposed in (Wong, Chan, 2009). Another study has been done to examine the combination of meta-heuristics and “multidisciplinary design optimization “(MDO) that can increase the efficiency of the design exploration, by taking into account the interactions between the different disciplines in (Strobbe et al., 2011). Other study on the computational architecture is tackled using a novel computational method that is able to deal with the softness of design requirements in (Bitterman, 2009). In another thesis (Chatzikonstantinou, 2011), the author generates parametric pattern for airport terminal design by using evolutionary computation.

In this application, design variables, constraints and objective functions were predefined and then it was referred to evolutionary computation to reach optimal design solutions by using optimization component in Grasshopper Environment.

5.1.1. Problem Definition of Application I

In this paper, it was considered to design a restaurant in order to maximize the total profit and to minimize the investment in multi-objective manner subject to some architectural constraints. To formulate the problem, required notations are in index of symbols. In this case, the decision variables are as follows:

$$x_k, y_k, x_s, y_s, WR, LR, WK, LK, RK, NT, NTA, NTN, p_j, w_j.$$

5.1.2. Objective Functions of Application I

The objectives aim to maximize the total profit and to minimize the total investment. Total investment is the cost for construction but total profit comes from

the restaurant in each month. It was considered that these two objectives are apart, but they are conflicting. The objectives are as follows:

$$\min\left(\frac{1}{TP}, INV\right) \quad (20)$$

where

$$TP = RV - RC \quad (21)$$

$$INV = KC + DC + BC \quad (22)$$

Subject to:

$$(WK * LK) - (NT * Area \text{ per table}) \geq 0 \quad (23)$$

$$(WR * LR) \cap (WK * LK) = (WK * LK) \quad (24)$$

$$10 < WR < 20 \quad (25)$$

$$10 < LR < 20 \quad (26)$$

$$0 < y_k < 38 \quad (27)$$

$$0 < x_k < 29 \quad (28)$$

$$3 < WK < 13 \quad (29)$$

$$3 < LK < 15 \quad (30)$$

$$0 < RK < 360 \quad (31)$$

$$0 < p_j < 1 \quad (32)$$

$$0 < y_j < 1 \quad (33)$$

Constraint (23) ensures that the kitchen size is related with the number of tables. Constraint (24) ensures that the kitchen size cannot exceed the restaurant boundaries. The remaining constraints are boundary constraints. Assumptions are given in Appendix 3. Details of calculations of the total profit and total investment are given as follows:

5.1.2.1. Revenue Calculation

In the model, revenue is the function of average number of customers (A), customer staying time, the price for customers and working hours monthly as follow:

$$RV = ((A/CST) * PR * WHM) \quad (34)$$

The tables which are away from the window have low occupancy rate, while the tables which are close to window have high occupancy rate because of the customer preferences. Average number of customers is the function of number of tables, customer per table, occupancy rate as follow:

$$A = (NTA * CPT * ORA) + (NTN * CPT * ORN) \quad (35)$$

5.1.2.2. Running Cost Calculation

The restaurant has some stuff cost and energy cost every month. For this reason, the running cost of the restaurant is the sum of stuff cost and energy cost as given below:

$$RC = SC + EC \quad (36)$$

For calculating stuff cost, first, the required number of stuff was calculated by using total distance between service point and each table i , number of orders per hour, walking distance per hour.

$$RNS = \frac{(\sum_1^{NT} d_i) * (noph)}{(wdp)} \quad (37)$$

It was tried to minimize total distance between service point and each table i which is the coefficient of the stuff cost function. If this distance will be minimized, quality of service will be increased due to the increasing speed of service. The stuff cost is the function of total distance between service point and each table i , number of orders per hour, walking distance per hour and unit stuff cost.

$$SC = \frac{(\sum_1^{NT} d_i) * (noph)}{(wdp)} * usc \quad (38)$$

$$\sum_{i=1}^{NT} d_i = |x_s - x_{t_i}| + |y_s - y_{t_i}| \quad i = 1, \dots, NT \quad (39)$$

Energy cost is the sum of lighting cost and heating cost as follows:

$$EC = LC + HC \quad (40)$$

Lighting cost is the function of number of tables away from windows, energy consumption for unit light bulb, working hours monthly and the cost of per kilowatt hour as follows:

$$LC = [NTA * (ecl) * WHM] * [cpk] \quad (41)$$

Customer satisfaction strategy of this paper is based on lighting cost. Since the customers generally tend to prefer the tables that are closer to windows, the number of tables that are away from windows will be decreased, as it is proportional with lighting cost.

Heating cost is a function of required heat input, given that heat losses occur through the windows, walls and floor (transmission, we suppose the floor above the restaurant is inhabited), as well as because of air exchange through ventilation. The associated monthly cost is given by:

$$HC = HL * WHM * \frac{3600}{1000} * cpk \quad (42)$$

Where HL is the total heat loss rate, given by:

$$HL = HT + HV \quad (43)$$

The transmission-induced heat loss rate is given by:

$$HT = LRW + LRG \quad (44)$$

$$LRW = (H * L_g * U_g + H * L_w * U_w) * \Delta T_{out} \quad (45)$$

$$LRG = WR * LR * U_F * \Delta T_F \quad (46)$$

$$L_g = w_1 + w_2 + w_3 + w_4 \quad (47)$$

$$w_j = (\sum_{j=1}^2 y_j * WR) + (\sum_{j=3}^4 y_j * LR) \quad (48)$$

$$L_w = 2(WR + LR) - L_g \quad (49)$$

The convection-induced heat loss is given by:

$$HV = V * d_a * c_a * \Delta T_{out} * n \quad (50)$$

5.1.2.3. Investment Calculation

$$INV = KC + DC + BC \quad (51)$$

$$KC = (WK * LK) + (\textit{Kitchen cost per } m^2) \quad (52)$$

$$DC = [(WR * LR) - (WK * LK)] + (\textit{Diner cost per } m^2) \quad (53)$$

$$BC = GC + TC + WC \quad (54)$$

$$GC = (L_g * H * \textit{Glass cost per } m^2) \quad (55)$$

$$TC = (NT * \textit{Unit cost of tables}) \quad (56)$$

$$WC = (L_w * H * \textit{Wall cost per } m^2) \quad (57)$$

5.1.3. Parametric Model of Application I

The parametric model for generating and evaluating the layouts has been implemented in the Grasshopper platform. Grasshopper is a part of Rhinoceros, a CAD program, and defining geometric entities and performing calculations on them with great ease, through visual programming (grasshopper3d.com). A screenshot of the interface of Grasshopper can be seen in Figure 5.1.

For the restaurant model, a simplified geometric representation was generated as a first stage. A rectangular floor plan, whose dimensions form decision variables, was considered. Within this area, the tables were placed in a rectangular grid with predefined density, as explained in the problem formulation. As the second step, the kitchen volume was placed, with location and dimensions also forming decision variables. Tables that intersect with the kitchen volume were removed from the layout. The starting point for the service on the kitchen perimeter, which is called service point, was determined through a single decision variable in the model. Finally, windows are placed on each of the four walls of the building with their

dimensions and position forming some other decision variables as well. The correspondence of decision variable to design dimensions is given in Figure 5.2. Assumptions for definition of the problem are given in Appendix-3.

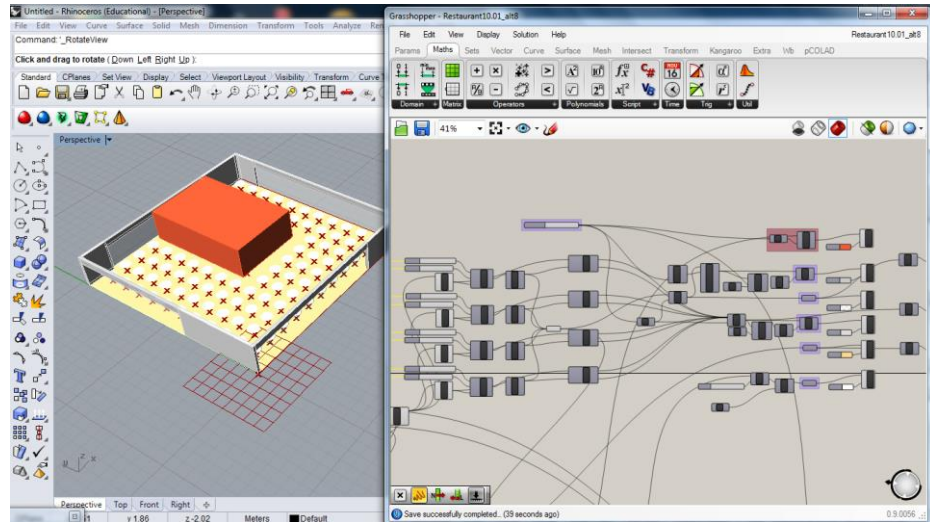


Figure 5.1. Grasshopper model of restaurant design

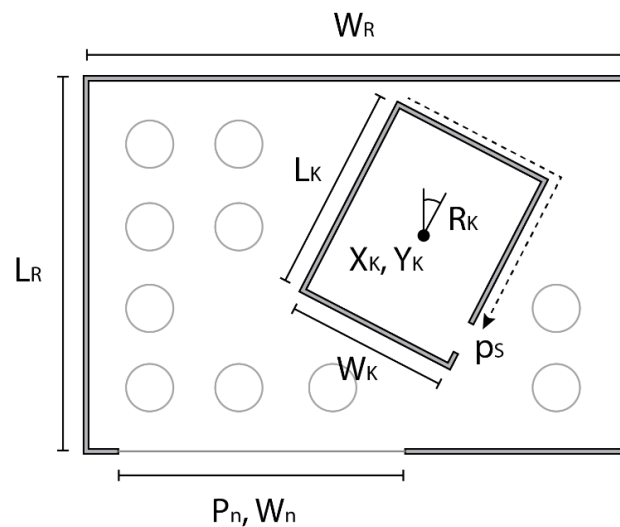


Figure 5.2. Correspondance of decision variables to design dimensions

As the second stage, numerical figures were derived from the geometry in order to formulate the objective functions and constraints. Several properties of the geometric model were considered. These are summarized as follows:

- Total areas for building components (floor, ceiling, walls, and windows), for calculation of building costs, as well as heat loss through the building envelope

- Total areas for kitchen, for calculation of equipment cost as well as minimum required area constraint
- Number of tables, for calculation of profit (along with occupancy rate), as well as furnishing cost and artificial lighting needs
- Distances between kitchen service point and each individual table, for calculation of the mean servicing distances covered by employees
- Distances between windows and each individual table, for calculation of each table's individual occupancy rate, as well as artificial lighting needs

Then, NSGA-II, JDE and EDE were individually implemented as a component for the Grasshopper environment. After setting up the model, it was connected appropriately to the decision variables, objective functions and constraints. The parametric model was concluded by including components that generate a 3D visualization of the arrangement. The complete model as is shown in the Grasshopper environment is available in Figure 5.3.

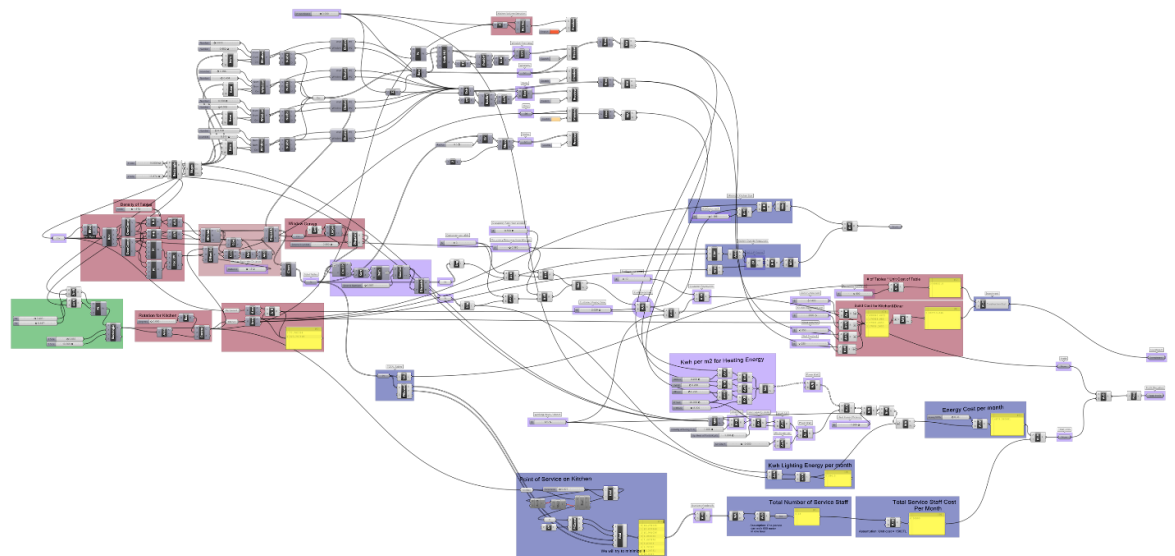


Figure 5.3. Complete model in Grasshopper

5.1.4. Computational Results of Application I

The NSGA-II, JDE and EDE algorithms with the parametric model were tested on an Intel Core-i5 computer, with 2 GB of RAM. The population converged after around 70 generations. The population size is taken as 250 and both algorithms are terminated after 100 generations. The Pareto front approximations are given in

Figure 5.4., Figure 5.5., and Figure 5.6., respectively for the NSGA-II, JDE and EDE algorithms.

Four designs that correspond to the ones with the highest crowding distance are demonstrated for both algorithms with the labels (C1, C2, C3, C4). Furthermore, additional four designs had been identified through visual inspection and in order to demonstrate the breadth of solutions separately for each algorithms with the labels (S1, S2, S3, S4). Same labelling format was selected but there is no relation between C1 in (NSGA-II) Figure 5.4. and C1 in (JDE) Figure 5.5., for instance.

According to the results of NSGA-II, the one with the least calculated investment, 204,700 TL, offers a calculated profit of 12,500 TL. On the other extreme, the design with an investment of 608,800 TL offers a profit of 32,780 TL. Trends with respect to some spatial characteristics can be observed in the resulting solutions.

For both algorithms, smaller designs tend to have the kitchen area in one of the four corners. This can be justified by the fact that such a placement presents an advantage for the serving position. At the same time the daylight obstruction due to the kitchen placement minimally affects the daylight levels in the dining area because of its small size. The windows in small designs are of moderate size and usually run along one or two walls, with smaller ones present on the rest of the walls.

On the other hand, designs with large dimensions feature a more central kitchen placement. Specifically, the kitchen is placed so that the service point is at the centre, but at the same time so that the least daylight obstruction was obtained. As such, corner placement is excluded. Furthermore, in some of the solutions; an interesting diagonal placement of the kitchen was obtained. This allows more space closer to the windows for the tables, and as such results in better daylighting conditions for the dining hall. Further investigation of the applicability of such designs is part of the ongoing research.

According to the hypervolume calculation, JDE algorithm seems to be slightly better in results. However, it should be justified statistically. To justify the performance of the algorithms, a paired-t test has been carried out. P-value between NSGA-II and JDE results is 0.913. Since the p-value is higher than the $\alpha=0.05$ level, these two algorithms are equivalent. However, p-values between NSGA-II & EDE

and JDE & EDE were all zero. It can be said that JDE and NSGA-II algorithms are better than EDE algorithm for this problem.

Results for the investment and total profit criteria for the algorithms that are sorted according to I_H are shown below:

ALGORITHMS	I_H
JDE	0.4556
NSGA-II	0.4561
EDE	0.4225

Table 3 Hypervolume Results of Application 1

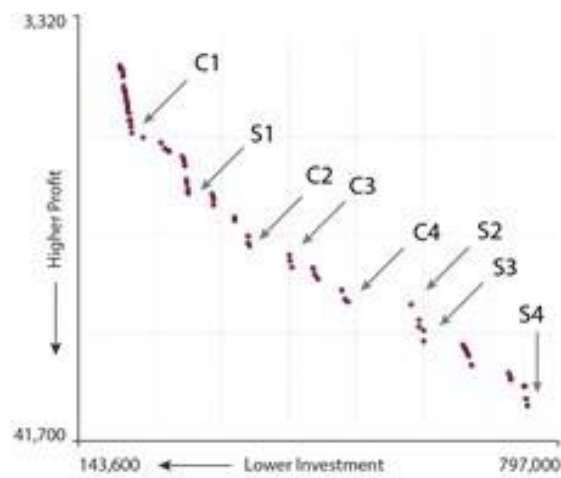


Figure 5.4. Pareto Front Approximation for NSGA-II

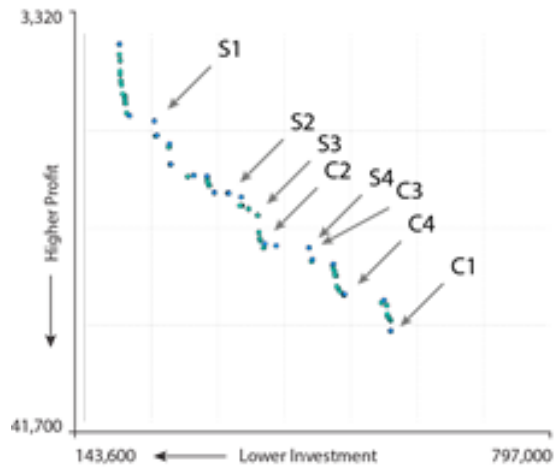


Figure 5.5. Pareto Front Approximation for DE Algorithm

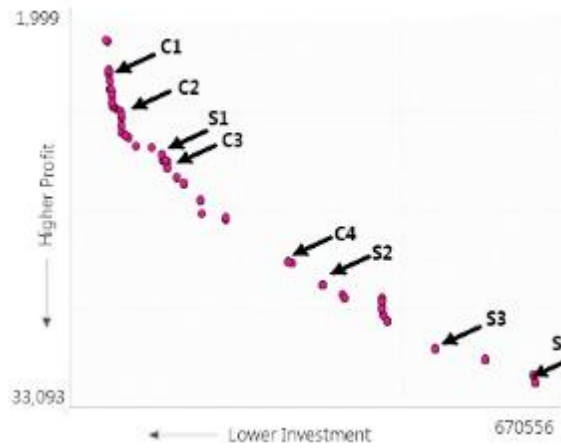


Figure 5.6. Pareto Front Approximation for EDE Algorithm

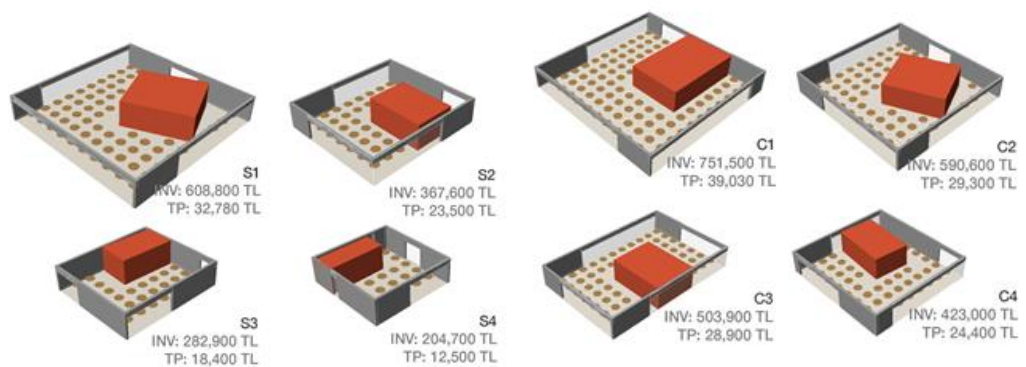


Figure 5.7. Pareto front solutions (highest crowding distances & visual inspection) obtained with the NSGA-II

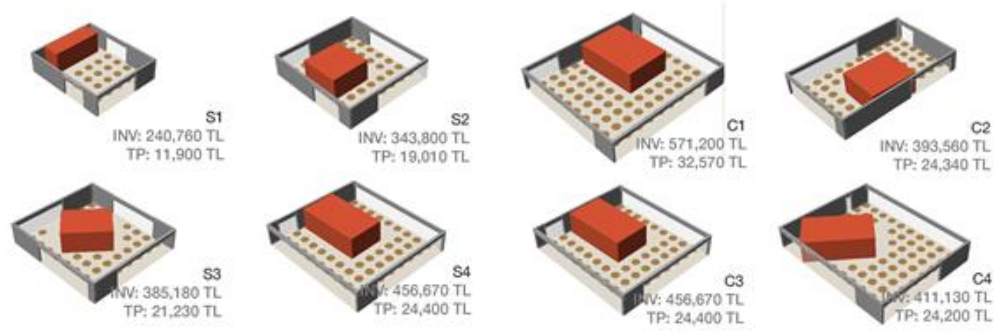


Figure 5.8. Pareto front solutions (highest crowding distance & visual inspection) obtained with the DE algorithm

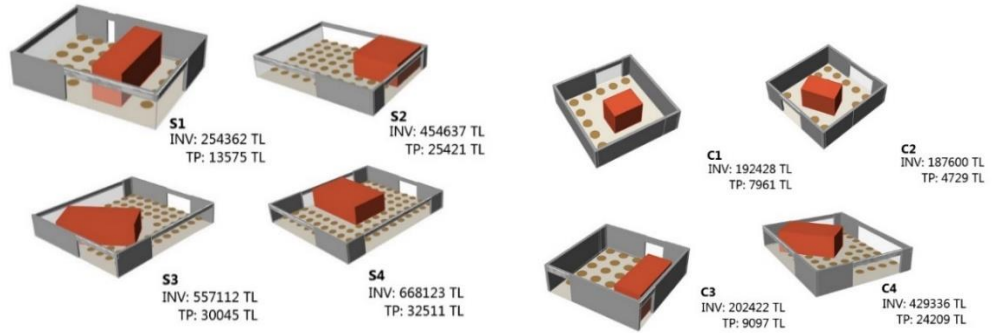


Figure 5.9. Pareto front solutions (highest crowding distance & visual inspection) obtained with the EDE algorithm

5.2. Application II: Sustainable Designs for Floating Settlements

During the last decade; due to the increasing number of disasters and rising sea level, floating settlements started to become an architectural trend. Floating settlements are relevant and innovative solutions for dealing with new challenges in development of cities and settlements. However, their design poses a lot of considerations and technical challenges. Computational tools, methods and techniques may be beneficial for tackling these complex issues in floating settlement design.

This part of the study focuses on the conceptual design and the development of a floating neighbourhood by taking advantage of computational methods. An application to a concept design of a floating neighbourhood in the region of Urla – a coastal town close to Izmir in Turkey, has been studied.

In the first phase of the study, the concept (yacht tourism development) was determined according to site analysis. Simultaneously, with respect to concept, the floating neighbourhood functions (accommodation, marine, yacht club, public area) have been identified. As an extension of these, with referring to computational methods due to the need of a distribution of the functions that includes desired distance relationships and sea water issues and a suitable form of the neighbourhood was obtained.

With respect to configuration of the functions, optimization concerns like maximization of accessibility, maximization of wind protection for keeping living spaces (houses, marine) from wind, as well as maximization of visibility for making commercial places (houses, marine, yacht club) noticeable. According to these objectives, it was tried to be find the most efficient location of the functions subject to both architectural and engineering constraints. Since wind protection and visibility objectives, as well as accessibility and visibility are conflicting with each other, the use of multi-objective evolutionary algorithms was made.

Applications on evolutionary computation in floating neighbourhood design have not been demonstrated in any previous study. Tartar (2012) discussed floating architectural design using fuzzy logic and a rule-based decision support system.

De Graaf (2009) focused on urban water management innovations to reduce vulnerability of urban areas and social aspects that are relevant to mainstreaming and application of innovations.

In Watanabe et. al (2004), design considerations for very large floating structures have been discussed. Authors mentioned good examples of floating structures such as,

- Floating Bridges in USA and Norway,
- Floating Island at Onomichi, Hiroshima, Japan,
- Floating Restaurant in Yokohoma, Japan,
- Shirashima Floating Oil Storage, Base, Japan,
- Kamigoto Floating Oil Storage Base, Nagasaki Prefecture, Japan,
- Concept Design of a Clean Energy Plant by Floating Structure Association of Japan,
- Floating Pier at Ujina, Japan,
- Proposed Floating Runway at Tokyo International Airport (Haneda).

According to the literature review, there is no floating architectural work that also makes use of evolutionary algorithms. Evolutionary algorithms allow to find a

lot of acceptably good design solutions and copes with the complexity of the problems since this study deals with the architectural problem which has a lot of parameters. (Blum et. al, 2014), (Chiong et. al, 2010)

5.2.1. Problem Definition of Application II

The scenario that has been addressed in this part concerns the development of an efficient floating settlement between four islands that are local to the studied region. The islands are Akca, Yassica, Incirli, Pirnarli and their locations are illustrated in Figure 5.10.

This application revolves around two issues. The one is about configuration of the functions (accommodation, marine, yacht club, public area) for a floating neighbourhood design in order to maximize accessibility, wind protection and visibility subject to both technical and nontechnical constraints. The second one is about how to connect functions where their places are certain, in other words, form finding.

There was a need of some notations to explain mathematical process of the problem in the index of symbols.



Figure 5.10. A top view from the islands of Urla

5.2.2. Optimization Model of Application II

It was tried to maximize the satisfaction of design goals that are accessibility, wind protection and visibility by controlling decision variables of the problem subject to constraints that are appropriate water depth, prohibit intersection,

maximum movement area. Detailed information about the objectives and constraints are explained below.

The decision variables are the coordinates of the functions $hx, hy, mx, my, yx, yy, px, py$ each corresponding to one of the following functions: accommodation, marine, yacht club, public area. Their values are real numbers, so the optimization model belongs to the real parameter optimization domain.

5.2.2.1. Objective Function I: Accessibility

In the model, the overall accessibility is determined by two factors: i. accessibility of the public space from all other modules (1), (2), (3) and ii. proximity between the yacht club and the marine (4), in order to satisfy yacht owners that may frequently make use of the two functions. Accessibility calculation strategy is given as follows:

$$db_{h,p} = \sqrt{|hx - px|^2 + |hy - py|^2} \quad (58)$$

$$db_{y,p} = \sqrt{|yx - px|^2 + |yy - py|^2} \quad (59)$$

$$db_{m,p} = \sqrt{|mx - px|^2 + |my - py|^2} \quad (60)$$

$$db_{m,y} = \sqrt{|mx - yx|^2 + |my - yy|^2} \quad (61)$$

After calculating the distances between functions, optimization process would try to keep these values between ranges. The distances between functions that should be kept in the specified accessibility input ranges. The best case is to have 300 meters distance, while the worst case is to have 1500 meters distance. We used a common formula for addressing it, as follows:

$$\max(0, \min\left(1, \frac{\text{current value} - \text{worst}}{\text{best} - \text{worst}}\right)) \quad (62)$$

$$d_1 = \max(0, \min\left(1, \frac{db_{h,p} - 1500}{300 - 1500}\right)) \quad (63)$$

$$d_2 = \max(0, \min\left(1, \frac{db_{y,p} - 1500}{300 - 1500}\right)) \quad (64)$$

$$d_3 = \max(0, \min\left(1, \frac{db_{m,p} - 1500}{300 - 1500}\right)) \quad (65)$$

$$d_4 = \max(0, \min(1, \frac{db_{m,y}-2000}{500-2000})) \quad (66)$$

$$\text{Maximize } z_1 = \min(\frac{1}{d_1}, \frac{1}{d_2}, \frac{1}{d_3}, \frac{1}{d_4}) \quad (67)$$

5.2.2.2. Objective Function II: Wind Protection

Another objective is to maximize wind protection for living spaces, such as housing places and marine. Due to the previous boat experience in project region, it was both experienced and established that the region has calm wind. Because of the dominant wind coming from north-east or south-east in project region, islands that are close to offshore protect the other islands that are close to coastline, and vice versa. Secluded areas between each two islands were considered as protected regions. It was tried to locate the living spaces (houses, marine) as close as to these protected regions. For instance, in Figure 5.11., houses are close to protected regions, but marine is not enough.

To apply this objective to the model, the distances between desired functions and the protected regions will try to minimize to increase wind protection. (68), (69), (70) show that the calculation strategy of the distances between functions and protected regions should be minimized.

$$db_{h,p_i} = \sqrt{|hx - pix|^2 + |hy - piy|^2} \quad (68)$$

$$db_{m,p_i} = \sqrt{|mx - pix|^2 + |my - piy|^2} \quad (69)$$

$$\text{Maximize } z_2 = (\frac{1}{db_{h,p_i} + db_{m,p_i}}) \quad (70)$$

5.2.2.3. Objective Function III: Visibility

The third objective is to maximize visibility to the functions. The aim is to make commercial functions (yacht club, marine, houses) attractive. Since the project region is close to the cruise-ships line, there was a tendency to locate commercial functions (yacht club, marine, houses) so that they are noticeable from the passing ships. Cruise ships passing line and area of interest are illustrated in Figure 5.12.

Calculation of this objective started with generating lines between cruise-ship passing line and the desired functions in the parametric model. If the visibility lines

intersect with the islands that are closer to offshore, visibility will be bad. For instance, the number of visibility lines that are intersected with the islands that are closer to offshore is 6 for a housing unit, in Figure 5.13. The aim is to maximize visibility along the path of the cruise ship by minimizing the visibility lines to the functions explained earlier. Calculation strategy of maximization of visibility is formulized in (71).

$$\text{Maximize } z_3 = \left(\frac{1}{(noil_h + noil_y + noil_m)} \right) \quad (71)$$

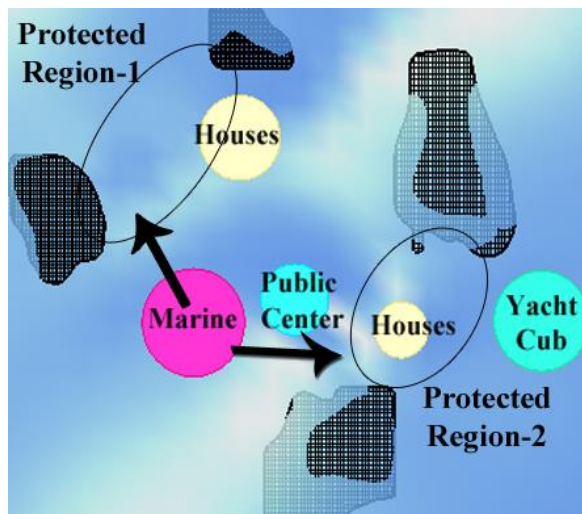


Figure 5.11. Top view of the project region to illustrate protected regions



Figure 5.12. A view of project region's cruise-ship passing line

5.2.2.4. Constraints

Location of the functions is restricted by the decision variable boundaries. Minimum and maximum values were generated not to exceed area of interest in

meters unit while the origin point is at Urla coast. These constraints were given as follows:

$$1101 < hx < 4000 \quad (72)$$

$$1184 < hy < 3184 \quad (73)$$

$$1101 < mx < 4000 \quad (74)$$

$$1184 < my < 3184 \quad (75)$$

$$1101 < yx < 4000 \quad (76)$$

$$1184 < yy < 3184 \quad (77)$$

$$1101 < px < 4000 \quad (78)$$

$$1184 < py < 3184 \quad (79)$$

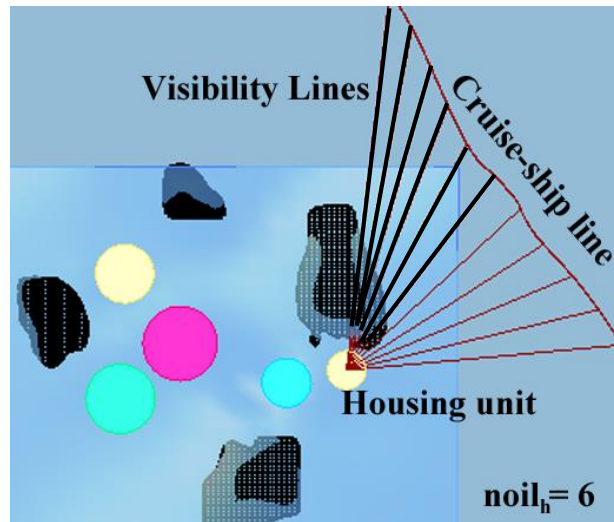


Figure 5.13. Example of generating the intersected lines for a housing unit

Distances for being closer to the wind protected areas should not exceed 300 meters.

$$z_2 < 300 \quad (80)$$

Each number of visibility lines should not exceed 5.

$$noil_h < 5 \quad (81)$$

$$noil_y < 5 \quad (82)$$

$$noil_m < 5 \quad (83)$$

The real depth data has taken from NAVIONICS navigation program. Interpolation was used to create an approximate sea bottom model. By this way, the location of the functions can be controlled to allow location at a specific water depth. The yacht club cannot be located on the water depth less than 20 meters.

$$wdy > 20 \quad (84)$$

The last constraint prevents all of the surface intersections, such as functions to functions and islands to functions.

5.2.3. Form Finding

The second part of the problem aims to create the street networks between city functions. In this step, the main question is how each function (defined by circle units) can be connected by generating meaningful paths between them, as in Figure 5.14.

The shortest walk algorithm has been used to generate the roads between functions. This algorithm finds the shortest distance between target points (functions) where their locations are already chosen from Pareto optimal solutions coming from the first optimization model and the other random points (represented the people). Meanwhile, this algorithm allows generating an organic network that is open to expansion and development.

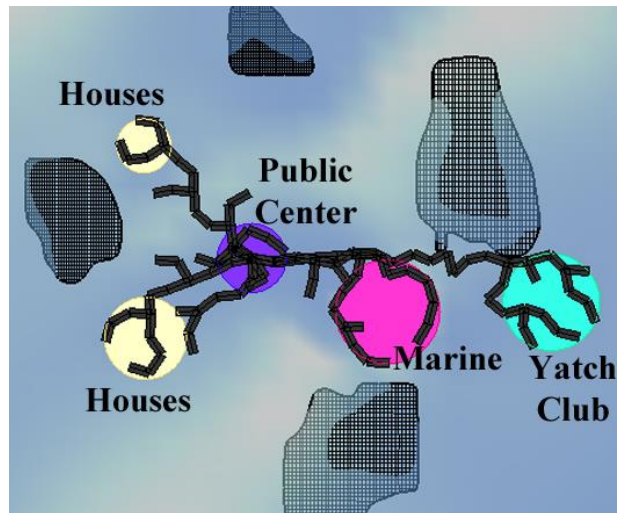


Figure 5.14. Generating the roads between functions with using shortest walk algorithm

5.2.4. Parametric Model of Application II

The parametric model has been created in the Grasshopper platform according to predefined objectives. The complete Grasshopper model is in Figure 5.15.

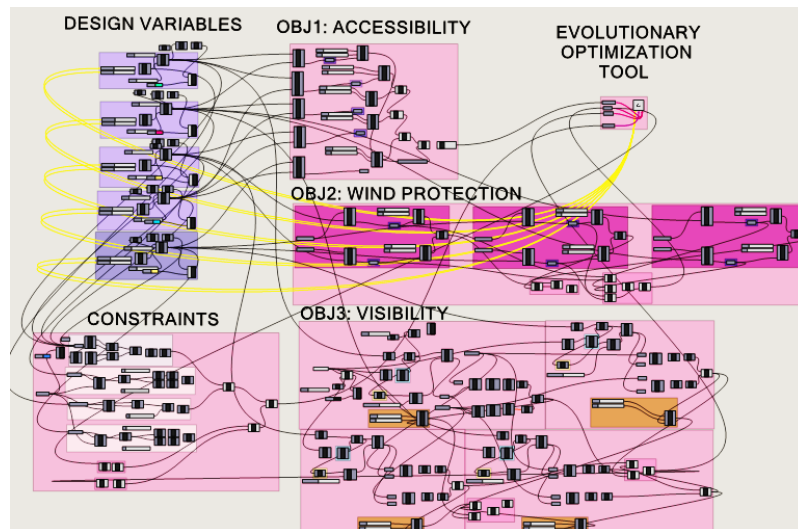


Figure 5.15. Overview of the Grasshopper model

5.2.5. Computational Results of Application II

NSGA-II and JDE with the parametric model of floating neighbourhood arrangement problem were tested on an Intel Core-i7 computer, with 8 GB of RAM. Population size was taken as 100 and both algorithms were terminated after 250

generations. The Pareto front approximations are given in Figure 5.16. for NSGA-II in Figure 5.17. for JDE, and Figure 5.18. for EDE, respectively.

Four designs for each algorithm have been identified through visual inspection. The selected solutions were randomly chosen from the Pareto front approximation for each algorithm. Their visual configuration illustrated in Figure 5.19. for NSGA-II, in Figure 5.20. for JDE and Figure 5.21. for EDE. All Pareto solutions have been satisfied the visibility of yacht club, because there was no need to yacht club to be wind protected. Despite the wind protection and visibility are conflicting for houses and marine, the configurations of them seem nice. Because the last two selected solutions (S3, S4, S3' and S4') for both algorithms have lower accessibility, marine and yacht club tend to locate far away from each other and the public place has less centralized. DE algorithm has wider breadth in Pareto front approximation, for each three axis. However, the last two solutions for JDE algorithm senseless tend to use the maximum bound value of the x-coordinate. Instead, it was tried to locate our functions in a way that scattered across the islands.

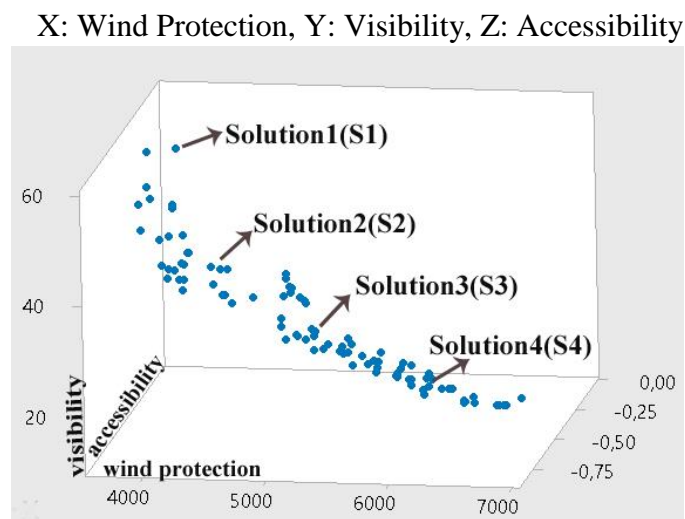


Figure 5.16. Pareto Front approximation for NSGA-II

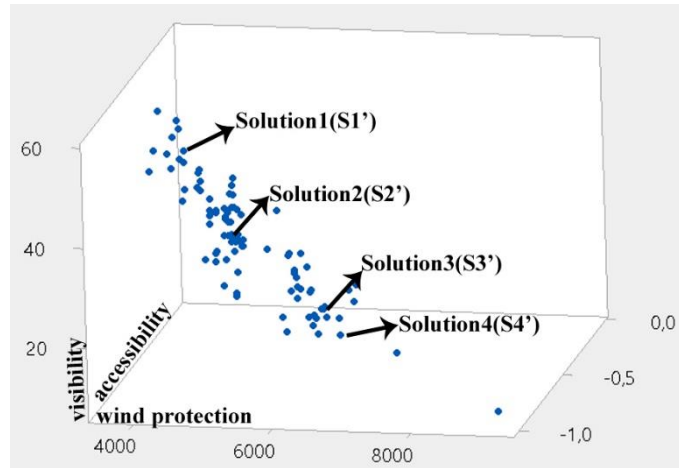


Figure 5.17. Pareto Front approximation for DE Algorithm

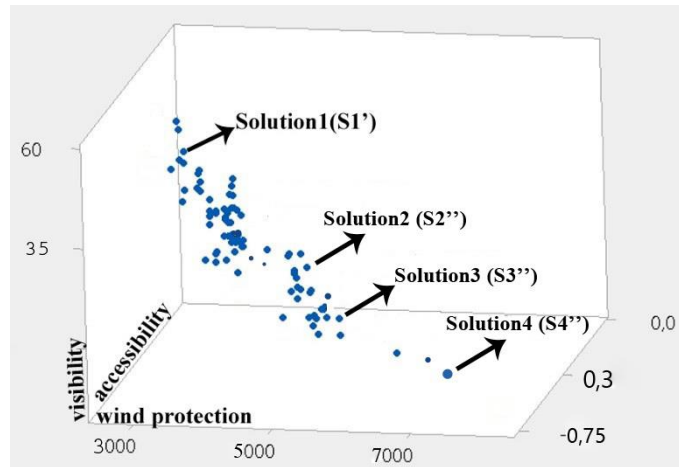


Figure 5.18. Pareto Front approximation for EDE Algorithm

After the objective function values gathered from all algorithms were taken, I_H values for each algorithm were calculated. NSGA-II performs better for 100 population and 250 generation according to hypervolume results. However, this conclusion should be tested statistically.

A paired-t test between hyper volume results was carried out and it was found that NSGA-II is significantly better than JDE because the p- value of 0.000 was less than the $\alpha = 0.05$ level. NSGA-II is also better than EDE because the p- value of 0.002 was less than the $\alpha = 0.05$ level. On the other hand, since the p-value is 0.695, JDE and EDE algorithms are equivalent.

ALGORITHMS	HYPER VOLUME RESULTS
NSGA-II	0.4476
EDE	0.4227
JDE	0.4172

Table 4 Hypervolume Results of Application 2

● Marine Unit
 ● Housing Unit
 ● Public Center
 ● Yacht Club

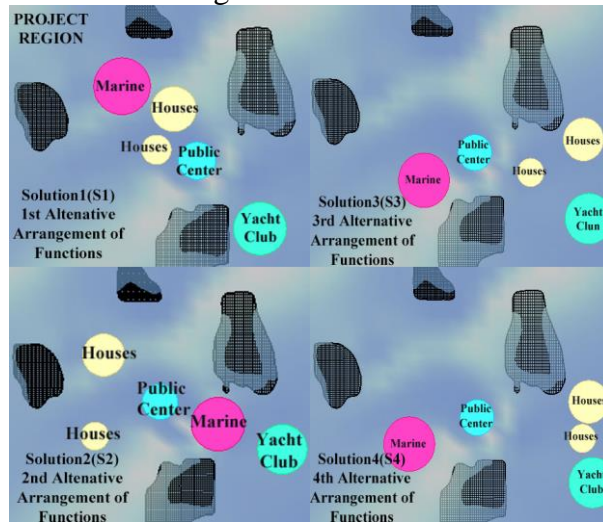


Figure 5.19. Alternative solutions for NSGA-II

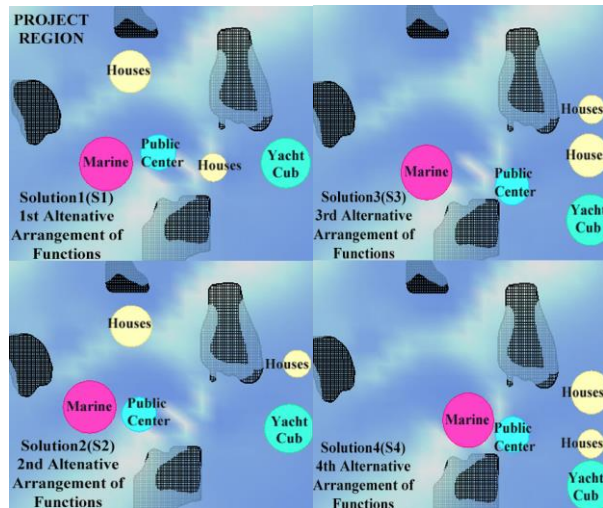


Figure 5.20. Alternative solutions for DE Algorithm

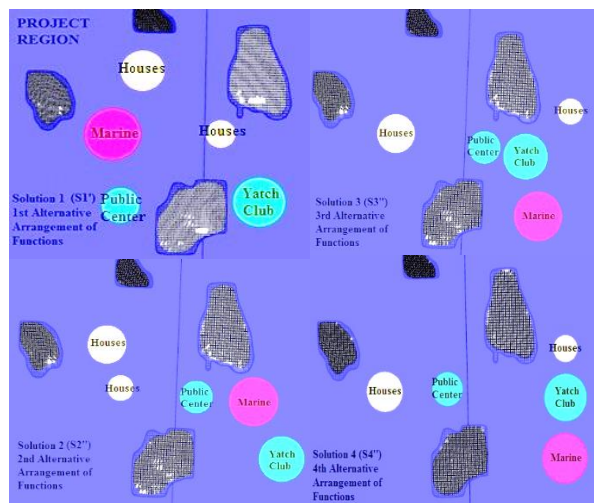


Figure 5.21. Alternative solutions for EDE Algorithm

Before the detailed visualization stage, one of the solutions of NSGA-II was taken and performed the shortest-walk algorithm to generate the form. The final whole floating neighbourhood concept design is shown in Figure 5.22. The proposed marine part of the floating settlement can be seen in Figure 5.27. The other renders can be found between Figure 5.23. and Figure 5.29.



Figure 5.22. Final whole design of floating neighbourhood



Figure 5.23. Other perspective of whole design of floating neighbourhood



Figure 5.24. Computer render of one of the housing places



Figure 5.25. One of the housing places, coral bay

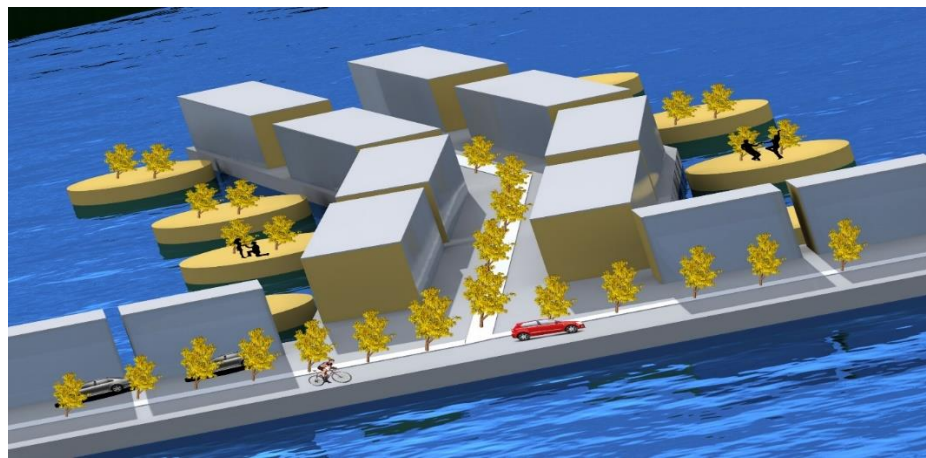


Figure 5.26. One of the housing places, mimosa bay

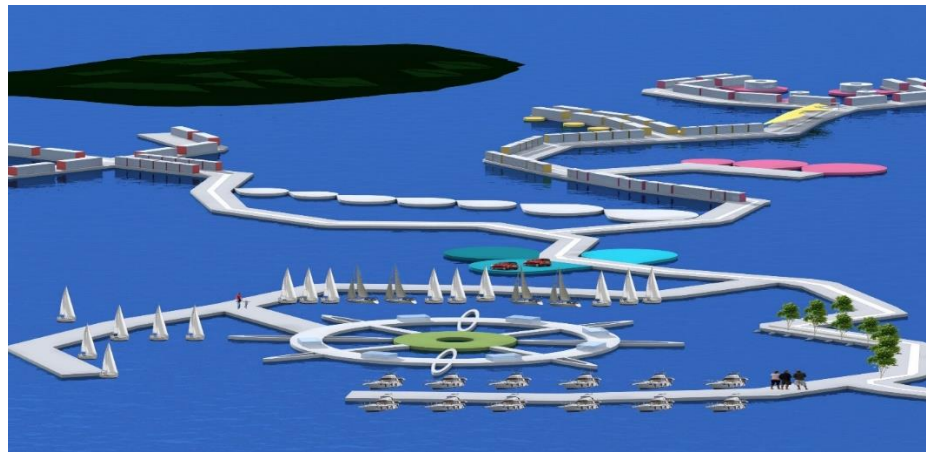


Figure 5.27. Marine part of the floating neighbourhood

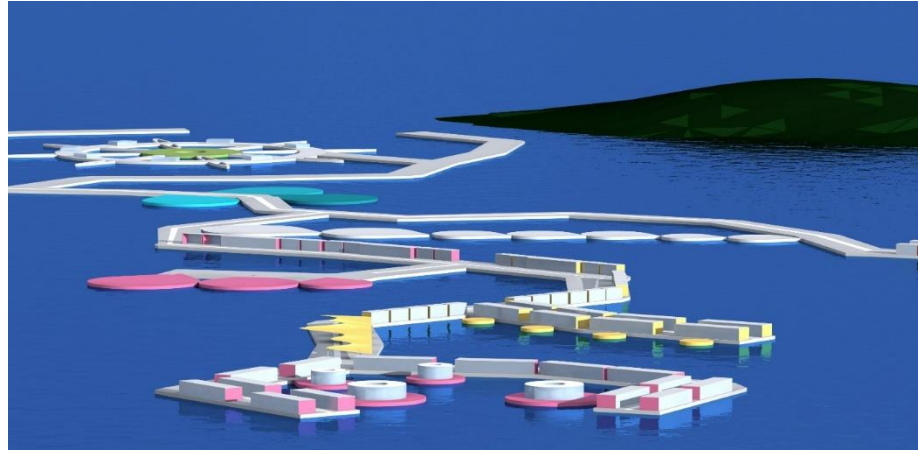


Figure 5.28. All housing places

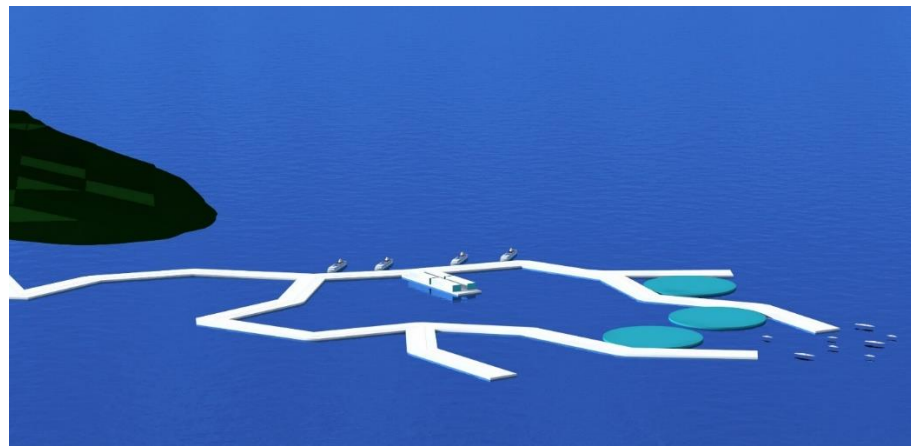


Figure 5.29. Yatch Club part of the floating neighbourhood

5.3. Application III: Floating Underwater Hotel Room Design

Alternative living spaces have been constantly sought because of land shortage and increasing population around the world. Floating structures can be considered as an alternative touristic destination to settlements that is surrounded by eighty percentages of salty water, as the seas and oceans are constantly observed by Yuksek and Arıkan (2009). In addition to this, although these kind of structures should be attractive touristic centers for people, the design of a floating structure is a challenging work because, both technical and architectural aspects need to be concerned. For this reason, a floating underwater hotel room design in the project region illustrated in Figure 5.30. Marmaris/ Bozburun, a sea resort of Izmir, had been proposed with using evolutionary computation techniques to handle a kind of complex problem.



Figure 5.30. A top view of Marmaris/ Bozburun coast

In this part of the study, a specific case study using floating underwater hotel room is presented. Design solutions for proposed underwater hotel room are trying to be found by considering minimization of total cost and maximization of shading performance. The reason why cost is minimized is that constructing structures on water are much more expensive than structures on land. According to the observations in the project region, there is an influx of foreign tourists in summer time. Project region addresses the limited space available in such a small territory.

Furthermore, shading performance maximization has been taken into account because the project region has a quite overheating problem in summertime. In this problem, the project was started as an individual hotel room design satisfying the design objectives and then this individual hotel room had multiplied around coast of project area with respect to the design constraints.

Evolutionary computation in the field of floating underwater structure is a unique approach. Any study that is tackled both floating structures and multi-objective optimization together is not encountered according to our researches in the literature. However, nowadays computational optimization is slowly started to be implemented on architectural design. For instance, Soh and Yong (1996) had done shape optimization. In another study (Camp et al., 1998), genetic algorithms were developed for discrete optimization of design of two dimensional structure with considering single total weight (cost) objective. Application of genetic algorithm for the support location optimization of beams and for cost optimization of industrial building has been studied by Wang and Chen (1996), Kumar (2013). On the other hand, genetic algorithms are used for construction site layout in project planning. (Mawdesley et.al, 2002) The design problem is used to demonstrate the

approach is a large office layout planning problem with its associated topological and geometrical arrangements of space elements by using evolutionary algorithms (Jo and Gero, 1998). More recent studies propose multi-objective design problems by using evolutionary algorithms. In the study of Sariyildiz (2008), multi-objective-optimization-based positioning of houses in a residential neighbourhood is described. Turrin et. al (2011) considered bi-objective, the solar heat gain and daylight transmittance of a long span roof. With respect to new review papers (Evins, 2013) and (Machairas, 2014), objective functions including sustainability are getting popular.

In this study, design variables, some of the constraints and objective functions are predefined and some are user defined. Then it was referred to evolutionary computation to reach optimal design solutions by using optimization component in Grasshopper Environment.

5.3.1. Problem Definition of Application III

The mathematical background of the problem is described in this section. Formulation of the problem is a collaborative work with Architect Ayca Kirimtat. The problem is to find optimal design solutions in order to maximize shading performance and minimize total cost subject to restricted floor areas, balance float, buoyancy, boundary limits of decision variables for the floating underwater hotel room case. The notations are needed to be explained in order to understand how we formulate the objectives and constraints in the model, as in the index of symbols.

5.3.1.1. Decision Variables

There are some design variables defined to control when optimization of the design. The design variables are as follows:

$$ur_x, ur_y, uf_x, uf_y, h_u, h_s, a_1, a_2, a_3, b_1, b_2, b_3, S, r_{win(1,..,3)}.$$

5.3.1.2. Objectives

The design goals are minimization of total cost while maximization of shading performance.

- Total Cost

Both underwater and superstructure part of the hotel room was considered while calculating total cost. Cost calculation includes multiplication of unit cost with floor areas, roof areas and each window areas.

$$Obj_1 = TC \quad (85)$$

$$TC = UFC + URC + UWC + UWAC + SBW + SBG + SRC \quad (86)$$

Where

$$UFC = (uf_x * uf_y) * ucc \quad (87)$$

$$URC = (ur_x * ur_y) * ucc \quad (88)$$

$$UWC = \pi * (r_{win(1,..,3)})^2 * ugc \quad (89)$$

$$UWAC = TLA * ucc \quad (90)$$

$$SBW = TSA * p_{wall(1,..,4)} * ucc \quad (91)$$

$$SBG = TSA * p_{windows(1,..,4)} * ugc \quad (92)$$

$$SRC = SRA * utc \quad (93)$$

- Shading Performance

Shading performance is conflict with total cost; because the more shading element is bigger, the more it effects total cost. There is a shading element above superstructure part and it protects the room from sunlight. The reason that the hotel room will run throughout whole summer, it was tried to find Pareto optimal shading component forms while also satisfying the total cost. For calculation of shading performance, illuminance (lux) of the superstructure floor is calculated by making use of DIVA component in Grasshopper Parametric Environment.

Sky condition is clear sky with sun and one of the summer day has been selected for illuminance simulations settings. The illuminance value of the superstructure floor (I_f) was affected from the materials (shading element, sea water, superstructure windows, and floor). Performance of the superstructure's shading component is quite important to provide shades on its floor. Formula (62) is used for this objective. It is assumed that the worst case is 5000 lux, the best case is 1000 lux with more shades as in (94).

¹ Diva-for-Rhino is a plug-in for the Rhinoceros-NURBS modeler making analysis of daylighting and energy modelling (www.diva4rhino.com).

$$Obj_2 = \max(0, \min\left(1, \frac{I_f - 5000}{1000 - 5000}\right)) \quad (94)$$

5.3.1.3. Constraints

Because of the project region requirements and standing on the sea water, there are some constraints for the design of floating underwater hotel room.

Buoyancy and float balance are the most essential ones. Moreover, height of the underwater part is limited to water depth of the project site which is between 0-6 meters, height of the superstructure part has some boundaries not to exceed maximum wave height of the region (1-1.5 meters).

Constraints (95) to (108) represent the boundary limits of design variables with the meters unit:

$$5 < ur_x < 10 \quad (95)$$

$$2 < ur_y < 5 \quad (96)$$

$$3 < uf_x < 5 \quad (97)$$

$$2 < uf_y < 4 \quad (98)$$

$$2 < h_u < 3 \quad (99)$$

$$1 < h_s < 2 \quad (100)$$

$$1 < h_b < 1.5 \quad (101)$$

$$1.5 < A_1 < 2 \quad (102)$$

$$2.8 < A_2 < 5 \quad (103)$$

$$1.5 < A_3 < 2 \quad (104)$$

$$1.5 < B_1 < 2 \quad (105)$$

$$2.8 < B_2 < 5 \quad (106)$$

$$1.5 < B_3 < 2 \quad (107)$$

$$0,2 < r_{win} < 2.5 \quad (108)$$

Constraints (109) and (110) ensure the area of underwater floor and roof. Floor area boundaries come from customer demand. The reason that roof area boundaries are bigger than floor's is to maximize volume of the underwater part.

$$7 \text{ m} < (uf_x * uf_y) < 10 \text{ m} \quad (109)$$

$$25 \text{ m} < (ur_x * ur_y) < 50 \text{ m} \quad (110)$$

Buoyancy is the most essential constraint as having a floating structure. Employing the buoyancy started with calculation of solid intersection of sea water which has 6m depth and room. Volume of the solid intersection was used to calculate displacement, multiplied with density of seawater. After this calculation, difference between total mass of the room and displacement was calculated. It was tried to hold the value of by between 0 and 0.1 in constraint (111).

$$0 < by < 0.1 \quad (111)$$

The other constraint is float balance of the room. According to this constraint, center of gravity and center of buoyancy of water were displaced by the room and they are lying in the same vertical line.

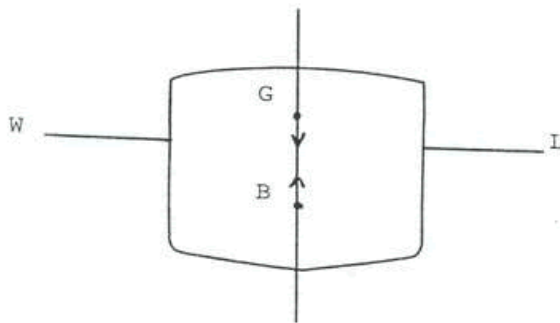


Figure 5.31. Representation for center of gravity and center of buoyancy
(<http://www.shipinspection.eu>)

For instance, in Figure 5.31., G (center of gravity) and B (center of buoyancy) lie in same vertical line amidships, then the floating structure is in equilibrium (www.shipinspection.eu).

To address this strategy, it was taken the volume of each element of the room. Then, the densities of them were multiplied to calculate the mass. For the windows plexiglass was used, so the density of the plexiglass was taken 1200 kg/m³. For the

floors and walls, the concrete structure-which has 2400 kg/m³ density was used. The material of the superstructure roof (shading component) is composite because of its strength. The density of the composite is 5000 kg/m³. From the weights, it was reached to the center of gravity. Center of buoyancy is coming from the solid intersection of sea and room. Then it was tried to keep their vertical vectors on the same line.

5.3.2. Generative Model of Application III

The parametric model for generating and evaluating the underwater floating hotel room design has been implemented in the Grasshopper platform. A screenshot of the interface of Grasshopper can be seen in Figure 5.32.

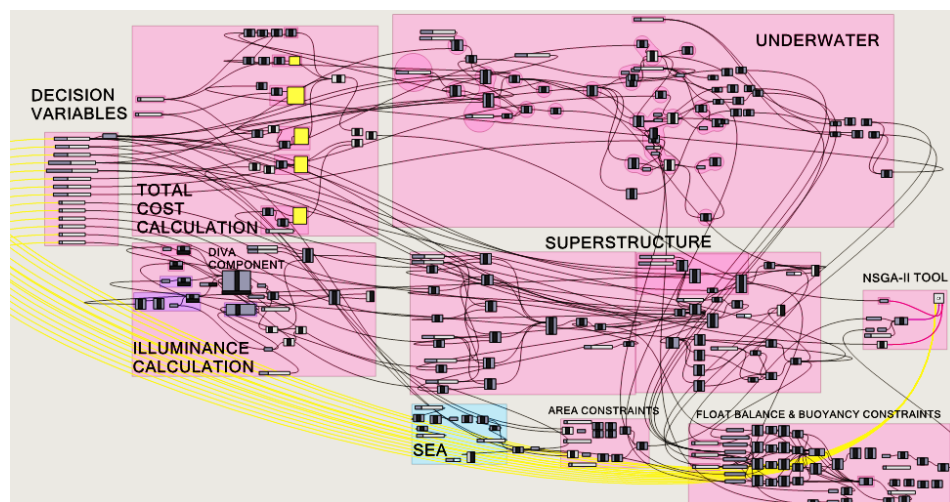


Figure 5.32. Grasshopper model of underwater floating hotel room

A rectangular underwater floor was generated to satisfy desired area of the room, as a first stage. In order to increase the volume of the room, the rectangular roof of the underwater part has been generated with bigger dimensions. The circular underwater windows were also considered in the model. As the second step, the superstructure floor and totally glass superstructure barriers were placed above water. Then, the shading component of the superstructure was formed with 6 control points that are shown as A1, A2, A3, B1, B2, B3 in Figure 5.33. Their heights (Z-axis) are some of our decision variables. Since they are important to provide shades on the superstructure floor, it was tried to find accurate values for these control points. The second stage of the problem is deriving the numerical figures to formulate the objectives and the constraints.

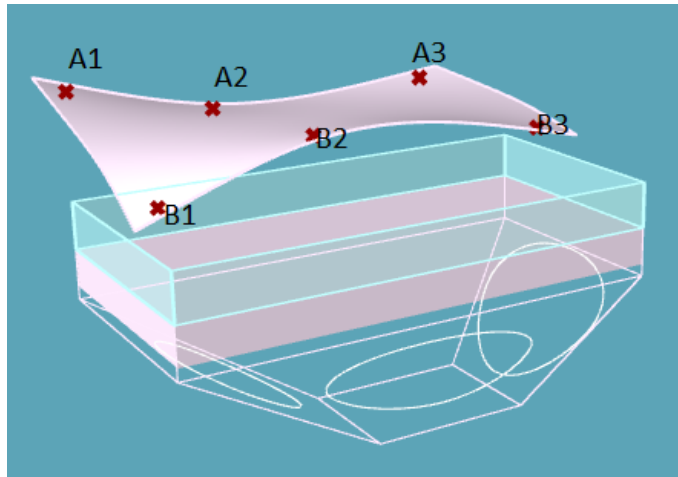


Figure 5.33. Superstructure shading device form illustrated by generative model

NSGA-II, JDE and EDE algorithms were implemented as a plug-in for the Grasshopper environment. After setting up the model, it was connected appropriately to the decision variables, objective functions and constraints. The parametric model was finalized by including components that generate a 3D visualization.

5.3.3. Computational Results of Application III

The algorithms were tested on an Intel Core-i5 computer, with 4 GB of RAM for the underwater floating hotel room design. The Pareto front approximation after 100 generations and a population of 100 had been obtained while all the final solutions are feasible. Some of the solutions from Pareto Front Approximations were picked.

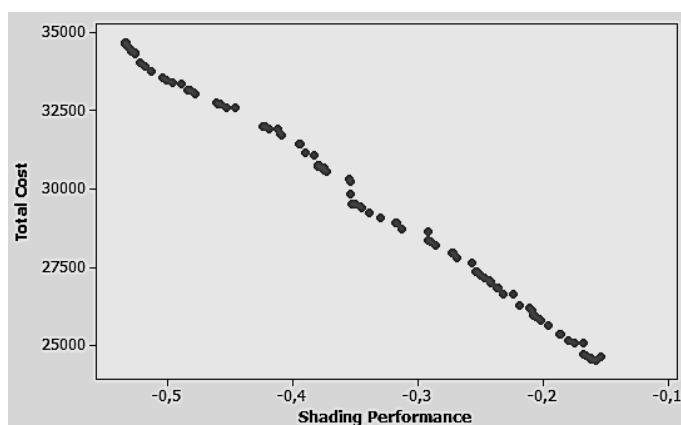


Figure 5.34. Pareto Front Approximation for NSGA-II

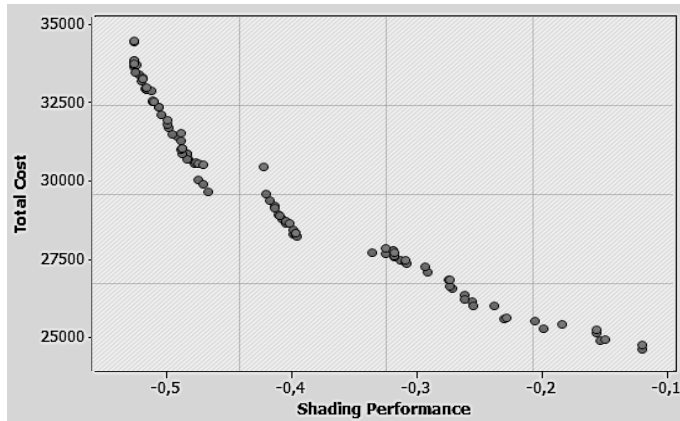


Figure 5.35. Pareto Front Approximation for JDE Algorithm

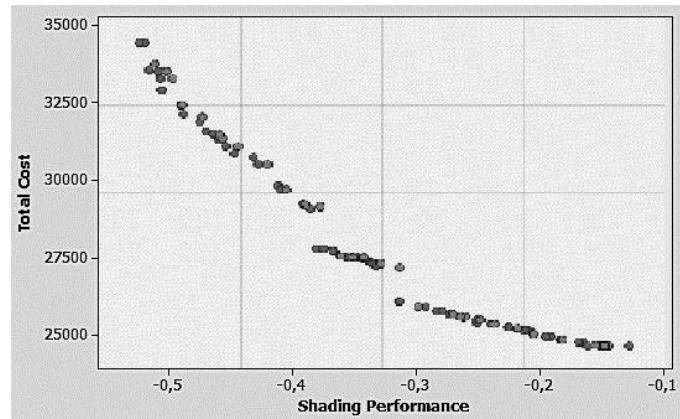


Figure 5.36. Pareto Front Approximation for EDE Algorithm

According to the hypervolume calculation, EDE algorithm seems to be slightly better in results. However, it should be again justified statistically. To justify the performance of the algorithms, a paired-t test has been carried out. Since the p-value is less than the $\alpha=0.05$ level between EDE & DE and EDE & NSGA-II, it can be concluded that EDE performs better for this problem.

ALGORITHMS	I_H
EDE	0.2734
JDE	0.2727
NSGA-II	0.2510

Table 5 Hypervolume Results of Application 3

Total cost results are changing according to Pareto front rank. For instance, one solution has the highest cost which is 34,549 TL. Another solution is 29,056 TL, and other solution has the lowest cost which is 24,539 TL. However, while a solution has the highest cost, its shading performance is much better than others. Also, the one with the lowest cost has the worst shading performance. The worst shading performance means that there are a lot of sunlight is collecting on the superstructure floor surface. On the other hand, the bigger shading component is in which the less sunlight comes in, it affects total cost negatively.

All the solutions are feasible according to buoyancy and float balance. It means the whole structure both underwater part and superstructure part is able to floating. Also, the results of the circular windows' semi-diameters on the walls of underwater part are quite satisfied. Thus, occupants can feel that they are in aquarium. One of the solutions from design alternatives gathered from optimization results is picked and detailed visualized.

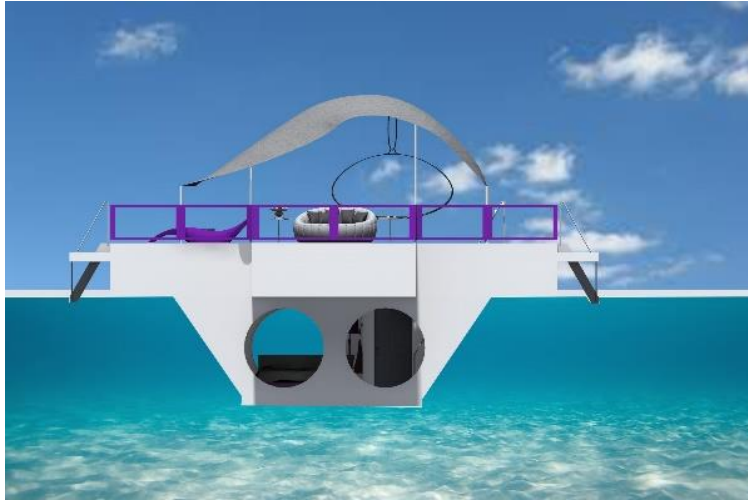


Figure 5.37. Elevation of the proposed hotel room

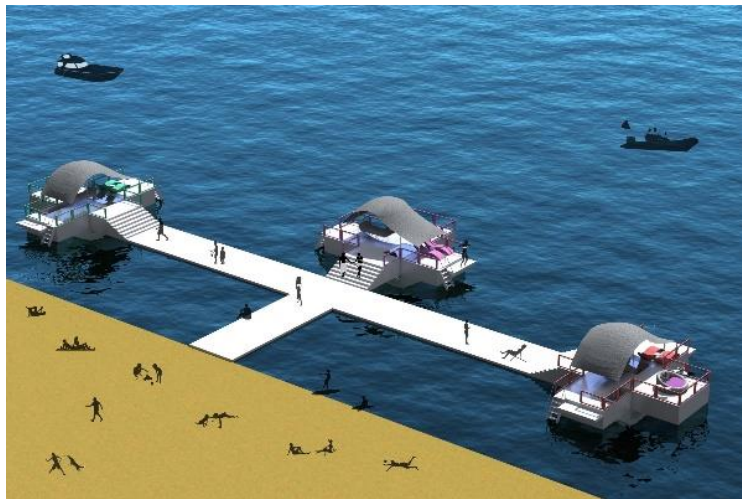


Figure 5.38. Multiple rooms



Figure 5.39. Superstructure part illustration

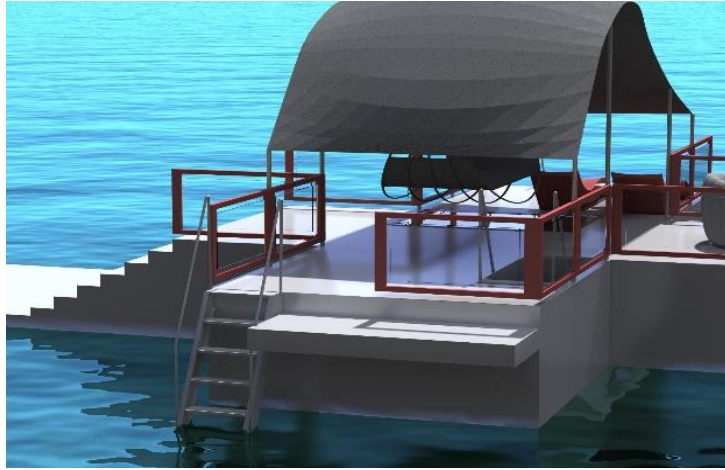


Figure 5.40. Superstructure part illustration



Figure 5.41. Underwater part illustration

6 CONCLUSIONS & FUTURE WORK

An investigation regarding the applications of optimization algorithms on architectural design problems after examination of these optimization algorithms on benchmark functions was presented in this thesis. This idea comes from the reason that nature of the architectural problems have real numbers to optimally find, since the term researched and analyzed over the years by engineers, called, “Real Parameter Optimization”.

First of all, in the context of constrained single real parameter optimization; 22 benchmark problems taken from CEC’2006 technical report were considered to solve by using EDE algorithm and various constrained handling methods (SF and ε -constraint). According to the solutions, EDE algorithm was very competitive to some of the best performing algorithms from the literature.

After the comparison of the algorithms, constrained multi-objective test functions were used to measure performance of the algorithms. Good experiences gathered from the benchmark problems encourages to apply them into architectural problems.

Three case studies regarding the design of restaurant layouts, floating settlement design and underwater hotel room design using the proposed approach had been studied.

Application 1 addresses a study of layout design for the architecture of restaurants. A multi-objective problem was setup to locate a kitchen and a set of tables which collectively increases the profit and decreases investment. The studied approach takes functional, economic, construction and architectural aspects of the layout into account. It was tried to achieve plausible layout solutions that may be considered as efficient designs that maximize profit while minimizing investment. With respect to the algorithm comparison, it was found that JDE performs better than NSGA-II and EDE in terms of hypervolume whereas NSGA was better than EDE.

In the application 2, it was focused on a comparison between the NSGA-II, JDE and EDE algorithms in the problem of design of efficient floating neighborhood. First optimization problem entails the configuration of the functions of the floating settlements. These functions were identified according to concept

“yacht tourism development”. After determining functions’ locations in order to maximize accessibility, wind protection and visibility; it was passed to the form generation step. The shortest walk algorithm had been used to create the streets. With respect to the algorithm comparison, it was found that NSGA-II performs better than DE algorithm while EDE has better formance than JDE.

In the application 3, an investigation regarding the design of floating underwater hotel room using an approach which blends parametric design with evolutionary computation to achieve optimal solutions had been presented. A parametric model that generates the underwater and above water part of the hotel room as well as shading device of the above water part was developed. Through the use of the proposed model, it can be obtained wide range of design alternatives. A case study regarding the design of a floating underwater hotel room using the proposed approach had been carried out. The site is that of the new hotel in Marmaris, Bozburun. The designs that were generated by the Evolutionary Algorithm that shows a good adaptation to the site specific constraints (buoyancy, float balance, size limits) and parameters (dimensions of the underwater part and superstructure part, sizes of windows and shading device). The algorithms (NSGA-II, JDE, EDE) were suitable to obtain feasible solutions. The application of the algorithms to the floating underwater hotel room problem also presents variety of interesting solutions. After analysis of one of the results, a model had been created by adding additional elements of the hotel room such as stairs, furniture, beams, etc.

Future development of the approach can include the investigation of state of the art optimization schemes, as well as improvements in the calculation detail of the objective function values. The other algorithms or constraint handling methods should be used for optimization and other strategies can be used for algorithm comparison. Future approach can include the other evolutionary algorithms, harmony search, particle swarm etc. It was tried to call for better performing algorithms or constraint handling techniques, because one algorithm that has good performance in one problem may not have a good performance on other problems. On the other hand, design of experiments can be implemented in future.

REFERENCES

- Brest J., Greiner S., Boskovic B., Mernik M., and Zumer V.,** 2006, “Self adapting control parameters in differential evolution: A comparative study on numerical benchmark problems,” *IEEE Trans. Evol. Comput.*, vol. 10, no. 6, pp. 646–657.
- Cai H. R., Chung C. Y., and Wong K. P.,** 2008, “Application of differential evolution algorithm for transient stability constrained optimal power flow”, *IEEE Transactions on Power Systems*, Vol. 23, No. 2.
- Chiou J.-P.,** 2009, “A variable scaling hybrid differential evolution for solving large-scale power dispatch problems”, *IET Generation, Transmission & Distribution*, Volume: 3, Issue: 2, Page(s): 154-163.
- Coelho L.S. and Mariani V.C.,** 2006, Combining of chaotic differential evolution and quadratic programming for economic dispatch optimization with valve-point effect, *IEEE Transactions on Power Systems*, Vol. 21, Issue 2, pp. 989–996.
- Abou El Ela A. A., Abido M. A., and Spea S. R.,** 2009, “Optimal power flow using differential evolution algorithm”, *Electrical Engineering (Archiv fur Elektrotechnik)*, Vol. 91, No. 2, Page(s): 69 – 78.
- Chang C., Wong J., Chiou J., and Su C-T.,** 2007, “Robust searching hybrid differential evolution method for optimal reactive power planning in large-scale distribution systems”, *Electric Power Systems Research*, Vol. 77, Issues 5-6, Pages 430-437..
- Chiou J., Chang C., and Su C-T.,** 2005, “Variable Scaling Hybrid Differential Evolution for Solving Network Reconfiguration of Distribution Systems”, *IEEE Transactions on Power Systems*, Vol. 20, No. 2, Page(s) 668 – 674.
- Chang Y-P. and Wu C-J.,** 2005, “Optimal multiobjective planning of large-scale passive harmonic filters using hybrid differential evolution method considering parameter and loading uncertainty”, *IEEE Transactions on Power Delivery*, Vol. 20, No. 1, Page(s) 408 – 416.

REFERENCES (CONTINUE)

Chang Y-P. and Low C., 2008, “An ant direction hybrid differential evolution heuristic for the largescale passive harmonic filters planning problem”, *Expert Systems with Applications*, Vol. 35, Issue 3, Pages 894-904.

Abbass H., 2002, “The self-adaptive Pareto differential evolution algorithm,” in *Proc. Congr. Evol. Comput.*, vol. 1., pp. 831–836.

Chang C.S., Xu D.Y., and Quek H.B., 1999, “Pareto-optimal set based multiobjective tuning of fuzzy automatic train operation for mass transit system”, *IEE Proceedings on Electric Power Applications*, 146(5):577–583.

Abbass H. A., 2001, “A memetic pareto evolutionary approach to artificial neural networks”, In *The Australian Joint Conference on Artificial Intelligence*, pages 1–12, Adelaide, Australia, Springer. *Lecture Notes in Artificial Intelligence* Vol. 2256.

Beyer H.-G. and Finck S., 2012, “HappyCat -- A Simple Function Class Where Well-Known Direct Search Algorithms Do Fail,” In *Proc. of Parallel Problem Solving from Nature~12*, pp. 367-376, Ed by C. A. Coello Coello et al., Springer, Berlin, 2012.

Coello Coello, C.A., (2002) “Theoretical and Numerical Constraint-Handling Techniques Used with Evolutionary Algorithms: A Survey of the State of the Art”, *Comput. Methods Appl. Mech. Engrg.*, 191(11-12), pp. 1245-1287.

Caldas L.G., Norford L.K., 2002, A design optimization tool based on a genetic algorithm, *Automation in Construction* 11 2002 173–184.

Bittermann, M., 2009, *Intelligent Design Objects (IDO)*, PhD Thesis, Delft: Delft University of Technology.

Chatzikonstantinou, I., 2011, *Evolutionary Computation and Parametric Pattern Generation for Airport Terminal Design*, Msc Thesis, Delft University of Technology.

REFERENCES (CONTINUE)

Corne D., Dorigo M., and Glover F. (eds.), 1999, "Part Two: Differential Evolution," *New Ideas in Optimization*, McGraw-Hill, pp. 77-158.

Babu B. V. and Onwubolu G. C. (eds.), 2004, "New Optimization Techniques in Engineering", Springer Verlag.

Baldock R., Shea K., 2006, Structural Topology Optimization of Braced Steel Frameworks Using Genetic Programming, I.F.C. Smith (Ed.): EG-ICE 2006, LNAI 4200, pp.54-61.

Basu M., 2008, "Optimal power flow with FACTS devices using differential evolution", *International Journal of Electrical Power and Energy Systems*, Vol. 30, Issue 2, Pages 150-156.

Blum C., Chiong R., DeJong K., Michalewicz Z., Neri F., Weise T., 2012, "Evolutionary optimization" in *Variants of Evolutionary Algorithms for Real-World Applications*, pp.1-29, Springer.

Camp C., Pezeshk S., and Cao G., 1998, "Optimized design of two-dimensional structures using a genetic algorithm", *Journal of Structural Engineering* 124, 551-559.

Chakraborty U K., 2008 (ed.) *Advances in Differential Evolution*. Berlin: Springer.

Chiong R., Neri F., McKay R.I., 2010, "Nature that breeds solutions," in *Nature-Inspired Informatics for Intelligent Applications and Knowledge Discovery: Implications in Business, Science and Engineering*, Chapter 1, pp.1-24, Hershey, PA: Information Science Reference.

Corne D., Dorigo M., and Glover F. (eds.), 1999, "Part Two: Differential Evolution," *New Ideas in Optimization*, McGraw-Hill, pp. 77-158.

REFERENCES (CONTINUE)

Das S., Abraham A., Chakraborty U. K., and Konar A., 2009, “Differential evolution using a neighborhood based mutation operator,” *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 526–553.

Das S., Konar A., and Chakraborty U., 2005, “Improved differential evolution algorithms for handling noisy optimization problems,” in *Proc. IEEE Congr. Evol. Comput.*, vol. 2. 2005, pp. 1691–1698.

Das S., Suganthan P.N., 2011, *Differential Evolution: A Survey of the State-of-the-Art*, *IEEE Trans. Evol. Comput.*, Vol 15, No.1.

Deb, K., 2000, An efficient constraint handling method for genetic algorithms, *Comput. Methods Appl. Mech. Eng.*, 186, pp. 311–338.

Deb K., Pratap A., Agarwal S., Meyarivan T., 2002, A Fast and Elitist Multiobjective Genetic Algorithm: Nsga-ii, *IEEE Transactions on Evolutionary Computation*, Vol.6, No.2, Pp. 182-197.

Deb K., Pratap A., Agarwal S., Meyarivan T., 2002, A Fast and Elitist Multiobjective Genetic Algorithm: Nsga-ii, *IEEE Transactions on Evolutionary Computation*, Vol.6, No.2, April 2002, Pp. 182-197.

Deb K., Pratap A., Meyarivan T., 2001, Constrained test problems for multi-objective evolutionary optimization, 1st International Conference of Evolutionary Multi-Criterion Optimization, Zurich Switzerland.

De Graaf R.E., 2009, “Innovations in urban water management to reduce the vulnerability of cities: Feasibility, case studies and governance”, PhD Thesis, TU Delft, Faculty of Civil Engineering and Geosciences, Department of Water Management.

REFERENCES (CONTINUE)

Derrac J., Garcia S., Hui S., Herrera F., Suganthan P. N., 2013, "Statistical analysis of convergence performance throughout the search: A case study with SaDE-MMTS and Sa-EPsDE-MMTS," IEEE Symp. on Differential Evolution 2013, IEEE SSCI 2013, Singapore.

Ekici B., 2014, "Multi-Performance Based Computational Design in Architecture: Integrated High-Rise Buildings " Master of Science Department of Architecture, Yasar University.

Epperly T., Global optimization test problems with solutions. Available at <http://citeseer.nj.nec.com/147308.html>.

Evins R., 2013, "A review of computational optimization methods applied to sustainable building design" Renewable and Sustainable Energy Reviews 22, 230-245.

Floudas C. and Pardalos P., 1987, A Collection of Test Problems for Constrained Global Optimization, volume 455 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, Germany.

Floudas C., 1999, Handbook of Test Problems in Local and Global Optimization. Nonconvex Optimization and its Applications. Kluwer Academic Publishers, The Netherlands.

Gagne J.M.L., Andersen M., 2010, A Multi-objective Façade Optimization for Daylighting Design Using a Genetic Algorithm, 4th National Conference of IBPSA-USA SimBuild 2010, New York, August 11-13.

Hansen N. and Ostermeier A., 2001, "Completely derandomized self-adaptation in evolution strategies", Evolutionary Computation, 9(2) pp. 159–195.

Hansen N., Finck S., Ros R. and Auger A., 2012, "Real-Parameter Black-Box Optimization Benchmarking 2010: Noiseless Functions Definitions" INRIA research report RR-6829, March 24, 2012.

REFERENCES (CONTINUE)

- Himmelblau D.**, 1972, Applied Nonlinear Programming. McGraw-Hill, New-York.
- Hock W. and Schittkowski K.**, 1981, Test Examples for Nonlinear Programming Codes. Lecture Notes in Economics and Mathematical Systems. Springer-Verlag, Berlin, Germany.
- Huang V. L., Qin A. K., and Suganthan P. N.**, 2006, "Self-adaptive Differential Evolution Algorithm for Constrained Real-Parameter Optimization", Nanyang Technological University, Singapore.
- Huang V. L., Qin A. K., Suganthan P. N., and Tasgetiren M. F.**, 2007, "Multiobjective optimization based on self-adaptive differential evolution algorithm," in Proc. Congr. Evol. Comput., pp. 3601–3608.
- Huang V. L., Zhao S. Z., Mallipeddi R., and Suganthan P. N.**, 2009, "Multiobjective optimization using self-adaptive differential evolution algorithm (special session and competition on 'performance assessment of constrained/bound constrained multiobjective optimization algorithms')," in Proc. Conf. Congr. Evol. Comput., pp. 190–194.
- Iorio A. W. and Li X.**, 2004 "Solving rotated multi-objective optimization problems using differential evolution," in Proc. AI: Adv. Artif. Intell., LNCS 3339. 2004, pp. 861–872.
- Jo J., and Gero J. S.**, 1998, "Space layout planning using an evolutionary approach", Artificial Intelligence in Engineering 12(3): 149-162.
- Jiménez F., Gomez-Skarmeta A.F., Sanchez G. and Deb K.**, 2002, "An evolutionary algorithm for constrained multiobjective optimization," Proceedings of Congress on Evolutionary Computation, Honolulu, HI, pp. 1133-1138.
- Karabulut K., Tasgetiren M. F.**, 2014, A variable iterated greedy algorithm for the traveling salesman problem with time windows, Information Sciences, Volume 279, 20 September 2014, Pages 383-39.

REFERENCES (CONTINUE)

Kannan S. and Murugan P., 2008, “Solutions to transmission constrained generation expansion planning using differential evolution”, European Transactions on Electrical Power.

Kozieland S., Michalewicz Z., 1999, Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evolutionary Computation*, 7(1):19–44.

Kukkonen S. and Lampinen J., 2004, “An Extension of Generalized Differential Evolution for Multiobjective Optimization with Constraints”, In *Parallel Problem Solving from Nature - PPSN VIII*, Page(s) 752–761, Springer-Verlag. Lecture Notes in Computer Science Vol. 3242, Birmingham, UK.

Kumar R., 2013, “Cost Optimization of Industrial Building using Genetic Algorithm”, *International Journal of Scientific Engineering and Technology* (ISSN : 2277-1581) Volume 2 Issue 4, pp : 185-191.

Lakshminarasimman L. and Subramanian S., 2008, “Applications of Differential Evolution in Power System Optimization”, U.K. Chakraborty (Ed.): *Advances in Differential Evolution*, SCI 143, pp. 257–273.

Landa Becerra R. and Coello Coello C. A., 2006, “Cultured differential evolution for constrained optimization,” *Comput. Methods Appl. Mech. Eng.*, vol. 195, nos. 33–36, pp. 4303–4322.

Landa Becerra R. and Coello Coello C. A., 2006, “Solving hard multiobjective optimization problems using ϵ -constraint with cultured differential evolution,” in *Proc. 9th Int. Conf. Parallel Problem Solving Nature*, LNCS 4193. Sep. 2006, pp. 543–552.

Liang C. H., Chung C. Y., Wong K. P., and Duan X. Z., 2007, “Parallel optimal reactive power flow based on cooperative co-evolutionary differential evolution and power system decomposition”, *IEEE Transactions on Power Systems*, Vol. 22, No. 1.

REFERENCES (CONTINUE)

Liang J. J., Qu B. Y., Suganthan P. N., Alfredo G. Hernández-Díaz, 2013, "Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session and Competition on Real-Parameter Optimization", Technical Report 201212, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, January 2013.

Liang J. J. and P. N. Suganthan, 2006, "Dynamic Multi-Swarm Particle Swarm Optimizer with a Novel Constraint-Handling Mechanism", Nanyang Technological University, Singapore 639798.

Liang J. J., Runarsson T. P., Mezura-Montes E., Clerc M., Suganthan P. N., Coello Coello C. A., Deb K., 2006, "Problem Definitions and Evaluation Criteria for the CEC 2006", Special Session on Constrained Real-Parameter Optimization, Technical Report, Nanyang Technological University, Singapore.

Li X., Tang K., Omidvar M. N., Yang Z., and Qin K., 2013, Benchmark Functions for the CEC'2013 Special Session and Competition on Large-Scale Global Optimization, Technical Report, 2013.

Liu J. and Lampinen J., 2005, "A fuzzy adaptive differential evolution algorithm," *Soft Computing—A Fusion of Foundations, Methodologies and Applications*, vol. 9, no. 6, pp. 448–462, 2005 [Online]. Available: <http://springerlink.metapress.com/index/10.1007/s00500-004-0363-x>

Lampinen J., 2001, "A Bibliography of Differential Evolution Algorithm," Technical Report, Lappeenranta University of Technology, Department of Information Technology, Laboratory of Information Processing.

Machairas V., Tsangrassoulis A., Axarli K., 2014, "Algorithms for optimization of building design: A review", *Renewable and Sustainable Energy Reviews* 31, 101-112.

REFERENCES (CONTINUE)

- Mawdesley M.J., Al-Jibouri S.H., and Yang H.**, 2002, “Genetic Algorithms for Construction Site Layout in Project Planning” *J. Constr. Engrg. And Mgmt.* 128, 418-426.
- Michalek J.J.**, 2011, *Interactive Layout Design Optimization*, Msc Thesis, University of Michigan.
- Michalewicz Z., Nazhiyath G., and Michalewicz M.**, 1996, A note on usefulness of geometrical crossover for numerical optimization problems. In L.J. Fogel, P.J. Angeline, and T. Back, editors, *Proc. of the 5th Annual Conference on Evolutionary Programming*, pages 305–312. MIT Press, Cambridge, MA.
- Michalewicz Z. and Schoenauer M.**, 1996, Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation*, 4(1):1–32.
- Minella G., Ruiz R., Ciavotta M.**, 2011, Restarted Iterated Pareto Greedy algorithm for multi-objective flow shop scheduling problems, *Computers & Operations Research* 38, 1521-1533.
- Neri F. and Tirronen V.**, 2010, “Recent advances in differential evolution: A review and experimental analysis,” *Artif. Intell. Rev.*, vol. 33, no. 1, pp. 61–106.
- Noman N. and Iba H.**, 2008, “Differential evolution for economic load dispatch problems”, *Electric Power Systems Research*, Vol. 78, Issue 8, Pages 1322-1331.
- Osyczka A. and Kundu S.**, 1995, “A new method to solve generalized multi-criteria optimization problems using the simple genetic algorithm,” *Structural Optimization*, Vol. 10, No. 2, pp. 94-99, 1995.
- Price K., Storn R., Lampinen J.**, 2005, *Differential Evolution: A Practical Approach to Global Optimization*, Springer-Verlag, Berlin.

REFERENCES (CONTINUE)

Price K. V., and Storn R., 1997, Differential evolution: A simple evolution strategy for fast optimization, Dr. Dobb's J., vol. 22, no. 4, pp. 18–24.

Qin A. K. and Suganthan P. N., 2005, “Self-adaptive Differential Evolution Algorithm for Numerical Optimization”, Proc. IEEE Congress on Evolutionary Computation.

Qin A. K., Huang V. L., and Suganthan P. N., 2009, “Differential evolution algorithm with strategy adaptation for global numerical optimization,” IEEE Trans. Evol. Comput., vol. 13, no. 2, pp. 398–417.

Qing A., 2003, “Electromagnetic inverse scattering of multiple two-dimensional perfectly conducting objects by the differential evolution strategy,” IEEE Transactions on. Antennas and Propagation, Vol. 51, Page(s) 1251–1262.

Rahnamayan S., Tizhoosh H. R., and Salama M. M. A., 2008, “Opposition based differential evolution,” IEEE Trans. Evol. Comput., vol. 12, no. 1, pp. 64–79.

Robic T. and Filipic B., 2005, “DEMO: Differential evolution for multi-objective optimization,” in Proc. 3rd Int. Conf. Evol. Multi-Criterion Optimization, LNCS 3410. 2005, pp. 520–533.

Sariyildiz I.S., Bittermann M.S., Ciftcioglu Ö., 2008, Multi-objective optimization in the construction industry. In: Proc. 5th Int. Conf. Innovation in Architecture, Engineering and Construction - AEC 2008, Antalya, Turkey 1-11.

Sayah S. and Zehar K., 2008, “Modified differential evolution algorithm for optimal power flow with non-smooth cost functions”, Energy Conversion and Management, Vol. 49, Issue 11, Pages 30363042.

Shipinspection **web** **site** **[online]**,
<http://www.shipinspection.eu/index.php/home/k2-item-view/item/572-metacentric-height>.

Soh C. K., and Yang J., 1996, ”Fuzzy controlled genetic algorithm search for shape optimization”, J. of Computing in Civil Engineering 10,143-150.

REFERENCES (CONTINUE)

Soh, C. K. and Yang, J., 1996, "Fuzzy controlled genetic algorithm search for shape optimization", *J. of Computing in Civil Engineering* 10 (1996) 143-150.

Storn R. and Price K. V., 1995 "Differential evolution: A simple and efficient adaptive scheme for global optimization over continuous spaces," ICSI, USA, Tech. Rep. TR-95-012, [Online]. Available: <http://icsi.berkeley.edu/~storn/litera.html>

Storn R. and Price K. V., 1997, Differential evolution: A simple and efficient heuristic for global optimization over continuous spaces, *J. Global Optimization*, vol. 11, no. 4, pp. 341–359.

Storn, R., (1999) "System Design by Constraint Adaptation and Differential Evolution," *IEEE Transactions on Evolutionary Computation*, vol. 3, pp. 22-34.

Strobbe T., Pauwels P., Verstraeten R., & R De Meyer, 2011, Metaheuristics in architecture, *Sustainable Construction and Design*.

Štumberger G., Seme S., Štumberger B., Polajžer B., and Dolinar D., 2008, "Determining magnetically nonlinear characteristics of transformers and iron core inductors by differential evolution", *IEEE Transactions on Magnetics*, Vol. 44, No. 6, Page(s) 1570 – 1573.

Su C-T. and Lee C-S., 2003, "Network reconfiguration of distribution systems using improved mixedinteger hybrid differential evolution", *IEEE Transactions on Power Delivery*, Vol. 18, No. 3, Page(s): 1022 – 1027.

Suganthan P. N., Hansen N., Liang J. J., Deb K., Chen Y.-P., Auger A. & Tiwari S., 2005, "Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization," Technical Report, Nanyang Technological University, Singapore, May 2005 and KanGAL Report #2005005, IIT Kanpur, India, 2005.

REFERENCES (CONTINUE)

Sum-Im T., Taylor G. A., Irving M.R., and Song Y.H., 2009, “Differential evolution algorithm for static and multistage transmission expansion planning”, *IET Generation, Transmission & Distribution*, Vol. 3, No. 4, Page(s) 365–384.

Toman M., Štumberger G., and Dolinar D., 2008, “Parameter identification of the Jiles–Atherton hysteresis model using differential evolution”, *IEEE Transactions on Magnetics*, Vol. 44, No. 6, Page(s) 1098 – 1101.

Tasgetiren M.F., Liang Y-C, Gencyilmaz G., Eker I., 2005, “A Differential evolution Algorithm for Continuous Function Optimization”, Department of Management, Fatih University, Turkey.

Takahama T. and Sakai S., 2006, "Constrained Optimization by the Constrained Differential Evolution with Gradient-Based Mutation and Feasible Elites," in *IEEE Congress on Evolutionary Computation* Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada, pp. 1-8.

Tasgetiren M. F., and Suganthan P. N., 2006, "A Multi-Populated Differential Evolution Algorithm for Solving Constrained Optimization Problems", Fatih University, 34500, Buyukcekmece, Istanbul, Turkey.

Tartar A., 2012, Floating Architecture Design Process Modelling Supported by Rule-Based Decision-Making, PhD Thesis, İstanbul Technical University.

Turrin M., Buelow P.V., Stouffs R., 2011, “Design explorations of performance driven geometry in architectural design using parametric modeling and genetic algorithms”, *Advanced Engineering Informatics* 25, 656-675.

Varadarajan M. and Swarup K.S., 2008, “Solving multi-objective optimal power flow using differential evolution”, *IET Generation, Transmission & Distribution*, Vol. 2, No. 5, pp. 720–730.

Wang B., and Chen J., 1996, “Application of genetic algorithm for the support location optimization of beams”, *Computers and Structures* 58, 797-800.

REFERENCES (CONTINUE)

Wang, B. and Chen, J., 1996, “Application of genetic algorithm for the support location optimization of beams”, *Computers and Structures* 58 (1996) 797-800.

Wang Z., Chung C.Y., Wong K.P., and Tse C.T., 2008, “Robust power system stabiliser design under multi-operating conditions using differential evolution”, *IET Generation, Transmission & Distribution*, Vol. 2, No. 5, pp. 690–700.

Watanabe et. al, 2004, *Very Large Floating Structures : Application, Analysis and Design*, National University of Singapore.

Wong S.S.Y., Chan K.C.C., 2009, *EvoArch: An Evolutionary Algorithm for Architectural Layout Design* *Computer-Aided Design* 41 (2009) 649–667.

Xia Q., Global optimization test problems. Available at <http://www.mat.univie.ac.at/neum/glopt/xia.txt>.

Xue F., Sanderson A. C., and Graves R. J., 2003, “Pareto-based multi-objective differential evolution,” in *Proc. Congr. Evol. Comput.*, vol. 2. pp. 862–869.

Yuan X., Wang L., Zhang Y., and Yuan Y., 2009, “A hybrid differential evolution method for dynamic economic dispatch with valve-point effects”, *Expert Systems with Applications*, 36, 2 Page(s) 4042-4048.

Yang G. Y., Dong Z. Y., and Wong K. P., 2008, “A Modified Differential Evolution Algorithm With Fitness Sharing for Power System Planning”, *IEEE Transactions on Power Systems*, Vol. 23, No. 2.

Yüksek G., Arıkan İ., 2009, "Marketing Floating Islands as a Destination: A Field Study for Determination of Potential Demand and Consumer Expectations of Floating Islands" *World Journal of Tourism, Leisure and Sports*, Volume 3, Issue 1.

Zamuda A., Brest J., Boskovic B., and Zumer V., 2007, “Differential evolution for multiobjective optimization with self adaptation,” in *Proc. Congr. Evol. Comput.*, pp. 3617–3624.

REFERENCES (CONTINUE)

Zhang J. and Sanderson A. C., 2009, “JADE: Adaptive differential evolution with optional external archive,” IEEE Trans. Evol. Comput., vol. 13, no. 5, pp. 945–958.

Zitzler E., Thiele L., Laumanns M., Fonseca CM., Grunert da Fonseca V., 2003, Performance assessment of multiobjective optimizers: an analysis and review. IEEE Transactions on Evolutionary Computation 2003; 7(2):117-32.

Zitzler E., Thiele L., 1999, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE Transactions on Evolutionary Computation 1999; 3(4):257-71.

(<http://eldar.mathstat.uoguelph.ca/dashlock/math4060/PDF/Reading2.pdf>)

CURRICULUM VITEA



Cemre Ugurlu was born in İzmir. She is currently a research assistant at the Department of Interior Architecture and Environmental Design at Yaşar University (Turkey), Architecture Faculty since December, 2013. She received her BSc degree in Industrial Engineering with the 3rd degree from Yaşar University Engineering Faculty in 2013 and she attended her MSc degree in Industrial Engineering from Yaşar University Engineering Faculty integrated with Architecture Program (Computational Design), respectively. She has taken the courses about various topics, including Computational Intelligence in Building Design, System Simulation, Optimization Models and Algorithms, Heuristic Optimization, Scheduling Theory, Probabilistic Analysis and Applied Stochastic Processes, Floating Cities, Advanced Computational Design, Design Studios and etc. During her Masters, she attended IEEE WCCI 2014 (World Congress on Computational Intelligence) in Beijing. She also presented two conference papers at IEEE CEC 2015 (Congress on Evolutionary Computation) in Sendai.

APPENDIX 1 CEC 2006 BENCHMARKS

Function1: g01 (Floudas C. and Pardalos P., 1987)

g01

Minimize

$$f(\vec{x}) = 5 \sum_{i=1}^4 x_i - 5 \sum_{i=1}^4 x_i^2 - \sum_{i=5}^{13} x_i$$

subject to:

$$g_1(\vec{x}) = 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \leq 0$$

$$g_2(\vec{x}) = 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \leq 0$$

$$g_3(\vec{x}) = 2x_2 + 2x_3 + x_{11} + x_{12} - 10 \leq 0$$

$$g_4(\vec{x}) = -8x_1 + x_{10} \leq 0$$

$$g_5(\vec{x}) = -8x_2 + x_{11} \leq 0$$

$$g_6(\vec{x}) = -8x_3 + x_{12} \leq 0$$

$$g_7(\vec{x}) = -2x_4 - x_5 + x_{10} \leq 0$$

$$g_8(\vec{x}) = -2x_6 - x_7 + x_{11} \leq 0$$

$$g_9(\vec{x}) = -2x_8 - x_9 + x_{12} \leq 0$$

where the bounds are $0 \leq x_i \leq 1$ ($i = 1, \dots, 9$), $0 \leq x_i \leq 100$ ($i = 10, 11, 12$) and $0 \leq x_{13} \leq 1$. The global minimum is at $\vec{x}^* = (1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 1)$ where six constraints are active (g_1, g_2, g_3, g_7, g_8 and g_9) and $f(\vec{x}^*) = -15$.

Function2: g02 (Kozieland S., Michalewicz Z., 1999)

g02

Minimize

$$f(\vec{x}) = - \left| \frac{\sum_{i=1}^n \cos^4(x_i) - 2 \prod_{i=1}^n \cos^2(x_i)}{\sqrt{\sum_{i=1}^n i x_i^2}} \right|$$

subject to:

$$g_1(\vec{x}) = 0.75 - \prod_{i=1}^n x_i \leq 0$$

$$g_2(\vec{x}) = \sum_{i=1}^n x_i - 7.5n \leq 0$$

where $n = 20$ and $0 < x_i \leq 10$ ($i = 1, \dots, n$). The global minimum $\vec{x}^* = (3.16246061572185, 3.12833142812967, 3.09479212988791, 3.06145059523469, 3.02792915885555, 2.99382606701730, 2.95866871765285, 2.92184227312450, 0.49482511456933, 0.48835711005490, 0.48231642711865, 0.47664475092742, 0.47129550835493, 0.46623099264167, 0.46142004984199, 0.45683664767217, 0.45245876903267, 0.44826762241853, 0.44424700958760, 0.44038285956317)$, the best we found is $f(\vec{x}^*) = -0.80361910412559$ (which, to the best of our knowledge, is better than any reported value), constraint g_1 is close to being active.

Function 3: g03 (Michalewicz Z., Nazhiyath G., and Michalewicz M., 1996)

g03

Minimize

$$f(\vec{x}) = -(\sqrt{n})^n \prod_{i=1}^n x_i$$

subject to:

$$h_1(\vec{x}) = \sum_{i=1}^n x_i^2 - 1 = 0$$

where $n = 10$ and $0 \leq x_i \leq 1$ ($i = 1, \dots, n$). The global minimum is at $\vec{x}^* = (0.31624357647283069, 0.316243577414338339, 0.316243578012345927, 0.316243575664017895, 0.316243578205526066, 0.31624357738855069, 0.316243575472949512, 0.316243577164883938, 0.316243578155920302, 0.316243576147374916)$ where $f(\vec{x}^*) = -1.00050010001000$.

Function 4: g04 (Himmelblau D., 1972)

g04

Minimize

$$f(\vec{x}) = 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141$$

subject to:

$$g_1(\vec{x}) = 85.334407 + 0.0056858x_2x_5 + 0.0006262x_1x_4 - 0.0022053x_3x_5 - 92 \leq 0$$

$$g_2(\vec{x}) = -85.334407 - 0.0056858x_2x_5 - 0.0006262x_1x_4 + 0.0022053x_3x_5 \leq 0$$

$$g_3(\vec{x}) = 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2 - 110 \leq 0$$

$$g_4(\vec{x}) = -80.51249 - 0.0071317x_2x_5 - 0.0029955x_1x_2 - 0.0021813x_3^2 + 90 \leq 0$$

$$g_5(\vec{x}) = 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4 - 25 \leq 0$$

$$g_6(\vec{x}) = -9.300961 - 0.0047026x_3x_5 - 0.0012547x_1x_3 - 0.0019085x_3x_4 + 20 \leq 0$$

where $78 \leq x_1 \leq 102$, $33 \leq x_2 \leq 45$ and $27 \leq x_i \leq 45$ ($i = 3, 4, 5$). The optimum solution is $\vec{x}^* = 78, 33, 29.9952560256815985, 45, 36.7758129057882073)$ where $f(\vec{x}^*) = -3.066553867178332e + 004$. Two constraints are active (g_1 and g_6).

Function 5: g05 (Hock W. and Schittkowski K., 1981)

g05

Minimize

$$f(\vec{x}) = 3x_1 + 0.000001x_1^3 + 2x_2 + (0.000002/3)x_2^3$$

subject to:

$$g_1(\vec{x}) = -x_4 + x_3 - 0.55 \leq 0$$

$$g_2(\vec{x}) = -x_3 + x_4 - 0.55 \leq 0$$

$$h_3(\vec{x}) = 1000 \sin(-x_3 - 0.25) + 1000 \sin(-x_4 - 0.25) + 894.8 - x_1 = 0$$

$$h_4(\vec{x}) = 1000 \sin(x_3 - 0.25) + 1000 \sin(x_3 - x_4 - 0.25) + 894.8 - x_2 = 0$$

$$h_5(\vec{x}) = 1000 \sin(x_4 - 0.25) + 1000 \sin(x_4 - x_3 - 0.25) + 1294.8 = 0$$

where $0 \leq x_1 \leq 1200$, $0 \leq x_2 \leq 1200$, $-0.55 \leq x_3 \leq 0.55$ and $-0.55 \leq x_4 \leq 0.55$. The best known solution [4] $\vec{x}^* = (679.945148297028709, 1026.06697600004691, 0.118876369094410433, -0.39623348521517826)$ where $f(\vec{x}^*) = 5126.4967140071$.

Function 6: g06 (Floudas C. and Pardalos P., 1987)

g06

Minimize

$$f(\vec{x}) = (x_1 - 10)^3 + (x_2 - 20)^3$$

subject to:

$$g_1(\vec{x}) = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \leq 0$$

$$g_2(\vec{x}) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \leq 0$$

where $13 \leq x_1 \leq 100$ and $0 \leq x_2 \leq 100$. The optimum solution is $\vec{x}^* = (14.09500000000000064, 0.8429607892154795668)$ where $f(\vec{x}^*) = -6961.81387558015$. Both constraints are active.

Function 7: g07 (Hock W. and Schittkowski K., 1981)

g07

Minimize

$$f(\vec{x}) = x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2 + 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45$$

subject to:

$$g_1(\vec{x}) = -105 + 4x_1 + 5x_2 - 3x_7 + 9x_8 \leq 0$$

$$g_2(\vec{x}) = 10x_1 - 8x_2 - 17x_7 + 2x_8 \leq 0$$

$$g_3(\vec{x}) = -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \leq 0$$

$$g_4(\vec{x}) = 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 \leq 0$$

$$g_5(\vec{x}) = 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \leq 0$$

$$g_6(\vec{x}) = x_1^2 + 2(x_2 - 2)^2 - 2x_1x_2 + 14x_5 - 6x_6 \leq 0$$

$$g_7(\vec{x}) = 0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 - 30 \leq 0$$

$$g_8(\vec{x}) = -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \leq 0$$

where $-10 \leq x_i \leq 10$ ($i = 1, \dots, 10$). The optimum solution is $\vec{x}^* = (2.17199634142692, 2.3636830416034, 8.77392573913157, 5.09598443745173, 0.990654756560493, 1.43057392853463, 1.32164415364306, 9.82872576524495, 8.2800915887356, 8.3759266477347)$ where $g07(\vec{x}^*) = 24.30620906818$ (The recorded results may suffer from rounding errors which may cause slight infeasibility sometimes in the best given solutions). Six constraints are active (g_1, g_2, g_3, g_4, g_5 and g_6).

Function 8: g08 (Kozieland S., Michalewicz Z., 1999)

g08

Minimize

$$f(\vec{x}) = -\frac{\sin^3(2\pi x_1) \sin(2\pi x_2)}{x_1^3(x_1 + x_2)}$$

subject to:

$$g_1(\vec{x}) = x_1^2 - x_2 + 1 \leq 0$$

$$g_2(\vec{x}) = 1 - x_1 + (x_2 - 4)^2 \leq 0$$

where $0 \leq x_1 \leq 10$ and $0 \leq x_2 \leq 10$. The optimum is located at $\vec{x}^* = (1.22797135260752599, 4.24537336612274885)$ where $f(\vec{x}^*) = -0.0958250414180359$.

Function 9: g09 (Hock W. and Schittkowski K., 1981)

g09

Minimize

$$f(\vec{x}) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7$$

subject to:

$$\begin{aligned} g_1(\vec{x}) &= -127 + 2x_1^2 + 3x_2^4 + x_3 + 4x_4^2 + 5x_5 \leq 0 \\ g_2(\vec{x}) &= -282 + 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 \leq 0 \\ g_3(\vec{x}) &= -196 + 23x_1 + x_2^2 + 6x_6^2 - 8x_7 \leq 0 \\ g_4(\vec{x}) &= 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \leq 0 \end{aligned}$$

where $-10 \leq x_i \leq 10$ for $(i = 1, \dots, 7)$. The optimum solution is $\vec{x}^* = (2.33049935147405174, 1.95137236847114592, -0.477541399510615805, 4.36572624923625874, -0.624486959100388983,$

$1.03813099410962173, 1.5942266780671519)$ where $f(\vec{x}^*) = 680.630057374402$. Two constraints are active (g_1 and g_4).

Function 10: g10 (Hock W. and Schittkowski K., 1981)

g10

Minimize

$$f(\vec{x}) = x_1 + x_2 + x_3$$

subject to:

$$\begin{aligned} g_1(\vec{x}) &= -1 + 0.0025(x_4 + x_6) \leq 0 \\ g_2(\vec{x}) &= -1 + 0.0025(x_5 + x_7 - x_4) \leq 0 \\ g_3(\vec{x}) &= -1 + 0.01(x_8 - x_5) \leq 0 \\ g_4(\vec{x}) &= -x_1x_6 + 833.33252x_4 + 100x_1 - 83333.333 \leq 0 \\ g_5(\vec{x}) &= -x_2x_7 + 1250x_5 + x_2x_4 - 1250x_4 \leq 0 \\ g_6(\vec{x}) &= -x_3x_8 + 1250000 + x_3x_5 - 2500x_5 \leq 0 \end{aligned}$$

where $100 \leq x_1 \leq 10000$, $1000 \leq x_i \leq 10000$ ($i = 2, 3$) and $10 \leq x_i \leq 1000$ ($i = 4, \dots, 8$). The optimum solution is $\vec{x}^* = (579.306685017979589, 1359.97067807935605, 5109.97065743133317, 182.01769963061534, 295.601173702746792, 217.982300369384632, 286.41652592786852, 395.601173702746735)$, where $f(\vec{x}^*) = 7049.24802052867$. All constraints are active (g_1, g_2 and g_3).

Function 11: g11 (Kozieland S., Michalewicz Z., 1999)

g11

Minimize

$$f(\vec{x}) = x_1^2 + (x_2 - 1)^2$$

subject to:

$$h(\vec{x}) = x_2 - x_1^2 = 0$$

where $-1 \leq x_1 \leq 1$ and $-1 \leq x_2 \leq 1$. The optimum solution is $\vec{x}^* = (-0.707036070037170616, 0.500000004333606807)$ where $f(\vec{x}^*) = 0.7499$.

Function 12: g12 (Kozieland S., Michalewicz Z., 1999)

g12

Minimize

$$f(\vec{x}) = -(100 - (x_1 - 5)^2 - (x_2 - 5)^2 - (x_3 - 5)^2)/100$$

subject to:

$$g(\vec{x}) = (x_1 - p)^2 + (x_2 - q)^2 + (x_3 - r)^2 - 0.0625 \leq 0$$

where $0 \leq x_i \leq 10$ ($i = 1, 2, 3$) and $p, q, r = 1, 2, \dots, 9$. The feasible region of the search space consists of 9^3 disjointed spheres. A point (x_1, x_2, x_3) is feasible if and only if there exist p, q, r such that the above inequality holds. The optimum is located at $\vec{x}^* = (5, 5, 5)$ where $f(\vec{x}^*) = -1$. The solution lies within the feasible region.

Function 13: g13 (Hock W. and Schittkowski K., 1981)

g13

Minimize

$$f(\vec{x}) = e^{x_1 x_2 x_3 x_4 x_5}$$

subject to:

$$h_1(\vec{x}) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0$$

$$h_2(\vec{x}) = x_2 x_3 - 5 x_4 x_5 = 0$$

$$h_3(\vec{x}) = x_1^3 + x_2^3 + 1 = 0$$

where $-2.3 \leq x_i \leq 2.3$ ($i = 1, 2$) and $-3.2 \leq x_i \leq 3.2$ ($i = 3, 4, 5$). The optimum solution is $\vec{x}^* = (-1.71714224003, 1.59572124049468, 1.8272502406271, -0.763659881912867, -0.76365986736498)$ where $f(\vec{x}^*) = 0.053941514041898$.

Function 14: g14 (Himmelblau D. M., 1972)

g14

Minimize

$$f(\vec{x}) = \sum_{i=1}^{10} x_i \left(c_i + \ln \frac{x_i}{\sum_{j=1}^{10} x_j} \right)$$

subject to:

$$h_1(\vec{x}) = x_1 + 2x_2 + 2x_3 + x_6 + x_{10} - 2 = 0$$

$$h_2(\vec{x}) = x_4 + 2x_5 + x_6 + x_7 - 1 = 0$$

$$h_3(\vec{x}) = x_3 + x_7 + x_8 + 2x_9 + x_{10} - 1 = 0$$

where the bounds are $0 < x_i \leq 10$ ($i = 1, \dots, 10$), and $c_1 = -6.089$, $c_2 = -17.164$, $c_3 = -34.054$, $c_4 = -5.914$, $c_5 = -24.721$, $c_6 = -14.986$, $c_7 = -24.1$, $c_8 = -10.708$, $c_9 = -26.662$, $c_{10} = -22.179$. The best known solution is at $x^* = (0.0406684113216282, 0.147721240492452, 0.783205732104114, 0.00141433931889084, 0.485293636780388, 0.000693183051556082, 0.0274052040687766, 0.0179509660214818, 0.0373268186859717, 0.0968844604336845)$ where $f(x^*) = -47.7648884594915$.

Function 15: g15 (Himmelblau D. M., 1972)

g15

Minimize

$$f(\vec{x}) = 1000 - x_1^2 - 2x_2^2 - x_3^2 - x_1x_2 - x_1x_3$$

subject to:

$$h_1(\vec{x}) = x_1^2 + x_2^2 + x_3^2 - 25 = 0$$

$$h_2(\vec{x}) = 8x_1 + 14x_2 + 7x_3 - 56 = 0$$

where the bounds are $0 \leq x_i \leq 10$ ($i = 1, 2, 3$). The best known solution is at $x^* = (3.51212812611795133, 0.216987510429556135, 3.55217854929179921)$ where $f(x^*) = 961.715022289961$.

Function 16: g16 (Himmelblau D. M., 1972)

g16

Minimize

$$f(\vec{x}) = 0.000117y_{14} + 0.1365 + 0.00002358y_{13} + 0.000001502y_{16} + 0.0321y_{12} \\ + 0.004324y_5 + 0.0001 \frac{c_{15}}{c_{16}} + 37.48 \frac{y_2}{c_{12}} - 0.0000005843y_{17}$$

subject to:

$$g_1(\vec{x}) = \frac{0.28}{0.72}y_5 - y_4 \leq 0$$

$$g_2(\vec{x}) = x_3 - 1.5x_2 \leq 0$$

$$g_3(\vec{x}) = 3496 \frac{y_2}{c_{12}} - 21 \leq 0$$

$$g_4(\vec{x}) = 110.6 + y_1 - \frac{62212}{c_{17}} \leq 0$$

$$g_5(\vec{x}) = 213.1 - y_1 \leq 0$$

$$g_6(\vec{x}) = y_1 - 405.23 \leq 0$$

$$g_7(\vec{x}) = 17.505 - y_2 \leq 0$$

$$g_8(\vec{x}) = y_2 - 1053.6667 \leq 0$$

$$g_9(\vec{x}) = 11.275 - y_3 \leq 0$$

$$g_{10}(\vec{x}) = y_3 - 35.03 \leq 0$$

$$g_{11}(\vec{x}) = 214.228 - y_4 \leq 0$$

$$g_{12}(\vec{x}) = y_4 - 665.585 \leq 0$$

$$g_{13}(\vec{x}) = 7.458 - y_5 \leq 0$$

$$g_{14}(\vec{x}) = y_5 - 584.463 \leq 0$$

$$g_{15}(\vec{x}) = 0.961 - y_6 \leq 0$$

$$g_{16}(\vec{x}) = y_6 - 265.916 \leq 0$$

$$g_{17}(\vec{x}) = 1.612 - y_7 \leq 0$$

$$g_{18}(\vec{x}) = y_7 - 7.046 \leq 0$$

$$g_{19}(\vec{x}) = 0.146 - y_8 \leq 0$$

$$g_{20}(\vec{x}) = y_8 - 0.222 \leq 0$$

$$g_{21}(\vec{x}) = 107.99 - y_9 \leq 0$$

$$g_{22}(\vec{x}) = y_9 - 273.366 \leq 0$$

$$g_{23}(\vec{x}) = 922.693 - y_{10} \leq 0$$

$$g_{24}(\vec{x}) = y_{10} - 1286.105 \leq 0$$

$$g_{25}(\vec{x}) = 926.832 - y_{11} \leq 0$$

$$g_{26}(\vec{x}) = y_{11} - 1444.046 \leq 0$$

$$g_{27}(\vec{x}) = 18.766 - y_{12} \leq 0$$

$$g_{28}(\vec{x}) = y_{12} - 537.141 \leq 0$$

$$g_{29}(\vec{x}) = 1072.163 - y_{13} \leq 0$$

$$\begin{aligned}
g_{30}(\vec{x}) &= y_{13} - 3247.039 \leq 0 \\
g_{31}(\vec{x}) &= 8961.448 - y_{14} \leq 0 \\
g_{32}(\vec{x}) &= y_{14} - 26844.086 \leq 0 \\
g_{33}(\vec{x}) &= 0.063 - y_{15} \leq 0 \\
g_{34}(\vec{x}) &= y_{15} - 0.386 \leq 0 \\
g_{35}(\vec{x}) &= 71084.33 - y_{16} \leq 0 \\
g_{36}(\vec{x}) &= -140000 + y_{16} \leq 0 \\
g_{37}(\vec{x}) &= 2802713 - y_{17} \leq 0 \\
g_{38}(\vec{x}) &= y_{17} - 12146108 \leq 0
\end{aligned}$$

where:

$$\begin{aligned}
y_1 &= x_2 + x_3 + 41.6 \\
c_1 &= 0.024x_4 - 4.62 \\
y_2 &= \frac{12.5}{c_1} + 12 \\
c_2 &= 0.0003535x_1^2 + 0.5311x_1 + 0.08705y_2x_1 \\
c_3 &= 0.052x_1 + 78 + 0.002377y_2x_1 \\
y_3 &= \frac{c_2}{c_3} \\
y_4 &= 19y_3 \\
c_4 &= 0.04782(x_1 - y_3) + \frac{0.1956(x_1 - y_3)^2}{x_2} + 0.6376y_4 + 1.594y_3 \\
c_5 &= 100x_2 \\
c_6 &= x_1 - y_3 - y_4 \\
c_7 &= 0.950 - \frac{c_4}{c_5} \\
y_5 &= c_6c_7 \\
y_6 &= x_1 - y_5 - y_4 - y_3 \\
c_8 &= (y_5 + y_4)0.995 \\
y_7 &= \frac{c_8}{y_1} \\
y_8 &= \frac{c_8}{3798} \\
c_9 &= y_7 - \frac{0.0663y_7}{y_8} - 0.3153 \\
y_9 &= \frac{96.82}{c_9} + 0.321y_1 \\
y_{10} &= 1.29y_5 + 1.258y_4 + 2.29y_3 + 1.71y_6 \\
y_{11} &= 1.71x_1 - 0.452y_4 + 0.580y_3 \\
c_{10} &= \frac{12.3}{752.3} \\
c_{11} &= (1.75y_2)(0.995x_1) \\
c_{12} &= 0.995y_{10} + 1998 \\
y_{12} &= c_{10}x_1 + \frac{c_{11}}{c_{12}} \\
y_{13} &= c_{12} - 1.75y_2 \\
y_{14} &= 3623 + 64.4x_2 + 58.4x_3 + \frac{146312}{y_9 + x_5}
\end{aligned}$$

$$c_{13} = 0.995y_{10} + 60.8x_2 + 48x_4 - 0.1121y_{14} - 5095$$

$$y_{15} = \frac{y_{13}}{c_{13}}$$

$$y_{16} = 148000 - 331000y_{15} + 40y_{13} - 61y_{15}y_{13}$$

$$c_{14} = 2324y_{10} - 28740000y_2$$

$$y_{17} = 14130000 - 1328y_{10} - 531y_{11} + \frac{c_{14}}{c_{12}}$$

$$c_{15} = \frac{y_{13}}{y_{15}} - \frac{y_{13}}{0.52}$$

$$c_{16} = 1.104 - 0.72y_{15}$$

$$c_{17} = y_9 + x_5$$

and where the bounds are $704.4148 \leq x_1 \leq 906.3855$, $68.6 \leq x_2 \leq 288.88$, $0 \leq x_3 \leq 134.75$, $193 \leq x_4 \leq 287.0966$ and $25 \leq x_5 \leq 84.1988$. The best known solution is at $x^* = (705.174537070090537, 68.5999999999999943, 102.899999999999991, 282.324931593660324, 37.5841164258054832)$ where $f(x^*) = -1.90515525853479$.

Function 17: g17 (Himmelblau D. M., 1972)

g17

Minimize

$$f(\vec{x}) = f(x_1) + f(x_2)$$

where

$$f_1(x_1) = \begin{cases} 30x_1 & 0 \leq x_1 < 300 \\ 31x_1 & 300 \leq x_1 < 400 \end{cases}$$

$$f_2(x_2) = \begin{cases} 28x_2 & 0 \leq x_2 < 100 \\ 29x_2 & 100 \leq x_2 < 200 \\ 30x_2 & 200 \leq x_2 < 1000 \end{cases}$$

subject to:

$$h_1(\vec{x}) = -x_1 + 300 - \frac{x_3x_4}{131.078} \cos(1.48477 - x_6) + \frac{0.90798x_3^2}{131.078} \cos(1.47588)$$

$$h_2(\vec{x}) = -x_2 - \frac{x_3x_4}{131.078} \cos((1.48477 + x_6) + \frac{0.90798x_4^2}{131.078} \cos(1.47588)$$

$$h_3(\vec{x}) = -x_5 - \frac{x_3x_4}{131.078} \sin((1.48477 + x_6) + \frac{0.90798x_4^2}{131.078} \sin(1.47588)$$

$$h_4(\vec{x}) = 200 - \frac{x_3x_4}{131.078} \sin((1.48477 - x_6) + \frac{0.90798x_3^2}{131.078} \sin(1.47588)$$

where the bounds are $0 \leq x_1 \leq 400$, $0 \leq x_2 \leq 1000$, $340 \leq x_3 \leq 420$, $340 \leq x_4 \leq 420$, $-1000 \leq x_5 \leq 1000$ and $0 \leq x_6 \leq 0.5236$. The best known solution is at $x^* = (201.784467214523659, 99.99999999999999005, 383.071034852773266, 420, -10.9076584514292652, 0.0731482312084287128)$ where $f(x^*) = 8853.53967480648$.

Function 18: g18 (Himmelblau D. M., 1972)

g18

Minimize

$$f(\vec{x}) = -0.5(x_1x_4 - x_2x_3 + x_3x_9 - x_5x_9 + x_5x_8 - x_6x_7)$$

subject to:

$$\begin{aligned} gg_1(\vec{x}) &= x_3^2 + x_4^2 - 1 \leq 0 \\ gg_2(\vec{x}) &= x_9^2 - 1 \leq 0 \\ gg_3(\vec{x}) &= x_5^2 + x_6^2 - 1 \leq 0 \\ gg_4(\vec{x}) &= x_1^2 + (x_2 - x_9)^2 - 1 \leq 0 \\ gg_5(\vec{x}) &= (x_1 - x_5)^2 + (x_2 - x_6)^2 - 1 \leq 0 \\ gg_6(\vec{x}) &= (x_1 - x_7)^2 + (x_2 - x_8)^2 - 1 \leq 0 \\ gg_7(\vec{x}) &= (x_3 - x_5)^2 + (x_4 - x_6)^2 - 1 \leq 0 \\ gg_8(\vec{x}) &= (x_3 - x_7)^2 + (x_4 - x_8)^2 - 1 \leq 0 \\ gg_9(\vec{x}) &= x_7^2 + (x_8 - x_9)^2 - 1 \leq 0 \\ gg_{10}(\vec{x}) &= x_2x_3 - x_1x_4 \leq 0 \\ gg_{11}(\vec{x}) &= -x_3x_9 \leq 0 \\ gg_{12}(\vec{x}) &= x_5x_9 \leq 0 \\ gg_{13}(\vec{x}) &= x_6x_7 - x_5x_8 \leq 0 \end{aligned}$$

where the bounds are $-10 \leq x_i \leq 10$ ($i = 1, \dots, 8$) and $0 \leq x_9 \leq 20$. The best known solution is at $x^* = (-0.657776192427943163, -0.153418773482438542, 0.323413871675240938, -0.946257611651304398, -0.657776194376798906, -0.753213434632691414, 0.323413874123576972, -0.346462947962331735, 0.59979466285217542)$ where $f(x^*) = -0.866025403784439$.

Function 19: g19 (Himmelblau D. M., 1972)

g19

Minimize

$$f(\vec{x}) = \sum_{j=1}^5 \sum_{i=1}^5 c_{ij}x_{(10+i)}x_{(10+j)} + 2 \sum_{j=1}^5 d_jx_{(10+j)}^3 - \sum_{i=1}^{10} b_ix_i$$

subject to:

$$g_j(\vec{x}) = -2 \sum_{i=1}^5 c_{ij}x_{(10+i)} - 3d_jx_{(10+j)}^2 - e_j + \sum_{i=1}^{10} a_{ij}x_i \leq 0 \quad j = 1, \dots, 5$$

where $\vec{b} = [-40, -2, -0.25, -4, -4, -1, -40, -60, 5, 1]$ and the remaining data is on Table 1. The bounds are $0 \leq x_i \leq 10$ ($i = 1, \dots, 15$). The best known solution is at $x^* = (1.66991341326291344e - 17, 3.95378229282456509e - 16, 3.94599045143233784, 1.06036597479721211e - 16, 3.2831773458454161, 9.99999999999999822, 1.12829414671605333e - 17, 1.2026194599794709e - 17, 2.50706276000769697e - 15, 2.24624122987970677e - 15, 0.370764847417013987, 0.278456024942955571, 0.523838487672241171, 0.388620152510322781, 0.298156764974678579)$ where $f(x^*) = 32.6555929502463$.

j	1	2	3	4	5
e_j	-15	-27	-36	-18	-12
c_{1j}	30	-20	-10	32	-10
c_{2j}	-20	39	-6	-31	32
c_{3j}	-10	-6	10	-6	-10
c_{4j}	32	-31	-6	39	-20
c_{5j}	-10	32	-10	-20	30
d_j	4	8	10	6	2
a_{1j}	-16	2	0	1	0
a_{2j}	0	-2	0	0.4	2
a_{3j}	-3.5	0	2	0	0
a_{4j}	0	-2	0	-4	-1
a_{5j}	0	-9	-2	1	-2.8
a_{6j}	2	0	-4	0	0
a_{7j}	-1	-1	-1	-1	-1
a_{8j}	-1	-2	-3	-2	-1
a_{9j}	1	2	3	4	5
a_{10j}	1	1	1	1	1

Data set for test problem g19

Function 20: g20 (Himmelblau D. M., 1972)

g20

Minimize

$$f(\vec{x}) = \sum_{i=1}^{24} a_i x_i$$

subject to:

$$g_i(\vec{x}) = \frac{(x_i + x_{(i+12)})}{\sum_{j=1}^{24} x_j + e_i} \leq 0 \quad i = 1, 2, 3$$

$$g_i(\vec{x}) = \frac{(x_{(i+3)} + x_{(i+15)})}{\sum_{j=1}^{24} x_j + e_i} \leq 0 \quad i = 4, 5, 6$$

$$h_i(\vec{x}) = \frac{x_{(i+12)}}{b_{(i+12)} \sum_{j=13}^{24} \frac{x_j}{b_j}} - \frac{c_i x_i}{40 b_i \sum_{j=1}^{12} \frac{x_j}{b_j}} = 0 \quad i = 1, \dots, 12$$

$$h_{13}(\vec{x}) = \sum_{i=1}^{24} x_i - 1 = 0$$

$$h_{14}(\vec{x}) = \sum_{i=1}^{12} \frac{x_i}{d_i} + k \sum_{i=13}^{24} \frac{x_i}{b_i} - 1.671 = 0$$

where $k = (0.7302)(530)(\frac{14.7}{40})$ and the data set is detailed on Table 2. The bounds are $0 \leq x_i \leq 10$ ($i = 1, \dots, 24$). The best known solution is at $x^* = (1.28582343498528086e - 18, 4.83460302526130664e - 34, 0, 0, 6.30459929660781851e - 18, 7.57192526201145068e - 34, 5.03350698372840437e - 34, 9.28268079616618064e - 34, 0, 1.76723384525547359e - 17, 3.55686101822965701e - 34, 2.99413850083471346e - 34, 0.158143376337580827, 2.29601774161699833e - 19, 1.06106938611042947e - 18, 1.31968344319506391e - 18, 0.530902525044209539, 0, 2.89148310257773535e - 18, 3.34892126180666159e - 18, 0, 0.310999974151577319, 5.41244666317833561e - 05, 4.84993165246959553e - 16)$. This solution is a little infeasible and no feasible solution is found so far.

i	a_i	b_i	c_i	d_i	e_i
1	0.0693	44.094	123.7	31.244	0.1
2	0.0577	58.12	31.7	36.12	0.3
3	0.05	58.12	45.7	34.784	0.4
4	0.2	137.4	14.7	92.7	0.3
5	0.26	120.9	84.7	82.7	0.6
6	0.55	170.9	27.7	91.6	0.3
7	0.06	62.501	49.7	56.708	
8	0.1	84.94	7.1	82.7	
9	0.12	133.425	2.1	80.8	
10	0.18	82.507	17.7	64.517	
11	0.1	46.07	0.85	49.4	
12	0.09	60.097	0.64	49.1	
13	0.0693	44.094			
14	0.0577	58.12			
15	0.05	58.12			
16	0.2	137.4			
17	0.26	120.9			
18	0.55	170.9			
19	0.06	62.501			
20	0.1	84.94			
21	0.12	133.425			
22	0.18	82.507			
23	0.1	46.07			
24	0.09	60.097			

Data set for test problem g20

Function 21: g21 (Epperly T.)

g21

Minimize

$$f(\vec{x}) = x_1$$

subject to:

$$g_1(\vec{x}) = -x_1 + 35x_2^{0.6} + 35x_3^{0.6} \leq 0$$

$$h_1(\vec{x}) = -300x_3 + 7500x_5 - 7500x_6 - 25x_4x_5 + 25x_4x_6 + x_3x_4 = 0$$

$$h_2(\vec{x}) = 100x_2 + 155.365x_4 + 2500x_7 - x_2x_4 - 25x_4x_7 - 15536.5 = 0$$

$$h_3(\vec{x}) = -x_5 + \ln(-x_4 + 900) = 0$$

$$h_4(\vec{x}) = -x_6 + \ln(x_4 + 300) = 0$$

$$h_5(\vec{x}) = -x_7 + \ln(-2x_4 + 700) = 0$$

where the bounds are $0 \leq x_1 \leq 1000$, $0 \leq x_2, x_3 \leq 40$, $100 \leq x_4 \leq 300$, $6.3 \leq x_5 \leq 6.7$, $5.9 \leq x_6 \leq 6.4$ and $4.5 \leq x_7 \leq 6.25$. The best known solution is at $x^* = (193.724510070034967, 5.56944131553368433e-27, 17.3191887294084914, 100.047897801386839, 6.68445185362377892, 5.99168428444264833, 6.21451648886070451)$ where $f(x^*) = 193.724510070035$.

Function 22: g22 (Epperly T.)

g22

Minimize

$$f(\vec{x}) = x_1$$

subject to:

$$g_1(\vec{x}) = -x_1 + x_2^{0.6} + x_3^{0.6} + x_4^{0.6} \leq 0$$

$$h_1(\vec{x}) = x_5 - 100000x_8 + 1 \times 10^7 = 0$$

$$h_2(\vec{x}) = x_6 + 100000x_8 - 100000x_9 = 0$$

$$h_3(\vec{x}) = x_7 + 100000x_9 - 5 \times 10^7 = 0$$

$$h_4(\vec{x}) = x_5 + 100000x_{10} - 3.3 \times 10^7 = 0$$

$$h_5(\vec{x}) = x_6 + 100000x_{11} - 4.4 \times 10^7 = 0$$

$$h_6(\vec{x}) = x_7 + 100000x_{12} - 6.6 \times 10^7 = 0$$

$$h_7(\vec{x}) = x_5 - 120x_2x_{13} = 0$$

$$h_8(\vec{x}) = x_6 - 80x_3x_{14} = 0$$

$$h_9(\vec{x}) = x_7 - 40x_4x_{15} = 0$$

$$h_{10}(\vec{x}) = x_8 - x_{11} + x_{16} = 0$$

$$h_{11}(\vec{x}) = x_9 - x_{12} + x_{17} = 0$$

$$h_{12}(\vec{x}) = -x_{18} + \ln(x_{10} - 100) = 0$$

$$h_{13}(\vec{x}) = -x_{19} + \ln(-x_8 + 300) = 0$$

$$h_{14}(\vec{x}) = -x_{20} + \ln(x_{16}) = 0$$

$$h_{15}(\vec{x}) = -x_{21} + \ln(-x_9 + 400) = 0$$

$$h_{16}(\vec{x}) = -x_{22} + \ln(x_{17}) = 0$$

$$h_{17}(\vec{x}) = -x_8 - x_{10} + x_{13}x_{18} - x_{13}x_{19} + 400 = 0$$

$$h_{18}(\vec{x}) = x_8 - x_9 - x_{11} + x_{14}x_{20} - x_{14}x_{21} + 400 = 0$$

$$h_{19}(\vec{x}) = x_9 - x_{12} - 4.60517x_{15} + x_{15}x_{22} + 100 = 0$$

where the bounds are $0 \leq x_1 \leq 20000$, $0 \leq x_2, x_3, x_4 \leq 1 \times 10^6$, $0 \leq x_5, x_6, x_7 \leq 4 \times 10^7$, $100 \leq x_8 \leq 299.99$, $100 \leq x_9 \leq 399.99$, $100.01 \leq x_{10} \leq 300$, $100 \leq x_{11} \leq 400$, $100 \leq x_{12} \leq 600$, $0 \leq x_{13}, x_{14}, x_{15} \leq 500$, $0.01 \leq x_{16} \leq 300$, $0.01 \leq x_{17} \leq 400$, $-4.7 \leq x_{18}, x_{19}, x_{20}, x_{21}, x_{22} \leq 6.25$. The best known solution is at $x^* = (236.430975504001054, 135.82847151732463, 204.818152544824585, 6446.54654059436416, 3007540.83940215595, 4074188.65771341929, 32918270.5028952882, 130.075408394314167, 170.817294970528621, 299.924591605478554, 399.258113423595205, 330.817294971142758, 184.51831230897065, 248.64670239647424, 127.658546694545862, 269.182627528746707, 160.000016724090955, 5.29788288102680571, 5.13529735903945728, 5.59531526444068827, 5.43444479314453499, 5.07517453535834395)$ where $f(x^*) = 236.430975504001$.

Function 23: g23 (Xia Q.)

g23

Minimize

$$f(\vec{x}) = -9x_5 - 15x_8 + 6x_1 + 16x_2 + 10(x_6 + x_7)$$

subject to:

$$g_1(\vec{x}) = x_9x_3 + 0.02x_6 - 0.025x_5 \leq 0$$

$$g_2(\vec{x}) = x_9x_4 + 0.02x_7 - 0.015x_8 \leq 0$$

$$h_1(\vec{x}) = x_1 + x_2 - x_3 - x_4 = 0$$

$$h_2(\vec{x}) = 0.03x_1 + 0.01x_2 - x_9(x_3 + x_4) = 0$$

$$h_3(\vec{x}) = x_3 + x_6 - x_5 = 0$$

$$h_4(\vec{x}) = x_4 + x_7 - x_8 = 0$$

where the bounds are $0 \leq x_1, x_2, x_6 \leq 300$, $0 \leq x_3, x_5, x_7 \leq 100$, $0 \leq x_4, x_8 \leq 200$ and $0.01 \leq x_9 \leq 0.03$. The best known solution is at $x^* = (0.00510000000000259465, 99.99470000000000514, 9.01920162996045897e - 18, 99.99990000000000535, 0.000100000000027086086, 2.75700683389584542e - 14, 99.999999999999574, 2000.0100000100000100008)$ where $f(x^*) = -400.055099999999584$.

Function 24: g24 (Floudas C., 1999)

g24

Minimize

$$f(\vec{x}) = -x_1 - x_2$$

subject to:

$$g_1(\vec{x}) = -2x_1^4 + 8x_1^3 - 8x_1^2 + x_2 - 2 \leq 0$$

$$g_2(\vec{x}) = -4x_1^4 + 32x_1^3 - 88x_1^2 + 96x_1 + x_2 - 36 \leq 0$$

where the bounds are $0 \leq x_1 \leq 3$ and $0 \leq x_2 \leq 4$. The feasible global minimum is at $x^* = (2.329520197477623, 1.7849307411774)$ where $f(x^*) = -5.50801327159536$. This problem has a feasible region consisting on two disconnected sub-regions.

APPENDIX 2 MULTI-OBJECTIVE TEST FUNCTIONS

Function 1: CONSTR (Deb et. Al, 2001)

Minimize

$$F_1(\mathbf{x}) = x_1$$
$$F_2(\mathbf{x}) = \frac{1 + x_2}{x_1}$$

Subject to:

$$g_1(\mathbf{x}) = x_2 + 9x_1 \geq 6 \quad x_1 \in [0.1, 1]$$
$$g_2(\mathbf{x}) = -x_2 + 9x_1 \geq 1 \quad x_2 \in [0, 5]$$

Function 2: SRN (Jiménez F. et. al, 2002)

Minimize

$$F_1(\mathbf{x}) = 2 + (x_1 - 2)^2 + (x_2 - 1)^2$$
$$F_2(\mathbf{x}) = 9x_1 - (x_2 - 1)^2$$

Subject to:

$$g_1(\mathbf{x}) = x_1^2 + x_2^2 \leq 225 \quad x_i \in [-20, 20]$$
$$g_2(\mathbf{x}) = x_1 - 3x_2 + 10 \leq 0 \quad i = 1, 2$$

Function 3: OSY (Osyezka & Kundu, 1995)

Minimize

$$F_1(\mathbf{x}) = - \left[\begin{array}{l} 25(x_1 - 2)^2 + (x_2 - 2)^2 + (x_3 - 1)^2 \\ + (x_4 - 4)^2 + (x_5 - 1)^2 \end{array} \right]$$
$$F_2(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2$$

Subject to:

$$g_1(\mathbf{x}) = x_1 + x_2 - 2 \geq 0$$

$$g_2(\mathbf{x}) = 6 - x_1 - x_2 \geq 0$$

$$g_3(\mathbf{x}) = 2 - x_2 + x_1 \geq 0$$

$$g_4(\mathbf{x}) = 2 - x_1 + 3x_2 \geq 0$$

$$g_5(\mathbf{x}) = 4 - (x_3 - 3)^2 - x_4 \geq 0$$

$$g_6(\mathbf{x}) = (x_5 - 3)^2 + x_6 - 4 \geq 0$$

$$x_i \in [0,10]$$

$$i = 1,2,6$$

$$x_j \in [1,5]$$

$$j = 3,5$$

$$x_4 \in [0,6]$$

APPENDIX 3 ASSUMPTIONS FOR APPLICATION I

Assumptions are given below.

- Kitchen has a rectangular shape.
- Kitchen can be rotated and can be changed its dimensions and coordinates.
- CPT is 3 but it can be changed.
- Tables are created by generating grids on the restaurant surface.
- ORN is 0.503.
- ORA is 0.204.
- Tables which have higher OR are represented by circles.
- Tables which have lower OR are represented with points and they need artificial light. We assumed each tables have a light bulb at top of the table. Each bulb consumes 0.06 Kw.
- Kwh per m² for heating energy is 0.023.
- Total energy consumption will be multiplied with 0.36 TL/Kwh.
- WHM assumed as 120. Therefore, restaurant serves 6 hours per day.
- For calculating staff cost, one staff can walk 600 meters per hour.
- Unit cost of a staff per month is 1500 TL. (TL: Turkish Lira)
- For calculating revenue, CST is 2.026 hour, PR is 10 TL.
- Kitchen cost per m² is 1000 TL.
- Diner cost per m² is 2500 TL.
- Glass cost per m² is 224 TL.
- Unit cost of tables is 15 TL.
- Wall cost per m² is 480 TL.
- Number of orders per hour is 16.
- In constraint (1), we assume there is a constant that determines the kitchen space requirements relation with number of tables.
- Mean of proximity term between tables and windows defined by being closer to window less than 7.614 meters value.
- Service point is travelling on boundaries of the kitchen while optimization process. While modelling part, we considered this point must not exceed the boundary line.