DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF SOCIAL SCIENCES
DEPARTMENT OF BUSINESS ADMINISTRATION
BUSINESS ADMINISTRATION PROGRAM

DOCTORAL THESIS

Doctor of Philosophy (PhD)

NATURE-INSPIRED EVOLUTIONARY ALGORITHMS AND A MODEL PROPOSAL

Gülin Zeynep ÖZTAŞ

Supervisor

Prof. Dr. Sabri ERDEM

İZMİR - 2021

THESIS APPROVAL PAGE

DOCTORAL THESIS
APPROVAL PAGE

University	: Dokuz Eylul University
Graduate School	: Graduate School of Social Sciences
Name and Surname	: GÜLİN ZEYNEP ÖZTAŞ
Title of the Thesis	: Nature-Inspired Evolutionary Algorithms and A Model Proposal
Defence Date	: 16/06/2021
Supervisor	: Prof. Sabri ERDEM

EXAMINING COMMITTE MEMBERS

Title,Name and Surname	University	Signature
Prof. Sabri ERDEM	- Dokuz Eylül Üniversitesi	.....
Assoc.Prof.Güzin ÖZDAĞOĞLU	- Dokuz Eylül Üniversitesi	.....
Prof. İpek DEVECİ KOCAKOÇ	- Dokuz Eylül Üniversitesi	.....
Prof. Şevkinaz GÜMÜŞOĞLU	- Yaşar Üniversitesi	.....
Assoc.Prof. Esra AYTAÇ ADALI	- Pamukkale Üniversitesi	.....

Unanimity ( )
Majority of votes ( )

The thesis titled as "**Nature-Inspired Evolutionary Algorithms and A Model Proposal** " prepared and presented by GÜLİN ZEYNEP ÖZTAŞ is accepted and approved.

Prof. Dr. Asuman ALTAY
Müdür

DECLARATION

I hereby declare that this doctoral thesis titled as “Nature-Inspired Evolutionary Algorithms and A Model Proposal” has been written by myself in accordance with the academic rules and ethical conduct. I also declare that all materials benefited in this thesis consist of the mentioned resourses in the reference list. I verify all these with my honour.

Date

16/06/2021

Gülin Zeynep ÖZTAŞ

ABSTRACT

Doctoral Thesis

Doctor of Philosophy (PhD)

Nature-Inspired Evolutionary Algorithms and A Model Proposal

Gülin Zeynep ÖZTAŞ

Dokuz Eylül University

Graduate School of Social Sciences

Department of Business Administration

Business Administration Program

This study introduces a new population-based evolutionary computing model for solving linear/nonlinear continuous unconstrained/constrained optimization problems. The proposed model includes two optimization algorithms. The first one is an initialization algorithm that provides adaptive initial solutions, to some extent, reducing the diversity of randomness in the initialization of the algorithms for problems that may have many local optimums. The prominent feature of the algorithm is the ability to narrow the search space adaptively without falling into local optimums and changing the nature of the problem. Unlike simple random approaches, the proposed algorithm escapes from inadvertently removing the global optimum in multi-modal problems. In terms of time and performance, the initialization algorithm doesn't add additional burden, on the contrary, it contributes to the problem-solving procedure.

The second proposed algorithm called Repulsive Forces Optimization (REF) depends on Newton's General Gravity Law and Coulomb's Law. Different from the first algorithm, the REF algorithm aims to reach optimum-like solutions by constraint-handling abilities. REF algorithm assumes that likely charged particles in a bounded space are possible solution points. The forces between the particle and its neighbors make the particle moved to a new location where a better solution may exist. The repulsive structure of the particles could be considered as the mimics of Coulomb's Law. Furthermore, Tabu Search

Algorithm and Elitism selection approach inspire the memory usage of the proposed algorithm. The inspirations of the REF algorithm are determined to create the best combination of features that provides better results. Besides, this algorithm is structured on the principle of multiplicative penalty approach that considers satisfaction rates and the total deviations of constraints as well as objective function value for constraint handling. For this reason, it can handle continuous constrained problems very well.

The performances of the algorithms are evaluated with unconstrained/bounded optimization benchmarks and engineering design problems that belong to the most commonly used cases by evolutionary optimization researchers. In addition, an economic dispatch problem is also applied for benchmarking. It is concluded that the initialization algorithm converges much better than random solutions and it is applicable for further studies focusing on better initial solutions that guide reaching an optimal solution. Experimental results of real-world problems show that the proposed algorithms produce satisfactory results compared to the methods published in the literature.

Keywords: Metaheuristics, Physics-based algorithms, Repulsive Forces, Random Search, Benchmark Problems

ÖZET

Doktora Tezi

Doğadan Esinlenen Evrimsel Algoritmalar ve Bir Model Önerisi

Gülin Zeynep ÖZTAŞ

Dokuz Eylül Üniversitesi

Sosyal Bilimler Enstitüsü

İngilizce İşletme Anabilim Dalı

İngilizce İşletme Yönetimi Programı

Bu çalışma, doğrusal/doğrusal olmayan sürekli kısıtsız/kısıtlı optimizasyon problemlerini çözmek için yeni bir popülasyon tabanlı evrimsel hesaplama modeli sunmaktadır. Önerilen model, iki optimizasyon algoritması içermektedir. Bunlardan ilki, birçok yerel optimuma sahip problemler için algoritmaların başlatılmasında rastgelelik çeşitliliğini bir dereceye kadar azaltan uyarlamalı başlangıç çözümleri sağlayan bir başlatma algoritmasıdır. Algoritmanın öne çıkan özelliği, yerel optimumlara takılmadan ve sorunun doğasını değiştirmeden arama alanını uyarlamalı olarak daraltma yeteneğidir. Basit rastgele yaklaşımlardan farklı olarak, önerilen algoritma çok modlu problemlerde global optimum noktasının yanlışlıkla ortadan kaldırılmasından kaçınabilmektedir. Başlangıç algoritması zaman ve performans açısından ek bir yük getirmek yerine, problemin çözülmesine katkı sağlamaktadır.

İtici Kuvvetler optimizasyonu (REF) olarak adlandırılan ikinci algoritma, Newton'un Genel Yerçekimi Yasasına ve Coulomb Yasasına dayanmaktadır. REF algoritması, önerilen ilk algoritmadan farklı olarak, kısıtlama becerileriyle optimum benzeri çözümlere ulaşmayı amaçlamaktadır. Önerilen algoritma, sınırlı bir uzaydaki muhtemel yüklü parçacıkların olası çözüm noktaları olduğunu varsaymaktadır. Parçacıklar ve komşuları arasındaki kuvvetler, parçacığı daha iyi bir çözümün bulunabileceği yeni bir konuma hareket ettirir. Parçacıkların itici yapısı Coulomb Yasasının taklidi olarak düşünülebilir. Ayrıca, Tabu Arama Algoritması ve Elitizm seçim yaklaşımı önerilen algoritmanın hafıza

kullanımına ilham vermektedir. Algoritmanın ilham kaynakları, daha iyi sonuçlar sağlayan en iyi özelliklerin bir kombinasyonunu oluşturmak için belirlenmiştir. Ayrıca, bu algoritma, amaç fonksiyonu değerinin yanısıra kısıtları sağlama oranlarını ve kısıtlamaların toplam sapmalarını da dikkate alan çarpımsal ceza yaklaşımı ilkesi üzerine yapılandırılmıştır. Bu nedenle sürekli kısıtlı problemleri çok iyi bir şekilde çözme yeteneğine sahiptir.

Algoritmanın performansı, evrimsel optimizasyon araştırmacıları tarafından en sık kullanılan kısıtsız / sınırlı optimizasyon problemleri ve mühendislik tasarım problemleri ile test edilmiştir. Ek olarak, kıyaslama için ekonomik sevkiyat problemi de uygulanmıştır. Başlangıç algoritmasının, rastgele çözümlerden çok daha iyi yakınsadığı ve en uygun çözüme ulaşmayı hedefleyen algoritmalar için de başlangıç algoritması olarak kullanılabileceği sonucuna varılmıştır. Gerçek dünya problemlerinin deneysel sonuçları ise önerilen algoritmaların literatürde yayınlanan yöntemlere göre tatmin edici sonuçlar verdiğini göstermektedir.

Anahtar Kelimeler: Meta-Sezgisel Algoritmalar, Fizik Temelli Algoritmalar, İtici Kuvvetler, Rastgele Arama, Karşılaştırma Problemleri

NATURE-INSPIRED EVOLUTIONARY ALGORITHMS AND A MODEL PROPOSAL

CONTENTS

CHAPTER TWO
LITERATURE REVIEW OF PHYSIC BASED ALGORITHMS

CHAPTER THREE

THE PROPOSED ALGORITHM

CHAPTER FOUR

EXPERIMENTAL STUDIES

ABBREVIATIONS

ABC	Artificial Bee Colony Algorithm
APO	Artificial Physics Optimization
BB-BC	Big Bang- Big Crunch
BH	Black Hole Algorithm
BOA	Butterfly Optimization Algorithm
CBO	Colliding Bodies Optimization
CCM	Canonical Coordinates Method
CFO	Central Force Optimization
CGO	Chaos Game Optimization
CGWO	Chaotic Grey Wolf Optimizer
CHPED	Combined Heat and Power Economic Dispatch
C-PSO	Co-evolutionary Particle Swarm Optimization
CPU	Central Processing Unit
CSA	Cuckoo Search Algorithm
CSS	Charged System Search
EABO	Enhanced adaptive butterfly optimization algorithm
EC	Evolutionary Computation
EFO	Electromagnetic Field Optimization
EM	Electromagnetism-like Mechanism
EO	Equilibrium Optimizer
ES	Evolution Strategies
FA	Firefly Algorithm
FES	Function Evalutations
GBMO	Gases Brownian Motion Optimization
GBO	Gradient-Based Optimizer
GbSA	Galaxy-based Search Algorithm
GeneAS	Genetic Adaptive Search
GFA	Gravitation Field Algorithm
GIO	Gravitational Interaction Optimization
GRSA	General Relativity Search Algorithm

GSA	Gravitational Search Algorithm
GTO	Group Teaching Optimization Algorithm
GWO	Grey Wolf Optimizer
IHS	Improved Harmony Search Algorithm
H-FPA	Hybrid Flower Pollination Algorithm
H-GSA-GA	Hybrid GSA-GA Algorithm
HGSO	Henry Gas Solubility Optimization
HO	Hysteretic Optimization
H-PSO-GA	Hybrid PSO-GA Algorithm
HS	Harmony Search Algorithm
HTS	Heat Transfer Search
IGWO	Improved Grey Wolf Optimizer
IMO	Ions Motion Algorithm
ISA	Interior Search Algorithm
KGMO	Kinetic Energy of Gas Molecules
KKO	Kho-Kho Optimization Algorithm
LRO	Light Ray Optimization
MADS	Mesh Adaptive Direct Search Algorithm
MCSS	Magnetic Charged System Search
MOA	Magnetic Optimization Algorithm
MOPM	Modified Oracle Penalty Method
MPA	Marine Predators Algorithm
MTCA	Modified T-Cell Algorithm
MUPE	Multiplicative Penalty based Method
MVO	Multi-Verse Optimizer
NP-hard	Non-deterministic Polynomial-time Hardness
N2F	Nuclear Fission-Nuclear Fusion Algorithm
OIO	Optics Inspired Optimization
P	Polynomial
PA	Pathfinder algorithm
PCA	Particle Collision Algorithm
PPA	Plant Propagation Algorithm

PSO	Particle Swarm Optimization
PSO-GA	Advanced particle swarm assisted genetic algorithm
QC	Quantum Computing
Q - GSA	Quantum Gravitational Search Algorithm
Q-ABC	Quantum Artificial Bee Colony
Q-ACO	Quantum Ant Colony Optimization
Q-AIS	Quantum Artificial Immune System
Q-BFA	Quantum Bacterial Foraging Algorithm
Q-BO	Quantum Bat Optimization
Q-BWPA	Quantum Binary Wolf Pack Algorithm
Q-CA	Quantum Cultural Algorithm
Q-CSA	Quantum Cuckoo Search Algorithm
Q-DE	Quantum Differential Evolution
Q-DSA	Quantum Dolphin Swarm Algorithm
Q-EA	Quantum Evolutionary Algorithm
Q-EM	Quantum EM
Q-FA	Quantum Firefly Algorithm
Q-GA	Quantum Genetic Algorithm
Q-GSO	Quantum Glowworm Swarm Optimization
Q-HS	Quantum Harmony Search
Q-ICA	Quantum Immune clonal algorithm
Q-PSO	Quantum Particle Swarm Optimization
Q-SWA	Quantum Sperm Whale Algorithm
Q-TLBO	Quantum Teaching-Learning-Based Optimization
Q-TS	Quantum Tabu Search
REF	Repulsive Forces Optimization
RO	Ray Optimization
RSAB	Random Search with Adaptive Boundaries
SA	Simulated Annealing
SAP	Self-Adaptive Penalty
SBO	Smell Bees Optimization Algorithm
SCO	Social Cognitive Optimization

SMA	Slime Mould Algorithm
SOA	Seagull Optimization Algorithm
SpOA	Spiral Optimization Algorithm
SRO	Search and Rescue Optimization Algorithm
TEO	Thermal Exchange Optimization
TLMPA	Teaching-Learning based Marine Predator Algorithm
TS	Tabu Search

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

INTRODUCTION

Optimization takes an important place in terms of providing solutions to the problems encountered in business life such as business activities, engineering problems, industrial design problems under some restrictions as time, money, resource. In the literature, there are different algorithms that can be defined as a set of sequential operations to solve problems. However, these algorithms vary according to the nature of the problem. Depending on the randomness in algorithms, optimization techniques can be classified as deterministic and stochastic algorithms. Deterministic algorithms include algebraic methods such as the Newtonian approach, Gauss eliminations, and gradient-based methods. Although these methods are good at solving smooth unimodal problems, they cannot handle discontinuity in objective functions. In such cases, gradient-free methods are preferred. However, according to the computational complexity of the problems, more sophisticated algorithms are required. Today, in parallel with the developments in computer technology, iterative and divide-and-conquer-based methods have gained more importance. Especially real-world problems which belong to NP-Hard class cannot be solved in a polynomial time and stochastic algorithms come to exist to provide approximate solutions.

Artificial intelligence (AI) attracts optimization researchers' attention for developing sophisticated heuristics. In terms of this computational technology, AI methods use local search (i.e. direct search) techniques very efficiently in a short time. But there are still some new developments in heuristics and AI methods for global search.

Stochastic algorithms comprise heuristic and metaheuristic algorithms. Although the principles are similar for both sub-classes, heuristic algorithms are defined as problem-specific algorithms whereas metaheuristic algorithms are more general kinds of stochastic algorithms and are generally structured on various metaphors. Especially in the last decade, the number of new metaheuristic algorithms based on metaphors has exploded. Most of the researchers have focused on nature-based algorithms inspired by interactions of living and non-living objects. The main idea behind this is a belief that nature solves its problem instinctively like finding the shortest path between foods and nests for ants and bees. In addition, non-living objects

also behave toward finding their better state naturally as in free-falling bodies, repulsion of same polarized magnets, and steady-state in the cooling process of materials from high temperature. In the state-of-the-art, many algorithms that imitate these behaviors and interactions for solving optimization problems in applied and social sciences such as traveling salesman, assignment, transportation, scheduling, layout, conflict resolution, optimum policy-making, portfolio optimization, etc. All these points of view come up with a new field of computation and optimization, namely evolutionary algorithms.

According to Wolpert and Macready (1995) proved with the "No Free Lunch Theorem", there cannot be an algorithm that is appropriate for all problems. In other words, there is always a better algorithm than the existing ones. Although many researchers criticize that plenty of metaheuristics have similarities although they introduce different metaphors, it would be better to develop algorithms that provide more "optimal-like" solutions without trapping the "novelty" concept.

In line with the innovations in information technologies, the main motivation for this thesis study is to provide a basis for the forthcoming studies that may be breakthroughs in the field of optimization. In this thesis, we develop a hybrid algorithm that includes two algorithms called Random Search with Adaptive Boundaries (RSAB), Repulsive Force Optimization (REF) Algorithm. The first one is structured as a generic method that can be applied in the initialization stage of any algorithm for optimization problems and it depends on updating given upper and/or lower boundaries dynamically according to parameters. The outstanding feature of the first algorithm is the ability to reach better initial solutions by reducing the diversity of pure randomness, to some extent, for continuous unconstrained/bounded and constrained nonlinear optimization problems that may have many local optimums. Unlike conventional random search algorithms, the proposed algorithm can eliminate the risk of missing the global optimum while narrowing search space. We assert that the RSAB algorithm has advanced competencies that relieve the workload of the global optimization algorithms and shorten the time to find solutions. The REF algorithm which can be defined as the main algorithm is based on physics to provide the best-known solutions for constraint handling. This algorithm assumes that the particles that are likely charged repel each other considering their neighbors to reach new locations

where better solutions may exist. The calculation of repulsive forces mimics Coulomb's Law and the movement is calculated according to the Momentum Law. Furthermore, Tabu Search and Elitism selection are also inspired in terms of the memory structure of the algorithm. This algorithm handles constraints with the help of a multiplicative penalty approach that considers satisfaction rate and the deviations of constraints besides objective function value.

The main purpose of this thesis is not to duplicate existing algorithms through a new metaphor, but to provide an algorithm that reaches the best-known solution values. However, to reach better solutions, it is very important to determine the best combination of features inspired by the literature or metaphors. Therefore, achieving better results in a specific problem type is the primary goal, regardless of metaphors and similarities. It is worth mentioning that, the proposed algorithms in this thesis are revised versions of the algorithm introduced by Erdem (2007). However, many improvements and modifications have been implemented. In the related sections, all details of modifications are explained about the proposed algorithms. With these prominent features, the proposed algorithm will contribute to the literature.

The framework of this thesis study consisting of four chapters is as follows: In the first chapter general information about optimization is provided by summarizing the background of the metaheuristic algorithms. In the second chapter, all physics-based algorithms in the literature are classified and their highlights are mentioned briefly. In the third chapter, the proposed algorithms are explained in detail. Finally, in the fourth chapter, the experimental studies including continuous unconstrained/bounded benchmark problems, engineering design problems, and an economic dispatch problem as a business case are reported.

CHAPTER ONE
METAHEURISTICS

1.1. OPTIMIZATION

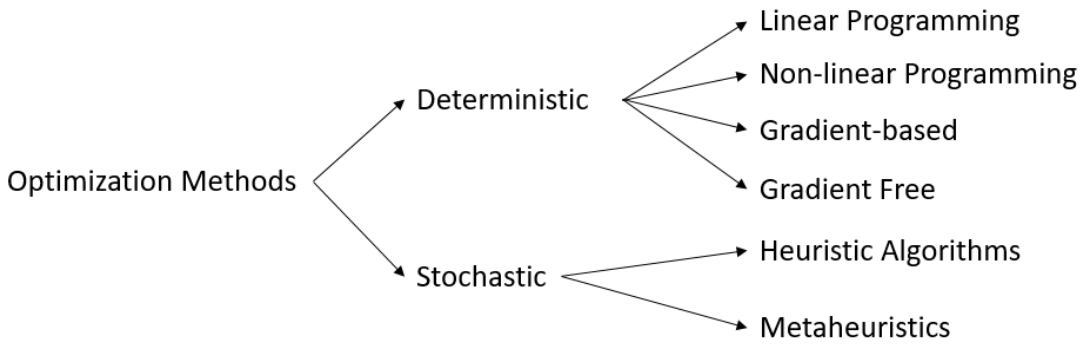
Optimization is a method for maximizing or minimizing a function by choosing the variables systematically (Kaveh, 2014: 1). Indeed, optimization aims at reaching the best possible solution under defined circumstances. The general model structure for optimization problems is given as follows (Astolfi, 2006: 3):

$$\begin{gathered} \text{[Min or Max] } f(x) \\ \text{subject to the constraints} \\ g_j(x) \leq 0 \text{ for } j=1, 2, \ldots, m \\ l_k(x) = 0 \text{ for } k = 1, 2, \ldots, p \end{gathered} \tag{1.1}$$

where $x = [x_1, x_2, \ldots, x_n]$ is a possible solution set tried to be found by minimizing or maximizing an objective function $f(x)$. x vector that includes decision variables can form continuous, discrete, or mixed decision space. This model also provides $g_j(x)$ for inequality and $l_k(x)$ for equality constraints. However, if the total number of constraints is zero, we can talk about an unconstrained optimization problem. Moreover, according to the function design of the $f(x)$, the algorithm for the solution of that model can change.

The optimization algorithms can be classified according to different principles. Sahab et al. (2013) gathered various classifications in terms of the number and design of objective functions, variables, constraints, landscape, determinacy. Astolfi (2006) also summarized the classification ways of optimization as the existence of constraints, the nature of equations, and admissible values of design variables. However, the classification of the optimization algorithms is carried out in terms of solution methods as shown in Figure 1 as well.

Figure 1: The Classification of Optimization Techniques



Source: Yang, 2010a: 15-20

The main difference between deterministic and stochastic algorithms lies in the solutions obtained from different runs (Siddique and Adeli, 2015a: 707). This means that in deterministic algorithms there is no randomness in generating new solutions. For that reason, while exploitation ability enhances, exploration capability remains inadequate (Yang, 2018: 5). Most conventional algorithms are deterministic and based on mathematical programming methods like linear programming, convex programming, integer programming, quadratic programming, dynamic programming, branch and bound methods (Nesmachnow, 2014: 321). They provide accurate solutions for problems in a continuous space. On the other hand, gradient-based algorithms are good at optimizing smooth unimodal problems by using function values and their derivatives. Newton-Raphson algorithm, steepest descent method, inexact line search can be given as examples for gradient-based method (Hendrix and Toth, 2010: 106). However, when the objective function has a discontinuity, it is not able to calculate the derivatives. Therefore, for these cases, it would be better to prefer gradient-free algorithms. As an example of gradient-free methods, Hooke - Jeeves pattern search and Nelder-Mead downhill simplex can be given (Yang, 2010a: 21). Although the ability of deterministic algorithms is too restricted in terms of problem types, they all provide exact optimal solutions to those problems every time. However, they are useless and time-consuming for real-world problems (Nesmachnow, 2014: 321).

1.1.1. Stochastic Optimization

Stochastic optimization class covers the algorithms and techniques that include some degree of randomness in themselves (Luke, 2011: 7). The hardest problem in the world can be described as an NP-hard (non-deterministic polynomial-time hardness) problem. Most real-world problems are NP-hard which cannot be solved in polynomial time (Caserta and Voß, 2009: 1). Those types of problems can only be solved by using stochastic algorithms. In stochastic optimization also called unconventional optimization, an objective function is tried to be maximized or minimized by improving possible solutions iteratively (Brownlee, 2011: 15). Stochastic algorithms can explore wide ranges at the same time without trapping out local optimums (Siddique and Adeli, 2015a: 707). Although they have competence in handling the hardest problems, they can just provide nearly optimal solutions because of the randomization. However, the randomization enables movements from local search to global search for algorithms (Gandomi et al., 2013a: 1). Therefore, in today's world, instead of classical optimization methods, approximate algorithms take place.

The stochastic optimization class covers heuristic and metaheuristic algorithms. Heuristic algorithms are defined as a method to discover a near-optimal solution for considered specific hard problems, whereas metaheuristics include an iterative process that integrates different concepts of exploring and exploiting the search space and they are not problem-specific algorithms (Kaveh, 2014: 2). The algorithms under the heuristic class also known as greedy heuristics provide the best local solutions. However, there is another chance to utilize these greedy approaches with global search approaches which create metaheuristics (Salcedo-Sanz, 2016: 12). For this reason, we can say that metaheuristic algorithms are more inclusive. Further details for metaheuristic algorithms will be given in the following sections.

In the literature, scholars call evolutionary algorithms, evolutionary computation, population-based algorithms, computational intelligence, soft computing, machine learning, heuristics refer to stochastic optimization algorithms. However, all the terms mentioned denote different topics (Simon, 2013: 2-3). Since some of them have a broad scope, some of them are subclasses, it is not appropriate to use these terms to refer to all stochastic optimization algorithms. For example;

evolutionary computation is a set of stochastic methods that are inspired by the evolutionary process of species in nature and it includes evolutionary algorithms, genetic programming, evolutionary strategies, genetic algorithm (Nesmachnow, 2014: 325). Moreover, evolutionary algorithms cover a large class of problem-solving methodologies that are based on the Darwinian principle of natural selection (Gendreau and Potvin, 2008: 76). On the other hand, computer intelligence represents technologies like neural networks, fuzzy systems, artificial life that can be applied other than optimization. As for machine learning and soft computing, evolutionary algorithms are a subset of them (Simon, 2013: 3). For that reason, in order not to be confused in terms, it is clear to use "metaheuristics" for the general name of the stochastic optimization techniques (Gandomi et al., 2013a: 1). The covering clusters are structured as given in Table 1.

Table 1: Covering Sets Examples

Evolutionary Computing ⊃ Evolutionary Algorithms
Computer Intelligence ⊃ Evolutionary Algorithms
Soft Computing ⊃ Evolutionary Algorithms
Machine Learning ⊃ Evolutionary Algorithms
Metaheuristics ⊃ Evolutionary Algorithms
Evolutionary Algorithms ⊃ Population-based Algorithms
Evolutionary Algorithms ⊃ Nature-Inspired Algorithms
Evolutionary Algorithms ⊃ Swarm Intelligence
Metaheuristics ⊃ Heuristics

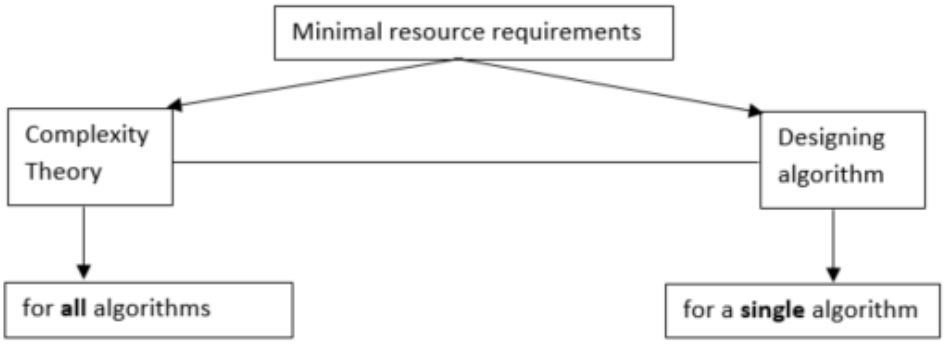
Source: Table is prepared by the Author

1.2. COMPLEXITY THEORY

Time and space are two concepts that are necessary for an algorithm to be able to solve a problem (Talbi, 2009: 9). Complexity theory deals with the time and space consumption of algorithms and the main classes of problems (Whitley and Watson, 2005: 317). According to Wegener (2005), "*The results of complexity theory have specific implications for the development of algorithms for practical applications.*" And they assert that complexity theory and designing an algorithm are two limitations of what can be done and cannot with certain resources in an algorithmic way. The relationship between complexity theory and designing an algorithm can be

summarized as in Figure 2. It would be better to clarify that resources are such things as computation time and storage space.

Figure 2: The Relationship Between Complexity Theory and Designing an Algorithm



Source: Adapted from Wegener (2005: 3)

1.2.1. Complexity of Algorithms

To obtain the computational complexity of an algorithm an asymptotic bound on the step count is used. The Big-*O* notation is used to compute the time or the space complexity of an algorithm. Various algorithms have different time complexities. Let *n* is the input length and *p* is a polynomial function, so we denote the time complexity of polynomial algorithm as $O(p(n))$; whereas when the time complexity function is not that much bounded which also includes non-polynomial time complexity, we can talk about exponential time algorithm. The comparison of these two function types is demonstrated in Table 2:

Table 2: The Time Complexity Comparisons

Time Complexity function	**Size (*n*)**		
	10	**30**	**50**
$O(n)$	0.00001 second	0.00003 second	0.00005 second
$O(n^2)$	0.001 second	0.009 second	0.0025 second
$O(n^5)$	0.1 second	0.27 second	0.125 second
$O(2^n)$	0.001 second	17.9 minutes	35.7 years
$O(3^n)$	0.59 second	6.5 years	$2x10^8$ centuries

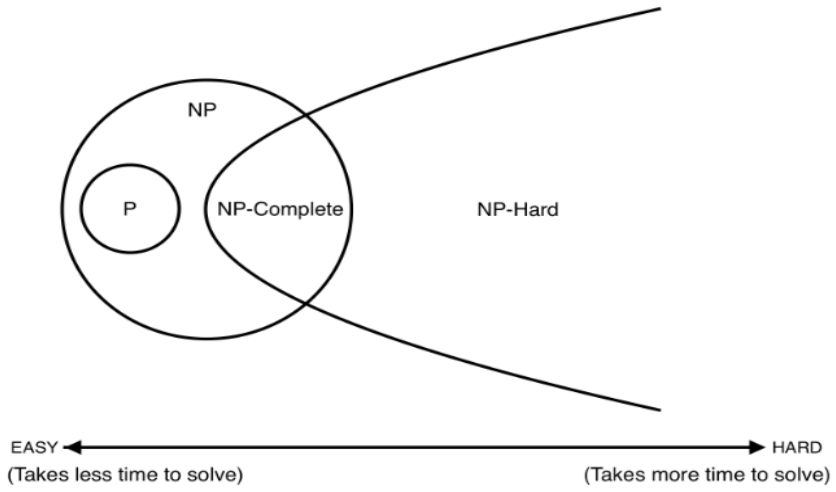
Source: Garey and Johnson, 1979: 7

As it is seen from Table 2, the execution times of different functions change enormously especially in exponential functions. It seems that polynomial-time algorithms can obtain deeper insight into the structure of a problem. For that reason, as mentioned by Garey and Johnson (1979), "*a problem has not been well-solved until a polynomial-time algorithm is known for it.*". If there is a polynomial algorithm for a problem, that problem is called tractable and all problems can be classified in terms of their complexities.

1.2.2. Complexity of Problems

The most popular problem classes handled by complexity theory are P (Polynomial) and NP (Non-deterministic Polynomial) problems. The representation of the classes is given in Figure 3 as a set structure.

Figure 3: The Classification of Problems



Source: Wu, 2016

As seen in Figure 3, the complexity of the problems increases as you go to the right side. Class P includes problems that are solvable in polynomial time which is a reasonable computation time and they are easy to solve. Continuous linear programming problems can be given as an example of class P (Talbi, 2009: 12). Whereas NP class which also covers class P denotes the problems that can be solved

in polynomial time but with a nondeterministic algorithm (Hedman, 2006: 299). All decision problems whose solution is either yes or no belong to the NP class (Garey and Johnson, 1979: 13). According to Figure 3, for any problem in class P, there are nondeterministic algorithms to solve because NP is a covering set. However, there is a striking question in the literature that addresses whether P=NP or not. Goldreich (2008) paraphrased this question as "*whether or not finding solutions is harder than checking the correctness of solutions*". Although the answer is unknown and controversial, generally it is thought that P≠NP means that finding a solution is harder than checking. Otherwise, P=NP means that if a problem can be solved in polynomial time with a non-deterministic computational model, then building a deterministic model is possible that solve the same problem in a polynomial time.

Furthermore, the NP-Complete class is also a subset of the NP class and it is related to reduction. This situation is explained by Goldreich (2008) as if any computational problem in NP class is reducible to another problem, that computational problem is NP-Complete. In other words, a problem in NP is also NP-Complete if all other problems in NP are reduced to the problem in NP class (Talbi, 2009: 14). Since NP-Complete problems are the hardest problems in NP class, Cook (1971) proved that the satisfiability problem belongs to NP-Complete class and then in 1972 Karp asserted that other computational problems such as set covering, Hamilton Circuit, Knapsack, Job sequencing are as hard as satisfiability problem by reducing satisfiability problem to the problems given. It would be better to clarify that NP-Complete problems are also subsets of NP-Hard class which is defined as the hardest problem class in complexity theory (Gonzalez, 2007: 4). It is clear from Figure 3 that NP-Hard problems do not have to be in NP class which means that there is no requirement for being a decision problem for NP-Hard problems. Most of the real-world optimization problems such as scheduling problems, vehicle routing problems, assignment problems belong to that class (Talbi, 2009: 14). However, we do not know how to find a solution to NP-Hard problems in polynomial time, algorithms that provide approximate solutions or the best possible solutions are needed. At this point, metaheuristics come to exist and will be handled in the following section.

1.3. METAHEURISTICS

The term "metaheuristics", firstly used by Glover in 1986, is a search framework that uses heuristic strategies (Gendreau and Potvin, 2008: 71). However, heuristic algorithms are problem-dependent and applicable to a particular problem, whereas metaheuristic algorithms are more general. Some of the scholars defined "metaheuristics" as below:

"*Metaheuristics are the methods of choice for solving complex, ill-defined problems*" (Gendreau and Potvin, 2008: 71).

"*Metaheuristics represent 'higher level', heuristic-based, soft computing algorithms that can be directed towards a variety of different optimization problems, by instantiating the generic schema to individual problems, needing relatively few modifications to be made in each specific case*" (Nesmachnow, 2014: 320).

"*Metaheuristic is a black box optimizer that can be applied to almost all optimization problems*" (Abdel-Basset et al., 2018: 185).

"*A metaheuristic is an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search spaces using learning strategies to structure information in order to find efficiently near-optimal solutions*" (Osman and Kelly, 1996: 3).

"*The word heuristic has its origin in the old Greek work heuriskein, which means the art of discovering new strategies (rules) to solve problems. The suffix meta, also is a Greek word, means "upper-level methodology*" (Kaveh, 2017: 2).

"*Metaheuristic algorithms are a higher-level heuristics with the use of memory, solution history and other forms of learning strategy*" (Yang, 2018: 4).

"*Metaheuristics are very popular family of solution methods for optimization problems and they are capable of finding acceptable solutions in a reasonable amount of time*" (Duarte et al., 2018: 29)

The definitions given above show us the general characteristics of metaheuristics and they can be summarized as below (Blum and Roli, 2003: 270-271); (Yang, 2018: 4):

- Metaheuristics are guiding the search process in order to find near-optimal solutions.

- Metaheuristics are approximate and stochastic algorithms.
- Metaheuristics are not problem-specific.
- Metaheuristics range from local search to complex learning processes.
- Metaheuristics use of domain-specific knowledge in the form of heuristics controlled by the upper-level strategy.
- Metaheuristics handle problems by treating them as a black box, thus they can solve a wide range of problems.

Although there are lots of algorithms, they can be evaluated in terms of their solution time, complexity, optimality, and the trade-off between diversification and intensification ability. They all present randomness and thus provide different solutions in different runs. The outstanding characterization of metaheuristics is that they explore several regions in search space and have the ability to escape from local optima (Siddique and Adeli, 2015a: 707).

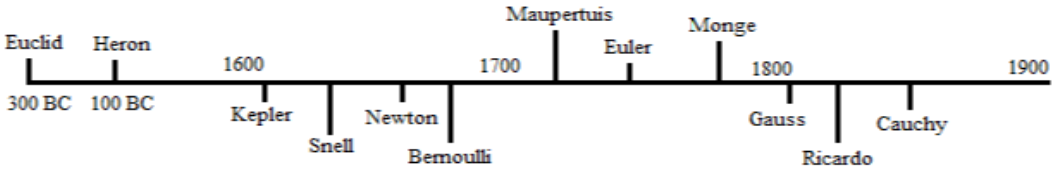
1.3.1. Historical Background

The birth of metaheuristics dates back to the years in the 20th century. However, it would be better to examine the periods separately when the foundations of metaheuristics were laid. For that reason, the historical background can be divided into four periods: The Early Period, 1900-1960, 1960-2000, and The Recent Past.

1.3.1.1. Early Period

The study of optimization problems is old as science itself. To summarize the breakthroughs in optimization, timelines are given chronologically.

Figure 4: Early Period



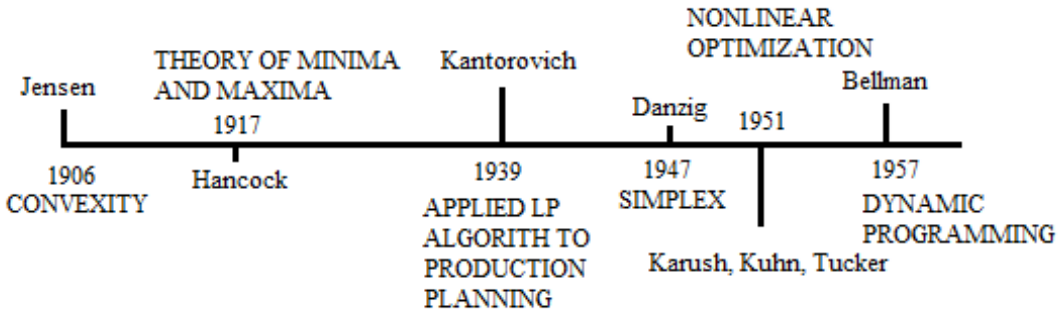
Source: Adapted from Yang, 2010a: 4-10

As shown in Figure 4, there are lots of contributors to the optimization field. For example, Greek Mathematicians Euclid and Heron used geometric entities and solved many optimization problems. Moreover, Kepler, Snell, Maupertuis who are known as astronomers discovered the laws of planetary motion, the law of reflection, and the principle of least action respectively. On the other hand, Newton solved the problem of the body shapes minimal resistance that was finalized as the resistance law of the body. As for Bernoulli, he made significant progress in calculus. Also, Monge analyzed a transportation problem and Gauss was the first person that used the least-squares analysis. Lastly, Cauchy developed an iterative method for equation systems (Yang, 2010a: 5-6). Although each of them has provided various theories for different subject areas, actually all of them are the milestones for optimization algorithms. Besides, most of the theories developed in these years are the foundations of most of the metaheuristic algorithms especially physics-based ones.

1.3.1.2. 1900-1960 Period

In the twentieth century, specific optimization topics have been revealed. At the beginning of the century, Jensen (1906) contributed to the literature by introducing the term called "convexity" which creates the basis of convex optimization. Throughout the years, many applications in combinatorial optimization and global optimization have been conducted by convex optimization (Boyd and Vandenberghe, 2004: xi). Moreover, Hancock (1917) published the first book about optimization that covers the extremum points of a function. Thereafter, Kantorovich (1939), a famous mathematician, and his colleague Koopmans developed a mathematical model which is awarded Nobel Prize in Economic Sciences for the production problem (Yang, 2010a: 6). Afterward, an algorithm called "Simplex" was created by Dantzig in 1947 to solve linear programs for planning and decision-making problems. The simplex algorithm has dominated applications in operations research for half a century (Nash, 2000: 29). Finally, in 1951 Karush-Kuhn-Tucker conditions were introduced in the literature for nonlinear optimization problems to be solved and in 1957, Bellman presented "Dynamic Programming" that provides solutions to multistage decision and planning problems (Yang, 2010a: 7).

Figure 5: Twentieth Century

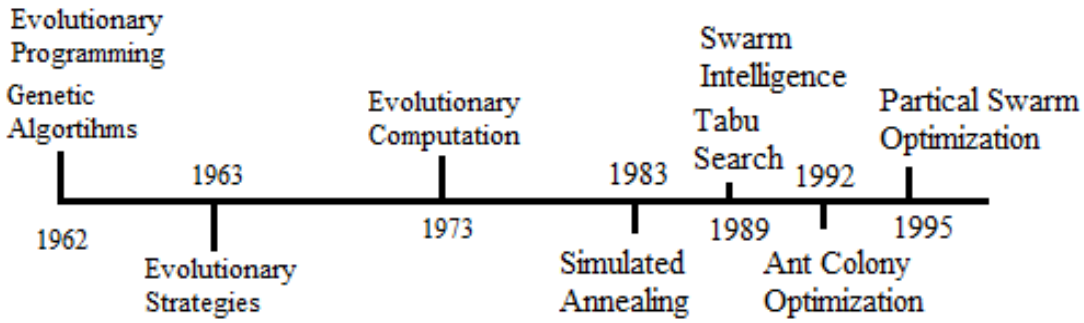


Source: Collected from Yang, 2010a: 4-10 and set on the timeline by the author

1.3.1.3. 1960-2000 Period

By the years of the 1960s, the literature on optimization broadened and the research line for problem-solving methods completely turned into a different format called "evolution" (Sörensen et al., 2018: 7). In Figure 6, the most commonly used and the best-known algorithms developed in the years between 1960-2000 are demonstrated. These algorithms can be defined as the cornerstones for the recently developed modern algorithms.

Figure 6: 1960-2000 Years



Source: Collected from Yang (2010a: 4-10) and set on the timeline by the author

All methods given in Figure 6 are based on different principles. For that reason, classification plays an important role. Before classifying, it would be better to discuss the starting points of the pioneer algorithms. Among these algorithms, Simulated

Annealing and Tabu Search are single solution methods; whereas the others are all population-based methods that keep around a set of candidate solutions. The population-based algorithms given in Figure 6 are inspired by biology and nature (Luke, 2011: 29).

Genetic Algorithm, Evolutionary Programming, Evolutionary Strategies, Genetic Programming belong to the field of Evolutionary Computation that depends on computational methods inspired by evolutionary processes (Brownlee, 2011: 87). The earliest well-known evolutionary algorithm is the Genetic Algorithm (Simon, 2013: 35), and they were presented by John Holland (1962). Holland is the first scholar that used the terms "crossover, recombination, mutation and selection" to adapt the natural selection theory of Darwin to optimization. Briefly, the Genetic Algorithm comprises encoding of an optimization function as character strings to denote chromosomes, using genetic operations and selecting the best chromosomes according to their fitness values. (Yang, 2014: 77-78).

Evolutionary Programming was developed in 1962 originally by Fogel (Erdem, 2007: 15), and it was proposed as an approach to artificial intelligence. Mutating the solutions and selecting the next generation from mutated solutions are two major steps (Yao et al., 1999: 82). On the other hand, Evolutionary Strategies were developed by Rechenberg and further improved by Schwefel in the 1960s (Xiong et al., 2015: 613). Different from other Evolutionary Algorithms, Evolution Strategies were inspired by the process of evolution (phenotype, hereditary, variation) rather than the genetic mechanism of evolution (genome, chromosome, genes, alleles) (Brownlee, 2011: 108). In other words, the evolutionary strategy does not use population or crossover, it just mutates one solution, and then the best two solutions become the parent for the next mutation round (Sörensen et al., 2018: 8). Genetic programming is based on the idea of evolving computer programs that depends on natural selection (Langdon et al., 2010: 185). It uses genetic operators (reproduction, crossover, and mutation). Although Genetic Programming is similar to the Genetic Algorithm, it differentiates in terms of the representation of individual computer programs in the population. It has structured trees to demonstrate functions or operators (Xiong et al., 2015: 613).

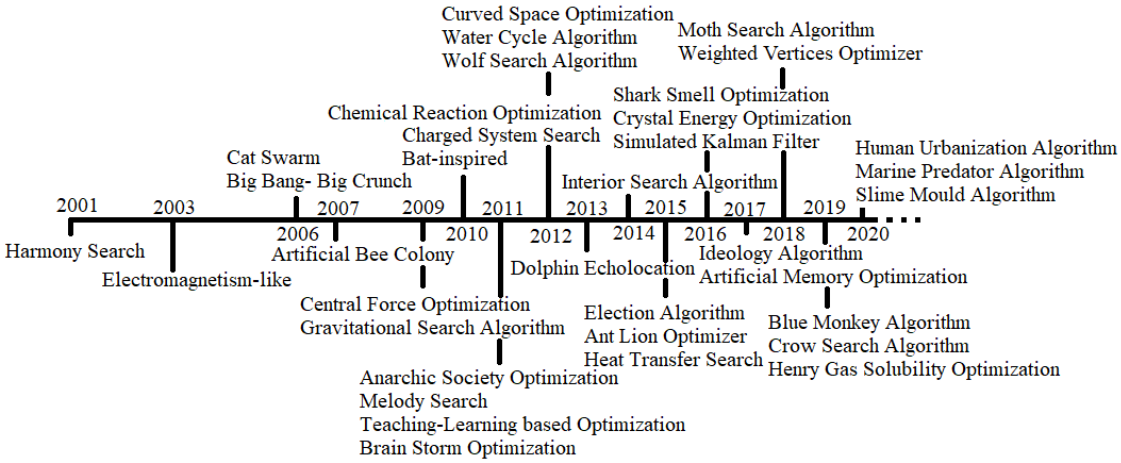
In the years of 1980s, more metaphors have started to be used in algorithms. As trajectory-based optimization algorithms, Simulated Annealing and Tabu Search

took their place in the past. Each of them has striking characteristics that are used as an inspiration for the other algorithms. For example, Simulated Annealing developed by Kirkpatrick et al. (1983) was inspired by the cooling and crystallizing behavior of chemical substances (Simon, 2013: 223). The objective is to reach the global minimum energy state during the process of annealing and move to any new set of design variables that corresponds to a change of the energy state (Sahab et al., 2013: 36). The basis of the Simulated Annealing dates back to the Metropolis Algorithm (Metropolis et al., 1953) which provides a simulation of a collection of atoms in equilibrium at a given temperature. Furthermore, both of them are the inspirations for most of the physics-based algorithms which will be covered in the following sections. Tabu Search proposed by Glover (1989; 1990) can be considered as an extension of a hill-climbing search (Xiong et al., 2015: 610). The most remarkable feature of this algorithm is that Tabu Search uses memory structures to store historical information in order not to visit the same candidate solutions again (Luke, 2011: 24).

As of the 2000s, Swarm Intelligence should be covered as a general title of bio-inspired algorithms. Swarm Intelligence mimics the evolutionary process of the behaviors of some species like ants, bees, birds, fish (Erdem, 2007: 19). The main idea of these kinds of algorithms is "collective intelligence", in other words, the cooperation of large numbers of homogeneous agents in the environment. The paradigm has two dominant sub-fields called Ant Colony Optimization and Particle Swarm Optimization (Brownlee, 2011: 229). Ant Colony Optimization algorithm (Dorigo et al., 1996) reflects the behaviors of ant colony while trying to find the food source, whereas in Particle Swarm Optimization (Eberhart and Kennedy, 1995) the location of particles is tracking in terms of their velocities (Sahab et al., 2013: 39). The recent swarm intelligence-based algorithms proposed in the literature are provided in the classification of metaheuristics section.

1.3.1.4. The Recent Past Period

Figure 7: Metaheuristic Algorithms Developed in the Recent Past



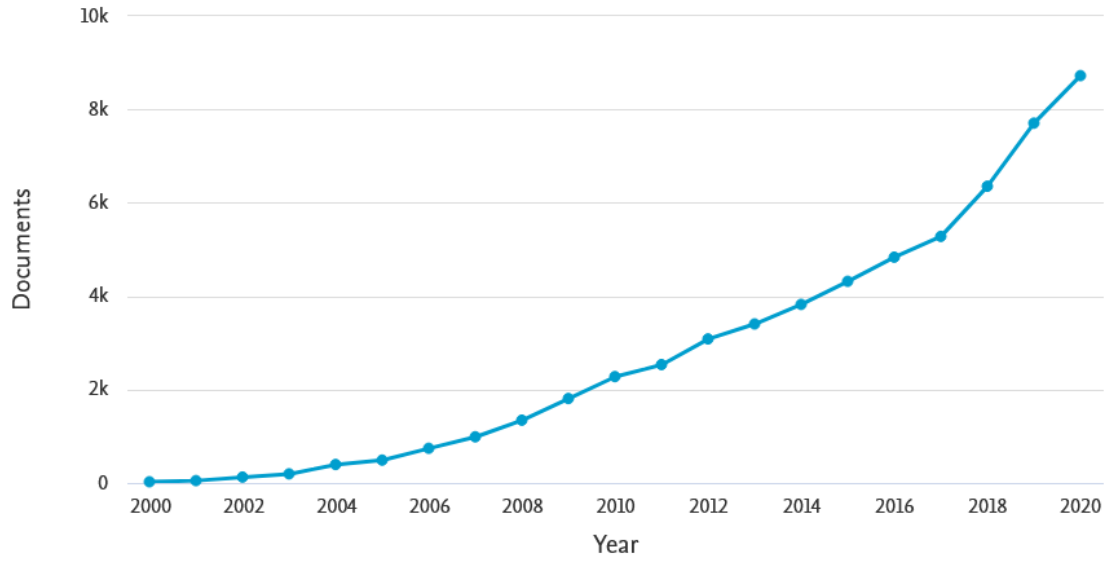
Source: Prepared by the Author

Figure 7 shows the various kinds of metaheuristic algorithms published between 2000 and 2020. Since 2000, researchers have focused more on developing new algorithms based on various kinds of metaphors (Sörensen et al., 2018: 14). The behavior of animals, laws in the field of science, facts of nature, social behaviors are some of the inspirations used in the literature. Cat Swarm Optimization (Chu et al., 2006), Artificial Bee Colony (Karaboga and Basturk, 2007), Bat-inspired (Yang, 2010b), Wolf Search Algorithm (Tang et al., 2012), Dolphin Echolocation (Kaveh and Farhoudi, 2013), Shark Smell Optimization (Abedinia et al., 2014), Ant Lion Optimizer (Mirjalili, 2015), Moth Search Algorithm (Wang, 2018), Blue Monkey Algorithm (Mahmood and Al-Khateeb, 2019), Crow Search Algorithm (Huang and Wu, 2019), Marine Predators Algorithm (Faramarzi et al., 2020a) are some of the animal-inspired algorithms; Central Force Optimization (Formato, 2007), Gravitational Search Algorithm (Rashedi et al., 2009), Charged System Search (Kaveh and Talatahari, 2010a), Chemical Reaction Optimization (Lam and Li, 2010), Curved Space optimization (Moghaddam et al., 2012), Heat Transfer Search (Patel and Savsani, 2015), Henry Gas Solubility Optimization (Hashim et al., 2019), Interior Search Algorithm (Gandomi, 2014), Weighted Vertices Optimizer (Dolatabadi, 2018), Simulated Kalman Filter (Ibrahim et al., 2016) are science-inspired algorithms;

Harmony Search (Geem et al., 2001), Melody Search (Ashrafi and Dariane, 2011) are music-inspired algorithms; Water Cycle Algorithm (Eskandar et al., 2012), Crystal Energy Optimization (Feng et al., 2016), Slime Mould Algorithm (Li et al., 2020) are nature-inspired algorithms; Anarchic Society Optimization (Ahmadi-Javid, 2011), Brain storm optimization (Shi, 2011a; 2011b), Teaching-Learning based optimization (Rao et al., 2011), Election algorithm (Emami and Derakhshan, 2015), Artificial Memory Optimization (Huang, 2017), Ideology Algorithm (Huan et al., 2017), Human Urbanization Algorithm (Ghasemian et al., 2020) are social-inspired algorithms.

In Figure 8, the number of articles published in the literature for each year is given (Scopus Database). It would be better to clarify that, "Bio-inspired", "Nature-inspired", "Swarm intelligence", "Physics-based", "Chemistry-based" and "Optimization" are used in Scopus Database as keywords ("*(TITLE-ABS-KEY (optimization)) AND (((((("Bio-inspired")) OR ("Nature-inspired")) OR ("Swarm intelligence")) OR ("Physics-based")) OR ("Chemistry-based")*"). The data include both theoretical and practical studies. However, the number of studies conducted is in an increasing trend.

Figure 8: Number of Documents Published Over the Period 2000-2020



Source: Scopus Database

Especially in recent years, the number of new algorithms has exploded. Although it seems good to have new algorithms, many researchers (Fister Jr et al., 2016; Piotrowski et al., 2014; Sörensen, 2015; Odili et al., 2018; Tovey, 2018; Lones, 2020) criticize that plenty of metaheuristics have similarities and they are not novel in literature. However, as mentioned by Wolpert and Macready (1995) there cannot be an algorithm that is appropriate for all problems. The possibility that there may always be a better algorithm leads researchers to develop algorithms using new metaphors. For this reason, new metaheuristics will continue to be introduced soon (Dokeroglu et al., 2019: 23).

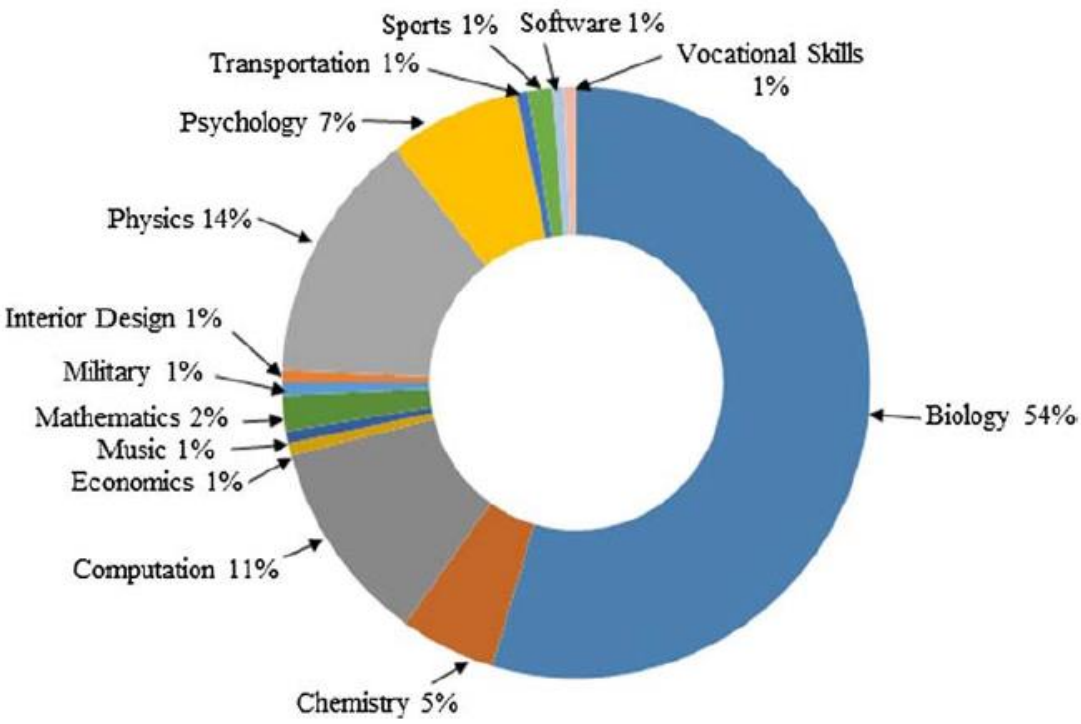
It is not possible to demonstrate all of the metaheuristic algorithms published recently in a figure. There are lots of review articles in literature such as Blum and Roli (2003), Fister et al. (2013), Siddique and Adeli (2015a), Yang et al. (2015), Sotoudeh-Anvari and Hafezalkotob (2018), Abdel-Basset et al. (2018). Detailed information about specific physics-based algorithms will be covered in the second chapter.

1.3.2. The Classification of Metaheuristics

The term "metaheuristics" comprises a wide range of techniques. There are lots of ways to classify metaheuristics. Fister et al. (2013) mentioned that classification of algorithms may depend on various criteria such as main principles, sources of inspiration, perspectives, and motivations. They classified nature-inspired algorithms in terms of the source of inspirations (Swarm-intelligence-based, bio-inspired, physics-based, chemistry-based). On the other hand, Blum and Roli (2003) summarized the most important classifications as Nature-inspired vs non-nature, population-based vs single point search, dynamic vs static objective function, one vs various neighborhood structures, memory usage vs memory-less methods. Echevarría et al. (2019) classified metaheuristics in terms of the number of solutions and inspiration sources. Beheshti and Shamsuddin (2013) handled metaheuristic algorithms in terms of inspiration sources, number of solutions, objective function, neighborhood structure, memory usage. Sotoudeh-Anvari and Hafezalkotob (2018) also classified the origins of inspirations as animals, physics, humans, plants, nature, and biology. They also demonstrated that the most popular foundations of inspiration

are animal and physic. Also, Hussain et al. (2019) classified all metaheuristics in terms of their metaphor disciplines, and the classification is shown in Figure 9 that biology and physics took the first two places respectively. Molina et al. (2020) proposed two taxonomies as the source of inspiration and the behavior of each algorithm. The source of inspiration includes Breeding-based Evolution, Swarm Intelligence, Physic and Chemistry based, Social Human Behavior, Plant-based and miscellaneous classes. On the other hand, the behavior of an algorithm is handled according to its principle in creating new solutions. The authors presented the behavior taxonomy as solution creation and Differential Vector Movement classes.

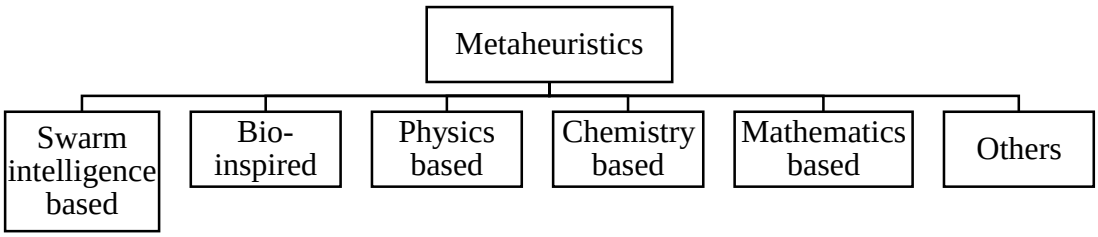
Figure 9: The Inspirations Adopted by Researchers



Source: Hussain et al., 2019: 2216

Finally, as mentioned above, algorithms can be classified according to the purposes of the researchers. For that reason, to include all types of algorithms, the classification conducted by Xing and Gao (2014) has been taken by adding one more class called Swarm Intelligence-based and given in Figure 10.

Figure 10: Classification of Metaheuristics



Source: Adapted from Xing and Gao, 2014

Swarm intelligence-based algorithms are the subsets of bio-inspired algorithms. For that reason, the algorithms classified as bio-inspired do not include swarming behavior (Fister et al., 2013: 2). On the other hand, Siddique and Adeli (2015a) also classified biology-based algorithms as Evolutionary Algorithms, Bio-inspired Algorithms, and Swarm Intelligence based Algorithms. Some of the studies that present new algorithms considering swarming behaviors of species can be summarized as Ant Colony Optimization (Dorigo and Di Caro, 1999), Cat Swarm (Chu et al., 2006), Monkey Search (Mucherino and Seref, 2007), Artificial Bee Colony (Karaboga and Basturk, 2007), Firefly Algorithm (Yang, 2009), Cuckoo Search (Yang and Deb, 2009), Eagle Strategy (Yang and Deb, 2010), Bat-inspired Algorithm (Yang, 2010b), Wolf Search Algorithm (Tang et al., 2012), Swallow swarm optimization algorithm (Neshat et al., 2013), Grey Wolf Optimizer (Mirjalili et al., 2014), Pigeon-inspired Optimization (Duan and Qiao, 2014), Crow Search, (Askarzadeh, 2016), Blue Monkey Algorithm, (Mahmood and Al-Khateeb, 2019), Marine Predators Algorithm (Faramarzi et al., 2020a), Jellyfish Search Optimizer (Chou and Truong, 2021).

The remaining classes belong to nature-inspired and non-nature-inspired algorithms. According to Kar (2016), bio-inspired algorithms can solve complex problems, since they can learn and adapt like biological organisms. For that reason, they are taking too much attention from the scientific community in recent years. Every day, the number of new algorithms increases. Artificial Plant Optimization (Zhao et al., 2011), Brain Storm Optimization (Shi, 2011a; 2011b), Flower Pollination Algorithm (Yang, 2012), Atmosphere Clouds Model Optimization (Gao-Wei and Zhanju, 2012), Great Salmon Run Algorithm (Mozaffari et al., 2012), Fruit Fly Optimization Algorithm (Pan, 2012), Cuttlefish Algorithm (Eesa et al., 2013), Dolphin

Echolocation (Kaveh and Farhoudi, 2013), Ant Lion Optimizer (Mirjalili, 2015), Laying Chicken Algorithm (Hosseini, 2017), Emperor Penguin Optimizer (Dhiman and Kumar, 2018), Seagull Optimization Algorithm (Dhiman and Kumar, 2019) are some of the algorithms developed in the recent years.

The physics-based class includes algorithms inspired by the theories developed in the field of physics. Biswas et al., 2013; Siddique and Adeli, 2016; Salcedo-Sanz, 2016 published articles focusing specifically on physics-based algorithms and provided comprehensive literature surveys. The principles, working structures, and the classification of physics-based algorithms will be covered in the literature survey section in detail. Some of the studies published in this field can be given as Electromagnetism-like Algorithm (Birbil and Fang, 2003), Particle Collision (Sacco and De Oliveira, 2005), The Big Bang- Big Crunch Algorithms (Erol and Eksin, 2006), Central Force Optimization (Formato, 2007), Magnetic Optimization Algorithm (Tayarani and Akbarzadeh, 2008), Gravitational Search Algorithm (Rashedi et al., 2009), Charged System Search (Kaveh and Talatahari, 2012), Ray Optimization (Kaveh and Khayatazad, 2012), Black Hole Search (Hatamlou, 2013), Thermal Exchange (Kaveh and Dadras, 2017), Henry Gas Solubility Optimization (Hashim et al., 2019).

Chemistry-based algorithms are the ones inspired by chemical reactions such as oxidation-reduction, combination, decomposition, displacement, gas formation, and metathesis reactions (Xing and Gao, 2014: 8). Chemical Reaction Optimization (Lam and Li, 2010), Gases Brownian Motion Optimization (Abdechiri et al., 2013), Chemotherapy Science Algorithm (Salmani and Eshghi, 2017), Vapor-Liquid Equilibrium Algorithm (Taramsco et al., 2020) are some of the examples published in various journals.

From now on, lots of metaheuristic algorithms have been mentioned separately. However, these algorithms can also be used together and are called “hybrid algorithms”. Metaheuristics can also be combined with exact methods. Boschetti et al. (2009) defined the term “matheuristics” as using mathematical techniques in metaheuristic frameworks. Salem (2012) proposed the Base Optimization Algorithm which works with the combinations of arithmetic operators. Moreover, Mirjalili (2016) developed an algorithm called Sine Cosine Algorithm which takes its name from the

sin and cos equations. Simulated Kalman Filter (Ibrahim et al., 2016), Golden sine algorithm (Tanyildizi and Demir, 2017) are the other mathematics-based algorithms published in recent years.

The remaining algorithms based on various inspirations put into the other subclass. Imperialist Competitive Algorithm (Atashpaz-Gargari and Lucas, 2007), Anarchic Society Optimization (Ahmadi-Javid, 2011), League Championship Algorithm (Kashan, 2009), Melody Search (Ashrafi and Dariane, 2011), Teaching–learning-based optimization (Rao et al., 2011), Jenga-inspired Optimization Algorithm (Lee et al., 2013), Golden Ball (Osaba et al., 2014), Artificial Cooperative Search Algorithm (Civicioglu, 2013), Interior Search Algorithm (Gandomi, 2014), Simulated Kalman Filter (Ibrahim et al., 2016), Ideology Algorithm (Huan et al., 2017), Weighted Vertices Optimizer (Dolatabadi, 2018), Human Urbanization Algorithm (Ghasemian et al., 2020) are some of the studies included in this class.

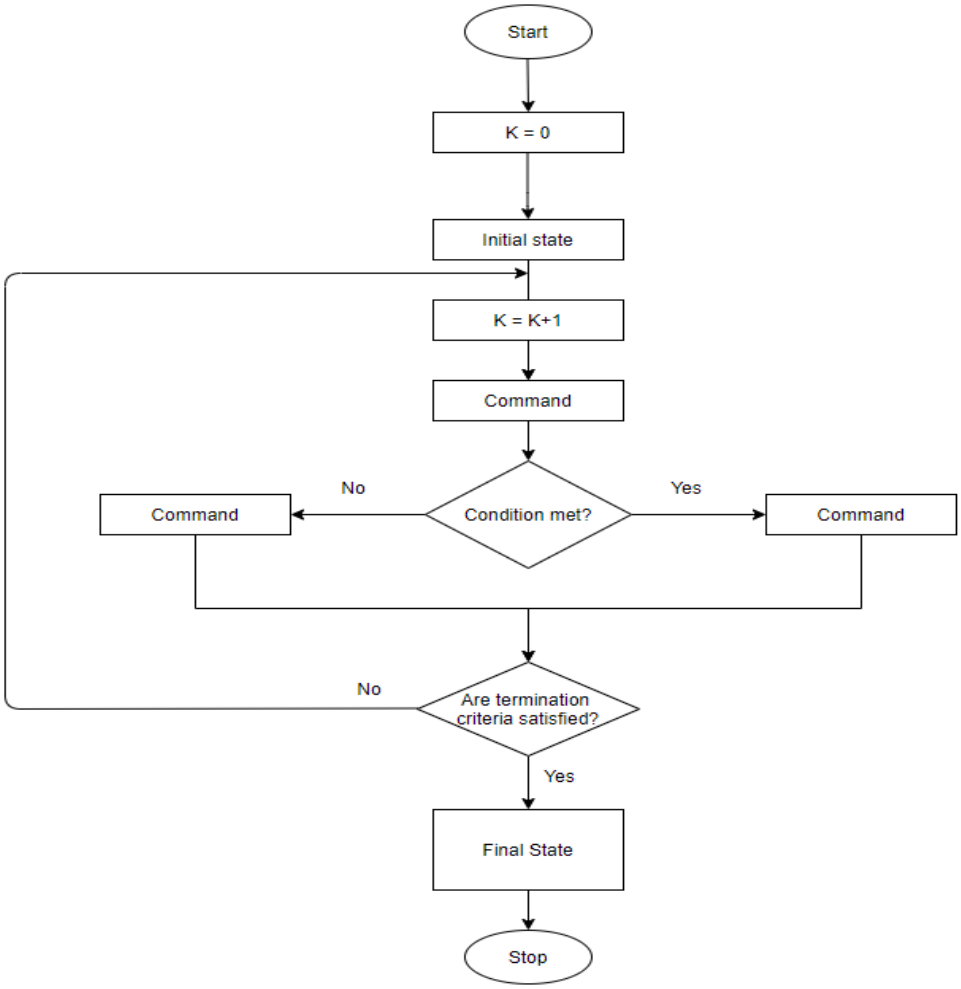
As summarized above, there are numerous metaheuristic algorithms in the literature. However, there is no way to determine which algorithm is the best one. As Wolpert and Macready (1995) proved with the "No Free Lunch Theorem", there cannot be an algorithm that is appropriate for all problems. In other words, there is always a better algorithm than the existing ones. According to Wolpert and Macready (1995), when the performances of any two algorithms are evaluated across all problems, the average performances will be equal. In simple terms, the only way to conclude that one algorithm outperforms another is related to considering a specific problem (Ho and Pepyne, 2002: 292). Since there is not any metaheuristic to arrive at the best solution for every problem, scholars should focus on developing an algorithm that can solve most types of problems. However, many researchers (Fister Jr. et al., 2016; Piotrowski et al., 2014; Sörensen, 2015; Odili et al., 2018) criticize that plenty of metaheuristics have similarities and they seem to be novel in literature. For that reason, it would be better to develop algorithms that provide more "optimal-like" solutions without trapping the "novelty".

1.4. EVOLUTIONARY COMPUTATION CONTEXT

Evolutionary Computation (EC) comprises building, applying, and studying metaphorical algorithms inspired by Darwinian principles of natural selection. The algorithms based on evolutionary principles utilize nature's capability to evolve living beings well adapted to their environment (Gonzalez, 2007: 9). The background of the current evolutionary algorithms will be handled in the historical background section. However, Evolutionary Programming, and Genetic Algorithms pioneer the existing metaphorical algorithms regardless of inspiration differences. While there are metaheuristics that are inspired by different subjects, they are basically all going through similar iterative evolutionary computation processes which include various strategies. All of these algorithms develop a population of individuals over generations, reproduce offspring through various operators, and select the most suitable for survival in each generation (Du and Swamy, 2016: 2).

Model specification (representation), model identification, initialization, fitness calculation, neighborhood strategies, memory usage, selection, reproduction, stopping condition, model reliability, and model validity are summarized in the following sections as components of the evolutionary computation for population-based metaheuristics. While in some algorithms all these components are applied; in some algorithms, these components can be used in different combinations. In this section, the general context of evolutionary computation is explained in brief. The schematic representation of any algorithm should follow is given in Figure 11.

Figure 11: General flow of an algorithm



Source: Bozorg-Haddad et al., 2017: 20

1.4.1. Model Specification

The model specification refers to the determination of the most suitable method for the problem to be solved. It is a decision that can be made according to the nature of the problem. As mentioned by Wolpert and Macready (1995) with the "No Free Lunch Theorem", there cannot be an algorithm that is appropriate for all problems. Although metaheuristic algorithms are not problem-based, choosing the appropriate algorithm that can be solved for specific problem groups is an important decision stage. For this reason, the representation of solutions is the initial design question to be answered according to the optimization problem in any iterative method. Representations that differ according to the problem types are as follows: binary

coding for the knapsack problem, vector of discrete values for the assignment problem, permutation for traveling salesman problem, and vector of real values for continuous optimization problems (Talbi, 2009: 35). Furthermore, even qubit representation and superposition states can be encountered in combinatorial optimization after merging evolutionary computation and quantum computing in the literature (Layeb, 2013: 15).

Choosing the most suitable algorithm among the algorithms that will provide solutions according to the representations (binary, discrete, permutation, continuous) is another decision-making problem. However, deciding the "good" method and parameter settings requires the know-how and experience of the user, rather than tracking well-laid-down rules as in the field of statistics (Siarry, 2016: 17). As mentioned by Yang (2014), choosing the best algorithm for a given type of problem is harder to be questioned than choosing problems for a given algorithm. For this reason, in many cases, we might not know the performance of the algorithm without testing. Therefore, choosing the right algorithm depends on the expertise of the decision-maker, available sources, and the problem types.

1.4.2. Model Identification

Identifying the model can result in the situation that reached at the end of the compliance between the problem and the method. This goodness-of-fit can be interpreted by considering how close the algorithm converged to the near-optimal point. The convergence of an algorithm may be checked by graphing or monitoring the fitness value of the best-so-far solution against iterations, run time, or function evaluation values (Bozorg-Haddad et al., 2017: 36). For example, the error level from the best-known solution can be regarded as an indicator of model identification. However, the identification of a model cannot be limited to a single iteration or trial. Descriptive statistics as summary measures should be considered among numerous trials. For this reason, several performance indicators can be also implemented for model identification of algorithms. Since these indicators are also related to the model reliability, the details about them will be given in the 1.4.10 section.

1.4.3. Initialization

The initialization step is related to generating a random population as a set of potential solution points. This initial process means taking population samples from the search space in each generation (Erdem, 2007: 12). According to McPhee and Hopper (1999), the lack of diversity may result in premature convergence. Therefore, especially in population-based metaheuristics, the diversity of the initial populations plays an important role in the efficiency of the algorithms. The strategies for generating the initial population are classified as random generation, sequential diversification, parallel diversification, and heuristic initialization (Talbi, 2009: 194). The population size is preferred constant in general and they are prepared to be affected by the reproduction strategies employed in metaheuristics.

1.4.4. Fitness Calculation

The optimality of the solutions is determined according to the value of the objective function as fitness value and it guides the algorithm to search for better solutions. However, it is not always a single indicator for the desirability of a solution (Bozorg-Haddad et al., 2017: 24). In some cases, if there is an expected value for the objective function, the sum of squared error values can be used as well (Erdem, 2007: 12).

Moreover, if the optimization model includes constraints, many different constraint handling methods are implemented to evaluate the optimality of the solution points by considering the feasibility as well. According to the purpose of the problem, the fitness value can be structured on the "shortest path", "lowest cost", "smallest penalty" or “highest utility” etc.

According to Montes et al. (2005), incorporating constraints into the fitness function in evolutionary algorithms is an open research area. For this reason, there are many approaches developed in the literature. The simplest approach is the “rejection strategy” which discards infeasible solutions regardless of the violation amount. The drawback of this approach is that there is no learning strategy about discarded solutions for further iterations (Duarte et al., 2018: 48). The most popular strategy for constraint

handling is the “penalty approach”. Different from rejecting directly, this approach penalizes solutions in terms of their degree of infeasibility (Maier et al., 2019: 207). Montes et al. (2005) summarized the most known penalty approaches as the death penalty, static penalty, dynamic penalty, annealing penalty, adaptive penalty, co-evolutionary penalty, Segregated genetic algorithm, and fuzzy penalty. However, penalty approaches have also some disadvantages in the determination of the corresponding weights via trial and error (Maier et al., 2019: 207). Another approach for constraint handling is “repairing algorithms”. They are applied to infeasible solutions to make them feasible and generally, they are greedy heuristics (Talbi, 2009: 53). Repair algorithms are widely applied in combinatorial optimization problems. The problem dependency is the weakness of repair strategies (Michalewicz, 2000: 56). “Preserving strategy” is another constraint handling approach that ensures feasibility by using problem-specific representation and operators. Nevertheless, this approach cannot be generalized for all optimization problems (Talbi, 2009: 53). Finally, hybrid methods can also be combined with evolutionary algorithms that employ heuristic rules or gradient methods to facilitate an efficient local search (Erdem, 2007: 34). Lagrangian multipliers, fuzzy logic, immune system models, cultural algorithms, ant colony algorithm-based methods are some of the hybrid methods used for constraint handling as well (Kicinger et al., 2005: 1943).

1.4.5. Neighborhood Strategies

Intelligence is another important concept that is utilized in population-based metaheuristics. Communication is a feature of intelligence and it arises from the neighborhood of individuals in the population. Communicating with each other leads individuals to reach better solution points by learning from each other’s successes and failures (Simon, 2013: 27). According to the neighborhood strategy employed for reproduction, the closest neighbor or the best-so-far solution is considered to generate better solutions. In some cases, different approaches can be developed. The concept of swarm intelligence explains the behaviors of some species in nature such as ant colonies, bird flocks, and fish schools (Erdem, 2007: 19). Proximity, quality, diverse response, stability, adaptability principles are some of the neighborhood principles which are employed to develop a branch of evolutionary problem-solving methods

(Eberhart and Shi, 1998). Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO) algorithms are the pioneers of metaheuristics based on cooperative behaviors of species (see Historical Background section). Cultural Algorithm, another swarm intelligence algorithm, differentiates from PSO and ACO in terms of knowledge types (normative knowledge, situational knowledge, domain knowledge, history knowledge, and topographical knowledge) used in problem-solving (Reynolds et al., 2005), and this algorithm models social evolution and learning (Du and Swamy, 2016: 7). Furthermore, there is also the Social and Civilization method developed by Ray and Liew (2003) that mimics the intra and intersociety interactions among society. There are many more algorithms based on different principles that take advantage of the communication between individuals in the population.

1.4.6. Memory Usage

Storing search history may provide information for future search in optimization algorithms. Memory-less algorithms guide the current solution by employing the Markov process to determine the next state of the solution points (Bozorg-Haddad et al., 2017: 28). However, memory is a component that improves the performance of the algorithms. Memory structure records information about solutions. Tabu Search (TS) is the first non-nature metaheuristic that uses attributive memory (Yang, 2014: 18). According to Bozorg-Haddad et al. (2017), there is an argument that memory usage qualifies TS as a nature-inspired algorithm. Although there are various ways to use memory, TS utilizes attributive memory which means that recording specific solution points not only prevents cycling in the search but also performs efficient exploration ability (Duarte et al., 2018: 94). In other words, TS holds the best results from search history as the Tabu list and uses this information to determine the next search strategies. Scatter Search (Glover, 1999) is the improved version of TS which involves best and worst search results in its memory list. The artificial Immune System is another memory-based algorithm that mimics the biological immune systems and utilizes its characteristics for learning and memorizing (Hunt and Cooke,

1996: 189). Many state-of-the-art algorithms in the literature include memory in their principles to improve their performance.

1.4.7. Selection

The selection operator is another component that may be preferred to use in evolutionary computation. In some of the algorithms, not all current solutions are employed to reproduce new solutions. The selection approach may differ randomly or deterministically (Bozorg-Haddad et al., 2017: 33). According to the Darwinian principle, the higher the survival of the fittest, the more often it is selected to reproduce (Petrowski and Hamida, 2016: 117). The simplest approach used in Genetic Algorithms called Elitism is letting the best solutions pass on to the next iterations (Yang, 2014: 30). Although not all algorithms implement selection operators, metaheuristics differentiate from evolutionary algorithms in the way of selecting solutions. The Boltzman selection, the Roulette wheel, the Tournament selection are the most common selection approaches used in the literature (Bozorg-Haddad et al., 2017: 34).

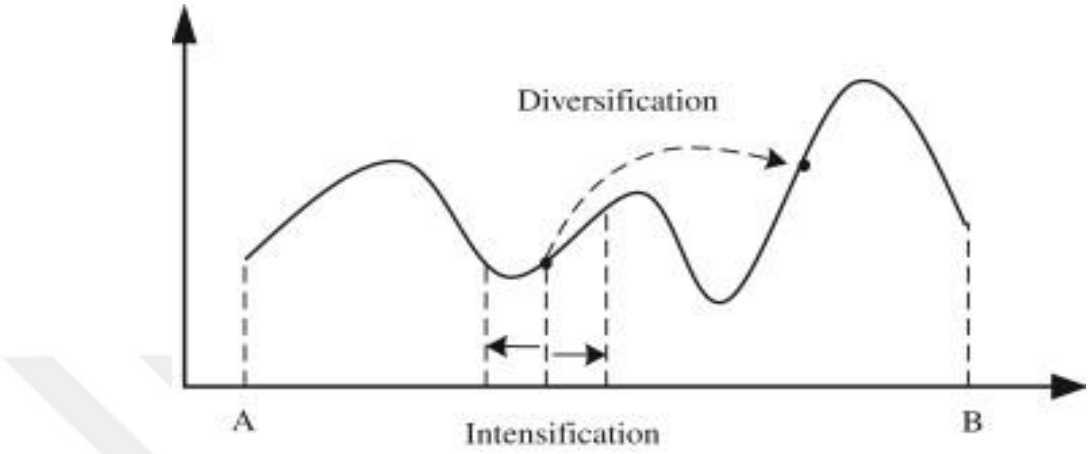
1.4.8. Reproduction

After the initialization and selection steps, new solutions should be generated according to the strategies that the algorithm employs. Generally generating new solutions is related to the solutions located around neighbors of the previous solution (Bozorg-Haddad et al., 2017: 34). All metaheuristics aim at finding feasible, good quality solutions efficiently and searching solution space without trapping into local optima (Nesmachnow, 2014: 323).

The key point for the efficiency of metaheuristics is the tradeoff between randomization and local search i.e. diversification and intensification. A good combination of diversification and intensification provides better global optimality for complex problems (Yang, 2018: 15). Exploration also called diversification means searching for new solutions, whereas exploitation means improving solutions as intensification (Simon, 2013: 28). The efficiency of metaheuristics depends on the

balance between exploration and exploitation (Sotoudeh-Anvari and Hafezalkotob, 2018: 84).

Figure 12: Diversification vs Intensification



Source: Purnomo and Wee, 2013: 386

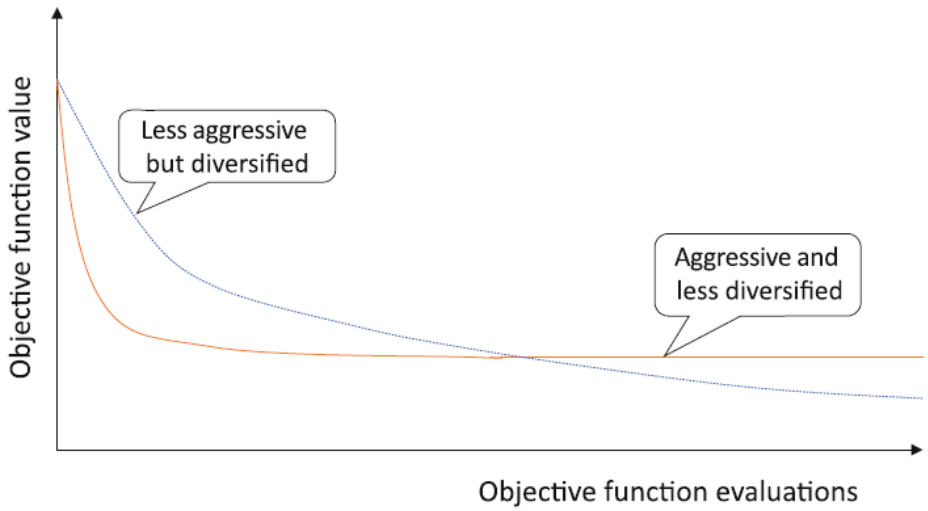
As shown in Figure 12, exploration is expressed as diversification and it allows the algorithm to visit diverse regions to explore the search space (Yang et al., 2014: 978). New regions within the solution space are searched with the help of randomization. Generally, random numbers are generated by using uniform distribution or Gaussian distribution (Yang, 2018: 14). Crossover, mutation, recombination are the evolutionary operators for generating novel solutions in Genetic Algorithms (Salcedo-Sanz, 2016: 15). Furthermore, various operators inspired by different subjects may be implemented to the algorithms with the same purpose. Exploration is a tool for increasing the probability of finding the global optimum, however, it also decreases the ability to converge the global optimum at the same time (Yang et al., 2015: 1988).

On the other hand, exploitation is also demonstrated in Figure 12 as intensification which is a way for converging the possible optimum. This time instead of searching for new regions, the algorithm concentrates on neighbors in the immediate region to increase the quality (Nesmachnow, 2014: 323). In other words, exploitation takes the advantage of the information that a good solution is found in this local region (Yang et al., 2014: 978). Moreover, it utilizes gradients for local information to

increase the convergence rate (Yang, 2018: 14). Hill Climbing is one of the local search methods that utilize derivative information for the search procedure (Yang, 2014: 27). As the opposite of exploration, exploitation can increase the convergence but move away from the possibility of finding the global optimum, which results in being stuck to the local optimum (Yang et al., 2015: 1988).

The balance between intensification and diversification plays an important role in the development of algorithms. The convergence rate of a metaheuristic is strongly related to the exploration-exploitation capacity (Morales-Castañeda et al., 2020: 2). As for Duarte et al. (2018), the opportunity for diversification depends on the length of the optimization process including several iterations, computational time, or function evaluations. However, these evaluations are related to the problem context. Metaheuristics are measured according to both the quality of the optimal solution and the computational time together. The representation of the balance for minimization problem is shown in Figure 13.

Figure 13: Performance profiles of two hypothetical metaheuristic procedures



Source: Duarte et al., 2018: 30

According to Figure 13, one can conclude that when the optimization horizon is short it would be better to prefer less aggressive but diversified algorithms, whereas when the optimization horizon gets longer aggressive and less diversified algorithms to provide better solutions for minimization problems. Although it is known that the

balance between intensification and diversification is very important, there is no definite way to measure objectively the rate of exploration and exploitation (Morales-Castañeda et al., 2020: 1).

1.4.9. Stopping Condition

In each iteration, new populations that lead the algorithm to reach an optimal-like solution are generated until the termination condition is satisfied. Bozorg-Haddard et al. (2017) summarized prevalent stopping criteria as the predefined number of iterations, the threshold of improvement in the fitness function, or the run time. Moreover, some statistics which show the diversity of the population are also utilized for stopping criteria as well (Talbi, 2009: 199). Apart from these, many termination conditions depending on some special feature may be implemented in algorithms.

1.4.10. Model Reliability

As mentioned in the title of Model-identification, it is not possible to interpret by looking at the result obtained in a single experiment in algorithms that offer a near solution. For this reason, it has made many trials and the summary statistics of these trials give an idea about the performance of the algorithm. The number of trials depends on statistics. This can be explained by the law of large numbers, which guarantees stable long-term results for averages of some random events (Dekking et al., 2005: 181). Trials such as 30, 50, and 100 are reported in the literature. However, achieving good results with less trial can also be interpreted as good in terms of performance. However, this must be obtained with the same trial number for each run.

Reliability shows the extent to which the algorithm can result in acceptable results. The percentage of trials terminated by converging the best fitness value under a predefined threshold value can be considered as a success rate for reliability (Du and Swamy, 2016: 20). A reliable algorithm is defined as converging similar near-optimal solutions with approximately the same function evaluation values which shows small variances among trials (Bozorg-Haddad et al., 2017: 38). Moreover, each experimental result can be reported by the best-so-far, the worst solutions, mean and standard

deviations to provide information about how the results distribute around the mean. By considering these descriptive statistics, information about how robust solutions the algorithm provides in each trial can be obtained.

Furthermore, the efficacy and efficiency of the algorithm should also be considered as performance indicators. The average of the best fitness values in the last population over all trials is used as an absolute measure for efficacy. Whereas efficiency is calculated by the average number of function evaluations for success which implies the speed for convergence (Du and Swamy, 2016: 20). Although a high success rate and low mean best fitness mean that the algorithm is reliable, there is a possibility to face bad results accidentally too. For this reason, it would be better to obtain a small mean best fitness value with low function evaluations in almost every trial which is challenging.

1.4.11. Model Validity

Validation which is another concept in the field of science is an indicator of the accuracy of the model. When this concept is evaluated in terms of algorithms, it can be considered as to whether the algorithm finds the right solution for the related problem. Especially in multimodal problems which have several local optimums, the algorithms may be trapped into the local optimums. Considering that this situation occurs in almost every trial, although the performance of the algorithm is evaluated as reliable, it is impossible to conclude that the algorithm is valid too. Since the algorithm cannot solve the problem correctly, which means that premature convergence, the algorithm is considered invalid. In other words, validity means “build the right model”, whereas reliability means “build the model right”. Moreover, according to Smit and Eiben (2010), the scope of validity is defined as the range of problems (a set of problems, more problems of a certain type, or all problems) that the algorithm can solve.

CHAPTER TWO
LITERATURE REVIEW OF PHYSIC BASED ALGORITHMS

As mentioned in the historical background section, with the theories developed by Newton, Gauss, Lagrange, and Fermat in the early period, Kantorovich applied linear programming to a production problem in 1939. According to Biswas et al. (2013), every technological development has to reach optimality in terms of time and complexity and the researchers have to propose algorithms that provide the best possible or better solutions. For that reason, seeking a better algorithm continues and new inspirations come to exist in literature.

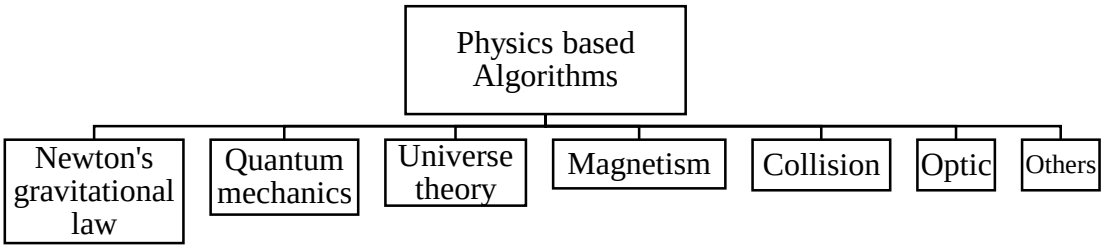
Holzner (2011) defined physics as all-encompassing science and clarified that the name of physics is derived from the Greek word "physika" which has the meaning of "natural things". According to Xing and Gao (2014), physics is the most fundamental science and it focuses on the basic principles of the universe. Physics includes various aspects of the natural world such as the objects in motion, energy, forces, gases, heat, and temperature. All these topics are the basis of the physics-based algorithms in the literature.

The origins of physics-based algorithms date back to the early periods. Biswas et al. (2013) asserted that physics-inspired optimization algorithms were paved the way by Feynman's proposal of a quantum computing system in 1982. On the other hand, in 1983 the Simulated Annealing algorithm inspired by the annealing process of metal was presented. However, this chapter addresses the developed algorithms especially in recent years and their common and distinctive features are discussed.

2.1. THE CLASSIFICATION OF PHYSICS-BASED ALGORITHMS

According to Sotoudeh-Anvari and Hafezalkotob (2018), physics-based algorithms are the most popular themes after animal-based algorithms. The classification of the physics-based algorithms adapted from Biswas et al. (2013) can be conducted as shown in Figure 14.

Figure 14: Classification of Physics-based Algorithms

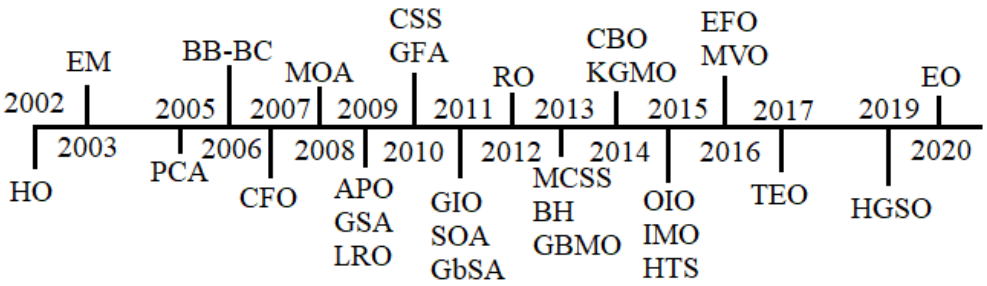


Source: Adapted from Biswas et al., 2013

Physics-based algorithms can be classified and grouped in a structured way. The metaphors used in each algorithm can be clarified by using main subjects as keywords. By doing so, the similarities and the differences in algorithms are put forward. Until today, numerous physics-based algorithms have been proposed to literature, the principles, and main subjects are given in the following section.

2.2. PHYSIC-BASED ALGORITHMS

Figure 15: Historical Perspective of Physics-based Algorithms



Source: Prepared by the author

Although the basis of physics-based algorithms dates back to Metropolis Algorithm (Metropolis et al., 1953) and Simulated Annealing (Kirkpatrick et al., 1983), as of 2000, algorithms with a wide range of inspiration are derived. After the pioneering physics-based algorithms, the first modern pure physics-based algorithms were developed in 2002 Hysteretic Optimization (Zarand et al., 2002) and Electromagnetism-like Optimization (Birbil and Fang, 2003) in 2003. In the following

section, the principles will briefly be mentioned and then the algorithms based on that principle will be reviewed.

2.2.1. Newton's Gravitational Law

The algorithms inspired by Newton's Gravitational Law are given in Table 3 in chronologic order. Each algorithm will be handled according to the common principles and distinctive features respectively and the application studies will be presented.

Table 3: Metaheuristic Algorithm Inspired by Newton's Gravitational Law

Algorithm	Main Subjects	Author(s)
Central Force Optimization (CFO)	Particle – Mass – Attraction	Formato (2007)
Artificial Physics Optimization (APO)	Particle – Mass – Hypothetical – Attraction – Repulsion	Xie et al. (2009a)
Gravitational Search Algorithm (GSA)	Particle – Mass – Attraction – Variable Hypothetical Gravity	Rashedi et al. (2009)
Gravitational Interaction Optimization (GIO)	Particle – Mass – Constant Gravity – Interaction	Flores et al. (2011)

2.2.1.1. Central Force Optimization

Central Force Optimization (CFO) developed by Formato (2007) is the first algorithm that is inspired by Newton's Gravitational Law. It focuses on particle kinematics in a gravitational field and utilizes particles' masses for the attraction between the particles in a deterministic way. In this principle, large particles attract small particles towards it, and large particles reflect the best possible solution. In this algorithm, each particle relocates by the masses of the other particles. However, the acceleration also depends on two parameters (α, β) which are not represented in nature but help the algorithm for a better exploration or exploitation. Moreover, this algorithm uses the difference between masses of particles instead of mass which eliminates the excessive attractions. Another important feature of CFO is that it does not let the particles repulse each other. Although this algorithm provides new insights into metaheuristics, the author left some questions behind the study. Thus, after publishing that article, the remaining questions have been answered one by one. For example,

Formato (2010a) proposed a modified CFO algorithm that works in the principle of parameter-free to avoid local trapping. Afterward, Formato (2011) introduced an improved CFO algorithm that provides better sampling and periodically shrinking decision space that is located around the best fitness. Moreover, Formato (2013) explained the concept of pseudo-randomness in CFO. In stochastic algorithms like PSO and ACO, their formulation is constituted true random variables, whereas in CFO the equations are completely deterministic. However, having a pseudorandom component in the algorithm provided better implementations although it is not required. Antenna optimization (Formato, 2010b; Qubati et al., 2010), design of multilayer microwave absorbers (Asi and Dib, 2010), leakage freshwater detection (Haghighi and Ramos, 2012), optimizing the location of diffusion spots (Eltokhey et al., 2016), water distribution network optimization (Jabbary et al., 2018), data clustering (Srinivasa Rao et al., 2015; Javadi and Zahiri, 2018) are some of the applications of CFO and its variants. A more detailed review of the CFO algorithm can be found in (Siddique and Adeli, 2015b).

2.2.1.2. Artificial Physics Optimization

Artificial Physics Optimization (APO) based on Physicomimetics is also inspired by Newton's Gravitational Law and it is developed by Xie et al. (2009a). In this algorithm, virtual physical forces drive a multi-robot system and each robot represents a physical particle. Different from the CFO algorithm, APO is structured as both attraction and repulsion forces. However, there is a threshold for this motion which is the radius of the particles. In addition to the distinctive feature, the power of the distance parameter used in Newton's Gravitational Law is flexible between -5 to 5 in the APO algorithm. Moreover, the APO algorithm allows all particles to control the velocity of the related one until there are no forces. The masses of particles are calculated using the best and the worst particle function values. The velocity and the positions of particles depend on two parameters; one is a number distributed randomly between $[0, 1]$, the other is adaptively changing from 0 to 1 according to the maximum iteration number. In line with the aim of science, improvements have been made in this algorithm that provides better results. For example, Xie et al. (2009b) developed

an APO algorithm that provides alternatives for mass functions (convex, linear, or concave) and concluded that an algorithm that utilizes concave mass function performs better. Furthermore, Xie and Zeng (2010) tested the APO algorithm with three different force laws (negative exponential, unimodal, and linear) and found out that linear force law resulted in more effective solutions. Ates and Yeroglu (2018) modified the APO algorithm by combining it with Base Optimization Algorithm for multi-parameter function minimization. The authors assert that the modified APO algorithm provides various mass-cost function combinations which results in fast convergence and less computational costs. Lastly, Aljohani et al. (2019) published very recently a hybrid algorithm that uses APO and Particle Swarm Optimization (PSO) together. They concluded that the new hybrid algorithm has an excellent search capacity and by doing so, being trapped in local optima can be overcome. Generally, engineering problems are encountered as real-world problems for APO and its developed versions such as hyperspectral imagery band selection (Wang and Wei, 2013), vulnerability assessment and reconstruction of micro-grid (Xie and Ma, 2018), security-constrained optimal power flow problem (Teeparthi and Kumar, 2018), spectrum detection problem in cognitive Internet of Things (Li et al., 2019).

2.2.1.3. Gravitational Search Algorithm

Gravitational Search Algorithm (GSA), the most popular algorithm with thousands of citations in Scopus, is developed by Rashedi et al. (2009). The main principles of this algorithm are Newton's Gravitational Law and Law of Motion. Differently from the other algorithms, each particle has four specifications as position (solution), inertial mass (resistance against its position changing), active and passive gravitational masses which represent force intensity now and previously (Zandevakili et al., 2019). Moreover, GSA assumes that any particle can attract any other particles in terms of their masses but inversely proportional to the Euclidean distance rather than distance square. As in the APO algorithm, the masses of particles are calculated using the best and the worst particle function values, but in a different form. Additionally, the forces on each particle depend on both passive and active gravitational masses which are assumed as equal, distance and gravitational constant

as well. However, in GSA gravitational constant is a function based on the total number of iterations and time and decreases with time like temperature. With this distinctive feature, the algorithm provides better exploitation. On the other hand, it uses random parameters for exploration and velocity. Furthermore, the authors also assert that GSA is a memoryless algorithm but works as effective as algorithms with memory. The algorithm is not only in its original form, but many new versions have been developed. For instance, Rashedi et al. (2010) modified GSA as for binary variables, Sarafrazi et al. (2011) integrated disruption operator which is inspired by astrophysics to GSA for a better exploitation and exploration ability, Zhang et al. (2011) improved GSA by immune system in biology and with the help of different inspiration falling into local optimum problem has been eliminated, Mirjalili and Lewis (2014) ameliorate slow exploitation which causes major weakness by adding memory to GSA and by doing so the exploitation phase is accelerated, Gao et al. (2014) improved GSA as for exploitation ability and local optima problem by integrating chaos which provides ergodicity and stochasticity, to eliminate trapping local optima problem, Huang and Qin (2019) used clustering method to divide whole population, Khan et al. (2019) modified exploitation strategy which provide adaptive velocity in terms of the number of iterations whereas Zheng et al. (2019) proposed improved GSA by adjusting gravitational constant. As for the recent applications of GSA and its modified versions, Sharma and Kumar (2017) applied discrete GSA for virtual machine placement in cloud computing, Priya et al. (2020) preferred to use GSA as a feature selection tool for phishing datasets, Han et al. (2020) used binary GSA, Shukla et al. (2020) integrated GSA with Teaching learning-based algorithm for cancer classification, Mosa (2020) hybridized Particle Swarm Optimization and GSA for mining social media data. For more application examples, Sabri et al. (2013) reviewed GSA and its applications comprehensively.

2.2.1.4. Gravitational Interactions Optimization

Another Newton's Gravitational Law inspired algorithm called Gravitational Interactions Optimization is developed by Flores et al. (2011). Unlike GSA, the GIO algorithm utilizes the Gravitational constant as a constant number because the opposite

of it destroys the underlying interaction metaphor. Besides, the GIO algorithm allows each particle to interact with all others and uses particles' masses for fitness values. Although mass functions are similar to GSA, GIO limits fitness values with an interval. Moreover, the authors asserted that the GIO algorithm provides all local and global optimums for multimodal functions instead of just global optimum. Also, GIO does not need a radius parameter or a maximum number of particles as in PSO. However, GIO has not received enough attention and has not been applied in engineering problems.

2.2.2. Magnetism

The algorithms inspired by magnetism are given in Table 4. Each algorithm will be handled according to the common principles and distinctive features respectively and application studies will be presented.

Table 4: Metaheuristic Algorithms Inspired by Magnetism

Algorithm	Main Subjects	Author(s)
Hysteretic optimization (HO)	Material – Energy – Magnetic - Glass demagnetization	Zarand et al. (2002)
Electromagnetism-like mechanism (EM)	Particle – Charge – Distance - Attraction	Birbil and Fang (2003)
Magnetic Optimization Algorithm (MOA)	Magnetic field - Particles - Distance	Tayarani and Akbarzadeh (2008)
Charged System Search (CSS)	Particle – Charge – Electrostatics – Attraction – Velocity - Force	Kaveh and Talatahari (2010a)
Magnetic Charged System Search (MCSS)	Magnetic forces - Particles - Attraction - Repulsion -Absorbing	Kaveh et al. (2013a)
Electromagnetic field optimization (EFO)	Attraction - Repulsion - Electromagnets	Abedinpourshotorban et al. (2016)

2.2.2.1. Hysteretic Optimization

Hysteretic optimization (HO) is one of the algorithms published in the very beginning and it is inspired by the magnetism concept in physics (Zarand et al., 2002: 1). It depends on the demagnetization of magnetic materials with changing the external field of decreasing amplitude (Pál, 2003: 287). The principle of gradually decreasing and reaching a steady-state reminds Simulated Annealing (SA) which is inspired by

the annealing processes of metals. However, this time, the focused object has a magnetic property and it is left to a magnetic field with a decreasing amplitude for reaching the global optimum that is the minimum level of magnetism. In this process, not only magnetism is lowered, but also a low energy level is reached. As for Pál (2006), the HO algorithm can be improved by repeatedly shaking up the system but with a smaller maximum amplitude. Although HO is one of the pioneers in Physic-based algorithms, it does not attract much attention after 2012. However, it is used for traveling salesman problem (Pál, 2003), spin glasses application (Gonçalves and Boettcher, 2008), capacitated vehicle routing problem (Yan and Wu, 2012), 3D protein folding problem with lattice model (Xiong et al., 2012).

2.2.2.2. Electromagnetism-like Mechanism

Birbil and Fang (2003) developed an algorithm inspired by electromagnetism theory and that method utilizes an attraction-repulsion mechanism as also mentioned in Newton's Gravitational Law section. However, in EM algorithm electrical forces are used instead of masses. The total electrical force on a particle is calculated vectorially according to the charges of the other particles in the population and the distances between the particles (Yurtkuran and Emel, 2010: 3428). Therefore, it is clear that the total electrical force determines the magnitude of attraction or repulsion and the higher the magnitude means the better the objective function value. Although the EM algorithm seems like other physics-based algorithms in some points, the authors asserted that EM provides less execution time and memory usage. The algorithm allows both attraction and repulsion of the particles moving in search space. The overall electrical charges on a particle are calculated in terms of Coulomb's law (Cuevas et al., 2012: 43). The authors assert that the algorithm converged global optimum on average 25 iterations per dimension (Birbil and Fang, 2003: 270). A year after, Birbil et al. (2004) modified EM by considering convergence with probability one. Although this algorithm can be used as a stand-alone approach, scholars have also hybridized the EM algorithm with other metaheuristics. For example, EM has been integrated with simulated annealing (Tavakkoli-Moghaddam et al., 2009; Naderi et al., 2010; Jamili et al., 2011), particle swarm optimization (Tian et al., 2009; Lee and Lee,

2012), differential evolution (Muhsen et al., 2015), firefly algorithm (Le et al., 2019), tabu search (Sels and Vanhoucke, 2014), back-propagation algorithm (Lee et al., 2012). Moreover, the EM algorithm has also attracted attention in the literature as a field of application. Traveling salesman problem (Javadian et al., 2008), flow shop scheduling (Yan et al., 2014), inventory control (Tsou and Kao, 2008), array pattern optimization (Jhang and Lee, 2009), course timetabling problem (Abdullah et al., 2012), layout problem (Jolai et al., 2012), job shop problem (Tavakkoli-Moghaddam et al., 2009), feature selection (Su and Lin, 2011), single machine scheduling problem (Chang et al., 2009), capacitated vehicle routing (Yurtkuran and Emel, 2010), automatic circle detection (Cuevas et al., 2012) are some of the most cited application studies in the literature.

2.2.2.3. Magnetic Optimization Algorithm

Magnetic Optimization Algorithm (MOA) is an algorithm inspired by magnetic field theory and it was firstly introduced in 2008 by Tayarani and Akbarzadeh. Differently from the other magnetism-inspired algorithms, in MOA each particle is located in the algorithm by its mass and magnetic field. Moreover, all particles are in a lattice-like environment for better exploitation and the force on a particle is limited to only a certain number of neighbors rather than all other particles to decrease the complexity (Tayarani and Akbarzadeh, 2008: 2664). Also, unlike algorithms based on gravity, MOA takes into account both repulsive and attractive forces. However, the authors mentioned that the attraction works in long-range force, whereas repulsion in short-range (Tayarani and Akbarzadeh, 2014: 83). In MOA when the distances between the particles reach infinity, the forces disappear on a particle (Kushwaha et al., 2018: 60). Tayarani and Akbarzadeh (2014) published a revised version of MOA which explains the algorithm in detail and includes an extension. Aziz and Tayarani (2016) improved MOA performance by considering the experience of the algorithm. Thereafter, Sadiq et al. (2018) integrated MOA with Particle Swarm Optimization for more accuracy and efficiency. Kushwaha et al. (2018) applied MOA for data clustering. Kushwaha and Pant (2018) modified the magnetic optimization clustering algorithm with fuzzy logic and applied it in the health care field.

2.2.2.4. Charged System Search

Kaveh and Talatahari (2010a) developed a method called Charged System Search (CSS). The principle of the method is based on physics and mechanics. The forces between the charged particles are calculated by utilizing Coulomb's law and Gauss laws, whereas the behavior of the particles is determined by Newtonian laws (Kaveh and Talatahari, 2012: 382). CSS considers a population of solutions and allows any particle to attract others according to the self-requirements. In addition, to provide optimal balance in intensification-diversification, the algorithm can remember the past. The authors mentioned that CSS provides good solutions especially when the domains are non-smooth and non-convex. Thereafter CSS became the center of attention and many scholars developed CSS to achieve better results. Kaveh and Talatahari (2011) improved CSS by introducing the concept of "field of forces" which provides a general model of physics-based algorithms. Kaveh and Ahmadi (2013) included supervisor agents in the CSS algorithm for a better exploration ability. Kaveh et al. (2013a) integrated magnetic forces besides electrical forces for enhancing the performance of the CSS algorithm. Chu and Tsai (2013) modified CSS by determining a moving strategy to solve the distribution system loss minimization problem. Precup et al. (2014) proposed an adaptive CSS that has engagement, exploration, explanation, elaboration, and evaluation stages. Niknam et al. (2014) implemented a self-adaptive reformation technique to achieve better performance and high speed. Prasad and Vinod Kumar (2017) developed a rule-based fuzzy inference system to control the parameters of CSS. In addition to being developed with the help of additional features, CSS has been presented to the literature by hybridizing it with different algorithms as well. For example, CSS hybridized with PSO (Talatahari et al., 2013; Kaveh and Laknejadi, 2011; Kaveh and Talatahari, 2014); Bayesian Optimization Algorithm (BOA) (Aryan and Alizadeh, 2016); Harmony Search (HS) (Kaveh and Hosseini, 2012); Colliding Body Optimization (CBO) (Khanzadi et al., 2016; Shirgir et al., 2020); Big Bang-Big Crunch (Talatahari et al., 2018). After introducing CSS and its extensions to the literature, they have been applied in many fields. Kaveh and Talatahari (2010b) applied CSS for the optimal design of skeletal structures, Özyön et al. (2012) utilized CSS algorithm for economic power dispatch problem with emission constrained, Kumar

and Sahoo (2014) found optimal cluster centers by CSS, Kaveh and Behnam (2013) provided an optimal design of reinforced concrete cantilever retaining walls by minimizing cost, Precup et al. (2015) used adaptive CSS for finding an optimal path for mobile robots, Kaveh et al. (2018a) compared CSS with MCSS by applying them to site layout planning problem, Khanzadi et al. (2016) provided a solution for resource allocation and resource leveling problem with CSS, Kaveh and Zolghadr (2015) used an improved version of CSS to detect damages in truss structures, Akbari et al. (2020) applied CSS and EM for fixed-charged solid transportation problem in supply chain network.

2.2.2.5. Magnetic Charged System Search

MCSS is a modified version of CSS as mentioned briefly in the CSS part. Different from CSS, MCSS considers magnetic forces as well by utilizing Biot-Savart Law. In physics, a magnetic field occurs when a charged particle moves. Kaveh et al. (2013a) developed the MCSS algorithm to make an algorithm closer to the nature of the movements of charged particles. In MCSS, the magnitude of the magnetic force on a particle is proportional to the charge and the speed of the particle. Moreover, this algorithm assumes that each particle can move in virtual straight wires. Although only attraction force is allowed in CSS, a repelling force is also added with a probability in terms of electrical forces. Besides, the magnetic force includes both attraction and repulsion forces to obtain better searchability. It would be better to clarify that, when the objective function value of a particle increases, that particle absorbs other particles. Furthermore, MCSS allows changes continuously within an iteration, rather than at the end of the iteration. According to the comparison results conducted by Kaveh et al. (2013a), the difference between CSS and MCSS becomes more obvious, when the number of particles (the dimension of the problem) is small. On top of all these improvements, other researchers have also worked on the MCSS and presented new versions. Kaveh et al. (2014a) developed an improved MCSS which includes an improved harmony search scheme for position correction and more effective convergence parameters. Kaveh et al. (2015a) modified MCSS by hybridizing MCSS and improved the scheme of the harmony search algorithm. D'Ambrosio et al. (2020)

modified MCSS by introducing a self-adaptive parameter setting and a chaotic local search for better performance. Moreover, MCSS has been also applied in various fields such as construction project planning (Tavakolan and Share, 2013); phase stability analysis, and phase equilibrium calculations (Elnabawy et al., 2014); damage detection (Kaveh and Maniat, 2015).

2.2.2.6. Electromagnetic Field Optimization

Abedinpourshotorban et al. (2016) proposed EFO inspired by the behavior of electromagnets. Different from the other algorithms based on magnetism, EFO takes the advantage of the golden ratio for the attraction-repulsion ratio to converge better. However, it would be better to clarify that, the attraction force is stronger than the repulsion force. Moreover, this algorithm divides the whole population into three subfields as positive, negative, and neutral which lead particles to the optimum solution. Although EFO is relatively new, its modifications have been presented in the literature as well. For example, Bouchekara et al. (2017) modified EFO in terms of the distribution of the random number generation and the rule in case of crossing the boundaries of search space. Song et al. (2019) improved EFO by implementing a fuzzy entropy criterion and by embedding a chaotic strategy to enhance searchability. Yurtkuran (2019) proposed two novel modifications for generating new solution principles and adaptive control of parameters. Bouchekara (2020) presented a new chaotic EFO algorithm that considers chaotic numbers rather than random and which provides a new generating procedure. In addition to introducing new versions, EFO has also found application areas. Talebi and Dehkordi (2018) utilized the EFO algorithm for sensitive association rules hiding in the data mining field; Kushwaha et al. (2018) enhanced the EFO algorithm for clustering; Şahin and Kellegöz (2019) applied the EFO algorithm for multi-manned assembly line balancing problem by modifying regeneration strategy.

2.2.3. Collision

Different from the algorithms inspired by Newton's Gravitational Law, the algorithms inspired by collision behaviors are given in Table 5. Each algorithm will be handled according to the common principles and distinctive features, respectively.

Table 5: Metaheuristic Algorithms Inspired by Collision

Algorithm	Main Subjects	Author(s)
Particle Collision Algorithm (PCA)	Nucleus - Absorption - Scattering	Sacco and De Oliveira (2005)
Colliding Bodies Optimization (CBO)	Colliding bodies – masses – velocity	Kaveh and Mahdavi (2014)
Kinetic Energy of Gas Molecules (KGMO)	Gas molecules - Collision - Kinetic Energy - Velocity	Moein and Logeswaran (2014)

2.2.3.1. Particle Collision Algorithm

PCA is a stochastic single-solution optimization algorithm developed by Sacco and De Oliveira (2005). It is inspired by the physics of nuclear particle collision reactions (Duderstadt and Hamilton, 1976: 3). The mechanism of PCA shows that when the particle moves towards to nucleus which has a high fitness value, the nucleus would absorb that particle, on the other hand when the particle hits a low fitness nucleus, it would be scattered to another region. This information means that PCA handles exploitation and exploration abilities with absorption and scattering. Moreover, PCA resembles the Metropolis algorithm in terms of the ability of the acceptance of a candidate solution with a certain probability and Simulated Annealing (SA) in terms of its structure. Since the early version of PCA is primitive, other modified versions have been presented in literature throughout the years. Sacco et al. (2007) published a population-based PCA that is hybridized with the Genetic Algorithm. Da Luz et al. (2008) proposed a multi-particle collision algorithm (M-PCA) that is based on canonical PCA by using several parameters instead of one. Abuhamdah and Ayob (2009) developed Multi-Neighborhood PCA which exploits hill-climbing local search and two-staged neighborhood structure. Sacco and Rios-Coelho (2016) introduced an enhanced PCA algorithm called Cross-Section PCA

which uses the probability of interaction between a neutron and target nucleus. Torres et al. (2018) hybridized Multi-Particle PCA with Rotation-Based Sampling that provides better global search. As for application studies, Da Luz et al. (2011) applied Multiple PCA to Radiative Transference and Pollutant Localization Inverse problems; Abuhamdah and Ayob (2011) solved course timetabling problem with Multi Neighborhood PCA and Adaptive Randomized Descent Algorithm; Domiciano et al. (2018) utilized PCA for automatic digital elevation model from images.

2.2.3.2. Colliding Bodies Optimization

Kaveh and Mahdavi (2014) proposed a single-solution CBO inspired by the law of momentum and energy. In this algorithm, colliding bodies are considered with their masses, besides there are no external forces on these bodies. In addition, colliding bodies have two subgroups (stationary and moving objects) (Kaveh and Mahdavi, 2016: 14). It would be better to mention that only collisions between moving and stationary bodies are allowed. Nevertheless, even stationary bodies move towards to better positions through that collision (Siddique and Adeli, 2016: 616). After the development of the algorithm, different versions of CBO have started to be introduced. Kaveh and Ghazaan (2014) implemented a memory to CBO and added a stochastic mechanism to escape from local optima. Bouchekara et al. (2016) proposed an improved CBO which has the number of colliding bodies as three instead of two. Panda and Pani (2016); Kaveh and Mahdavi (2019) modified CBO for multi-objective problems. Cheng and Zhao (2020) proposed Chaotic Enhanced CBO by changing the generation pattern of bodies which yields better convergence. Chen et al. (2020) hybridized CBO with Teaching-Learning based optimization algorithm to provide high convergence speed and to eliminate premature convergence. CBO and its extensions have also been applied in the engineering literature, for example, cost optimization of a concrete ribbed slab (Kaveh and Bijari, 2014); clustering model for seismic catalog (Nanda and Panda, 2015); resource allocation (Kaveh et al., 2015b; Khanzadi et al., 2016); optimal power flow problem (Pulluri et al., 2016); cost and CO_2 emission optimization (Kaveh, 2017); optimization of fractional order PID controller (Asl et al., 2018); construction site layout (Kaveh et al., 2018b); designing

a microwave filter (Gupta et al., 2020a); structural reliability analysis (Cheng and Zhao, 2020).

2.2.3.3. Kinetic Energy of Gas Molecules

Moein and Logeswaran (2014) developed an algorithm inspired by the behavior of gas molecules. In the KGMO algorithm, each gas molecule is handled with its position, kinetic energy, velocity, and mass. The velocity and the position are determined according to the kinetic energy of the molecule. The gas molecules are moving until reaching the lowest temperature and kinetic energy. Besides, movements depend on Newton's law and molecules are allowed to move in a straight line. Unlike algorithms based on Newton's gravitational law and magnetism, there are no attractive or repulsive forces between molecules, and energy is not gained or lost as a result of a collision. When the literature studies are reviewed, an algorithm that improves KGMO has not been encountered yet. However, there are hybrid studies made with KGMO. Moein et al. (2016) hybridized KGMO with Neural Network (NN) for the detection of heart disorders; Vinay Kumar et al. (2019) utilized KGMO with fuzzy c-means clustering; Hemachandra Reddy et al. (2019) proposed a hybrid algorithm that includes the KGMO and PSO; Asha and Gowrishankar (2020) presented an efficient clustering and routing algorithm that covers both KGMO and Glowworm Swarm Optimization. Moreover, scholars also applied the KGMO algorithm in various problems such as nonconvex economic dispatch problem (Basu, 2016); allocation problem in engineering (Panthagani and Rao, 2017); performance optimization (Reddy and Reddy, 2019); optimal reactive power dispatch problem (Panthagani and Rao, 2020).

2.2.4. Quantum Mechanics

Different from the other physics-inspired algorithms, Quantum Mechanics (QM) inspired algorithms have been developed as hybrid algorithms. These algorithms are the combinations of quantum computing and metaheuristics (Karmakar et al., 2018: 272). Quantum Computing (QC) has been one of the most interesting topics after the studies conducted by Feynman (1982) (Dey et al., 2016: 678). Moreover, Dey

et al. (2016) also mentioned that because of the parallelism capability of QC that reduces complexity, it has become popular in various fields like engineering, artificial intelligence, and so on. It would be better to clarify that quantum metaheuristics and quantum-inspired metaheuristics do not represent the same concepts. Quantum metaheuristics are algorithms executed in a quantum computer, whereas quantum-inspired metaheuristics are developed as algorithms that imitate the principle of quantum physics for classical computers (Ross, 2020: 815).

Unlike classical computing, a quantum bit (Q-bit) can be represented by 0, 1, or both which is called a superposition state. This state simulates the superposition of electrons. In Quantum mechanics, electrons are moving in orbits according to their angular momentum and energy level. A lower energy electron can jump to another orbit that has a higher level by absorbing the energy and the visa versa is possible as well. The orbit which includes electrons can be known with a probability and this situation can be expressed with a superposition state. Therefore, the position of an electron is defined as a quantum state vector with Q-bits (Biswas et al., 2013: 6).

Since the principle of QM has not been utilized as single-handed in literature, the Quantum-inspired algorithms can be considered as semi-physics-based metaheuristics in literature. Indeed, QC is preferred when the metaheuristics are unable to handle some NP-Hard problems. Moore and Narayanan (1995) presented the potential use of QC in NP-Hard problems.

Numerous studies are proposed algorithms hybridized with QM in literature. The review studies focus on quantum-inspired metaheuristics on a specific subject that can be found in Dey et al., 2014; Dey et al., 2016; Mozaffari et al., 2017; Karmakar et al., 2018; Ross, 2020. In Table 6, pioneer Quantum-inspired metaheuristics are given.

Table 6: Metaheuristic Algorithms Inspired by Quantum Mechanics

Quantum-inspired Algorithm	**Author(s)**
Q - Genetic Algorithm (GA)	Narayanan and Moore (1996)
Q - Evolutionary Algorithm (EA)	Han and Kim (2002)
Q - PSO	Sun et al. (2004)
Q - Immune clonal algorithm (ICA)	Li and Jiao (2005)
Q - Ant Colony Optimization (ACO)	Wang et al. (2007)
Q - GSA	Su and Yang (2008)
Q - Differential Evolution (DE)	Su and Yang (2008)
Q - Artificial Immune System (AIS)	Gao et al. (2010)
Q - Tabu Search (TS)	Chiu et al. (2011)

Q - Cuckoo Search Algorithm (CSA)	Layeb (2011)
Q - Firefly Algorithm (FA)	Manju and Nigam (2012)
Q - EM	Chou et al. (2012)
Q - Bacterial Foraging Algorithm (BFA)	Huang and Zhao (2012)
Q - Harmony Search (HS)	Layeb (2013)
Q - Artificial Bee Colony (ABC)	Bouaziz et al. (2013)
Q - Cultural Algorithm (CA)	Guo and Zhang (2015)
Q - Glowworm Swarm Optimization (GS)	Gao et al. (2017)
Q - Bat Optimization (BO)	Dey et al. (2019a)
Q - Dolphin Swarm Algorithm (DSA)	Qiao and Yang (2019)
Q - Binary Wolf Pack Algorithm (BWPA)	Gao et al. (2019a)
Q - Teaching-Learning-Based Optimization (TLO)	Gao et al. (2019b)
Q - Sperm Whale Algorithm (SWA)	Dey et al. (2019b)

When the list is examined, it is clear that generally animal-based algorithms are modified by quantum mechanics and these algorithms can be considered as semi-physics. However, there are also physics-based algorithms such as SA, GSA, EM that are hybridized with QM for better performance.

2.2.5. Universe Theory

The algorithms inspired by the universe theory are given in Table 7. Each algorithm will be handled according to the common principles and distinctive features respectively and application studies will be presented.

Table 7: Metaheuristic Algorithms Inspired by Universe Theory

Algorithm	Main Subjects	Author(s)
Big Bang- Big Crunch (BB-BC)	Center of mass – Collapse – Blackhole	Erol and Eksin (2006)
Gravitation Field Algorithm (GFA)	Solar Nebular Disk Model - Mass - Dust - Motion - Absorption	Zheng et al. (2010)
Spiral Optimization Algorithm (SOA)	Logarithmic Spiral	Tamura and Yasuda (2011a; 2011b; 2011c)
Galaxy-based Search Algorithm (GbSA)	Spiral movements - Galaxies	Shah-Hosseini (2011a; 2011b)
Black Hole Algorithm (BH)	Stars – Black Hole	Hatamlou (2013)
General Relativity Search Algorithm (GRSA)	Velocity – Curved Space – Geodesic Trajectory	Beiranvand and Rokrok (2015)
Multi-verse optimizer (MVO)	White hole - Black hole - Wormhole - Colliding	Mirjalili et al. (2016)

2.2.5.1. Big Bang- Big Crunch

The universe was born after the Big-Bang. However, this leads us to be curious about the ultimate fate of the universe. According to Einstein's Theory of Relativity, the ultimate fate of the universe depends on the density of mass and energy in the universe. This means that the universe continues to exist when there is enough attraction to expand by getting colder and result in a single point called Big Crunch (Holzner, 2011: 358). Erol and Eksin (2006) developed the BB-BC optimization algorithm that is inspired by the theories of the evolution of the universe. The Big-Bang phase is constructed to obtain randomness that is a result of energy dissipation. Moreover, candidate solutions scatter to search space with uniform distribution. Thereafter, the Big Crunch phase is constituted for intensification. In this part, the center of mass is calculated which represents the inverse of the fitness function value. Furthermore, the center of mass is obtained as a result of the collision of all masses and it has gravitational force for the attraction (Siddique and Adeli, 2016: 607). Eventually, this algorithm iterates between these two phases. Afterward, modified versions of BB-BC have been proposed in the literature. Kripka and Luvezute Kripka (2008) proposed Big Crunch optimization that works with a population of variables; Kaveh and Talatahari (2010c) modified BB-BC for discrete problems; Alatas (2011) utilized chaos for speed convergence and uniform population method; Hasançebi and Azad (2012) modified BB-BC by utilizing exponential distribution in random number generation; Sedighizadeh and Ghalambor (2014) presented a modified BB-BC for reconfiguration of multi-objective distribution networks in fuzzy sets; Kumar et al. (2018) developed multi-population version of BB-BC; Yin et al. (2018) improved BB-BC by changing the exploding radius and generating multiple mass centers; Bijari et al. (2018) improved BB-BC by implementing enriched memory; Yalcin and Pekcan (2020) developed a derivative version of BB-BC called Nuclear Fission- Nuclear Fusion. Also, applications of BB-BC and its modifications in various fields have been published, such as the design of space trusses (Camp, 2007; Kaveh and Talatahari, 2009); economic dispatch (Labbi and Attous, 2010; Rao and Yesuratnam, 2015; Labbi and Attous, 2017; Ieng et al., 2019); scheduling (Jaradat and Ayob, 2010; Kang et al., 2016); data clustering (Hatamlou et al., 2011; Bijari et al., 2018); damage detection

(Tabrizian et al., 2013); path planning (Yılmaz and Gökaşan, 2015); cloud computing (Mahdavi and Ghaffari, 2019); passive building design (Robic et al., 2020).

2.2.5.2. Gravitation Field Algorithm

Zheng et al. (2010) developed GFA which is an algorithm inspired by the famous astronomy theory of planetary formation called the Solar Nebular Disk Model. According to that theory, everything was a cloud of dust in the beginning. Later, this dust cloud formed the rocks by gravitational force. Throughout this process, big rocks grabbed small rocks and finally planets took place. From this point, GFA imitates that process by considering the dust cloud as candidate solutions and the planet as a global optimal solution. Moreover, assembling of the dust clouds occurs by taking into account the assigned masses and the power of attraction caused by the other clouds of dust. In the following process, the algorithm has been improved. Zheng et al. (2012a) modified GFA in terms of both the rule of random division and rotation factor. Rong et al. (2013) proposed Parallel GFA that is based on the island model for better computing ability. Hu et al. (2019) implemented dust sampling that can locate more likely the space containing the optimal solutions and explosion operator for better accuracy. GFA and its other versions have not been applied in various fields yet. However, the reconstruction of gene regulatory networks problem in biology (Zheng et al., 2012b) and the navigability analysis (Liu et al., 2019) are the published studies that utilized GFA.

2.2.5.3. Spiral Optimization Algorithm

The concept of spiral dynamics found in nature was firstly utilized in the optimization algorithm by Jin and Tran (2010) and Tamura and Yasuda (2011a; 2011b 2011c). Jin and Tran (2010) proposed a nature-inspired evolutionary algorithm that employs attraction based on spiral movements and dynamic mutation for better convergence. A year later, Tamura and Yasuda (2011a) developed a two-dimensional Spiral Optimization algorithm inspired by logarithmic spiral phenomena. The logarithmic spirals can be observed in nature as whirling current, a low pressure, a

nautilus shell, arms of spiral galaxies. The authors asserted that these examples provide efficient searchability in metaheuristics. Immediately after, Tamura and Yasuda, 2011b; 2011c) modified SOA for n-dimensional problems. Moreover, the novelty in the algorithm is not only the analogy but also the structure that has no randomness and fewer parameters. Tsai et al. (2014) proposed a novel SOA that divides the population into subpopulations to increase the diversity for clustering performance. Kaveh and Mahjoubi (2019) modified SOA by altering the movement operator implementing a mechanism to escape from local optima. Cao et al. (2020a) improved SOA for multi-objective problems which include both minimizing cost and maximizing energy efficiency. Furthermore, Stability analysis (Tamura and Yasuda, 2013); economic and emission dispatch (Benasla et al., 2014); clustering (Tsai et al., 2014); image segmentation (Man et al., 2014) sizing and layout optimization problem (Kaveh and Mahjoubi, 2019) are some of the studies utilized SOA and its modifications.

2.2.5.4. Galaxy-based Search Algorithm

Shah-Hosseini (2011a) got inspired by the spiral arm of spiral galaxies in outer space and proposed a novel metaheuristic called GbSA. In this algorithm, spiral movement is enriched with chaos to eliminate trapping into local optima. The advantage of chaos is that it prevents the algorithm move to the same point. Moreover, GbSA also includes a mechanism that stores the best solution and utilized modified Hill Climbing for local search (Shah-Hosseini, 2011b: 383). After the algorithm was published, several modifications have been made. Tolabi et al. (2016); Ara et al. (2016) modified GbSA by utilizing fuzzy sets. Sardari and Moghaddam (2016) modified GbSA in terms of local search procedure which results in more precise results. Apart from theoretical improvements, GbSA has been applied in different areas. Economic and emission dispatch problem (Zerigat et al., 2013; Zerigat et al., 2014), minimization of real power losses (Kumar et al., 2016), capacity optimization (Recioui, 2016), symmetric traveling salesman problem (Phu-Ang, 2018) are some of the applied studies published in the literature.

2.2.5.5. Black Hole Algorithm

Hatamlou (2013) developed an algorithm inspired by the black hole phenomenon. A black hole in space has enormous gravitational power that absorbs everything. BH starts with a randomly generated star population and a black hole becomes the best-so-far solution among all-stars. Later, according to the event horizon (the threshold for the distance between the star and the black hole) and a random number, the black hole starts to swallow up and in each iteration, a new located black hole is aimed which denotes a better optimal solution. Additionally, when there is a star absorbed by a black hole, a new random solution is generated until the population size remains the same. The authors asserted that BH has advantages in terms of both its simple structure and free-parameter tuning. However, Piotrowski et al. (2014) criticized BH that it is not a novel metaheuristic rather it is a simplified version of PSO and the only difference is the event horizon that limits the exploration ability; Gupta et al. (2016) compared BH with PSO and found out that PSO outperformed than BH. Many studies suggesting improvements have also been published, as the algorithm is immediately criticized. Jeet et al. (2016), Yaghoobi and Mojallali (2016) hybridized BH with Genetic Algorithm to prevent trapping into local optima; Wang et al. (2016) implemented Euclidean distance in the initialization of star locations to provide better exploration ability; Aslani et al. (2016) modified BH by introducing chaotic inertia weight for enhancing global searchability; Wu et al. (2017) proposed an adaptive BH that is less computational and has better intensification-diversification balance; Pashaei and Aydin (2017), García et al. (2017), Qasim et al. (2020) modified BH for binary variable problems; Gao (2017) presented a modified BH by implementing limit equilibrium mechanism; Xie et al. (2019) improved exploration and exploitation performance of BH by implementing Golden Sine and Levy Flight Operator respectively. Furthermore, BH and its improved versions have been utilized in problem-solving in various fields. For example, power flow optimization (Hasan and El-Hawary, 2014), job scheduling (Jeet et al., 2016), set covering (García et al., 2017; Soto et al., 2018), facility location and assignment problem (Veres et al., 2017), feature selection (Pashaei and Aydin, 2017; Qasim et al., 2020), a traveling salesman (Hatamlou, 2018), gene selection (Pashaei et al., 2019).

2.2.5.6. General Relativity Search Algorithm

Beiranvand and Rokrok (2015) developed GRSA as for global optimization approach inspired by Einstein's General Relativity Theory. GRSA is handled under the Universe class rather than Newton's Gravitational Law class. Because Einstein's General Relativity Theory generalizes Newton's Law of universal gravitation by explaining several effects that are unexplained by Newton's law. In GRSA, it is assumed that particles are moving in a non-gravitational space and they tend to be less active. This algorithm allows particles to move along geodesic trajectories in curved space and their velocities are calculated according to their energy momentums. Moreover, the velocities and the geodesic tangent are utilized for the determination of step length and step direction in the position-changing process. Although it presents a different principle, it has not been widely applied in the literature. Until now, Kumar et al. (2017) applied GRSA for Automatic Generation control; Ehsan-Maleki et al. (2018) utilized GRSA for parameter optimization in the design of WAPPSs.

2.2.5.7. Multi-verse Optimizer

Another metaheuristic algorithm called Multi-verse Optimization has been developed by Mirjalili et al. (2016). MVO mimics the theory of multi-verse in physics which says that there is more than one universe and they can interact and collide with each other. Mirjalili et al. (2016) utilized white holes, black holes, and wormholes concepts that have distinctive characteristics and capabilities. White holes attract everything whereas black holes absorb and wormholes provide a connection between white and black holes. Moreover, each universe has an inflation speed which constitutes every component in it. Hence, these principles are employed in the modeling of the MVO algorithm. For example, the inflation rate is assigned to each universe (candidate solution) according to the fitness value, and when the rate is high enough the universe may have a white hole that denotes a better solution. Also, having white hole results in sending objects to universes that have black holes through wormholes. This process has been achieved by utilizing a roulette wheel procedure. Moreover, it would be better to mention that wormholes appear randomly unlike the

others, and ensure sudden changes in white and black holes to eliminate local optima stagnations. As seen in other algorithms as well, improved versions of MVO have been published. Meshkat and Parhizgar (2017) improved the performance of MVO by implementing an updated genetic reproduction scheme; Valenzuela et al. (2017), Abdel-Basset et al. (2019), Al-Madi et al. (2019) modified MVO for binary variables; Mirjalili et al. (2017), Geng et al. (2019), Elaziz et al. (2019) proposed multi-objective version of MVO; Ewees et al. (2019), Sahoo and Panda (2020) developed Chaotic MVO to avoid local optima stagnation and slow convergence; Wang et al. (2020a) presented a novel MVO that provides parallel sessions to avoid premature convergence; Abasi et al. (2020) enhanced the exploitation ability of MVO by introducing neighbor operator. Furthermore, the application fields could be summarized as power flow optimization (Bentouati et al., 2016), classification problem (Faris et al., 2016), feature selection (Faris et al., 2018), knapsack problem (Abdel-Basset et al., 2019), flow-shop scheduling (Wang et al., 2019a), damage detection (Ghannadi and Kourehli, 2020).

2.2.6. Optic

The algorithms inspired by optics are given in Table 8 in chronological order. Each algorithm will be handled according to the common principles and distinctive features respectively and application studies will be presented.

Table 8: Metaheuristic Algorithms Inspired by Optics

Algorithm	Main Subjects	Author(s)
Light Ray Optimization (LRO)	Optical phenomena - Refraction - Reflection - Position	Shen and Li (2009)
Ray Optimization (RO)	Rays of light - Travels - Ray tracing	Kaveh and Khayatazad (2012)
Optics Inspired Optimization (OIO)	Behaviors of light - mirror	Kashan (2015a)

2.2.6.1. Light Ray Optimization

According to Fermat's principle, a ray takes a path between two points in the least time. Therefore, Shen and Li (2009) developed the Light Ray Optimization

algorithm based on optical principles. This algorithm mimics the behavior of rays in an uneven transparent medium. The searching process starts with dividing the search space into rectangular grids in which light rays go at different velocities (Shen and Li, 2010: 919). The propagation velocity of light rays is calculated as the objective function value of the center of the related grid and that the propagation path represents the searching path in the problem solution. Besides, refraction or reflection occurs while trans-passing the grids for searching the global optima according to the characteristics of the corresponding grid (Shen and Li, 2012: 435). Although LRO is one of the first algorithms inspired by optics, it does not attract much attention in the literature. Wang and Shen (2012) modified LRO for multi-objective problems; Shen and Li (2012) improved the local searchability of LRO by implementing a simulated annealing strategy are some of the studies focusing on LRO and its extensions.

2.2.6.2. Ray Optimization

Another optic-inspired algorithm is RO developed by Kaveh and Khayatazad (2012). Unlike LRO, RO utilized Snell's light reflection law. According to that law, refraction of light occurs, and the direction changes when rays of light move from a lighter medium to a darker medium. By exploiting this principle, the candidate solutions approach the global optimum in the RO algorithm. Different from other algorithms mentioned in this chapter, the RO algorithm handles high dimension problems by dividing solution vectors into 2 or 3-dimensional spaces and then joining them together. Also, RO utilizes a definite movement vector in case of being out of the search space. After proposing the RO to the literature, its modifications have been developed. Kaveh et al. (2013b) improved the RO algorithm for high dimensional problems by changing the procedure for the division of search space into 2 or 3 and the boundary violation rule; Beirami et al. (2020) modified RO for the multi-objective problems. Moreover, RO has been also applied in designing truss structures (Kaveh et al., 2013b), damage assessment (Kaveh et al., 2014b), optimization of the thickness of granular layers in railway tracks (Esmaeili et al., 2015), layout and size optimization (Kaveh and Ghazaan, 2015), economic generation scheduling (Beirami et al., 2020).

2.2.6.3. Optics Inspired Optimization

OIO is another algorithm inspired by the law of reflection and it models the behavior of light and its interactions with instruments. According to the law of reflection, a concave surface makes the light be converged whereas, a convex surface makes the light diverge. By utilizing these principles metaphorically, Kashan (2015a) developed OIO which assumes that the surface of the objective function is a wavy mirror including peaks and valleys as convex and concave mirrors. Thereafter, a candidate solution (image) is generated based on the mirror equations adopted in optics. Moreover, Kashan (2015a) developed two variants of OIO as well. Since OIO has the ability to handle unconstrained optimization problems, Kashan (2015b) developed a modified version of OIO to handle constraints. In addition, Wang et al. (2017) proposed a Self-adaptive OIO for better convergence speed and accuracy; Wang et al. (2020b) proposed an estimation method that integrates Support Vector Machine and Quantum OIO. Besides the theoretical developments, some application articles as clustering and routing in wireless sensor network (Lalwani et al., 2017), traveling tournament (Alatas and Bingol, 2019), design of truss structures (Jalili and Kashan, 2019), scheduling of batch processing machine (Alizadeh and Kashan, 2019) have also been published.

2.2.7. Others

Until now, the physics-based algorithms classified in terms of their principles have been reviewed and summarized. In Table 9, physics-based algorithms that are not put into a class are listed. In the following, each one will be reviewed and its recent developments will be summarized.

Table 9: Other Algorithms

Principle	Algorithm	Main Subjects	Author(s)
Law of motion	Gases Brownian Motion Optimization (GBMO)	Gas molecule – Mass – Velocity – Radius of Turbulent	Abdechiri et al. (2013)
Ions motion	Ions Motion Algorithm (IMO)	Attraction - Repulsion - Anions - Cations - Motion – Charged	Javidy et al. (2015)

Law of Thermodynamics	Heat Transfer Search (HTS)	Heat - Temperature - Balance	Patel and Savsani (2015)
Newton's Law of Cooling	Thermal exchange Optimization (TEO)	Heat - Temperature - Position	Kaveh and Dadras (2017)
Henry's Law	Henry gas solubility optimization HGSO)	Gas-Liquid - Temperature -	Hashim et al. (2019)
Balance	Equilibrium optimizer (EO)	Control Volume- Mass-Balance	Faramarzi et al. (2020b)

2.2.7.1. Gases Brownian Motion Optimization

GBMO is an algorithm inspired by the law of motion and mimics the gas molecules according to their positions, masses, velocities, and radiuses of turbulent (Abdechiri et al., 2013: 2934). In GBMO, the temperature has a crucial role in setting up the balance between exploitation and exploration. Molecules in small masses denote better candidate solutions and they can move faster which provides good exploitation. Moreover, in the beginning, the molecules are in high-temperature space and they have kinetic energies for exploring search space with the help of Gases Brownian Motion. However, with a lapse of time, temperature decreases, and turbulent rotational motion provides local searchability. Therefore, it is clear that the power of GBMO lies under the policy of changing the roles of Gases Brownian Motion with turbulent rotational motion. Afterward, GBMO has been improved with different procedures. For example, Rathore and Roy (2014) applied GBMO in transmission network expansion planning problem; Rahchamani et al. (2019) proposed an adaptive neuro-fuzzy inference system for classification by utilizing GBMO; Zamani et al. (2016) applied GBMO for Fractional Order PID controller; Nayak et al. (2019) designed digital differentiator by GBMO.

2.2.7.2. Ions Motion Algorithm

Javidy et al. (2015) developed the IMO algorithm that utilizes the behavior of ions towards each other in terms of their charges. Although various algorithms employ charged particles under the classification of "Newton's Gravitational Law" and "Magnetism", IMO does not belong to both classes. In IMO, ions are treated as the candidate solutions in two groups (anions: negative ions and cations: positive ions). It

is clear that an attraction may occur between these two groups and repulsion may occur within the groups. Namely, anions tend to move toward the best cation, whereas cations move toward the best anion. The magnitudes of the charges are calculated in terms of the objective function values and the movements that occur by considering their charges. Furthermore, for exploration and exploitation procedures two phases (liquid and solid) of ions are employed. The process of the transition from liquid to solid yields IMO to converge the global optimum. Afterward, modified versions have been published in literature asserting that they provide better performances. Wang and Ma (2018) improved the convergence speed and accuracy by implementing an opposition-based learning strategy and changing random perturbations in the solid phase. Wang et al. (2019b) eliminated the local optimum stagnation and premature convergence by introducing a cloud adaptive inertia weight quantum chaotic IMO algorithm. Buch and Trivedi (2020) modified IMO as Non-Dominated Sorting IMO which utilizes selective crowding distance and non-dominated sorting method to preserve the diversity of the best solutions set. Besides, IMO has also attracted attention in engineering applications such as optimization of Least Squares Support Vector Machine parameters for temperature compensation approach (Li et al., 2016); optimization of the pump position in a water distribution network (Tahani et al., 2019).

2.2.7.3. Heat Transfer Search

HTS is an algorithm inspired by the law of Thermodynamics and it is developed by Patel and Savsani (2015). This algorithm is built on the concept of thermal equilibrium. According to that principle, any system always tends to reduce the thermal imbalance between the system (candidate solutions) and the surrounding (best solution) by conducting heat transfer in the form of conduction, convection, and radiation. HTS employs clusters of molecules that are at different temperature levels and represent variables and the energy levels of molecules are treated as the objective function value of a problem. HTS includes conduction, radiation, and convection phases with equal probability to neutralize thermal imbalance, respectively. If the updated solution has a better fitness value, it will be accepted and a greedy selection procedure will be employed. In addition, the balance between intensification and

diversification is controlled for each phase with the help of the factors used. After the algorithm is introduced to the literature, different versions have been presented as well as in others. Savsani et al. (2017) developed multi-objective HTS which includes a non-dominated sorting method and diversity preserving crowding distance approach. Tejani et al. (2019a) improved HTS by incorporating the interactions between molecules as well as with surrounding and by introducing the population regenerator procedure. Alnahari et al. (2020) solved a dynamic optimization problem in chemical engineering with the help of HTS which is modified in terms of simultaneous heat transfer search, quadratic interpolation method, and population regeneration mechanism. In addition to theoretical developments, optimization of truss structures (Degertekin et al., 2017; Tejani et al., 2019b; Kumar et al., 2020a), optimization of semi-active suspension system (Garg et al., 2017), economic dispatch problem (Hazra et al., 2018; Pattanaik et al., 2020), optimum design of distribution networks (Mohamadi et al., 2020) are some of the application studies published in the literature.

2.2.7.4. Thermal Exchange Optimization

Kaveh and Dadras (2017) developed the TEO algorithm inspired by Newton's law of cooling. This law states that the rate of heat loss of a body is proportional to the temperature difference between the body and its surroundings. Differently from SA, TEO employs the temperature of bodies as their position and new positions are determined by grouping the bodies with new temperatures. As a distinctive feature, TEO has thermal memory that saves best-so-far solutions and removes the worst objects at the same time to improve the performance. Since the algorithm has been introduced relatively new, it has not been studied much yet. A year after, Kaveh and Dadras (2018b) modified TEO by implementing an offline parameter-tuning method to identify structural damage. Kaveh et al. (2018b) eliminated shortcomings of TEO and applied it for the optimization of the design of skeletal structures. Afterward, Xing and Jia (2020) implemented the Levy flight algorithm to TEO to obtain a better exploration and exploitation balance for image segmentation.

2.2.7.5. Henry Gas Solubility Optimization

HGSO is another physics-based algorithm inspired by Henry's Law (Hashim et al., 2019: 646). According to Henry's Law, the dissolved gas amount in a liquid is proportional to the partial pressure of that gas in equilibrium with that liquid. In HGSO, the solubility can be affected by temperature and pressure. For example, the solubility of gasses increases at less temperature but in a high-pressure environment. This algorithm utilizes clustering after the initialization step. Although HGSO proposed relatively new, it attracts attention. Saranya and Saravanan (2020) modified HGSO by implementing langrage relaxation; Shehabeldeen et al. (2020) hybridized HGSO and Artificial Neural Network (ANN); Hashim et al. (2020) improved HGSO by inserting a new section for accurate detection of target motif. Besides, HGSO applied for feature selection (Neggaz et al., 2020), shape optimization of a vehicle brake pedal (Yıldız et al., 2020), parameter optimization of Support Vector Regression (Cao et al., 2020b).

2.2.7.6. Equilibrium Optimizer

Faramarzi et al. (2020b) proposed EO as one of the most recently published physic-based algorithms. It is inspired by control volume mass balance models. It is assumed that each agent changes its position according to the equilibrium candidates which are the best-so-far solutions. Although EO has been presented this year, it takes too much attention and is cited more than 25 times within a couple of months. For instance, Gupta et al. (2020b) implemented Gaussian mutation and exploratory mechanism into EO; Wunnava et al. (2020) proposed another version of EO which provides an adaptive position update strategy. Furthermore, modeling of a fuel cell (Menesy et al., 2020); economic dispatch problem (Agnihotri et al., 2020); prediction of laser cutting parameters (Elsheikh et al., 2020) are the first application studies.

CHAPTER THREE
THE PROPOSED ALGORITHM

In this thesis, two optimization algorithms that are adapted from the thesis prepared by Erdem (2007) are proposed. The first one called Random Search with Adaptive Boundaries (RSAB) is an initialization algorithm that provides adaptive initial solutions rather than pure random. The main power of the proposed algorithm is the ability to eliminate the local optimums by narrowing the search space. However, the proposed algorithm is not the one that guarantees the global optimum solution. Indeed, it is a generic methodology that can be applied in the initialization stage of any algorithm for unimodal or multi-modal problems. In other words, it provides an adaptive initial solution for both continuous unconstrained/bounded and constrained nonlinear optimization problems demonstrated in Eq. (3.1) and Eq. (3.2) that may have many local optima.

$$\begin{gathered} Min\ f(\vec{x}) \\ \forall x \in [x_l, x_u], x_l \in \mathbb{R}, x_u \in \mathbb{R} \\ where\ f: \mathbb{R}^n \longrightarrow \mathbb{R} \\ x \in \mathbb{R}, \vec{x} \in \mathbb{R}^n \end{gathered} \tag{3.1}$$

where $\vec{x}$ is the solution vector, x_l and x_u are the boundaries of the related variable, lastly $f(\vec{x})$ is the objective function whether in a linear or non-linear form.

$$\begin{gathered} Min\ f(\vec{x}) \\ \text{Subject to} \\ g_i(\vec{x}) \geq b_i\ i = 1,2, \dots, k \\ h_j(\vec{x}) = 0\ j = 1,2, \dots, l \\ \forall x \in [x_l, x_u], x_l \in \mathbb{R}, x_u \in \mathbb{R} \\ where\ f: \mathbb{R}^n \longrightarrow \mathbb{R}, g: \mathbb{R}^n \longrightarrow \mathbb{R}\ and\ h: \mathbb{R}^n \longrightarrow \mathbb{R} \\ x \in \mathbb{R}, \vec{x} \in \mathbb{R}^n\ and\ k, l \in \aleph \end{gathered} \tag{3.2}$$

where $\vec{x}$ is the solution vector, k is the number of inequality constraints, l is the number of equality constraints, x_l and x_u are boundary constraints, and all the functions could be linear or non-linear.

The second algorithm is called Repulsive Forces (REF). Since the main principles depend on Newton's General Gravity Law and Coulomb's Law, that

algorithm can be regarded as a physics-based algorithm. REF model uses Coulomb's Law to implement the repulsive structure of the particles in that particles are like charged. Different from the other algorithm, the REF algorithm aims to reach optimum-like solutions by constraint-handling abilities.

In the following subsections, the details of both algorithms are explained. The pseudo-codes and their principles are given.

3.1. RANDOM SEARCH WITH ADAPTIVE BOUNDARIES (RSAB)

In this section, we introduce an algorithm that provides an adaptive initial (better than pure random initial) solution for continuous nonlinear optimization problems that may have many local optima. It is structured as a generic methodology that can be applied in the initialization stage of any algorithm for unimodal or multi-modal problems. The stages of this algorithm were converted from the codes that Erdem (2007) prepared in Visual Basic. However, some structural changes have been implemented for the dynamic procedure used in the updated domains of variables in this thesis.

The RSAB algorithm depends on updating given upper and/or lower intervals dynamically (i.e., boundaries iteratively). The main procedure includes two steps. The first step is called "Determine Intervals". In this step, a search space is constituted by using the domain of decision variables used in the problem. This step provides determined intervals that satisfy boundaries or constraints for each variable. Thereafter, an initial solution is obtained by evaluating constraints if any. After the first run, a better solution is tried to be obtained in each iteration. The second step called "Update Intervals" depends on updating the intervals in terms of best-so-far solutions to reach a better solution. The pseudo-code of the algorithm and the detailed codes of updating interval are given below, respectively. Figure 16 shows the general pseudo-code and it starts with determining intervals. In Figure 17, the pseudo-code of "Determine Intervals" is given. Although the general flow is adapted from Erdem (2007), the approach for finding upper and lower limits is modified as explained below.

Figure 16: The Pseudo-code of Random Search with Adaptive Boundaries

```
 1:  Determine intervals (Initial Limits)
 2:  Create initial 1000 sized random solution vectors
 3:  For each solution vector
 4:         Evaluate constraints
 5:  For each iteration
 6:         If Improvement = FALSE
 7:              Initial Limits
 8:              If unconstrained problem
 9:                   Update intervals by using the midpoint
10:                 Else
11:                   Update intervals by using the holdbest
12:         Else
13:              If unconstrained problem
14:                   Update intervals by using the holdbest
15:                 Else
16:                   Update intervals by using the midpoint
17:         For each variable
18:                Create random values based on new intervals
19:         For each solution vector
20:               Evaluate constraints
21:         Store Updated Interval
22:  Loop Until maximum iteration given
23:  For each variable
24:      Calculate means, modes, medians of lower-upper limits
25:      Updated Lower Limit = min (Mean_L, Mode_L, Median_L)
26:      Updated Upper Limit = max (Mean_U, Mode_U, Median_U)
```

In the first step, the search space is determined by considering given boundaries of variables regardless of having constraints. The lower limit and the upper limit are found out by checking concurrently all boundaries in an iterative manner. In the case of constrained problems, all constraints are considered for each variable. It is worth mentioning that, since evaluating boundaries as constraints in the "Determine Intervals" step wastes time in visual basic codes, boundaries defined in the optimization model are excluded from evaluating loop. Instead, boundaries are considered as initial limits directly before checking constraints.

Finding an upper limit or lower limit depends on the current boundaries of the variables except the related one. The only rule is that constraints should not be in the form of a numerator/denominator. The details of these two functions are given below:

Find Upper Limit:

- Replace the related variable with "x" and replace all the others with "**lower limits**"
- If the sign of the related variable is positive while the others are negative:

  Replace the related variable with "x" and replace all the others with "**upper limits**"

Find Lower Limit:

- Replace the related variable with "x" and replace all the others with "**upper limits**"
- If the sign of the related variable is positive while the others are negative:

  Replace the related variable with "x" and replace all the others with "**lower limits**"

After replacing the variables, finding root operation is conducted. If the root is in the determined intervals, the new lower or upper limit is updated before considering the other constraint. This process continues until all constraints are reviewed. In the end, the maximum value in a lower limit set and the minimum value in an upper limit set will be stored as determined intervals of variables.

Figure 17: The Pseudo-code of "Determine Intervals"

```
 1: Create Intervals by considering boundaries
 2: If constrained problem
 3:    For each variable
 4:       For each constraint
 5:         If sign <=
 6:            If rhs >= 0
 7:               If all variables are positive sign
 8:                  Find_upper_limit
 9:               Else
10:                  Find_coefficient
11:                  If coefficient > 0
12:                     change = TRUE
13:                     Find_upper_limit
14:            Else
15:               If not all variables are positive sign
16:                  Find_coefficient
17:                  If coefficient < 0
18:                     Find_lower_limit
19:         Else
20:            If rhs >= 0
21:               If all variables are positive sign
22:                  Find_lower_limit
23:               Else
24:                  Find_coefficient
25:                  If coefficient > 0
26:                     change = TRUE
27:                     Find_lower_limit
28:            Else
29:                If not all variables are positive sign
30:                   Find_coefficient
31:                   If coefficient < 0
32:                      Change_negative_signs
33:                      Find_upper_limit
```

*rhs = Right-hand side value of the constraint

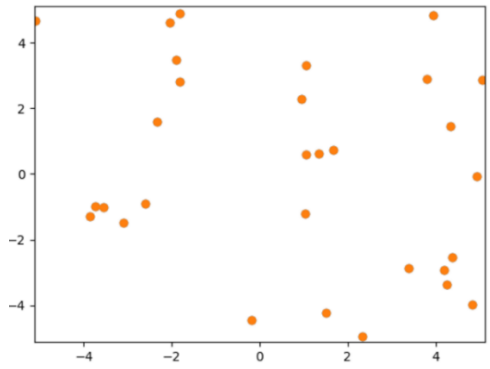
After determining intervals, firstly 1000-sized random solution vectors are generated for once concerning equal chances according to the boundaries of variables in the hyperspace that has dimensions as much as the number of variables in the optimization model. To some extent, this method has been inspired by the "*Scatter Search Algorithm*" by Glover (1999). The pseudocode for the initialization is given in Figure 18.

Figure 18: The Pseudo-code of Generating Random Numbers

```
k = int (4 * random () + 1)
If k = 1:
      x_i = ((δ_i^+ − δ_i^−)/4) * random() + δ_i^−
Else if k = 2:
      x_i = ((δ_i^+ − δ_i^−)/4) * random() + (δ_i^+ − δ_i^−)/4 + δ_i^−
Else if k = 3:
      x_i = ((δ_i^+ − δ_i^−)/4) * random() + 2 * (δ_i^+ − δ_i^−)/4 + δ_i^−
Else:
      x_i = ((δ_i^+ − δ_i^−)/4) * random() + 3 * (δ_i^+ − δ_i^−)/4 + δ_i^−
```

Where δ_i^-, δ_i^+: Boundaries of each variable; x_i: Generated random number for i^{th} variable; *random ()*: A function generates random number between [0,1].

Figure 19: Initial Particles (2-Dimension)



Source: Prepared by the author

A demonstration of generated particles is shown in Figure 19. This step also includes evaluating each particle individually in terms of their goal function values. It means that not only objective function value is calculated but also the amount of like charge for each particle is determined in the employer model so that repulsive forces can be assessed. In the case of unconstrained problems, the constraint satisfied rate is

taken as “1” and the maximum deviation as 0 directly. However, in the case of constrained problems, a penalty approach (explained in the REF algorithm section) is used. Among them, the best-so-far solution is stored as “holdbest”. In the first iteration, it is assumed that a better solution is not reached.

Subsequently, the “Update Intervals” step is ready to be implemented. This step is designed to obtain robust and stable solutions. The underlying purpose of that step is to eliminate unnecessary search space that does not include the global optimum solution. The details of “Update Intervals” which includes two approaches are given in Figure 20-21 in the following.

Figure 20: The Pseudo-code of Update Intervals by Using Midpoint

```
For each variable
    Calculate mid_point
    If x_bestsofar < mid_point
        δ⁻_new = δ⁻_updated
        δ⁺_new = mid_point
    Else
        δ⁻_new = mid_point
        δ⁺_new = δ⁺_updated
```

where mid_point is the middle point of the related variable interval, $x_{bestsofar}$ is the best-so-far value of the related variable, δ^-, δ^+ are the lower-upper limits. Update interval by using midpoint is running when there is no improvement for unconstrained problems and when there is an improvement for constrained problems. Since there are no restrictions in unconstrained problems, focusing on the holdbest may cause a wrong direction in case of no improvement. Furthermore, updating intervals by using midpoint helps unconstrained problems to converge the right area faster.

Figure 21: The Pseudo-code of Update Intervals by Using Holdbest

```
For each variable
    d_u = δ⁺_updated − x_bestsofar
    d_l = x_bestsofar − δ⁻_updated
    If d_u > d_l
        δ⁻_new = δ⁻_updated
        δ⁺_new = x_bestsofar + d_l
    Else
        δ⁻_new = x_bestsofar − d_u
        δ⁺_new = δ⁺_updated
```

where d_u is the distance between best-so-far value and upper limit of related variable, d_l is the distance between the best-so-far value and the lower limit of the related variable. Updating intervals by using holdbest is running when there is an improvement for unconstrained problems and when there is no improvement for constrained problems. Since the ability of the penalty approach is a strength, focusing on holdbest in updating intervals improves the solution when there is no improvement for constrained problems. On the other hand, when there is an improvement updating intervals by considering holdbest provides better convergence for unconstrained problems.

Consequently, the updated intervals are stored until the maximum iteration is reached. The final updated intervals are determined by considering all generated intervals for having robust, stable solutions. For that reason, mean-mode-median values of lower and upper limits are calculated. In the end, the minimum of lower limits and the maximum of upper limits will be the final updated intervals for the related problem.

- Mean_L = The mean value of lower limits for the related variable
- Mode_L = The mode value of lower limits for the related variable
- Median_L = The median value of lower limits for the related variable
- Mean_U = The mean value of upper limits for the related variable
- Mode_U = The mode value of upper limits for the related variable
- Median_U = The median value of upper limits for the related variable

Updated Intervals = [Min (Mean_L, Mode_L, Median_L), Max (Mean_U, Mode_U, Median_U)]

It is worth mentioning that there is a parameter using dynamically. Generally, the algorithms use constant set size/ population in literature, however, in our random search algorithm the set size can be changed according to the counter parameter. This feature reveals the difference of the RSAB algorithm from other algorithms in the literature. If there is no improvement in fitness value, the θ parameter is also increasing as the counter. However, the increasing amount is another parameter that can be modified. If there is an improvement, the counter remains the same as the last value, otherwise, the set size is increased by the defined amount.

In this algorithm there is no stopping criterion, instead, it will continue until the last iteration. However, for further studies, different stopping conditions can be considered. It would be better to clarify that the RSAB algorithm is not an algorithm that provides optimum-like solutions. It aims to narrow the search space without trapping local optimums and provides reduced candidate solutions to the main algorithms. In other words, the RSAB algorithm is thought of as an initialization algorithm before reaching the optimum-like solutions.

3.2. REPULSIVE FORCES ALGORITHM (REF)

In this part, an algorithm based on repulsive forces of particles is presented and it is called the Repulsive Forces Algorithm (REF). That method is a revised version of the algorithm introduced by Erdem (2007). It takes into account the natural forces that cause repulsion between bodies such as atomic and sub-atomic particles that may be like electrical charged, polarized in that their impact may cause shifting to the new locations and of course new minimum objective function value. These natural facts focus especially on physical and chemical phenomena as explained in the following section. Thereafter, the pseudo-code of the algorithm and its initialization, neighborhood, repulsive forces, displacement, duplication principles of the REF algorithm are given in the following sections.

3.2.1. Theoretical Background

In nature, there is much evidence for optimization that is encountered as a minimum energy state, equilibrium point, zero compound forces. Based on these inspirations, many physics and chemistry-based optimization algorithms have been proposed especially after the year 2000. According to the literature of physics-based algorithms given in the second chapter, it is seen that the motions, interactions, or dynamics of the particles that occurred in nature are used as inspirations. However, magnetism, quantum, and universe concepts are utilized as inspirations for physics-based algorithms in general.

In the REF algorithm, *Coulomb's Law* is used to implement the repulsive structure of the particles. The question to be answered by this model is to find the value of charged units in terms of the solution points, the distance between the particles, and the assignment of the Coulomb's Constant that changes whether dynamically or statically. According to *Coulomb's Law* (Coulomb, 1785) between the two identical/opposite charged units, they attract or repel each other due to their electrical charge amount proportionally, polarity directionally, distance as inverse proportionally. The Coulomb Constant is similar to Universal Gravity Constant which is used in *Newton's General Gravity Law*. Moreover, the impact of the distance factor can be explained by Inverse Square Law proposed by Newton (Newton, 1999: 238). The calculation of forces is demonstrated in Table 10.

Table 10: Newton's General Gravity versus Coulomb's Law

Newton's General Gravity Law	**Coulomb's Law**
$F = G\frac{m_1 m_2}{d^2}$	$F = k\frac{Q_1 Q_2}{d^2}$
G: universal gravity constant m_1, m_2: the mass of two bodies d: the distance between two bodies	k: Coulomb's Constant. Q_1, Q_2: electrical charge of two units d: the distance between two elementary units

REF algorithm will stop when it reaches some predefined steady state. The question on the stopping condition is to find the level of zero or near zero combined net force exerting to each particle. This principle can be explained with equilibrium states in thermodynamics. The word meaning implies a state of balance which means all elements in a system have the same value whether it is temperature or pressure. There are various kinds of equilibrium such as thermal, mechanical, chemical, phase. Each principle asserts a state of balance in terms of different indicators (Çengel et al., 2019: 14). Universal gravity, atomic and molecular structure, and magnetism sing the same song. They state something about forces and energy levels. Some particles may be suffered all the effects and of course forces. These forces influence the particles until a new equilibrium point in terms of combined forces by extracting some amount of energy that can be potential, kinetic, heat, chemical, and electrical.

Moreover, in nature, nobody can control the instant location of atomic particles uniquely because of the uncertainty of locations. This is explained by *Heisenberg's Uncertainty Principle in Quantum Mechanics* (Busch et al., 2007: 155). The

uncertainty principle is adapted to the REF model in that initial scattering is selected as a random process and exact locations of the particles should be calculated with some uncertainty (i.e., probability) that should not be taken place in the model significantly.

Besides, *Pauli's Exclusion Principle* is another principle utilized in REF. According to that principle, no more than one particle can exist in the same state. This means that none of the particles can have the same repulsive forces. REF method ensures diversification like *Tabu Search* (Glover, 1989; Glover, 1990) and *Scatter Search* (Glover, 1999) especially at early stages before converging to the global optimum by giving no extra effort rather than applying repulsive forces.

All particles tend to move to a new location under the forces if it is a better place. Displacements of each particle are calculated by considering the *Law of Momentum* in that total momentum must be conserved. The unit displacement is structured by considering Newton's Second Law (Chandrasekhar, 2003: 18).

3.2.2. Assumptions of REF

REF algorithm is a population-based algorithm inspired by the principles mentioned above. It is assumed that the particles represent solution vectors and the global optimum solution is aimed to be reached by considering the interactions of particles in terms of their charges and distances. Other assumptions considered in REF are listed below.

- Solution points are assumed as like particles that repel each other; they are differently charged elementary particles (i.e., magnitude of charges).
- Particles move in hyperspace through only a path but not like a wave.
- Magnitudes of particles are determined through corresponding objective function values.
- Amount of replacements are computed by momentum law where both magnitudes of the particles and distance between the particles are considered as well as the direction of the particles after exerting the repulsive forces.
- No attractions and merging are allowed for particles.
- Repulsive forces occur between the solution point and its neighbors.

- No negative improvements are allowed, which means that if the new location after the repulsive effects on a selected particle has a higher energy level, it is assumed that the particle is forced to a higher energy level immediately back to its previous stationary state by emitting the energy.
- The stopping conditions of the algorithm will not meet if the desired improvements and number of iterations are satisfied.
- If a particle has greater Δx than it must be, it is assumed that as if particle hits the wall of the constrained boundaries and goes back until $\Delta x_i^{k+1} < x_i^k/2$. Therefore, the only remainder of Δx_i^{k+1} is available to use.
- The multiplicative penalty-based method is used for constraint handling.
- No particle can be found at the same location up to a certain degree of precision.
- The best position of each particle in the population is maintained, and with each relocation, an attempt is made to reach a better than "*best-so-far*" position.
- Each particle produced in the related run, whether it is a better solution or not; saved in a database. Thus, extra function evaluation will not be required for a previously evaluated particle.

3.2.3. The Pseudocode of the Algorithm

Before explaining each step of the REF algorithm, the pseudo-code of the algorithm is given in Figure 22. REF algorithm comprises Determine Intervals, Initialization, Multiplicative Penalty based Method, Repulsive Forces, Neighborhood, Displacement and Duplication check steps, respectively. Determine Intervals and Initialization steps are handled in the “Random Search with Adaptive Boundaries” algorithm section. The remaining modules will be mentioned in the following.

Figure 22: The Pseudo-code of REF Algorithm

```
 1: Determine Intervals
 2: Create Initial Particles
 3: Evaluate Constraints
 4: For Each Iteration
 5:      Until stopping condition is met
 6:      For Each Particle
 7:          Find Neighbors
 8:          For Each Neighbor
 9:              Find Incremental Replacements
10:              If f(x) reduces Go to New Location
11:              Update New State
12:      Check Duplication
```

REF algorithm utilizes memory for two purposes. The first one is the creation of a database. Each particle is recorded in a memory along with the evaluation scores (constraint satisfied rates, total deviations). This procedure is inspired by the principle that each step in the *Tabu Search* algorithm is kept in a "history" mechanism and the repetition of previous solutions is prohibited by looking at this memory (Glover, 1989). However, in the REF algorithm, memory is used not as a banned list, but to get rid of unnecessary repetitive function evaluations. The second one is about recording the best-so-far positions of the particles. This approach is inspired by the elitism principle in the literature. The *Elitism* principle is one of the selection techniques that saves the best solution in the population to eliminate the risk of losing the best solution between iterations (Simon, 2013: 188). In the REF algorithm, best-so-far solutions of every particle are stored in order not to lose them in case of displacements. However, differently from the original elitism principle, there is no limitation for the number of elitist particles. Namely, the best-so-far particles are kept separately from the relocated particle set.

3.2.4. Multiplicative Penalty based Method (MUPE)

As stated by Erdem (2007), the early study by Yokota et al. (1995); Deb (2000); Coello (2002); Oyama et al. (2005) motivated and led as for penalty approach developed for the REF algorithm. The traditional approach for constraint handling for the single objective nonlinear programming is based on penalty functions where the fitness of a design candidate is determined based on a new aggregate function *F*, which

is a weighted sum of the objective function value and the amount of design constraint violations.

In the REF algorithm, a multiplicative penalty approach is used for handling constraints. The multiplicative penalty-based constraint handling (MUPE) method developed by Erdem (2007) assumes that the goal function employs a single objective function to be minimized under the constraints to be satisfied that are joined in an objective function as in other penalty-based methods. In the case of unconstrained/boundary conditions, the goal function would be the objective function itself. Herein goal function combines both objective function and all constraints if it is a constrained nonlinear optimization. In this situation, objective function has three goals that can be expressed as:

- Goal 1: Minimize or maximize $f(\vec{x})$
- Goal 2: Minimize the total deviations from all constraints
- Goal 3: Maximize the ratio of the satisfied constraints

Here Goal 2 and Goal 3 provide a benchmarking on the condition that two infeasible solution points that have different constraint violation level in terms of both Goal 2 and Goal 3. Goal 2 deals with the relative summary measure of constraint violations of solution points. On the other hand, Goal 3 incorporates the ratio of satisfied constraints among overall ones. It can be concluded that selecting the "*a solution point that has great violation on a single constraint but the others satisfied*" against "*a solution point that has little violations for all the constraints*" is a benchmarking interest of this method. If all of the constraints were satisfied for the two solution points, Goal 1 would be the only criteria for comparing these solution points. Experimental studies will give us acceptable results on benchmarking and trade-offs about weighting values among these three parts.

Apart from the previous studies, the MUPE method is interested in not only the ratio of satisfied constraints but also the total amount of violations regarding corresponding upper or lower constrained values. If the goal function acts as a heuristic function of the model of the combined objective function to be minimized, then the general form of the proposed heuristic fitness function will be:

$$H(\vec{x}) = H(f(\vec{x}), d(\vec{x}), t(\vec{x})) \tag{3.3}$$

where

$f(\vec{x})$: objective function to be minimized/maximized

$d(\vec{x})$: the total amount of violations of the constraints

$t(\vec{x})$: the ratio of the satisfied constraints

The total amount of violation is calculated as a relative violation according to the right-hand-side value of the related constraints. In case of less than 1.00E-11 deviation in the constraints, the candidate solution is considered feasible.

$$d_i = \begin{cases} -|g_i(\vec{x}) - b_i|/b_i, & b_i \neq 0 \\ -|g_i(\vec{x}) - b_i|, & otherwise \end{cases} \quad (3.4)$$

It is worth noting that, each goal is considered according to different importance scores, namely constants (c_1, c_2, c_3). Additionally, the sign of the objective function value is added to this heuristic function as multiplication as well. The pseudo-code for the calculation of the heuristic fitness function is given in Figure 23.

Figure 23: The Pseudo-code of Heuristic Fitness Function

```
If f(x) > 0
    goalSign = 1
Else
    goalSign = -1
Goal1 = |f(x)|^(c1)
If |d(x)| < η
    Goal2 = 1 / e^(c2 d(x))
Else
    Goal2 = λ
If goalSign =1
    Goal3 = 1 / t(x)^(c3)
Else
    Goal3 = t(x)^(c3)
fitness = goalSign * Goal1 * Goal2 * Goal3
```

*For maximization problems; Goal1, Goal2, and Goal3 should be considered inverse.

where η is a defined total amount of deviations from constraints, λ (1.0E+100) is the penalty score for deviation in case of the total amount of deviation is bigger than η.

3.2.5. Repulsive Forces on Particles

When the particles take their places in hyperspace by their randomly assigned values in the initialization part, they start to repulse each other according to their forces. Repulsive forces are exerted on each particle employing Equation 3.5 where $f(\overrightarrow{x_i})$ corresponds to the amount of charge as similar to Coulomb's Q_i.

$$F = C\frac{f(\overrightarrow{x_i})f(\overrightarrow{x_l})}{d_{il}^2} \tag{3.5}$$

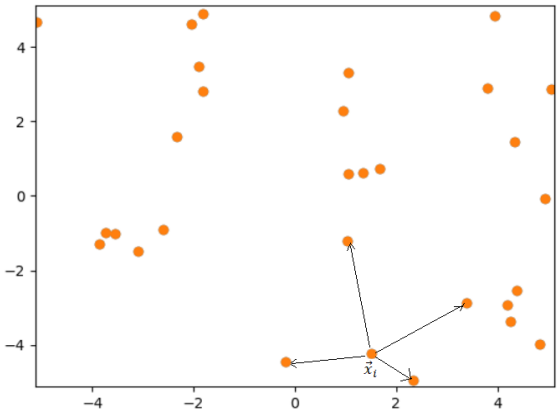
where *F* is a repulsive force between *i* and *l* particle; *C* is a repulsive force constant; $f(\overrightarrow{x_i}), f(\overrightarrow{x_l})$ are the fitness values of particle i and l respectively; d_{il} is the Euclidean distance between particle *i* and particle *l* and the calculation is demonstrated below.

$$d_{il} = \sqrt{\sum_1^n (x_{in} - x_{ln})^2} = \sqrt{(x_{i1} - x_{l1})^2 + (x_{i2} - x_{l2})^2 + \cdots + (x_{in} - x_{ln})^2} \tag{3.6}$$

3.2.6. Neighborhood

After calculating the fitness values of each particle, neighbor sets are constructing for each solution set. However, the determination of the number of neighbors is also another issue to be addressed. According to *Pareto's Principle*, roughly 80% of the effects come from 20% of the causes (Sanders, 1987: 37). Namely, when neighborhood vectors are sorted in descending order, the magnitudes of forces decrease sharply after 2nd-5th vectors. For that reason, we prefer to ignore the rest of them. Moreover, when the number of variables is large finding neighbors for each particle becomes very time-consuming. Therefore, the number of neighbors must be limited. In the REF algorithm, the best (the closest) two neighbors for each particle are stored for the next steps. The positions of particles visualizing the process of choosing neighbors are given in Figure 24. This means that the most three closest particles are determined as neighbors of the corresponding particle. Thereafter, these neighbors are utilized for the displacement procedure of the particle.

Figure 24: Neighborhood



Source: Prepared by the author

3.2.7. Displacement

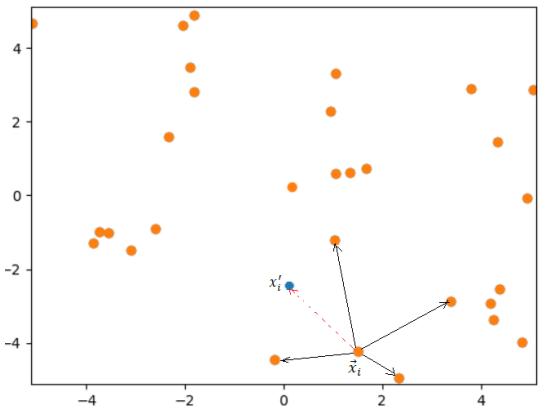
After identifying neighbors, new locations of each particle must be calculated by considering the *Impulse-Momentum Law* in that total momentum must be conserved. Particles are assumed as the same charge not as magnitude but as a pole that is why all the interactions between neighbors are repulsive as demonstrated along one dimension.

Figure 25: The Pseudo-code of Displacement

```
For each particle
    Find Neighbor
    For each Neighbor
        Do
           For each dimension
               Calculate the unified net force
               If improvement=True
                  Update New State
                  Update amount of displacement
        While improvement=True
```

After all the repulsive forces exerted on the selected particle $\overrightarrow{x_i}$, the unified (or compound) net force is calculated, and the new location of the selected particle is demonstrated as shown in Figure 26. Therefore, the distance between $\overrightarrow{x_i}$ and $\vec{x}_i'$ is equal to the net displacement for the selected particle.

Figure 26: Displacement of The Particle



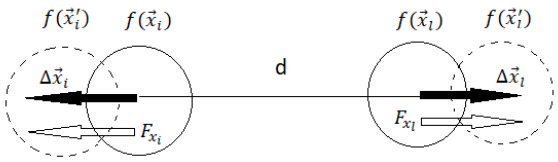
Source: Prepared by the author

REF algorithm employs *Pauli's Exclusion Principle* as a diversification procedure on the principle that two particles cannot occupy the same state in a closed system. Thus, particles can explore hyperspace without sticking to one point. Indeed, REF guarantees the dissimilarity not only for different charges $f(\vec{x})$ in similar locations but also similar charges in different locations. This situation can be satisfied by controlling and finding new iterative locations for particles via repulsive forces.

In line with the net force, the particles are exposed, they can go to their new location in case it is a better position. If the possible location of the particle has a worse fitness value; the particle retains its current position. It would be better to clarify that particles can move within the allowed space which is determined by the boundary constraints of variables.

Once each particle decides its neighbors, displacements for each particle must be calculated by considering the impulse-momentum theorem in that total momentum must be conserved. Particles are assumed as the same charge not as magnitude but as a pole that is why all the interactions between neighbors are repulsive as demonstrated along one dimension. This means that any attraction between particles is not allowed. In Figure 27, a demonstration is given for the repulsive forces along one dimension.

Figure 27: Repulsive Forces Along One-Dimension



Source: Erdem, 2007: 60

$$\sum mv = 0, \quad \sum m\frac{\Delta v}{\Delta t} = 0 \tag{3.7}$$

According to the formulation given in Equation 3.6 (Holzner, 2011: 190), fitness values and relocation can also be used as m and velocity change respectively. However, in our approach time is omitted and revised as shown below:

$$\sum f_i \Delta\vec{x}_i = 0 \tag{3.8}$$

Since it is known by Newton's Third Law (Newton, 1999), two repulsive forces F_{x_1} and F_{x_2} are equal to each other, the displacement depends on the current value of the particles. After having done the computations for all the displacements, for all neighbors in all dimensions in the hyperspace, each particle has a new state in both magnitude and direction.

$$x_i^k = x_i^{k-1} + [\Delta x_i^{k-1}\alpha + x_b^{k-1}(1-\alpha)], i = 1, 2, .., n \tag{3.9}$$

where n is the number of dimensions and k is the iteration number,

As mentioned above, relocation runs when the new possible location has a lower fitness value for each particle which is given below:

$$f\left(x_i^k\right) < f\left(x_i^{k-1}\right) \tag{3.10}$$

All particles have the intention to lower energy levels by changing their locations but that is not to say global minimum-level, it may be saddle point, as in the quantum state in the natural processes.

The first iteration is completed after all the repulsions are considered for each particle in hyperspace. After the effect of the compound/net forces exerting on a particle, the unit displacement may be found by the momentum law. The amount of displacement for a given particle that is forced by compound forces (i.e., net forces) would be proportional to related dimensional force and its mass (i.e., objective function value) as shown below.

$$\Delta x_i' = \frac{F\frac{\Delta x_i^2}{r^2}}{f(x_i)} = \frac{F}{f(x_i)}\frac{\Delta x_i^2}{r^2} \quad (3.11)$$

After all final locations are determined for each particle in hyperspace, the first iteration is completed. These process chains are repeated until no remarkable movements and displacements occur. However, if a particle on the new location is under the effect of repulsive forces still, it changes its location as a small increased amount of displacement as shown in Eq. (3.12).

$$\Delta x_i^k = \Psi \Delta x_i^{k-1} \quad (3.12)$$

where $\Psi \in [1.01,1.1]$ and a subjective parameter. In case of improvements Δx_i^{k-1} continues to be multiplied by Ψ. After considering the forces caused by the neighbors, the unified net force is checked lastly and in case of remarkable change, new locations for particles are determined.

3.2.8. Duplication

As a diversification procedure, the REF algorithm employs *Pauli's Exclusion Principle* that two particles cannot occupy the same state in a closed system. Thus, particles can explore hyperspace without trapping into a single point. Indeed, REF guarantees the dissimilarity not only for different charges $f(\vec{x})$ in similar locations but also similar charges in a different location. This situation can be satisfied by controlling and finding new iterative locations for particles via repulsive forces.

After each particle in the population moves to a new location as a result of the effect of its neighbors, duplication control is made in the population to meet Pauli's Exclusion principle. However, a degree of precision is important to determine if the particles are in the same location. In our experiments, locations are assumed as duplication in case of first three digits are the same. In such cases, the following procedures are applied to determine the new position of the same particle heuristically. The new location is determined within the updated domain by considering the best-so-far particle. At this point, the updated range is used instead of the originally defined range because of the balance between diversification and intensification. While duplication check provides diversification, repositioning in a wide range can create too much diversity which disrupts the balance. For this reason, making the repositioning

around the best-known solution has an intensification purpose to balance the exploration-exploitation ability of the REF algorithm. The updated domain for each variable is calculated as below:

Figure 28: The Pseudo-code of Update Interval

```
For each variable
    randInterval = (0.21*log(random()+0.009)+1)* x_bestsofar
    If δ⁻<(x_bestsofar − randInterval) < δ⁺
        δ⁻_new = x_bestsofar − randInterval
    Else
        δ⁻_new = δ⁻
    If δ⁻<( x_bestsofar + randInterval) < δ⁺
        δ⁺_new = x_bestsofar + randInterval
    Else
        δ⁺_new = δ⁺
```

where δ^- and δ^+ are the determined lower-upper limits in the beginning; δ^-_{new} and δ^+_{new} are the updated lower and upper limits; *randInterval* is a random number and $x_{bestsofar}$ is the variable value in the best solution so far. It is worth mentioning that, duplication is checked for each new location found. This procedure continues until there is no more than one particle left from the same location.

3.2.9. Stopping Condition

The stopping condition is related to the number of function evaluations (FES). The pseudocode for the stopping condition is given in Figure 29. Although there is a limit for the number of function evaluations, the algorithm will stop in most of the cases where the main stopping condition is met.

Figure 29: Stopping Condition

```
For each trial
   Set Improvement = FALSE
   Set Counter = 0
   Set φ = 100000
   Do
     Apply REF
     If Improvement = FALSE
        Counter += 1
        If Counter = 1 // In case of first no-improvement
           φ = FES
     Else
        Counter = 0 // In case of finding an improvement
        φ = 100000
   While FES < (1+β)*φ OR FES < Max_FES
```

CHAPTER FOUR
EXPERIMENTAL STUDIES

4.1. MATERIALS AND MODELING ENVIRONMENT

In the evolutionary optimization software literature, various kinds of frameworks have been developed. These frameworks have various advantages and disadvantages and are written in different languages. A comprehensive study reviewed 32 numbers of platforms for optimization problems (Oztas and Erdem, 2021: 3832). According to that study, there is no overwhelming superiority between Java and C-like (C++, C#, and MATLAB) languages and the Python language attracts attention because of its popularity in the field of optimization. Moreover, the authors also reported that management scientists in social sciences tend to prefer easy-to-use platforms which are approximate to human language. In light of this information, we preferred to code the RSAB and REF algorithms in Python language, and PyCharm developed by the Czech company JetBrains was utilized as IDE. Besides, the general structure of Visual Basic codes prepared by Erdem (2007) has been converted into a Python coding environment by utilizing Python libraries and they have been created by using the class structure and object-oriented programming. The experiments are executed on an Intel Core i7 computer with a 2.60GHz CPU and 12 GB RAM under the windows operating system. RSAB-REF algorithm uses approximately 20% CPU and 1GB memory.

4.2. BENCHMARK PROBLEMS

There are standard benchmark optimization problems (i.e. unconstrained, single-objective, multi-objective, combinatorial) in the literature to validate the performance of the algorithms (Jamil and Yang, 2013: 1). As mentioned by Collins and Eaton (1997), test functions can be classified as continuous-discontinuous, convex-non-convex, unimodal-multimodal, quadratic-non-quadratic, low dimensionality-high dimensionality, deterministic-stochastic. According to Talbi (2009), the problems that are high dimension, multimodal and non-differentiable

cannot be solved by deterministic optimization algorithms. However, test functions are generally preferred according to the needs of the algorithms. For that reason, measuring the performance of stochastic optimization algorithms with appropriate test functions has an important place.

In this thesis study, the performance of the proposed algorithm is tested with unconstrained/bounded benchmark problems, engineering design problems, and economic dispatch problem, respectively. The main reason to choose these benchmark problems is that they are the most commonly used test functions and have broad characteristics. The details of the benchmarks are given in the following sub-sections.

4.2.1. Unconstrained/Bounded Problems

The list of the 22 benchmark functions used in this study is given in Appendix 1 with the information of the input domain and the optimal-like solutions. Moreover, we provide the characteristics (dimension, continuity, convexity, multimodality, differentiability, separability) of the benchmarks as below in Table 11. General profiles of the preferred test cases have been tried to be kept quite diverse.

Table 11: The characteristics of test cases

Name	N-dimensional	Continuous	Multimodal	Convexity	Differentiable	Separable
De JongF1	+	+	+	+	+	+
AckleyF1	+	+	+	-	+	-
Rastrigin	+	+	+	+	+	+
Cosine Mixture	+	-	+	-	-	+
Exponential	+	+	+	+	+	-
Cb3	-	+	+	-	+	-
Bohachevsky2	-	+	+	-	+	-
Griewank	+	+	-	-	+	-
Alpine 1	+	+	+	-	+	-
Egg Crate	-	+	+	-	+	-
3-D Paraboloid	-	+	-	+	+	-
Price 2	+	+	+	-	+	-
Schaffer 1	-	+	-	-	+	-
Schwefel 1.2	+	+	-	-	+	-
Xin-She Yang F2	+	-	+	-	-	-
Bird	-	+	+	-	+	-
Beale	-	+	+	-	+	-
McCormick	-	+	+	+	+	-
Giunta	+	+	+	-	+	+
Himmelblau	-	+	+	-	+	-
Branin	-	+	+	-	+	-
Adjiman	-	+	+	-	+	-

4.2.2. Constrained Problems

In the real world, especially in engineering design, optimization problems may have complex constraints. The highlight of the REF algorithm is its ability to deal with constraints very well. Constrained problems are much more important to show the effect of the multiplicative penalty method used in the REF algorithm. Therefore, the algorithm will be tested with the most common real engineering design application benchmarks (Pressure Vessel, Himmelblau's Function, Welded Beam, Tension/Compression Spring Design). In addition, although these benchmarks are referred to as engineering design problems in the literature, problems that are essentially aimed at "cost minimization" can also be regarded as a possible operational problem. Also, Combined Heat and Power Economic Dispatch (CHPED) problem which is an allocation problem in the production sector is applied. The numbers of variables and constraints for each problem are given in Table 12. Their models are provided in Appendix 2-9.

Table 12: Information about constrained problems

Problems	Variables	Constraints
Pressure Vessel	4	4
Himmelblau's Function	5	6
Welded Beam	4	6
Tension/Compression Spring Design	3	4
Combined Heat and Power Economic Dispatch	6	10*

* There are two equality constraints. We consider them as $\leq$ and $\geq$ which become 4 constraints.

In general, the algorithms to which the developed algorithm will be compared are randomly selected and re-run, and reported independently from the scholars who developed the algorithm. However, this situation may cause manipulation by using different parameters, different software hardware, or even different programming languages which give rise to biased results. For this reason, it will be better to compare the developed algorithm with the results of other algorithms as reported in the literature. At this point (if specified), population size, number of iterations, or function evaluation value will be sufficient indicators for comparison. Unfortunately, in some studies in the literature, even these parameters are not shared when making these comparisons.

While choosing the algorithms to be compared, care has been taken to ensure that they are up-to-date and published in high-quality journals. Similar physics-based algorithms are also included if any are executing the same benchmarks. The algorithms are listed in Table 13.

Table 13: The algorithms published in the related literature

Algorithm	Population	Max Iteration	Trial	FES
Genetic Adaptive Search (Deb, 1997) (GeneAS)	100	NA	NA	NA
Self-adaptive Penalty Approach (Coello, 2000) (SPA)	NA	NA	11	NA
Co-evolutionary Particle Swarm Optimization (He and Wang, (2007) (C-PSO)	NA	NA	30	200000
Improved Harmony Search Algorithm (Mahdavi et al., 2007) (IHS)	NA	NA	NA	200000
Harmony Search Algorithm (Vasebi et al., 2007) (HS)	NA	40000	30	25000
Evolution Strategies (Mezura-Montes and Coello, 2008) (ES)	200	NA	30	350000
Modified T-Cell Algorithm (Aragon et al., 2010) (MTCA)	20	30-50	50	320000 ⁞
Charged System Search (Kaveh and Talatahari, 2010b) (CSS)	NA	NA	30	NA
Firefly Algorithm (Gandomi et al., 2011) (FA)	25	1000	NA	50000Ψ
Mesh Adaptive Direct Search Algorithm (Hosseini et al., 2011) (MADS)	NA	NA	50	1000
Ray Optimization (Kaveh and Khayatazad, 2012) (RO)	40	NA	50	NA
Magnetic Charged System Search (Kaveh et al., 2013a) (MCSS)	NA	NA	30	NA
Cuckoo search algorithm (Gandomi et al., 2013b) (CSA)	25	NA	NA	5000
Firefly Algorithm for CHPED (Yazdani et al., 2013) (FA)	40	200	100	NA
Particle Swarm Optimization (Mohammadi-Ivatloo et al., 2013) (PSO)	500	300	NA	NA
Advanced particle swarm assisted genetic algorithm (Dhadwal et al., 2014) (PSO-GA)	NA	NA	30	5000
Artificial Bee Colony Algorithm (Garg, 2014) (ABC)	20*D	500	30	NA
Plant Propagation Algorithm (Sulaiman et al., 2014) (PPA)	40	25	100	30000
Hybrid Flower Pollination Algorithm (Abdel-Raoufi et al., 2014) (H-FPA)	50	1000	30	NA
Modified Oracle Penalty Method (Dong et al., 2014) (MOPM)	30	NA	100	90000
Interior Search Algorithm (Gandomi, 2014) (ISA)	NA	NA	30	30000*
Grey Wolf Optimizer (Mirjalili et al., 2014) (GWO)	NA	NA	NA	NA
Colliding Bodies Optimization (Kaveh and Mahdavi, 2014) (CBO)	20	200	30	4000
Canonical Coordinates Method (Chang and Lin, 2014) (CCM)	NA	400	NA	NA
Optics Inspired Optimization (Kashan, 2015b) (OIO)	19	NA	30	5000
Cuckoo search algorithm for CHPED (Nguyen et al., 2016) (CSA)	25	2000	100	NA
Hybrid PSO-GA Algorithm (Garg, 2016) (H-PSO-GA)	20*D	NA	30	NA
Thermal Exchange Optimization (Kaveh and Dadras, 2017) (TEO)	30	10000	30	NA
Social Cognitive Optimization (Sun and Li, 2018) (SCO)	100	100	20	NA
Seagull Optimization Algorithm (Dhiman & Kumar, 2019) (SOA)	100	1000	30	NA
Pathfinder algorithm (Yapici and Cetinkaya, 2019) (PA)	60	100	NA	NA
Hybrid GSA-GA Algorithm (Garg, 2019) (H-GSA-GA)	20*D	200	30	NA
Butterfly Optimization Algorithm (Arora and Singh, 2019) (BOA)	50	NA	30	NA
Kho-Kho Optimization (Srivastava and Das, 2020) (KKO)	200	500	15	NA
Nuclear Fission-Nuclear Fusion Algorithm (Yalcin and Pekcan, 2020) (N2F)	40	NA	30	30000
Marine Predators Algorithm (Faramarzi et al., 2020a) (MPA)	NA	500	30	25000
Equilibrium Optimizer (Faramarzi et al., 2020b) (EO)	30	500	NA	15000
Search and Rescue Optimization Algorithm (Shabani et al., 2020) (SRO)	20	NA	50	30000‡
Chaotic Grey Wolf Optimizer (Lu et al., 2020) (CGWO)	100	NA	30	40000
Slime mould Algorithm (Li et al., 2020) (SMA)	NA	NA	NA	NA
Chaos Game Optimization (Talatahari & Azizi, 2020) (CGO)	NA	NA	25	NA
Group Teaching Optimization Algorithm (Zhang & Jin, 2020) (GTO)	50	NA	30	10000
Teaching-learning based Marine Predator Algorithm (Zhong et al., 2020) (TLMPA)	NA	NA	30	NA
Improved Grey Wolf Optimizer (Nadimi-Shahraki et al., 2021) (IGWO)	20	$(D*10^4)/20^{\ddagger}$	10	NA

*5000 for Pressure/8900 for Tension; ‡15000 for Welded Beam/25000 for Spring Design; $13000 for Spring Design and Welded Beam; †30 for Welded Beam; ‡D is the number of variables; Ψ25000 for Pressure Vessel; ⁞36000 for Tension and 80000 for Pressure Vessel

4.2.3. Parameter Settings

The parameters used in the RSAB-REF algorithms are determined as a result of trial and error test and presented in Table 14.

Table 14: Parameter Settings

The number of trials	30
The importance constants (c_1, c_2, c_3) (MUPE)	(2.05, 11, 11)
The number of the particles (population size) (n)	20
The incremental parameter (Ψ)	1.02
The stopping parameter (β)	0.5
The neighborhood size (κ)	2
The precision number (ρ)	3
Maximum number of Function Evaluations (Max FES)	30000

The parameters shown in Table 14 are utilized for all benchmarks except some special cases. For example, the importance constants used in MUPE are (2.05, 8, 3), population size is 50 and stopping parameter is 2.5 for CHPED problem; Max FES differs for high dimensional benchmarks in unconstrained/bounded problems (10000 for unconstrained problems with 2-3 dimensions and 100000 for 30 dimensions) and the number of particles is structured as (Dimension*10) for 10&30 dimensional problems.

4.3. EXPERIMENTS ON HYBRID REF ALGORITHM

The main purpose of the RSAB-REF algorithm is to reach the best-known solution in the related literature. The total working time required for this varies according to the complexity of the problem, the number of constraints, and the number of variables. Since the total running time is directly proportional to the number of iterations and the number of FES, this information is presented in detail for each problem, respectively.

4.3.1. Initialization

It is worth mentioning that the RSAB algorithm is repeated with 50 iterations for each trial. Moreover, θ, the initial set size is defined as constant 20. Furthermore, η is taken as 200 in the MUPE approach. In case of fitness value reaches 1.00E-30 in the initialization algorithm (in unconstrained problems), the REF algorithm will be bypassed.

Different from the other studies published in the literature, we report the averages of the variables' updated lower and upper limits as an output of the RSAB algorithm. These limits show narrowed domains as an initialization step before reaching the optimum-like solution.

The initially defined domains and the updated lower and upper limits for each unconstrained/bounded problem after the RSAB algorithm are presented in Table 15. The updated domains are the averages of 30 trials. The main reason to apply RSAB has been achieved when the global optimum values of variables are within the updated ranges. According to Table 15, it is seen that the RSAB algorithm reduced the search space by approximately half.

Table 15: Updated lower and upper limits for unconstrained/bounded problems

	Problem	**Domain**	**Dimension**	**Updated Domains (Averages of 30 trials for each variable)**
1	De JongF1/ Sphere	[-100,100]	3	$x_i \in$ [-43.819476, 35.991105]*
			10	$x_i \in$ [-40.186921, 41.304585]*
			30	$x_i \in$ [-38.259419, 36.537807]*
2	AckleyF1	[-32,32]	2	$x_i \in$ [-13.359408, 13.284727]*
			10	$x_i \in$ [-15.004468, 14.105291]*
			30	$x_i \in$ [-16.73248, 15.463085]*
3	Rastrigin	[-5.12,5.12]	2	$x_i \in$ [-1.942695, 2.239764]*
			10	$x_i \in$ [-2.749607, 2.418027]*
			30	$x_i \in$ [-2.456009, 2.53529]*
4	Cosine Mixture	[-1,1]	10	$x_i \in$ [-0.466412, 0.460306]*
5	Exponential	[-1,1]	2	$x_i \in$ [-0.443979, 0.349052]*
			10	$x_i \in$ [-0.373108, 0.493297]*
6	Cb3	[-5,5]	2	$x_i \in$ [-2.331631, 1.557997]*
7	Bohachevsky2	[-50,50]	2	$x_i \in$ [-17.106277, 25.187521]*
8	Griewank	[-100,100]	2	$x_i \in$ [-47.614122, 54.182451]*
			10	$x_i \in$ [-41.601211, 51.415394]*
			30	$x_i \in$ [-32.454793, 38.934842]*
9	Alpine 1	[-10,10]	2	$x_i \in$ [-6.833573, 2.643768]*
			10	$x_i \in$ [-5.044324, 4.540503]*
			30	$x_i \in$ [-5.029153, 5.028612]*
10	Egg Crate	[-5,5]	2	$x_i \in$ [-1.740414, 2.05632]*
			10	$x_i \in$ [-2.306609, 2.588454]*
			30	$x_i \in$ [-2.572602, 2.340737]*
11	3-D Paraboloid	[-10,10]	3	$x_1 \in$ [-2.092071, 7.723993] ; $x_2 \in$ [-3.117435, 6.751627]; $x_3 \in$ [-5.095768, 4.768861]
12	Price 2	[-10, 10]	2	$x_i \in$ [-4.523815, 4.186838]*

13	Schaffer 1	[-100, 100]	2	$x_i \in$ [-53.175061, 40.762511]*
14	Schwefel 1.2	[-100, 100]	2	$x_i \in$ [-32.507195, 45.576833]*
			10	$x_i \in$ [-49.695003, 44.56004]*
15	Xin-She Yang	[-2π, 2π]	2	$x_i \in$ [-2.593346, 2.516497]*
16	Himmelblau	[-5, 5]	2	$x_1 \in$ [-2.16667, 2.833333]; $x_2 \in$ [-2.00693, 3.005908]
17	Giunta	[-1,1]	2	$x_1 \in$ [-0.001405, 1]; $x_2 \in$ [-0.001544, 1]
18	Adjiman	$x_1 \in$ [-1,2]; $x_2 \in$ [-1,1]	2	$x_1 \in$ [0.538884, 2]; $x_2 \in$ [-0.036695, 0.97748]
19	Branin	$x_1 \in$ [-5,10]; $x_2 \in$ [0,15]	2	$x_1 \in$ [-0.516649, 7]; $x_2 \in$ [3, 10.5]
20	Beale	[-4.5,4.5]	2	$x_1 \in$ [0, 4.5]; $x_2 \in$ [-0.070877, 4.5]
21	Bird	[-2π,2π]	2	$x_1 \in$ [-3.560472, 2.760253] ;$x_2 \in$ [-3.56138, 2.7235]
22	McCormick	$x_1 \in$[-1.5,4]; $x_2 \in$ [-3, 3]	2	$x_1 \in$ [1.5, 1.25]; $x_2 \in$ [-3, 4.53E-05]

* The averages of variables (x_i)

The same procedure is also conducted for constrained problems. It is more difficult to handle RSAB for constrained problems because of the possibility of losing global optimum or violating constraints. However, according to Table 16, the RSAB algorithm reduced search space successfully in case of constraints as well.

Table 16: Updated lower and upper limits for constrained problems

Problem	**Domains**	**Updated Domains (Averages of 30 trials for each variable)**
Pressure Vessel	$x_1 \in$ [0.0625, 10] $x_2 \in$ [0.0625, 10] $x_3 \in$ [0, 100] $x_4 \in$ [0, 240]	$x_1 \in$ [0.0625, 2.090642] $x_2 \in$ [0.0625, 1.244526] $x_3 \in$ [37.866315, 59.989319] $x_4 \in$ [88.292303, 196.400645]
Himmelblau's Function	$x_1 \in$ [78, 102] $x_2 \in$ [33, 45] $x_3 \in$ [27, 45] $x_4 \in$ [27, 45] $x_5 \in$ [27, 45]	$x_1 \in$ [78.013988, 79.141126] $x_2 \in$ [33, 33.861337] $x_3 \in$ [27.019917, 28.188766] $x_4 \in$ [43.607864, 44.8915] $x_5 \in$ [42.857521, 44.971922]
Welded Beam	$x_1 \in$ [0.125, 5] $x_2 \in$ [0.1, 10] $x_3 \in$ [0.1, 10] $x_4 \in$ [0.1, 5]	$x_1 \in$ [0.128382, 0.30487] $x_2 \in$ [1.107596, 7.629883] $x_3 \in$ [8.048392, 9.967568] $x_4 \in$ [0.202204, 0.33574]
Tension/Compression Spring Design	$x_1 \in$ [0.05, 1] $x_2 \in$ [0.25, 1.3] $x_3 \in$ [2, 15]	$x_1 \in$ [0.05, 0.062052] $x_2 \in$ [0.25, 0.655427] $x_3 \in$ [5.765613, 12.91294]
CHPED	$x_1 \in$ [0, 150] $x_2 \in$ [81, 274] $x_3 \in$ [40, 125.8] $x_4 \in$ [0, 180] $x_5 \in$ [0, 135.6] $x_6 \in$ [0, 2695.2]	$x_1 \in$ [0.099725, 16.743839] $x_2 \in$ [115.028577, 157.065205] $x_3 \in$ [40.253933, 69.307731] $x_4 \in$ [10.848522, 84.045112] $x_5 \in$ [15.303951, 99.741783] $x_6 \in$ [0, 22.829386]

RSAB algorithm is not an algorithm that guarantees the optimum solution. Indeed, it provides an adaptive initial (better than pure random initial) solution for continuous unconstrained/bounded or constrained nonlinear optimization problems that may have many local optima. For that reason, it is not proper to compare the results of the global optimization algorithms. Rather, after hybridizing RSAB with REF the best and the worst solutions among 30 trials will be compared with many other algorithms. According to the updated domains presented in Tables 15 and 16, the

RSAB algorithm reduced the search space by adaptively narrowing the boundaries. It is worth mentioning that, this procedure provides a contribution to problem-solving in terms of time and efficiency rather than adding burden to the REF algorithm. Mathematically speaking, the RSAB algorithm consumes approximately 10% of the total function evaluations.

4.3.2. Hybrid REF Algorithm

Since RSAB and REF algorithms are structured as successive procedures, after initializing RSAB, the REF algorithm is ready to be implemented. The experimental study of the Hybrid REF (RSAB-REF) algorithm is conducted for 22 unconstrained, 5 constrained benchmark problems. The findings for each problem are documented in the following sections. The results will be discussed in the Conclusion section.

4.3.2.1. Findings of Unconstrained Problems

The experimental results for 22 unconstrained/bounded benchmarks are given in Table 17. Different dimensions have been applied for these benchmarks to see the capability of RSAB-REF. 13 of them can be regarded as small-sized (2-3 dimension), whereas the rest of them are 10&30 dimensions.

Table 17: Updated lower and upper limits for constrained problems

	Problem	Dimension	Global Min	Best Objective Value	MEAN			Worst Objective Value	Average FES
					Objective Value	Standard Deviation	Standard Error		
1	De JongF1/Sphere	3*	**0.0E+00**	**0.0E+00**	1.54E-09	6.05E-09	1.1E-09	2.97E-08	6753.77
		10		**0.0E+00**	7.47E+01	2.09E+02	3.82E+01	1.05E+03	28380.1
		30		**1.74E-05**	4.44E+03	6.95E+03	1.27E+03	2.69E+04	96355.7
2	AckleyF1	2*	**0.0E+00**	**0.0E+00**	9.84E-11	5.39E-10	9.85E-11	2.95E-09	3571.1
		10		**3.11E-15**	1.07E+00	2.65E+00	4.83E-01	1.03E+01	30385.73
		30		**7.13E-04**	1.14E+00	1.4E+00	2.56E-01	4.34E+00	101381.6
3	Rastrigin	2*	**0.0E+00**	**0.0E+00**	9.96E-02	3.04E-01	5.55E-02	9.96E-01	4078.77
		10		**0.0E+00**	7.79E-01	1.94E+00	3.54E-01	8.96E+00	29697.27
		30		**2.84E-03**	2.08E+01	3.54E+01	6.46E+00	1.76E+02	101404.2
4	Cosine Mixture	10	**-1.0E+00**	**-1.0E+00**	-9.72E-01	1.05E-01	1.91E-02	-5.45E-01	30361.63
5	Exponential	2	**-1.0E+00**	**-1.0E+00**	-1.0E+00	0.0E+00	0.0E+00	-1.0E+00	5338.73
		10		**-1.0E+00**	-9.93E-01	1.58E-02	2.88E-03	-9.33E-01	26431.63
6	Cb3	2*	**0.0E+00**	**0.0E+00**	1.95E-15	1.02E-14	1.87E-15	5.61E-14	3634
7	Bohachevsky2	2*	**0.0E+00**	**0.0E+00**	4.37E-02	8.89E-02	1.62E-02	2.18E-01	6097.63
8	Griewank	2*	**0.0E+00**	**0.0E+00**	1.42E-02	1.81E-02	3.3E-03	6.66E-02	8215.1
		10		**3.33E-16**	8.92E-02	1.67E-01	3.05E-02	6.41E-01	29819.83
		30		**5.16E-08**	9.26E-01	1.68E+00	3.08E-01	6.28E+00	98399
9	Alpine 1	2*	**0.0E+00**	**0.0E+00**	6.68E-09	3.22E-08	5.88E-09	1.76E-07	6204.3
		10		**1.26E-07**	1.38E-02	4.41E-02	8.06E-03	1.96E-01	30431.33

		30		**3.61E-04**	9.17E-01	1.64E+00	2.99E-01	7.51E+00	101561.3
10	Egg Crate	2*	**0.0E+00**	**0.0E+00**	1.6E-21	8.75E-21	1.6E-21	4.9E-20	2736.47
		10		**4.82E-19**	6.48E-01	3.55E+00	6.48E-01	1.94E+01	26419.97
		30		**6.41E-08**	1.75E+01	3.64E+01	6.64E+01	1.75E+02	101321.6
11	3-D Paraboloid	3	**0.0E+00**	**3.2E-09**	1.73E-01	3.14E-01	5.74E-02	8.39E-01	9354.73
12	Price 2	2	**9.0E-01**	**9.0E-01**	9.0E-01	7.23E-06	1.32E-06	9.0004E-01	5291.27
13	Schaffer	2*	**0.0E+00**	**0.0E+00**	2.95E-05	1.62E-04	2.95E-05	8.85E-04	6684.47
14	Schwefel 1.2	2*	**0.0E+00**	**0.0E+00**	1.71E-10	7.89E-10	1.44E-10	4.26E-09	4777.37
		10		**9.88E-18**	1.9E+02	5.53E+03	1.01E+02	2.93E+03	30383.53
15	Xin-She Yang	2*	**0.0E+00**	**0.0E+00**	2.04E-09	1.12E-08	2.04E-09	6.11E-08	3672.63
16	Himmelblau	2	**0.0E+00**	**7.81E-18**	4.59E-10	1.52E-09	4.59E-10	1.38E-08	7419.7
17	Giunta	2	**6.04E-02**	**6.04E-02**	6.04E-02	4.23E-17	7.73E-18	6.04E-02	9271.5
18	Adjiman	2	**-2.02E+00**	**-2.02E+00**	-2.02E+00	1.49E-11	2.71E-12	-2.02E+00	5164.47
19	Branin	2	**3.98E-01**	**3.98E-01**	3.98E-01	4.25E-11	7.75E-12	3.98E-01	9096
20	Beale	2	**0.0E+00**	**1.92E-16**	8.98E-11	2.99E-10	5.48E-11	1.5E-09	9619.73
21	Bird	2	**-1.07E+02**	**-1.07E+02**	-1.07E+02	2.16E-10	3.94E-11	-1.07E+02	9133.57
22	McCormick	2	**-1.91E+00**	**-1.91E+00**	-1.91E+00	1.69E-09	3.08E-10	-1.91E+00	9428.13

In some test cases (*), the global optimum solution is achieved by the RSAB algorithm before starting REF in some of the 30 trials (fitness value reaches 1.00E-30 in RSAB). For this reason, the REF algorithm is bypassed in these trials.

According to the results in Table 17, especially benchmarks with 2-3 dimensions and even 10 dimensions achieved global minimum solutions successfully. For 30 dimensional problems with wide ranges, it can be said that it approaches the global minimum solution value to a certain degree. The standard deviations are noteworthy in problems with 30 dimensions and also large search spaces. However, it can be said that at least one out of 30 trials reached the global solution even in problems with 30 dimensions. Nevertheless, the problems with high dimensions can yield better results with more function evaluation values or stretching the stopping condition.

4.3.2.2. Pressure Vessel

The design of a Pressure Vessel is one of the most commonly used cost optimization problems. This problem has four variables where x_3 and x_4 are continuous while x_1 and x_2 are integer multiplies of 0.0625 inch which are the available thickness of the material (Sandgren, 1990: 227). The model and the figure of the problem are given in Appendix 2-3. The solution for the Pressure Vessel obtained by the RSAB-REF algorithm is given in Table 18. The left-hand side values of each constraint are also provided which show the feasibility of the solution. Only the first constraint value is obtained as 9.94393E-12 which can be ignored.

Table 18: Best solution for Pressure Vessel

Objective Value	x_1	x_2	x_3	x_4
5850.38306	0.75	0.375	38.860104	221.365471
Constraints	$g_1(x)$	$g_2(x)$	$g_3(x)$	
	9.94393E-12	-0.004275	-1.26292E-05	

The descriptive statistics of the 30 trials are given in Table 19. Although maximum FES is limited to 30000, the minimum, maximum, and average FES values are reported as well.

Table 19: Experimental results of Pressure Vessel

Best	5850.38306	**Average Iteration**	145.5
Mean	6344.250094	**Minimum FES**	5681
Worst	7962.557339	**Average FES**	22288.47
Standard Deviation	652.4	**Maximum FES**	30124
Standard Error	119.11		

The Pressure Vessel problem has been handled with many other metaheuristic algorithms previously. Since there are too many algorithms in the literature, only the recently published in high-quality journals are presented chronologically in Table 20. Most of the solutions could not be able to satisfy "having x_1 and x_2 as integer multiplies of 0.0625". The best-known solution for Pressure Vessel seems obtained firstly by Mahdavi et al. (2007) and then by Gandomi et al. (2011). However, according to the reported values of variables, the true objective value should be 5850.384 in IHS and FA presents 0.27 total deviation from constraints. RSAB-REF algorithm has provided the best-known solution with a smaller set size, fewer function evaluations, and no violation compared to FA and IHS.

Table 20: Comparisons for Pressure Vessel (Best-so-far solution)

Reference	***x1***	***x2***	***x3***	***x4***	**Objective**
GeneAS	0.9375	0.5	48.329	112.679	6410.3811
SAP	0.8125	0.4375	40.3239	200	6288.7445
C-PSO	0.8125	0.4375	42.091266	176.7465	6061.0777
IHS	*0.75*	*0.375*	*38.8601*	*221.36553*	*5849.76169*
ES	0.8125	0.4375	42.098087	176.640518	6059.745605
MTCA	0.8125	0.4375	42.098429	190.787695	6390.554
CSS	0.8125	0.4375	42.103624	176.572656	6059.09
FA	*0.75*	*0.375*	*38.8601*	*221.36547*	*5850.38306 (0.27)*
MCSS	0.8125	0.4375	42.10455	176.560967	6058.97
ISA	0.8125	0.4375	42.09845	176.6366	6059.7143
GWO	0.8125	0.4345	42.089181	176.758731	6051.5639
CBO	*0.779946*	*0.38556*	*40.49065*	*198.76232*	*5889.911 (0.001)*
TEO	0.779151	0.385296	40.369858	199.301899	5887.511073
SOA	*0.77808*	*0.383247*	*40.31512*	*200*	*5879.5241 (319.82)*

PA	*0.778168*	*0.3846489*	*40.31964*	*199.9999*	*5885.3351 (1e-06)*
N2F	1.125	0.625	58.290155	43.6926562	7197.72893
MPA	0.8125	0.4375	42.098445	176.636607	6059.7144
EO	0.8125	0.4375	42.098446	176.636596	6059.7143
SRO	0.8125	0.4375	42.098446	176.636596	6059.714335
SMA	0.7931	0.3932	40.6711	196.2178	5994.1857
CGO	0.778169	0.51	40.319619	200	6247.672819
GTO	*0.778169*	*0.38465*	*40.3196*	*200*	*5885.333 (1.33)*
TLMPA	*0.778169*	*0.384649*	*40.319618*	*200*	*5885.332774 (0.05)*
IGWO	0.779031	0.385501	40.36313	199.4017	5888.34
Proposed Algorithm	**0.75**	**0.375**	**38.860104**	**221.365471**	**5850.38306**

The descriptive statistics of the experiments are shown in Table 21. As it is seen, the performance of the RSAB-REF algorithm is better than the solutions reached by other algorithms. Although the worst and the standard deviation are relatively higher than the others, the feasible best-known solution is obtained by the proposed algorithm.

Table 21: Comparisons for Pressure Vessel (Descriptive Statistics)

Reference	Best	Mean	Worst	Std dev
GeneAS	6410.3811	N/A	N/A	N/A
SAP	6288.7445	6293.843232	6308.149652	7.413285
C-PSO	6061.0777	6147.1332	6363.8041	86.4545
IHS	*5849.76169*	N/A	N/A	N/A
ES	6059.745605	6850.004948	7332.879883	4.26E+02
MTCA	6390.554	6737.065147	7694.066881	3.57E+02
CSS	6059.09	6,067.91	6085.48	10.2564
FA	5850.38306 *(0.27)*	N/A	N/A	N/A
MCSS	6058.97	6063.18	6074.74	9.73494
ISA	6059.714	6410.087	7332.846	384.6
GWO	6051.5639	N/A	N/A	N/A
CBO	5889.911 *(0.002)*	5934.201	6213.006	63.5417
TEO	5887.511073	5942.565917	6134.187981	62.2212
SOA	5879.5241 *(319.82)*	5883.0052	5893.4521	256.415
PA	5885.3351 *(1e-06)*	N/A	N/A	N/A
N2F	7197.72893	7197.72905	7197.72924	7.90E-05
MPA	6059.7144	6102.8271	6410.0929	106.61
EO	6059.7143	6668.114	7544.4925	566.24
SRO	6059.714335	6091.32594	6410.0868	8.03E+01
SMA	5994.1857	N/A	N/A	N/A
CGO	6247.672819	6250.957354	6330.958685	10.759156
GTO	5885.333 *(1.33)*	N/A	N/A	N/A
TLMPA	5885.332774 *(0.05)*	N/A	N/A	N/A
IGWO	5888.34	N/A	N/A	N/A
Proposed Algorithm	**5850.38306**	**6344.25**	**7962.557**	**652.4**

4.3.2.3. Himmelblau's Function

Himmelblau proposed a non-linear constrained optimization problem in 1972 and it is regarded as a mechanical engineering problem (Kumar et al., 2020b: 25). This well-known problem has five variables and six constraints. The model of the problem is given in Appendix 4. The solution for Himmelblau's Function obtained by the

RSAB-REF algorithm is given in Table 22. As it is seen, all constraints are within the defined range which means that there are no violations.

Table 22: Best solution for Himmelblau's Function

Objective	x_1	x_2	x_3	x_4	x_5
-31025.55751	78.000017	33.000008	27.071017	45	44.969175
Constraints	$g_1(x)$*	$g_2(x)$*	$g_3(x)$*		
	91.999991	100.404777	20		

*$0 \leq g_1(x) \leq 92;\ 90 \leq g_2(x) \leq 110; 20 \leq g_3(x) \leq 25$

The experimental results for Himmelblau's Function are given in Table 23 as descriptive statistics. Maximum FES is limited to 30000, the minimum and the average FES values are reported as well.

Table 23: Experimental results of Himmelblau's Function

Best	-31025.55751	**Average Iteration**	69.83
Mean	-31017.015168	**Minimum FES**	5583
Worst	-30904.583815	**Average FES**	10841.87
Standard Deviation	25.89	**Maximum FES**	18655
Standard Error	4.73		

In Table 24, the feasible best-known solutions obtained for Himmelblau's Function are given. It is clear that the best-known solution is reached by H-GSA-GA with -31027.64076 which was published in 2019. However, the H-GSA-GA utilizes 20*Dimension as population size which becomes 100. Besides, other algorithms reached the -31025.5 objective value in general. ABC and H-PSO-GA have negligible violations in some constraints shown in brackets.

Table 24: Comparisons for Himmelblau's Function (Best-so-far solution)

Reference	***x1***	***x2***	***x3***	***x4***	***x5***	**Objective**
CSA	78	33	29.99616	45	36.77605	-30665.233
PSO-GA	78	33	29.99525	45	36.77582	-30665.5389
ABC	78	33	27.070979	45	44.969024	-31025.57569 (3.5e-05)
PPA	78	33	29.9952	45	36.7758	-30665.54
H-FPA	N/A	N/A	N/A	N/A	N/A	-31025.5654
OIO	N/A	N/A	N/A	N/A	N/A	-31025.50178
H-PSO-GA	78	33	27.070951	45	44.969167	-31025.57471 (2.8e-05)
H-GSA-GA	77.961	32.99948	27.072836	45	44.973943	-31027.64076
Proposed Algorithm	**78.000017**	**33.000008**	**27.071017**	**45**	**44.969175**	**-31025.55751**

The descriptive statistics for Himmelblau's Function are listed in Table 25. As it is seen, the performance of the REF algorithm for Himmelblau's Function is considerably good.

Table 25: Comparisons for Himmelblau's Function (Descriptive Statistics)

Reference	Best	Mean	Worst	Std dev
CSA	-30665.233	N/A	N/A	11.6231
PSO-GA	-30665.5389	-30665.53697	-30665.48996	8.76E-03
ABC	-31025.57569 (3.5e-05)	-31025.55841	-31025.49205	0.0153528
PPA	-30665.54	N/A	N/A	N/A
H-FPA	-31025.5654	NA	NA	NA
OIO	-31025.50178	-31024.5348	-31020.60517	1.2092
H-PSO-GA	-31025.57471 (2.8e-05)	-31025.55782	-31025.49205	0.01526
H-GSA-GA	-31027.64076	-31026.07246	-31025.38705	0.01803
Proposed Algorithm	**-31025.6**	**-31017**	**-30904.6**	**25.89**

4.3.2.4. Welded Beam

The welded beam is a structural optimization problem that is generally preferred as a benchmark (Ragsdell and Phillips, 1976: 1021). This cost optimization problem consists of four design variables and six constraints. The model and its figure are given in Appendix 5-6. The solution for Welded Beam obtained by the RSAB-REF algorithm is given in Table 26. Furthermore, to show the feasibility of the solution the left-hand side values of each constraint are also provided. As it is seen from the constraint values, the solution is feasible as well.

Table 26: Best solution for Welded Beam

Objective	x_1	x_2	x_3	x_4
1.724867	0.205734	3.470438	9.036532	0.205734
Constraints	$g_1(x)$	$g_2(x)$	$g_3(x)$	
	-1.52501E-05	-0.000243	-1.72409E-07	
	$g_4(x)$	$g_5(x)$	$g_6(x)$	
	-3.390645	-0.23554	-0.327636	

The experimental results for Welded Beam are given in Table 27 as descriptive statistics. Although maximum FES is limited with 30000 and actual FES values are also reported.

Table 27: Experimental results of Welded Beam

Best	1.724867	**Average Iteration**	162.53
Mean	1.79025	**Minimum FES**	9163
Worst	2.078897	**Average FES**	25511.57
Standard Deviation	0.08	**Maximum FES**	30129
Standard Error	0.02		

The feasible best-known solution for Welded Beam has been obtained as 1.724852 in the literature. Most of the algorithms have reached the feasible best-known solution, however, the solution values obtained differ after the 5th decimals as shown in Table 28.

Table 28: Comparisons for Welded Beam (Best-so-far solution)

References	***x1***	***x2***	***x3***	***x4***	**Objective**
GeneAS	0.2489	6.173	8.1789	0.2533	2.433116
SAP	0.2088	3.4205	8.9975	0.21	1.748309
C-PSO	0.202369	3.544214	9.04821	0.205723	1.728024
IHS	*0.20573*	*3.047049*	*9.03662*	*0.20573*	*1.7248 (1505.76)*
ES	0.19972	3.61206	9.0375	0.206082	1.7373
MTCA	0.244369	6.218613	8.291474	0.244369	2.38113
CSS	0.20582	3.468109	9.038024	0.205723	1.724866
RO	0.203687	3.528467	9.004233	0.207241	1.735344
MCSS	0.20573	3.470489	9.036624	0.20573	1.724855
ISA	0.2443303	6.219931	8.291521	0.244369	2.3812
GWO	0.205676	3.478377	9.03681	0.205778	1.72624
CBO	*0.205722*	*3.47041*	*9.037276*	*0.205735*	*1.724663*
TEO	0.205681	3.472305	9.035133	0.205796	1.725284
SOA	*0.205408*	*3.472316*	*9.035208*	*0.201141*	*1.723485 (1104.85)*
PA	*0.0205795*	*3.470495*	*9.036624*	*0.20573*	*1.724853 (124635.3)*
N2F	0.20573	3.470489	9.036624	0.20573	1.724852
MPA	*0.205728*	*3.470509*	*9.036624*	*0.20573*	*1.724853 (0.05)*
EO	0.2057	3.4705	9.03664	0.2057	1.7549
SRO	0.20573	3.470489	9.036624	0.20573	1.724852
CGWO	0.20573	3.470499	9.036637	0.20573	1.724854
SMA	*0.2054*	*3.2589*	*9.0384*	*0.2058*	*1.69604 (725.21)*
CGO	*0.198856*	*3.337244*	*9.191454*	*0.198858*	*1.670336 (1252.9)*
GTO	0.20573	3.470489	9.036624	0.20573	1.724852
TMPA	0.20573	3.470489	9.036624	0.20573	1.724852
IGWO	0.20573	3.47049	9.036624	0.20573	1.724853
Proposed Algorithm	**0.205734**	**3.470438**	**9.036532**	**0.205734**	**1.724867**

Different from Pressure Vessel and Himmelblau's function, there are some infeasible solutions reported in the literature as can be seen in Table 28. According to the table, IHS, SOA, PA, MPA, SMA, and CGO reached infeasible solutions and their total deviations are given in brackets. Furthermore, CBO documented its best solution as 1.724663 which is wrong. According to the variables, its best reached objective value is 1.724983 which is not a rounding error.

In Table 29, the descriptive statistics for Welded Beam are reported. Table 29 indicates that the RSAB-REF algorithm achieves the best-known solution nearly (with a 2.3E-05 deviation from the best-known solution which is negligible).

Table 29: Comparisons for Welded Beam (Descriptive Statistics)

Reference	Best	Mean	Worst	Std dev
GeneAS	2.433116	N/A	N/A	N/A
SAP	1.748309	1.771973	1.785835	0.011223
C-PSO	1.728024	1.748831	1.782143	0.012926
IHS	*1.7248 (1505.76)*	N/A	N/A	N/A
ES	1.7373	1.81329	1.994651	7.05E-02
MTCA	2.38113	2.439811	2.710406	0.093146
CSS	1.724866	1.739654	1.759479	0.008064
RO	1.735344	1.9083	N/A	0.173744
MCSS	1.724855	1.735374	1.750127	0.007571
ISA	2.3812	2.4973	2.67	1.02E-01
GWO	1.72624	N/A	N/A	N/A
CBO	*1.724662*	*1.725707*	*1.725059*	*0.000244*
TEO	1.725284	1.76804	1.931161	0.058166
SOA	1.723485*(1104.85)*	1.724251	1.727102	1.724007
PA	*1.724853(124635.3)*	N/A	N/A	N/A
N2F	1.724852	1.725	1.726147	3.39E-04
MPA	*1.724853 (0.05)*	*1.724861*	*1.724873*	*6.41E-06*
EO	1.724853	1.726482	1.736725	0.003257
SRO	1.724852	1.724852	1.724852	2.22E-11
CGWO	1.724854	1.724854	1.724854	3.36E-16
SMA	*1.69604 (725.21)*	N/A	N/A	N/A
CGO	1.670336 *(1252.9)*	1.670378	1.670903	9.30E-05
GTO	1.724852	N/A	N/A	N/A
TLMPA	1.724852	N/A	N/A	N/A
IGWO	1.724853	N/A	N/A	N/A
Proposed Algorithm	**1.724875**	**1.7708**	**1.949681**	**0.05**

4.3.2.5. Tension/Compression Spring Design

Tension/ Compression spring design problem is an optimization benchmark that aims to minimize the weight of the spring by satisfying requirements (Arora, 2017: 47). There are three variables and four constraints. Its model and figure are given in Appendix 7-8. The best solution for Tension/Compression Spring Design obtained by the RSAB-REF algorithm is given in Table 30. As it is seen from Table 30, all constraints are satisfied.

Table 30: Best solution for Tension/Compression Spring Design

Objective	x_1	x_2	x_3	
0.012665	0.05183	0.360125	11.091955	
Constraints	$g_1(x)$	$g_2(x)$	$g_3(x)$	$g_4(x)$
	6.07936E-12	-2.54827E-12	-4.060463	-0.725363

The experimental results for Tension/Compression Spring Design are given in Table 31 as descriptive statistics. Maximum FES is limited to 30000, the minimum and the average FES values are also reported.

Table 31: Experimental results of Tension/Compression Spring Design

Best	0.012665	**Average Iteration**	180.93
Mean	0.012858	**Minimum FES**	6917
Worst	0.013368	**Average FES**	27053.13
Standard Deviation	0.0002	**Maximum FES**	30122
Standard Error	4.27E-05		

According to Table 32, the feasible best-known solution is 0.012665 reported in the literature. The RSAB-REF algorithm also reached the value of the best-known solution. It is worth noting that, the solutions less than 0.012665 (CSS, MCSS, and SOA) violated some constraints and their total deviations which are relatively small are given in brackets.

Table 32: Comparisons for Tension/Compression Spring Design (Best-so-far solution)

Reference	***x1***	***x2***	***x3***	**Objective**
GeneAS	0.5148	0.351661	11.632201	0.012705
SAP	0.051728	0.357644	11.244543	0.012675
IHS	0.051154	0.349871	12.076432	0.012671
ES	0.051643	0.35536	11.397926	0.012698
MTCA	0.051622	0.355105	11.384534	0.012665
CSS	*0.051744*	*0.358532*	*11.165704*	*0.012638 (0.01)*
RO	0.05137	0.349096	11.7679	0.012679
MCSS	*0.051627*	*0.35629*	*11.275456*	*0.012607 (0.02)*
ISA	N/A	N/A	N/A	0.012665
GWO	0.05169	0.356737	11.28885	0.012666
CBO	0.051894	0.361674	11.007846	0.01267
TEO	0.051775	0.358792	11.16839	0.012665
SOA	*0.051065*	*0.342897*	*12.0885*	*0.012645 (0.04)*
PA	0.051727	0.35763	11.235724	0.012665
MPA	0.051725	0.35757	11.239196	0.012665
IGWO	0.05162	0.355055	11.387968	0.012666
SRO	0.051689	0.356723	11.288648	0.012665
GBO	0.05203	0.36509	10.81456	0.012667
CGO	0.051663	0.356078	11.326575	0.012665
TLMPA	0.051681	0.356533	11.299823	0.012665
Proposed Algorithm	**0.05183**	**0.360125**	**11.091955**	**0.012665**

The descriptive statistics for Tension/Compression Spring Design are listed in Table 33. As it is seen, the performance of the RSAB-REF algorithm is better than the solutions reached by other algorithms.

Table 33: Comparisons for Tension/Compression Spring Design (Descriptive Statistics)

Reference	Best	Mean	Worst	Std dev
GeneAS	0.012705	0.012769	0.012822	3.94E-05
SAP	0.012675	0.01273	0.012924	5.20E-05
IHS	0.012671	N/A	N/A	N/A
ES	0.012698	0.013461	0.016485	9.66E-04
MTCA	0.012665	0.012732	0.013309	9.40E-05
CSS	*0.012638 (0.01)*	*0.012852*	*0.013626*	*8.3564e−5*
RO	0.012679	0.013547	N/A	0.001159
MCSS	*0.012607 (0.02)*	*0.012712*	*0.012982*	*4.7831e−5*
ISA	0.012665	0.013165	0.012799	1.59E-02
GWO	0.012666	N/A	N/A	N/A
CBO	0.01267	0.01273	0.0128808	5.00E-05
TEO	0.012665	0.012685	0.012715	4.41E-06
SOA	*0.012645 (0.04)*	*0.012666*	*0.012666*	*0.001108*
PA	0.012665	N/A	N/A	N/A
MPA	0.012665	0.012665	0.012665	5.55E-08
IGWO	0.012666	0.013017	0.013997	3.91E-04
SRO	0.012665	0.012665	0.012668	1.26E-07
GBO	0.012667	0.012696	N/A	3.36E-05
CGO	0.012665	0.01267	0.012719	1.09E-05
TLMPA	0.012665	N/A	N/A	N/A
Proposed Algorithm	**0.012665**	**0.012858**	**0.013368**	**0.0002**

4.3.2.6. Combined Heat and Power Economic Dispatch Problem

The combined heat and power economic dispatch problem (CHPED) is a minimization problem that includes two sub-problems as heat dispatch and power dispatch. This problem aims to calculate the unit heat and power production while minimizing total cost and satisfying power and heat demands (Guo et al., 1996: 1779). This benchmark problem has six decision variables (P_1, P_2, P_3, H_2, H_3, H_4) and twelve constraints (Sun and Li, 2018: 8), and its model is given in Appendix 9. However, since the proposed algorithm is structured on inequality constraints, the heat and power demand constraints are regarded as both " $\leq$ " and " $\geq$ ".

Since this problem has many constraints and includes demand equations which mean four inequalities, finding a stable solution is a challenging problem for the RSAB-REF algorithm. The best solution and constraints values obtained by the RSAB-REF algorithm are given in Table 34.

Table 34: Best solution for CHPED

Objective	P_1	P_2	P_3	
9257.075	0	160	40	
	H_2	H_3	H_4	
	40	75	0	
Constraints	$g_1(x)$	$g_2(x)$	$g_3(x)$	$g_4(x)$
	0	0	0	0
	$g_5(x)$	$g_6(x)$	$g_7(x)$	$g_8(x)$
	-79.888889	-67.993893	-194.468083	-79.360456
	$g_9(x)$	$g_{10}(x)$		
	-1.25E-07	-3.0E-08		

It is an expected result that the variables are integers due to the nature of the CHPED problem. Therefore, even if it is not defined as an integer at the beginning, the best-known solution (0, 160, 40, 40, 75, 0) has integer variables. The experimental results for the CHPED problem are presented in Table 35.

Table 35: Experimental results of CHPED

Best	9257.075	**Average Iteration**	85.97
Mean	9625.02155	**Minimum FES**	9922
Worst	10241.75	**Average FES**	27442.4
Standard Deviation	274.69	**Maximum FES**	30305
Standard Error	50.15		

In the literature, the best-known solution for the CHPED problem is also obtained as 9257.075. Moreover, as it is seen from Table 36, there are also violated solutions such as CCM and KKO. Their total deviations from constraints are given in brackets. According to Table 13, the algorithms that reach the best solution in the literature were performed with particles between 25 and 500. Accordingly, the performance of RSAB-REF with 50 particles can be evaluated as relatively good.

Table 36: Comparisons for CHPED (Best solution)

Reference	P_1	P_2	P_3	H_2	H_3	H_4	**Objective**
HS	0	160	40	40	75	0	9257.07
MADS	0	160	40	40	75	0	9257.07
FA	0.0014	159.9986	40	40	75	0	9257.1
PSO	0	160	40	40	75	0	9257.07
CCM	*0*	*200*	*0*	*0*	*115*	*0*	*8606.07 (123.63)*
CSA	N/A	N/A	N/A	N/A	N/A	N/A	9257.075
SCO	0	160	40	40	75	0	9257.07
KKO	*0.0282*	*155.015*	*44.9568*	*18.1301*	*96.8699*	*0*	*9217.03 (20.38)*
Proposed Algorithm	**0**	**160**	**40**	**40**	**75**	**0**	**9257.075**

The descriptive statistics for CHPED are listed in Table 37. As can be seen from the table, the solution of this problem has reached the best-known solution with the RSAB-REF algorithm.

Table 37: Comparisons for CHPED (Descriptive Statistics)

Reference	Best	Mean	Worst	Std dev
HS	9257.07	NA	NA	NA
MADS	9257.07	9257.515	9260.432	1.0743
FA	9257.1	N/A	N/A	N/A
PSO	9257.07	N/A	N/A	N/A
CCM	*8606.07 (123.63)*	N/A	N/A	N/A
CSA	9257.075	9259.165	9327.972	9.886
SCO	9257.07	9263.34	9276.23	NA
KKO	*9217.03 (20.38)*	N/A	N/A	N/A
Proposed Algorithm	**9257.075**	**9625.022**	**10241.75**	**274.69**

CONCLUSION

Optimization comprises many techniques to provide optimal solutions for problems including scheduling, allocation, designing in a wide range of fields. Regardless of the complexity of problems, optimization is a process for minimizing or maximizing a function under given conditions. However, as the complexity of the problems is getting higher, advanced optimization techniques are required. A problem can be defined as tractable in case of solving it in a reasonable computation time. This kind of problem belongs to Class P which means they can be solved by traditional optimization techniques in a polynomial time. Most of the real-world optimization problems are classified in NP-Hard where the problems have nonlinear characteristics and cannot be solved in polynomial time. Nevertheless, there are no efficient deterministic algorithms for the problems denoted as NP-Hard. Therefore, stochastic algorithms which are constructed on randomization somehow in their principles are needed. Especially with the developments of artificial intelligence and computation technology, modern optimization techniques become metaheuristic algorithms that provide approximate solutions to NP-Hard problems.

The breaking point in problem-solving literature dates back to the early 1960s with the "evolution" concept. Genetic Algorithm, Evolutionary Programming, Evolutionary Strategies, Genetic Programming are pioneers for computational methods inspired by evolutionary processes. In the years of 1980s, with the developments of Simulated Annealing and Tabu Search, more metaphors have started to be used in algorithms. After 2000, the behaviors of living organisms, facts in nature, laws in the field of science have encouraged researchers to mimic these mechanisms for developing algorithms. In the state-of-the-art, many algorithms imitate these behaviors and interactions for solving real-world optimization problems. The number of new metaheuristic algorithms based on metaphors has exploded especially in the last decade. Although this topic attracts attention in the literature, some researchers asserted that the newly introduced metaheuristics have similarities regardless of different metaphors.

However, according to the "No Free Lunch Theorem", there cannot be an algorithm that is appropriate for all problems. For this reason, as real-world

optimization problems exist, scholars will continue to focus on developing novel algorithms that can solve most types of problems.

Discussion about Findings

The performance of the RSAB-REF algorithm has been tested with the most common unconstrained/bounded benchmark problems, engineering design problems, and economic dispatch problem respectively. Although these benchmarks are referred to as engineering design problems in the literature, problems that are essentially aimed at "cost minimization" can also be regarded as a business problem. The updated domains as the output of the RSAB algorithm and the best solutions obtained as a result of the RSAB-REF algorithm have been presented in Table 15-37.

According to the findings of the RSAB algorithm, it is clear that the search space was reduced by approximately half for each unconstrained problem. For the RSAB algorithm, the initial set size was used as 20 regardless of the dimension size of the problems, and the amount of increase in case of no improvement was used as 0 since RSAB is hybridized with the REF algorithm. It is recommended that the initial set size can be increased when the RSAB algorithm is applied only. For this reason, there is no radical difference between dimensions in test cases in terms of narrowing the search space. However, since 30 trials were run for each test function, the reporting of narrowed areas was only made as to the average of all trials and variables. As for constrained problems, it is more difficult to handle the RSAB algorithm because of the possibility of losing the global optimum or violating constraints. However, according to the findings, the RSAB algorithm reduced search space successfully in case of constraints as well. It is worth noting that, the reported domains succeed in between 60%-90% approximately among 30 trials for constrained problems. The reason for the updated range to miss the global minimum is thought to be since the constraints do not contain the relevant variable. However, this situation does not prevent the REF algorithm from reaching the global minimum. This procedure provides a contribution to problem-solving in terms of time and efficiency rather than adding a burden to the REF algorithm. Mathematically speaking, the RSAB algorithm consumes approximately 10% of the total function evaluations.

According to the findings of Hybrid RSAB-REF for unconstrained problems, especially benchmarks with 2-3 sizes and even 10 sizes achieved global minimum solutions successfully. In addition to achieving a global minimum, standard deviations and errors indicate that the findings are consistent among 30 trials. As for 30-dimensional problems, it can be said that at least one out of 30 trials reached the global solution to a certain degree. However, the standard deviations are noteworthy in problems with 30 dimensions and large search spaces. Therefore, the problems with high dimensions can yield better results with more function evaluation values or when stretching the stopping condition.

It is very important to test constrained benchmark problems to show the capability of the MUPE approach employed in the REF algorithm. Although the maximum FES value is limited to 30000 for all constrained problems, in some cases the algorithm terminated with lower FES values because of the stopping condition. This helps the REF algorithm to quickly complete that trial in case the global solution is missed in the RSAB algorithm.

According to the experimental results for the Pressure Vessel problem, the best feasible solution was reached as 5850.38306 by satisfying all constraints with the RSAB-REF algorithm. The result compared with many algorithms published in high-quality journals. According to the solutions published in the literature, it seems that HIS and FA have obtained the best-known solution so far. However, after a detailed examination, it was recognized that the reported values for IHS and FA did not reflect the truth as explained in the previous section. Besides, most of the studies could not be able to satisfy "having x_1 and x_2 as integer multiplies of 0.0625". Although the worst and the standard deviation are relatively higher than the others, the feasible best-known solution in the literature is obtained by the proposed algorithm. Moreover, the RSAB-REF algorithm has provided the best-known solution with a smaller set size, fewer function evaluations, and no violation compared to FA and IHS.

According to the findings for Himmelblau's function, the best-known solution is reached by H-GSA-GA with -31027.64076 which is published in 2019. However, the H-GSA-GA utilizes 20*Dimension as population size which becomes 100. Besides, other algorithms reached the objective value of -31025.5 in general. Besides, there are also infeasible solutions reached by other algorithms in Table 24-25. The

performance of the RSAB-REF algorithm for Himmelblau's Function is considerably good since -31025.5 was obtained and all constraints were satisfied.

According to the experimental results for Welded Beam, 1.724867 was obtained by the RSAB-REF algorithm which is feasible. The feasible best-known solution for Welded Beam has been obtained as 1.724852 in the literature. Most of the algorithms have reached the feasible best-known solution, however, the solution values obtained differ after the 5th decimals. The RSAB-REF algorithm achieved the best-known solution with a 2.3E-05 deviation from the best-known solution which is negligible. According to the comparison table, IHS, SOA, PA, MPA, SMA, and CGO reached infeasible solutions. Although CBO documented its best solution as 1.724663 which seems the best-known solution in the literature, its adjusted objective value is 1.724983 which cannot be a rounding error. As a result, it can be concluded that the RSAB-REF algorithm performed quite well for the Welded Beam problem.

Another unconstrained benchmark problem is Tension/Compression Spring Design problem. The RSAB-REF algorithm was reached 0.012665 by satisfying all constraints which is the best-known solution in the literature as well. However, according to the comparison table, the solutions violated some constraints less than 0.012665 (CSS, MCSS, and SOA) with relatively smaller deviations. Therefore, it is clear that the RSAB-REF algorithm also performed well for this problem too.

Except for engineering design problems, CHPED was preferred to be tested as a different business problem. This problem aims to calculate the unit heat and power production. Although it is not defined as integer programming, the best-known solution in the literature has integer values. However, this result is not surprising because of the nature of the allocation problem. Finding a stable solution for the CHPED problem is challenging for the RSAB-REF algorithm since this problem has many constraints and includes demand equalities. Since some deviation of constraints is allowed in MUPE, constraints should be treated as inequalities in the RSAB-REF algorithm. For this reason, considering equality constraints as two inequalities ($\leq, \geq$) makes CHPED a more challenging problem. According to the findings for the CHPED problem, the RSAB-REF algorithm was reached the best-known solution (9257.075) without any violation. However, it would be better to mention that variables were restricted to integers. According to the solutions published in the literature, CCM and

KKO violated some constraints. When the population sizes (changing between 25-500) used by the algorithms obtained 9257.075 are examined, it can be concluded that the performance of RSAB-REF with 50 particles is relatively good. However, it is worth mentioning that the RSAB-REF algorithm is open to improvement to obtain more stable results with fewer particles for problems like CHPED.

Discussion about Similar Algorithms

As an overall assessment, it is obvious that the REF algorithm is similar to algorithms in both the magnetism class and Newton's gravitational law class since it employs Coulomb's Law, Pareto's Principle, Pauli's Exclusion Principle, Newton's Third Law, Momentum Law. Considering the physics-based algorithms introduced in Section 2, it is seen that the REF algorithm belongs to both the magnetism class and Newton's gravitational law class.

The most prominent feature of REF is the "repulsive forces between particles". No algorithm relies solely on repulsive forces, as in the REF algorithm. GSA, CFO, CSS allow particles only to attract each other, whereas EM, MOA, MCSS, EFO, APO, GIO are the algorithms that allow particles to attract and repulse each other. Among them, MOA, MCSS, and EFO consider attraction between particles strongly than repulsion.

Another prominent principle in the REF algorithm is Coulomb's Law. EM and CSS, which belong to the magnetism class, are also based on Coulomb's Law as well. However, EM differentiates in terms of allowing attraction whereas CSS also utilizes Gauss Law additionally.

Moreover, REF employs memory. Thus, in each iteration of the algorithm, it is aimed to beat the previous best. In this way, it is easier to reach the global optimum. As for the others, EM and CSS also use memory in their principle. On the other hand, although memory usage provides better convergence, Rashedi et al. (2009) asserted that GSA is a memoryless algorithm but works as effectively as algorithms with memory.

Furthermore, REF calculates repulsive force by considering a Gravity Constant as a function that depends on the logarithm of the distance between particles. GSA

also incorporates Gravitational Constant as a function based on the total number of iterations and the time and it decreases dynamically through the iterations. Unlike REF and GSA, GIO algorithm employs a constant number for Gravitational Constant.

As for the neighborhood principle, REF takes into account a defined number of neighbors which are the most-closest ones and this means that a particle can be affected only by its neighbors. Furthermore, REF employs Euclidean distance between particles as in GSA. Among the similar algorithms, MOA considers repulsion and attraction until the distance approaches infinity between particles. Besides, the power of the distance parameter used in Newton's Gravitational Law is flexible between -5 to 5 in the APO algorithm rather than square which is used generally.

The highly dissociative property is whether the mass or energy of the particles is taken into account in algorithms. Since the REF algorithm is inspired by Coulomb's Law, the particles get involved in force calculation as electrical charges. However, the algorithms belong to Newton's Gravitational Law and MOA utilizes particle mass. On the other hand, HO, EM, CSS, MCSS consider charges of particles.

Furthermore, the REF algorithm assumes that particles can move in hyperspace through only a path but not like a wave. A similar principle is also employed by the MCSS algorithm by allowing only straight-line movements.

Although the REF algorithm and the algorithms mentioned above have both common and distinctive features, it should be noted that the main purpose of this study is not to duplicate existing algorithms through a new metaphor, but to provide an algorithm that converges the best-known solution values. To reach better solutions it is very important to determine the best combination of features inspired from the literature or metaphors. Therefore, achieving better results in a specific problem type is the primary goal, regardless of metaphors and similarities. It is also worth noting that physics-based algorithms, which have common features with REF, have been generally applied for unconstrained problems. Those developed for constrained real-world problems have been included in the comparison list within the scope of this thesis.

Theoretical Implications

In this thesis study, an evolutionary computation model for unconstrained/bounded and constrained continuous optimization problems has been developed. This model is constituted as a hybrid algorithm that includes RSAB as an initialization algorithm and REF as the main algorithm. These two algorithms presented in this thesis study are adapted from the thesis study conducted by Erdem (2007). However, certain structural modifications have been implemented. A comprehensive preliminary study has been carried out to decide on which framework and which programming language will be used for coding the proposed algorithm (see Oztas and Erdem, 2021). According to the findings of this study, management scientists in social sciences tend to prefer easy-to-use platforms which are approximate to human language. For this reason, we preferred to code the RSAB and REF algorithms in Python language on the PyCharm framework. The developed algorithms have been coded by the author under object-oriented programming and class structure to provide more readable and systematic codes. Moreover, code optimization has been applied numerous times to the code prepared within the scope of the study and improvements have been achieved by getting rid of the existing bottlenecks.

RSAB algorithm can be thought of as a procedure for banning regions in the search space where there is no global solution. It has been developed to provide adaptive initial solutions by reducing the diversity of randomness in the initialization procedure. The outstanding feature of this algorithm is the ability to escape from eliminating accidentally global optimum in multi-modal problems unlike in straightforward methods. Moreover, this initialization procedure does not add additional burden to existing solution methods, and on the contrary, it provides a contribution to problem-solving in terms of time and efficiency. For this initialization algorithm, the "Update Intervals" procedure has been newly developed as a modification to obtain robust narrowed intervals. This module includes two sub-modules that are running in case of the improvements in solutions for unconstrained/bounded and constrained problems. This algorithm runs for the defined number of iterations. The final updated intervals are determined by considering all generated intervals for having robust, stable solutions. For that reason, mean-mode-

median values of lower and upper limits are calculated. In the end, the minimum of lower limits and the maximum of upper limits will be the final updated intervals for the related problem. In this way, the updated domains as the output of RSAB will be the search space as input in the REF algorithm.

REF Algorithm has been structured on the principle that the forces between a particle and its neighbors make the particle moved to a new location where a better solution may exist. The repulsive structure of the particles and the movements can be considered as the mimics of Coulomb's Law and Momentum Law respectively. Furthermore, Tabu Search Algorithm and Elitism selection approach have inspired the memory usage of the main algorithm. Moreover, the REF algorithm includes the MUPE approach which is developed for constraint handling by considering satisfaction rates, the total deviations of constraints, and objective function in a multiplicative manner.

REF algorithm comprises “Determine Intervals”, “Initialization”, “Neighborhood”, “Displacement”, and “Duplication” steps. Modifications have been applied to each step except the "Neighborhood". “Determine Intervals” and “Initialization” are the common steps with RSAB which are executed only once when operating as a hybrid algorithm. Modifications have been applied to each procedure within the scope of this thesis study except the neighborhood procedure.

Some modifications have been applied to increase the efficiency of the algorithm by considering unified (net) forces. Another prominent change in the proposed algorithms is the duplication module. The duplication has been newly developed in this thesis as a control mechanism for duplications in population to meet Pauli’s Exclusion principle. However, a degree of precision is important to determine if the particles are in the same location. In our experiments, locations are assumed as duplication in case of first three digits are the same. In such cases, different displacement procedures are applied to determine the new position of the same particle heuristically. While duplication check provides diversification, repositioning in a wide range can create too much diversity which disrupts the balance. For this reason, making the repositioning around the best-known solution has an intensification purpose to balance the exploration-exploitation ability of the REF algorithm.

Furthermore, a database structure has been created to reduce the function evaluation load in the algorithm. Memory capacity has been implemented to the REF algorithm for two purposes. This database can be thought of as a memory capability. The first one is the creation of a database. Each particle is recorded in a memory along with the evaluation scores (constraint satisfied rates, total deviations). It is worth mentioning that, in the REF algorithm, memory is used not as a banned list as in Tabu Search, but to get rid of unnecessary repetitive function evaluations. The second one is about recording the best-so-far positions of the particles. In the REF algorithm, best-so-far solutions of every particle are stored in order not to lose them in case of displacements. However, differently from the original elitism principle, there is no certain ratio for the elitist particles. Namely, the best-so-far particles are kept separately from the relocated particle set.

Limitations

Some limitations of this thesis study should be noted. Although numerous code optimization was conducted, the latest version of the algorithm consists of operations that contain several for loops which is time-consuming because of its working principles and the nature of object-oriented programming. The performance of the developed algorithms is limited by the tested benchmarks. Moreover, the performance of the RSAB-REF is limited for large-dimensional problems, since the objective function value is the single control mechanism in unconstrained problems. In addition, since MUPE, which is the prominent capability of the algorithm, developed for constraint handling, is a penalty-based approach, determining importance scores by trial and error can be considered as a disadvantage. Apart from this, hardware qualifications such as central processing unit (CPU), computer data storage, the motherboard can also be considered as the constraints of this thesis.

Further Studies

Suggestions for future studies can be examined based on the proposed algorithm, programming language, and alternative computing approaches. There may

be structural changes that can be implemented to the algorithm. The problem range solved by the algorithm can be expanded. Modifications can be made to increase the performances for large-dimensional unconstrained problems and to achieve the desired result with fewer function evaluation values. Also, the developed approaches provided in the modules can be integrated with different algorithms and performance comparison can be achieved. Moreover, it can be integrated with various algorithms as a different hybrid algorithm. Within the scope of this thesis, the python codes of the algorithm will be made available to those concerned from platforms such as GitHub as open source. Thereafter, it is also planned to develop a user interface for the proposed algorithms. Consequently, researchers will be provided with an optimization tool that can apply the RSAB-REF algorithm easily.

Improvements can be applied to the programming language. It is always addressed that NP-Hard problems cannot be solved in polynomial time. However, another issue that needs to be addressed is that iterative methods should be developed with fewer loops and provide solutions in polynomial time. For this reason, it will be beneficial to develop programming languages suitable for functional programming using matrix-based associated memory.

Challenges in the field of optimization facing today comprise solving NP-Hard problems, which entails finding the best solution out of an enormous number of trials regardless of algorithms and even programming languages. Therefore, technologies may be inadequate in increasing the size of the problems that can be handled and in reducing the amount of time required to find a solution at the same time. Although parallel programming offers multiple solutions simultaneously which achieve results much faster by dividing the required transaction volume and time by the processors, even supercomputers may become impractical because of the complexity. With the latest technology of quantum computing, tools are being developed for researchers to work beyond classical capabilities. IBM Quantum, Google Quantum AI are the main examples of the technology which leads the world in quantum computing to solve complex problems the world's most powerful supercomputers cannot solve. Furthermore, specially designed Ising machines using a mathematical model have been developed for cases where even supercomputers cannot handle. The latest example of this technology is the Simulated Bifurcation machine developed by

Toshiba for large-scale, complex combinatorial optimization problems in a short time (Toshiba Corporation, 2019). According to Tatsumura et al. (2019), Simulated Bifurcation provides an opportunity to solve complex combinatorial problems very fast by massively parallel processing. According to Toshiba, this speed is 1000 times faster than when using standard optimized simulated annealing software. Moreover, Toshiba also considers developing general-purpose – to handle other problem types- Simulated Bifurcation Ising Machines as well. (Boyd, 2020). Therefore, these recent developments show us that there will be breakthroughs for optimization problems in the near future.

REFERENCES

Abasi, A. K., Khader, A. T., Al-Betar, M. A., Naim, S., Makhadmeh, S. N. and Alyasseri, Z. A. A. (2020). Link-Based Multi-Verse Optimizer for Text Documents Clustering. *Applied Soft Computing Journal*. 87, 106002. 1-24.

Abdechiri, M., Meybodi, M. R. and Bahrami, H. (2013). Gases Brownian Motion Optimization: an Algorithm for Optimization (GBMO). *Applied Soft Computing*. *13*(5): 2932–2946.

Abdel-Basset, M., Abdel-Fatah, L. and Sangaiah, A. K. (2018). Metaheuristic Algorithms: A Comprehensive Review. *Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications* (pp. 185–231). Elsevier.

Abdel-Basset, M., El-Shahat, D., Faris, H. and Mirjalili, S. (2019). A Binary Multi-Verse Optimizer for 0-1 Multidimensional Knapsack Problems with Application in Interactive Multimedia Systems. *Computers and Industrial Engineering*. *132*: 187–206.

Abdel-Raoufi, O., Abdel-Baset, M. and El-henawy, I. (2014). A New Hybrid Flower Pollination Algorithm for Solving Constrained Global Optimization Problems. *International Journal of Applied Operational Research*. *4*(2): 1–13.

Abdullah, S., Turabieh, H., Mccollum, B. and Mcmullan, P. (2012). A hybrid metaheuristic approach to the university course timetabling problem. *Journal of Heuristics*. 18: 1–23.

Abedinia, O., Amjady, N. and Ghasemi, A. (2014). A New Metaheuristic Algorithm Based on Shark Smell Optimization. *Complexity*. *21*(5): 97–116.

Abedinpourshotorban, H., Mariyam Shamsuddin, S., Beheshti, Z. and Jawawi, D. N. A. (2016). Electromagnetic Field Optimization: A Physics-Inspired Metaheuristic

Optimization Algorithm. *Swarm and Evolutionary Computation*. *26*: 8–22.

Abuhamdah, A. and Ayob, M. (2009). Multi-Neighbourhood Particle Collision Algorithm for Solving Course Timetabling Problems. *2009 2nd Conference on Data Mining and Optimization,* (pp. 21–27). IEEE.

Abuhamdah, A. and Ayob, M. (2011). MPCA-ARDA For Solving Course Timetabling Problems. *Conference on Data Mining and Optimization* (pp. 171–177). IEEE.

Agnihotri, S., Atre, A. and Verma, H. K. (2020). Equilibrium Optimizer for Solving Economic Dispatch Problem. *PIICON 2020 - 9th IEEE Power India International Conference* (pp. 1-5). IEEE.

Ahmadi-Javid, A. (2011). Anarchic Society Optimization: A Human-Inspired Method. *2011 IEEE Congress of Evolutionary Computation (CEC)* (pp. 2586–2592). IEEE.

Akbari, M., Molla-Alizadeh-Zavardehi, S. and Niroomand, S. (2020). Meta-Heuristic Approaches for Fixed-Charge Solid Transportation Problem in Two-Stage Supply Chain Network. *Operational Research*. *20*(1): 447–471.

Al-Madi, N., Faris, H., and Mirjalili, S. (2019). Binary Multi-Verse Optimization Algorithm for Global Optimization And Discrete Problems. *International Journal of Machine Learning and Cybernetics*. *10*(12): 3445–3465.

Alatas, B., and Bingol, H. (2019). A Physics-Based Novel Approach for Travelling Tournament Problem: Optics Inspired Optimization. *Information Technology and Control*. *48*(3): 373–388.

Alatas, B. (2011). Uniform Big Bang–Chaotic Big Crunch Optimization. *Communications in Nonlinear Science and Numerical Simulation, 16(9):* 3696–3703.

Alizadeh, N., and Kashan, A. H. (2019). Enhanced Grouping League Championship

And Optics Inspired Optimization Algorithms for Scheduling A Batch Processing Machine with Job Conflicts and Non-Identical Job Sizes. *Applied Soft Computing Journal*. *83*: 105657.

Aljohani, T. M., Ebrahim, A. F. and Mohammad, O. (2019). Single and Multiobjective Optimal Reactive Power Dispatch Based on Hybrid Artificial Physics – Particle Swarm Optimization. *Energies*. *12*(2333): 1–24.

Alnahari, E., Shi, H. and Alkebsi, K. (2020). Quadratic Interpolation Based Simultaneous Heat Transfer Search Algorithm and Its Application to Chemical Dynamic System Optimization. *Processes*: *8*(4): 478.

Ara, A. L., Tolabi, H. B. and Hosseini, R. (2016). Dynamic Modeling and Controller Design of Distribution Static Compensator in A Microgrid Based On Combination Of Fuzzy Set and Galaxy-Based Search Algorithm. *International Journal of Engineering-Transactions A: Basics.* 29(10): 1392-1400.

Aragon, V. S., Esquivel, S. C. and Coello, C. A. C. (2010). A Modified Version of a T-Cell Algorithm for Constrained Optimization Problems. *International Journal For Numerical Methods In Engineering*. *84*: 351–378.

Arora, J. (2017). *Introduction to Optimum Design*. Elsevier.

Arora, S. and Singh, S. (2019). Butterfly Optimization Algorithm: A Novel Approach for Global Optimization. *Soft Computing*. *23*(3): 715–734.

Aryan, M. and Alizadeh, B. A. M. (2016). Bayesian Charged System Search: A Hybrid Method for Multi-Modal Optimization Problems. *2016 IEEE Congress on Evolutionary Computation* (pp. 1501–1508).

Asha, G.R. (2020). An Efficient Clustering and Routing Algorithm for Wireless Sensor Networks Using GSO and KGMO Techniques. In Smart Computing

Paradigms: New Progresses and Challenges (pp. 75-85). Singapore: Springer.

Ashrafi, S. M. and Dariane, A. B. (2011). A Novel and Effective Algorithm for Numerical Optimization: Melody Search (MS). *2011 11th International Conference on Hybrid Intelligent Systems (HIS)* (pp. 109–114). IEEE.

Asi, M. J. and Dib, N. I. (2010). Design of Multilayer Microwave Broadband Absorbers Using Central Force Optimization. *Progress In Electromagnetics Research B*. *26*: 101–113.

Askarzadeh, A. (2016). A Novel Metaheuristic Method for Solving Constrained Engineering Optimization Problems: Crow Search Algorithm. *Computers & Structures*. 169: 1–12.

Asl, R. M., Pourabdollah, E. and Salmani, M. (2018). Optimal Fractional Order PID for A Robotic Manipulator Using Colliding Bodies Design. *Soft Computing*. *22*(14): 4647–4659.

Aslani, H., Yaghoobi, M. and Akbarzadeh-T, M.R. (2016). Chaotic inertia weight in black hole algorithm for function optimization. *2nd International Congress on Technology, Communication and Knowledge, ICTCK 2015* (pp. 123–129). IEEE.

Astolfi, A. (2006). Optimization: An Introduction. https://pdf4pro.com/view/optimization-an-introduction-imperial-college-2d633c.html, (18.02.2019).

Atashpaz-Gargari, E. and Lucas, C. (2007). Imperialist Competitive Algorithm: An Algorithm For Optimization Inspired by Imperialistic Competition. *2007 IEEE Congress on Evolutionary Computation* (pp. 4661–4667). IEEE.

Ates, A. and Yeroglu, C. (2018). Modified Artificial Physics Optimization for Multi-parameter Functions. *Iranian Journal of Science and Technology - Transactions of*

Electrical Engineering. *42*(4): 465–478.

Aziz, M. and Tayarani-N, M.-H. (2016). Opposition-Based Magnetic Optimization Algorithm with Parameter Adaptation Strategy. *Swarm and Evolutionary Computation*. 26: 97–119.

Basu, M. (2016). Kinetic Gas Molecule Optimization for Nonconvex Economic Dispatch Problem. *International Journal of Electrical Power and Energy Systems*, 80: 325–332.

Beheshti, Z. and Shamsuddin, S. M. H. (2013). A Review of Population-Based Meta-Heuristic Algorithm. *International Journal of Advances in Soft Computing and Its Applications*. 5(1): 1-35.

Beirami, A., Vahidinasab, V., Shafie-khah, M. and Catalão, J.P.S. (2020). Multiobjective Ray Optimization Algorithm as a Solution Strategy for Solving Non-Convex Problems: A Power Generation Scheduling Case Study. *International Journal of Electrical Power and Energy Systems*. 119: 105967.

Beiranvand, H. and Rokrok, E. (2015). General Relativity Search Algorithm: A Global Optimization Approach. *International Journal of Computational Intelligence and Applications*. 14(3): 1550017.

Benasla, L., Belmadani, A. and Rahli, M. (2014). Spiral Optimization Algorithm for solving Combined Economic and Emission Dispatch. *International Journal of Electrical Power and Energy Systems*. 62: 163–174.

Bentouati, B., Chettih, S., Jangir, P. and Trivedi, I. N. (2016). A Solution to The Optimal Power Flow Using Multi-Verse Optimizer. *Journal of Electrical Systems*. 12(4): 716–733.

Bijari, K., Zare, H., Veisi, H., and Bobarshad, H. (2018). Memory-Enriched Big Bang–

Big Crunch Optimization Algorithm for Data Clustering. *Neural Computing and Applications*. 29: 111–121.

Birbil, Ş. I. and Fang, S. C. (2003). An Electromagnetism-Like Mechanism for Global Optimization. *Journal of Global Optimization*. 25: 263–282.

Birbil, Ş. I., Fang, S. C. and Sheu, R.-L. (2004). On the Convergence of a Population-Based Global Optimization Algorithm. *Journal of Global Optimization*. 301–318.

Biswas, A., Mishra, K.K., Tiwari, S. and Misra, A.K. (2013). Physics-Inspired Optimization Algorithms: A Survey. *Journal Of Optimization*. 2013: 438152.

Blum, C. and Roli, A. (2003). Metaheuristics in Combinatorial Optimization: Overview And Conceptual Comparison. *ACM Computing Surveys (CSUR)*. 35(3): 268–308.

Boschetti, M. A., Maniezzo, V., Roffilli, M. and Bolufé Röhler, A. (2009). Matheuristics: Optimization, Simulation and Control. *Hybrid Metaheuristics* (pp. 171–177). Editors M.J. Blesa, C. Blum, L. Di Gaspero, A. Roli, M. Sampels and A. Schaerf, Berlin, Heidelberg: Springer.

Bouaziz, A., Draa, A. and Chikhi, S. (2013). A Quantum-Inspired Artificial Bee Colony Algorithm for Numerical Optimisation. *11th International Symposium on Programming and Systems (ISPS)* (pp. 81–88). IEEE.

Bouchekara, H. (2020). Solution of The Optimal Power Flow Problem Considering Security Constraints Using An Improved Chaotic Electromagnetic Field Optimization Algorithm. *Neural Computing and Applications*. 32(7): 2683–2703.

Bouchekara, H. R. E. H., Chaib, A. E., Abido, M. A. and El-Sehiemy, R. A. (2016). Optimal power flow using an Improved Colliding Bodies Optimization algorithm. *Applied Soft Computing Journal*. 42: 119–131.

Bouchekara, H. R. E. H., Zellagui, M. and Abido, M. A. (2017). Optimal Coordination of Directional Overcurrent Relays Using a Modified Electromagnetic Field Optimization Algorithm. *Applied Soft Computing Journal*. 54: 267–283.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

Boyd J. (2020). *Toshiba's Optimization Algorithm Sets Speed Record for Solving Combinatorial Problems*, https://spectrum.ieee.org/tech-talk/computing/software/toshiba--optimization-algorithm-speed-record-combinatorial-problems?s=08, (25.04.2021).

Bozorg-Haddad, O., Solgi, M. and Loáiciga, H. A. (2017). *Meta-Heuristic and Evolutionary Algorithms for Engineering Optimization*. John Wiley & Sons.

Brownlee, J. (2011). *Clever Algorithms: Nature-Inspired Programming Recipes*. Australia: Jason Brownlee.

Buch, H. and Trivedi, I.N. (2020). A New Non-Dominated Sorting Ions Motion Algorithm: Development and Applications. *Decision Science Letters*. 9(1): 59–76.

Busch, P., Heinonen, T. and Lahti, P. (2007). Heisenberg's Uncertainty Principle. *Physics Reports*. 452(6): 155–176.

Cagnina, L. C., Esquivel, S. C., Nacional, U., Luis, D. S., Luis, S. and Coello, C. A. C. (2008). Solving Engineering Optimization Problems with the Simple Constrained Particle Swarm Optimizer. *Informatica*. 32: 319–326.

Camp, C. V. (2007). Design of Space Trusses Using Big Bang – Big Crunch Optimization. *Journal of Structural Engineering*. 133(7): 999–1008.

Cao, Y., Nikafshan Rad, H., Hamedi Jamali, D., Hashemian, N. and Ghasemi, A.

(2020a). A Novel Multi-Objective Spiral Optimization Algorithm for An Innovative Solar/Biomass-Based Multi-Generation Energy System: 3e Analyses, and Optimization Algorithms Comparison. *Energy Conversion and Management*, 219: 112961.

Cao, W., Liu, X. and Ni, J. (2020b). Parameter Optimization of Support Vector Regression Using Henry Gas Solubility Optimization Algorithm. *IEEE Access*. 8: 88633–88642.

Caserta, M. and Voß, S. (2009). Matheuristics. *Matheuristics: Hybridizing Metaheuristics and Mathematical Programming* (pp. 1–32). Editors V. Maniezzo, T. Stützle, and S. Voß, New York: Springer.

Çengel, Y. A., Boles, M. A. and Kanoğlu, M. (2019). *Thermodynamics An Engineerıng Approach* (Ninth). New York: McGraw Hill.

Chandrasekhar, S. (2003). *Newton's Principia for the Common Reader*. Oxford University Press.

Chang, H. C. and Lin, P. C. (2014). A Demonstration of The Improved Efficiency of The Canonical Coordinates Method Using Nonlinear Combined Heat and Power Economic Dispatch Problems. *Engineering Optimization*. 46(2): 261–269.

Chang, P.-C., Chen, S.-H. and Fan, C.-Y. (2009). A Hybrid Electromagnetism-Like Algorithm for Single Machine Scheduling Problem. *Expert Systems With Applications*. 36(2): 1259–1267.

Chen, D., Lu, R., Li, S., Zou, F. and Liu, Y. (2020). An Enhanced Colliding Bodies Optimization and Its Application. *Artificial Intelligence Review*. 53(2): 1127-1186.

Cheng, J., and Zhao, W. (2020). Chaotic Enhanced Colliding Bodies Optimization Algorithm for Structural Reliability Analysis. Advances in Structural Engineering.

23(3): 438-453.

Chiu, C., Yang, Y. and Chou, Y. (2011). Quantum-Inspired Tabu Search Algorithm for Solving 0 / 1 Knapsack Problems, In *Proceedings of the 13th annual conference companion on Genetic and evolutionary computation* (pp. 55-56).

Chou, J. S. and Truong, D. N. (2021). A Novel Metaheuristic Optimizer Inspired by Behavior of Jellyfish in Ocean. *Applied Mathematics and Computation*. *389*: 125535.

Chou, Y. H., Chen, C. Y., Chiu, C. H. and Chao, H. C. (2012). Classical and Quantum-Inspired Electromagnetism-Like Mechanism And Its Applications. *IET Control Theory and Applications*. 6(10): 1424–1433.

Chu, C. C. and Tsai, M.S. (2013). Application of Novel Charged System Search with Real Number String for Distribution System Loss Minimization. *IEEE Transactions on Power Systems*. 28(4): 3600–3609.

Chu, S.-C., Tsai, P. and Pan, J.-S. (2006). Cat Swarm Optimization. *Trends in Artificial Intelligence* (pp. 854–858). Editors Q. Yang and G. Webb. Berlin, Heidelberg: Springer.

Civicioglu, P. (2013). Artificial Cooperative Search Algorithm for Numerical Optimization Problems. *Information Sciences*. *229*: 58–76.

Coello, C. A. C. (2000). Use of A Self-Adaptive Penalty Approach for Engineering Optimization Problems. *Computers in Industry*. 41(2): 113–127.

Coello, C. A. C. (2002). Theoretical and Numerical Constraint Handling Techniques used with Evolutionary Algorithms: A Survey of the State of the Art. *Computer Methods in Applied Mechanics and Engineering*, 191(11-12):1245-1287.

Collins, J. J. and Eaton, M. (1997). *Genocodes for Genetic Algorithms*.

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.51.2349&rep=rep1&type=pdf, (21.09.2019)

Cook, S. A. (1971). The Complexity of Theorem-Proving Procedures. *Proceedings of the third annual ACM symposium on Theory of computing* (pp. 151–158).

Coulomb, C.A. (1785). Premier Mémoire Sur L'électricité Et Le Magnétisme. Histoire de l'Académie Royale des Sciences. (pp. 569–577) Imprimerie Royale, Paris.

Cuevas, E., Oliva, D., Zaldivar, D., Pérez-Cisneros, M. and Sossa, H. (2012). Circle Detection Using Electro-Magnetism Optimization. *Information Sciences*. 182(1): 40–55.

D'Ambrosio, A., Spiller, D. and Curti, F. (2020). Improved Magnetic Charged System Search Optimization Algorithm with Application to Satellite Formation Flying. *Engineering Applications of Artificial Intelligence*, 89: 103473.

Da Luz, E. F. P., Becceneri, J. C. and De Campos Velho, H. F. (2011). Multiple Particle Collision Algorithm Applied to Radiative Transference and Pollutant Localization Inverse Problems. In *IEEE International Symposium on Parallel and Distributed Processing* (pp. 347–351).

Da Luz, E. F. P., Becceneri, J. C. and Velho, H. F. de C. (2008). A New Multi-Particle Collision Algorithm For Optimization in A High Performance Environment. *Journal of Computational Interdicriplinary Sciences.* 1(1): 3–10.

Deb, K. (1997). GeneAS: A Robust Optimal Design Technique for Mechanical Component Design. *Evolutionary Algorithms in Engineering Applications*. 497–514.

Deb, K. (2000). An Efficient Constraint-handling Method for Genetic Algorithms. *Computer Methods in Applied Mechanics and Engineering*. 186(2-4): 311-338.

Degertekin, S.O., Lamberti, L. and Hayalioglu, M.S. (2017). Heat Transfer Search Algorithm for Sizing Optimization of Truss Structures. *Latin American Journal of Solids and Structures*. 14(3): 373–397.

Dekking, F.M., Kraaikamp, C., Lopuhaä, H. P. and Meester, L. E. (2005). *A Modern Introduction to Probability And Statistics: Understanding Why and How*. Springer Science & Business Media.

Dey, S., Bhattacharyya, S. and Maulik, U. (2016). New Quantum Inspired Meta-Heuristic Techniques for Multi-Level Colour Image Thresholding. *Applied Soft Computing Journal*. 46: 677–702.

Dey, A., Bhattacharyya, S., Dey, S., Platos, J. and Snasel, V. (2019a). Quantum-Inspired Bat Optimization Algorithm for Automatic Clustering of Grayscale Images. *Recent Trends in Signal and Image Processing. Advances in Intelligent Systems and Computing* (pp. 23–30). Editors S. Bhattacharyya, S. Pál, I. Pan, and A. Das, Springer Singapore.

Dey, S., De, S., Ghosh, D., Konar, D., Bhattacharyya, S. and Platos, J. (2019b). A Novel Quantum Inspired Sperm Whale Meta-Heuristic for Image Thresholding. *2019 2nd International Conference on Advanced Computational and Communication Paradigms, ICACCP 2019*. IEEE.

Dey, S., Saha, I., Bhattacharyya, S. and Maulik, U. (2014). Multi-Level Thresholding Using Quantum Inspired Meta-Heuristics. *Knowledge-Based Systems*. 67: 373–400.

Dhadwal, M. K., Jung, S. N. and Kim, C. J. (2014). Advanced Particle Swarm Assisted Genetic Algorithm for Constrained Optimization Problems. *Computational Optimization and Applications*. 58(3): 781–806.

Dhiman, G. and Kumar, V. (2018). Emperor Penguin Optimizer: A Bio-Inspired Algorithm for Engineering Problems. *Knowledge-Based Systems*. 159: 20–50.

Dhiman, G. and Kumar, V. (2019). Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. *Knowledge-Based Systems*. 165: 169–196.

Du, K. L. and Swamy, M. N. S. (2016). *Search and optimization by metaheuristics. Techniques and Algorithms Inspired by Nature.* Birkauser.

Dokeroglu, T., Sevinc, E., Kucukyilmaz, T. and Cosar, A. (2019). A Survey on New Generation Metaheuristic Algorithms. *Computers and Industrial Engineering*, 137: 106040.

Dolatabadi, S. (2018). Weighted vertices optimizer (WVO): A Novel Metaheuristic Optimization Algorithm. *Numerical Algebra, Control and Optimization*. 8(4): 461–479.

Domiciano, M. A. P., Shiguemori, E. H., Vieira Dias, L. A. and da Cunha, A. M. (2018). Particle Collision Algorithm Applied to Automatic Estimation of Digital Elevation Model from Images Captured by UAV. *IEEE Geoscience and Remote Sensing Letters*, *15*(10), 1630–1634.

Dong, M., Wang, N., Cheng, X. and Jiang, C. (2014). Composite Differential Evolution with Modified Oracle Penalty Method for Constrained Optimization Problems. *Mathematical Problems in Engineering*. 2014: 617905.

Dorigo, M. and Di Caro, G. (1999). Ant Colony Optimization: A New Meta-Heuristic. *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)* (pp. 1470–1477). IEEE.

Dorigo, M., Maniezzo, V. and Colorni, A. (1996). The Ant System: Optimization by a Colony of Cooperating Agents IEEE Transaction on Systems. *IEEE Transactions on Systems, Man and Cybernetics-Part B.* 26(1): 1–13.

Duan, H. and Qiao, P. (2014). Pigeon-Inspired Optimization: A New Swarm İntelligence Optimizer for Air Robot Path Planning. *International Journal of Intelligent Computing and Cybernetics*. 7(1): 24–37.

Duarte, A., Laguna, M. and Marti, R. (2018). *Metaheuristics for Business Analytics*. Springer.

Duderstadt, J. J. and Hamilton, L. J. (1976). *Nuclear-Reactor Analysis*. *Nuclear Technology*. Wiley.

Eberhart, R. and Kennedy, J. (1995). A New Optimizer Using Particle Swarm Theory. *Proceedings of the Sixth International Symposium on Micro Machine and Human Science* (pp. 39–43). IEEE.

Eberhart, R.C. and Shi, Y. (1998). Introduction to Computational Intelligence. *WCCI Conference*, Alaska: Anchorage.

Echevarría, L. C., Santiago, O. L., Antônio, H. F. de C. V. and Neto, A. J. da S. (2019). *Fault Diagnosis Inverse Problems: Solution with Metaheuristics*. Springer.

Eesa, A. S., Brifcani, A. A. M. and Orman, Z. (2013). Cuttlefish Algorithm – A Novel Bio-Inspired Optimization Algorithm. *International Journal of Scientific & Engineering Research*. 4(9): 1978–1986.

Ehsan-Maleki, B., Naderi, P. and Beiranvand, H. (2018). A Novel 2-Stage WAPSS Design Method to Improve Inter-Area Mode Damping in Power Systems. *International Transactions on Electrical Energy Systems*. 28(3): 1–17.

Elaziz, M. A., Oliva, D., Ewees, A. A. and Xiong, S. (2019). Multi-Level Thresholding-Based Grey Scale İmage Segmentation Using Multi-Objective Multi-Verse Optimizer. *Expert Systems with Applications*. 125: 112–129.

Elnabawy, A.O., Fateen, S.E.K. and Bonilla-Petriciolet, A. (2014). Phase Stability Analysis and Phase Equilibrium Calculations in Reactive and Nonreactive Systems Using Charged System Search Algorithms. *Industrial and Engineering Chemistry Research*. 53(6): 2382–2395.

Elsheikh, A. H., Shehabeldeen, T. A., Zhou, J., Showaib, E. and Abd Elaziz, M. (2020). Prediction of laser cutting parameters for polymethylmethacrylate sheets using random vector functional link network integrated with equilibrium optimizer. *Journal of Intelligent Manufacturing*. 1-12.

Eltokhey, M. W., Mahmoud, K. R. and Obayya, S. S. A. (2016). Optimised Diffusion Spots ' Locations for Simultaneous Improvement in SNR and Delay Spread. *Photonic Network Communications*. 31: 172–182.

Emami, H. and Derakhshan, F. (2015). Election Algorithm: A New Socio-Politically Inspired Strategy. *AI Communications*. 28(3): 591–603.

Erdem, S. (2007). *Evolutionary Algorithms for the Nonlinear Optimization*. (Unpublished PhD Thesis). İzmir: Dokuz Eylul University Graduate School of Natural and Applied Sciences.

Erol, O. K. and Eksin, I. (2006). A New Optimization Method: Big Bang-Big Crunch. *Advances in Engineering Software*. 37(2): 106–111.

Eskandar, H., Sadollah, A., Bahreininejad, A. and Hamdi, M. (2012). Water Cycle Algorithm - A Novel Metaheuristic Optimization Method for Solving Constrained Engineering Optimization Problems. *Computers and Structures*. 110–111: 151–166.

Esmaeili, M., Zakeri, J. A., Kaveh, A., Bakhtiary, A. and Khayatazad, M. (2015). Designing Granular Layers for Railway Tracks Using Ray Optimization Algorithm. *Scientia Iranica*. 22: 47–58.

Ewees, A. A., El Aziz, M. A. and Hassanien, A. E. (2019). Chaotic Multi-Verse Optimizer-Based Feature Selection. *Neural Computing and Applications*. 31(4): 991–1006.

Faramarzi, A., Heidarinejad, M., Mirjalili, S. and Gandomi, A. H. (2020a). Marine Predators Algorithm: A Nature-Inspired Metaheuristic. *Expert Systems with Applications*. 152: 113377.

Faramarzi, A., Heidarinejad, M., Stephens, B. and Mirjalili, S. (2020b). Equilibrium Optimizer: A Novel Optimization Algorithm. *Knowledge-Based Systems*. 191: 105190.

Faris, H., Aljarah, I. and Mirjalili, S. (2016). Training Feedforward Neural Networks Using Multi-Verse Optimizer for Binary Classification Problems. *Applied Intelligence*. 45(2): 322–332.

Faris, H., Hassonah, M. A., Al-Zoubi, A. M., Mirjalili, S. and Aljarah, I. (2018). A Multi-Verse Optimizer Approach for Feature Selection and Optimizing Svm Parameters Based on A Robust System Architecture. *Neural Computing and Applications*. 30(8): 2355–2369.

Feng, X., Ma, M. and Yu, H. (2016). Crystal Energy Optimization Algorithm. *Computational Intelligence*. 32(2): 284–322.

Feynman, R. P. (1982). Simulating Physics with Computers. *International Journal of Theoretical Physics*. 21(6/7): 467–488.

Fister, I., Yang, X. S., Brest, J. and Fister, D. (2013). A Brief Review of Nature-Inspired Algorithms for Optimization. *Elektrotehniski Vestnik/Electrotechnical Review*. 80(3): 116–122.

Fister Jr, I., Mlakar, U., Brest, J. and Fister, I. (2016). A new population-based nature-

inspired algorithm every month : Is the current era coming to the end ? *Proceedings of the 3rd Student Computer Science Research Conference* (pp. 33–37).

Flores, J. J., López, R. and Barrera, J. (2011). Gravitational Interactions Optimization. *Learning and Intelligent Optimization* (pp. 226–237). Editors C. A. C. Coello. Berlin, Heidelberg: Springer.

Formato, R. A. (2007). Central Force Optimization: A New Metaheuristic with Applications in Applied Electromagnetics. *Progress in Electromagnetics Research*. 77: 425–491.

Formato, R. A. (2010a). Improved CFO Algorithm for Antenna. *Progress In Electromagnetics Research B*. 19: 405–425.

Formato, R. A. (2010b). Parameter-Free Deterministic Global Search with Simplified Central Force Optimization. *Advanced Intelligent Computing Theories and Applications* (pp. 309–318). Berlin, Heidelberg: Springer.

Formato, R. A. (2011). Central Force Optimization with Variable Initial Probes and Adaptive Decision Space. *Applied Mathematics and Computation*. 217(21): 8866–8872.

Formato, R. A. (2013). Pseudorandomness in Central Force Optimization. *British Journal of Mathematics & Computer Science*. 3(3): 241–264.

Gandomi, A.H. (2014). Interior Search Algorithm (Isa): A Novel Approach for Global Optimization. *ISA Transactions*. 53(4): 1168–1183.

Gandomi, A. H., Yang, X. S., Talatahari, S. and Alavi, A. H. (2013a). Metaheuristic Algorithms in Modeling and Optimization, *Metaheuristic Applications in Structures and Infrastructures* (pp. 1-24). Elsevier.

Gandomi, A. H., Yang, X. S. and Alavi, A. H. (2013b). Cuckoo Search Algorithm: A Metaheuristic Approach to Solve Structural Optimization Problems. *Engineering with Computers*. 29(1): 17–35.

Gandomi, A. H., Yang, X. S. and Alavi, A. H. (2011). Mixed Variable Structural Optimization Using Firefly Algorithm. *Computers and Structures*. 89(23–24): 2325–2336.

Gao-Wei, Y. and Zhanju, H. (2012). A Novel Atmosphere Clouds Model Optimization Algorithm. In *2012 International Conference on Computing, Measurement, Control and Sensor Network* (pp. 217–220). IEEE.

Gao, H., Du, Y. and Diao, M. (2017). Quantum-Inspired Glowworm Swarm Optimisation and Its Application. *International Journal of Computing Science and Mathematics*. 922: 91–100.

Gao, Y. J., Zhang, F. M., Zhao, Y. and Li, C. (2019a). A Novel Quantum-Inspired Binary Wolf Pack Algorithm for Difficult Knapsack Problem. *International Journal of Wireless and Mobile Computing*. 16(3): 222–232.

Gao, H. Y., Zhang, X. T., Du, Y. N. and Diao, M. (2019b). Quantum-Inspired Teaching-Learning-Based Optimization for Linear Array Pattern Synthesis. *Communications, Signal Processing, and Systems. CSPS 2017. Lecture Notes in Electrical Engineering* (pp. 2106–2115). Editors Q. Liang, J. Mu, M. Jia, W. Wang, X. Feng, and B. Zhang, Singapore: Springer.

Gao, J., Fang, L. and He, G. (2010). A Quantum-Inspired Artificial Immune System for Multiobjective 0-1 Knapsack Problems. *Lecture Notes in Computer Science* (Vol. 6063, pp. 161–168). Editors L. Zhang, B. L. Lu, and J. Kwok. Berlin, Heidelberg: Springer.

Gao, S., Vairappan, C., Wang, Y., Cao, Q. and Tang, Z. (2014). Gravitational Search

Algorithm Combined with Chaos for Unconstrained Numerical Optimization. *Applied Mathematics And Computation*. 231: 48–62.

Gao, W. (2017). Investigating The Critical Slip Surface of Soil Slope Based on an Improved Black Hole Algorithm. *Soils and Foundations*. 57(6): 988–1001.

García, J., Crawford, B., Soto, R. and García, P. (2017). A Multi Dynamic Binary Black Hole Algorithm Applied to Set Covering Problem. *Harmony Search Algorithm* (Vol. 2) (pp. 42-51). Editor J. Del Ser. Singapore: Springer.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability A Guide to the Theory of NP-Completeness*. Bell Telephone Laboratories, Incorporated.

Garg, A., Arvind, A. and Gadhvi, B. (2017). Optimum Control for the Vehicle Semi-active Suspension System. *Mechatronics and Robotics Engineering for Advanced and Intelligent Manufacturing. Lecture Notes in Mechanical Engineering* (pp. 421–430). Editors D. Zhang and B. Wei. Cham: Springer.

Garg, H. (2014). Solving Structural Engineering Design Optimization Problems Using an Artificial Bee Colony Algorithm. *Journal of Industrial and Management Optimization*. 10(3): 777–794.

Garg, H. (2016). A Hybrid PSO-GA Algorithm for Constrained Optimization Problems. *Applied Mathematics and Computation*. 274: 292–305.

Garg, H. (2019). A Hybrid GSA-GA Algorithm for Constrained Optimization Problems. *Information Sciences*. 478: 499–523.

Geem, Z. W., Kim, J. H. and Loganathan, G. V. (2001). A New Heuristic Optimization Algorithm: Harmony Search. *Simulation*. 76(2): 60–68.

Gendreau, M. and Potvin, J.-Y. (2008). Metaheuristics: A Canadian Perspective.

INFOR: Information Systems and Operational Research. 46(1): 71–80.

Geng, K., Ye, C., Cao, L. and Liu, L. (2019). Multi-Objective Reentrant Hybrid Flowshop Scheduling with Machines Turning on and off Control Strategy Using Improved Multi-Verse Optimizer Algorithm. *Mathematical Problems in Engineering*.

Ghannadi, P. and Kourehli, S.S. (2020). Multiverse Optimizer for Structural Damage Detection: Numerical Study and Experimental Validation. *Structural Design of Tall and Special Buildings*. 29(13): 1–27.

Ghasemian, H., Ghasemian, F. and Vahdat-Nejad, H. (2020). Human Urbanization Algorithm: A Novel Metaheuristic Approach. *Mathematics and Computers in Simulation*. 178: 1–15.

Glover, F. (1989). Tabu Search -Part I. *ORSA Journal on Computing*. 1(3): 190–206.

Glover, Fred. (1990). Tabu Search- Part II. *ORSA Journal on Computing*. 2(1): 4–32.

Glover, Fred. (1999). Scatter Search and Path Relinking. *New Ideas in Optimization* (pp. 297–316). Editors D. Corne, M. Dorigo, and F. Glover. McGraw Hill.

Goldreich, O. (2008). *Computatinal Complexity A Conceptual Perspective*. Cambridge University Press.

Gonçalves, B. and Boettcher, S. (2008). Hysteretic Optimization for Spin Glasses. *Journal of Statistical Mechanics: Theory and Experiment*.

Gonzalez, T.F. (Ed.). (2007). Introduction, Overview, and Notation, *Handbook of Approximation Algorithms and Metaheuristics* (pp. 1-17). Chapman & Hall/CRC.

Google Quantum AI, https://quantumai.google, (3.05.2021).

Guo, T., Henwood, M. I. and Van Ooijen, M. (1996). An Algorithm for Combined Heat and Power Economic Dispatch. *IEEE Transactions on Power Systems*. 11(4): 1778–1784.

Guo, Y. N. and Zhang, P. (2015). Multi-objective Quantum-Inspired Cultural Algorithm. In *Proceedings - 2015 2nd International Conference on Soft Computing and Machine Intelligence, ISCMI 2015* (pp. 25–29). IEEE.

Gupta, H., Gupta, A., Gupta, S. K., Nayak, P. and Shrivastava, T. (2016). How effective is Black Hole Algorithm? *Proceedings of the 2016 2nd International Conference on Contemporary Computing and Informatics, IC3I 2016* (pp. 474–478). IEEE.

Gupta, M., Kansal, M., Thyagarajan, S., Chauhan, P. S. and Upadhyay, D. K. (2020a). Design of an Optimal Microstrip Butterworth Low-Pass Filter Using Colliding Bodies Optimization. *Advances in VLSI, Communication, and Signal Processing* (pp. 125–131). Editors D. Dutta, H. Kar, C. Kumar, and V. Bhadauria. Singapore: Springer.

Gupta, S., Deep, K. and Mirjalili, S. (2020b). An Efficient Equilibrium Optimizer with Mutation Strategy for Numerical Optimization. *Applied Soft Computing Journal*. 96: 106542.

Haghighi, A. and Ramos, H. M. (2012). Detection of Leakage Freshwater and Friction Factor Calibration in Drinking Networks Using Central Force Optimization. *Water Resources Management*. 26(8): 2347–2363.

Han, K. H. and Kim, J. H. (2002). Quantum-Inspired Evolutionary Algorithm for a Class of Combinatorial Optimization. *IEEE Transactions on Evolutionary Computation*. 6(6): 580–593.

Han, X., Li, D., Liu, P. and Wang, L. (2020). Feature Selection by Recursive Binary Gravitational Search Algorithm Optimization for Cancer Classification. *Soft*

Computing. 24(6): 4407–4425.

Hancock, H. (1917). *Theory of maxima and minima*. Boston: Ginn.

Hasan, Z. and El-Hawary, M. E. (2014). Optimal power flow by black hole optimization algorithm. *Proceedings - 2014 Electrical Power and Energy Conference, EPEC 2014* (pp. 134–141). IEEE.

Hasançebi, O. and Azad, S. K. (2012). An exponential big bang-big crunch algorithm for discrete design optimization of steel frames. *Computers and Structures*. 111: 167–179.

Hashim, F. A., Houssein, E. H., Hussain, K., Mabrouk, M. S. and Al-Atabany, W. (2020). A Modified Henry Gas Solubility Optimization for Solving Motif Discovery Problem. *Neural Computing and Applications*. 32(14): 10759–10771.

Hashim, F. A., Houssein, E. H., Mabrouk, M. S., Al-Atabany, W. and Mirjalili, S. (2019). Henry Gas Solubility Optimization: A Novel Physics-Based Algorithm. *Future Generation Computer Systems*. 101: 646–667.

Hatamlou, A. (2013). Black Hole: A New Heuristic Optimization Approach for Data Clustering. *Information Sciences*. 222: 175–184.

Hatamlou, A. (2018). Solving Travelling Salesman Problem Using Black Hole Algorithm. *Soft Computing*. 22(24): 8167–8175.

Hatamlou, A., Abdullah, S. and Hatamlou, M. (2011). Data Clustering Using Big Bang–Big Crunch Algorithm. *Communications in Computer and Information Science* (pp. 383–388). Editors P. Pichappan, H. Ahmadi, and E. Ariwa. Berlin, Heidelberg: Springer.

Hazra, A., Das, S. and Basu, M. (2018). Heat Transfer Search Algorithm for Non-

convex Economic Dispatch Problems. *Journal of The Institution of Engineers (India): Series B*. 99(3): 273–280.

He, Q. and Wang, L. (2007). An Effective Co-Evolutionary Particle Swarm Optimization for Constrained Engineering Design Problems. *Engineering Applications of Artificial Intelligence*. 20(1): 89–99.

Hedman, S. (2006). *A First Course in Logic An introduction to model theory, proof theory, computability, and complexity*. Oxford University Press.

Hemachandra Reddy, K., Ram Kishore Kumar Reddy, P. and Ganesh, V. (2019). Optimal Allocation of Multiple Facts Devices with Hybrid Techniques for Improving Voltage Stability. *International Journal on Emerging Technologies*. 10(4): 76–84.

Hendrix, E.M.T. and G.-Toth, B. (2010). *Introduction to Nonlinear and Global Optimization*. Springer.

Ho, Y.-C. and Pepyne, D. L. (2002). Simple Explanation of The No Free Lunch Theorem of Optimization. *Cybernetics and Systems Analysis*. 38(2): 4409–4414.

Holland, J.H. (1962). Outline for a Logical Theory of Adaptive Systems. *Journal of the ACM*.

Holzner, S. (2011). *Physics I For Dummies* (Second). Wiley.

Hosseini, E. (2017). Laying Chicken Algorithm: A New Meta-Heuristic Approach to Solve Continuous Programming Problems. *Journal of Applied & Computational Mathematics*. 06(1): 1–8.

Hosseini, S. S. S., Jafarnejad, A., Behrooz, A. H. and Gandomi, A. H. (2011). Combined heat and power economic dispatch by mesh adaptive direct search algorithm. *Expert Systems with Applications*. 38(6): 6556–6564.

Hu, X., Huang, L., Wang, Y. and Pang, W. (2019). Explosion Gravitation Field Algorithm with Dust Sampling for Unconstrained Optimization. *Applied Soft Computing Journal*. 81: 105500.

Huan, T.T., Kulkarni, A.J., Kanesan, J., Huang, C.J. and Abraham, A. (2017). Ideology Algorithm: A Socio-Inspired Optimization Methodology. *Neural Computing and Applications*. 28(1): 845-876.

Huang, G.Q. (2017). Artificial Memory Optimization. *Applied Soft Computing Journal*. 61: 497–526.

Huang, K. W. and Wu, Z. X. (2019). CPO: A Crow Particle Optimization Algorithm. *International Journal of Computational Intelligence Systems*. 12(1): 426–435.

Huang, L. and Qin, C. (2019). A Novel Modified Gravitational Search Algorithm for The Real World Optimization Problem. *International Journal of Machine Learning and Cybernetics*. 10(11): 2993–3002.

Huang, S. and Zhao, G. (2012). A Comparison Between Quantum Inspired Bacterial Foraging Algorithm and GA-Like Algorithm for Global Optimization. *International Journal of Computational Intelligence and Applications*. 11(3): 1–20.

Hunt, J. E. and Cooke, D. E. (1996). Learning using an artificial immune system. *Journal of network and computer applications*, 19(2), 189-212.

Hussain, K., Najib, M., Salleh, M., Cheng, S. and Shi, Y. (2019). Metaheuristic Research : A Comprehensive Survey. *Artificial Intelligence Review*. 52(4): 2191–2233.

IBM Quantum Computing, https://www.ibm.com/quantum-computing/quantum-computing-at-ibm, (3.05.2021).

Ibrahim, Z., Aziz, N.H.A., Aziz, N.A.A., Razali, S. and Mohamad, M. S. (2016). Simulated Kalman Filter: A Novel Estimation-Based Metaheuristic Optimization Algorithm. *Advanced Science Letters*. 22(10): 2941–2946.

Ieng, S., Akil, Y. S. and Gunadin, I. C. (2019). Hydrothermal Economic Dispatch Using Hybrid Big Bang-Big Crunch Algorithm. *Journal of Physics: Conference Series*. 8–14.

Jabbary, A., Podeh, H. T., Younesi, H. and Haghiabi, A. H. (2018). Water Distribution Network Optimisation Using a Modified Central Force Optimisation Method. *Proceedings of the Institution of Civil Engineers - Water Management* (pp. 136–12). Thomas Telford Ltd.

Jalili, S. and Kashan, A.H. (2019). An Optics Inspired Optimization Method for Optimal Design of Truss Structures. *Structural Design of Tall and Special Buildings*. 28(6): 1–23.

Jamil, M. and Yang, X. S. (2013). A Literature Survey of Benchmark Functions for Global Optimisation Problems. *International Journal of Mathematical Modelling and Numerical Optimisation*. 4(2): 150–194.

Jamili, A., Shafia, M.A. and Tavakkoli-Moghaddam, R. (2011). A Hybridization of Simulated Annealing and Electromagnetism-Like Mechanism for A Periodic Job Shop Scheduling Problem. *Expert Systems with Applications*. 38(5): 5895-5901.

Jaradat, G. M. and Ayob, M. (2010). Big Bang-Big Crunch Optimization Algorithm to Solve the Course Timetabling Problem. *Proceedings of the 2010 10th International Conference on Intelligent Systems Design and Applications* (pp. 1448–1452). IEEE.

Javadi, S. H. and Zahiri, S. H. (2018). A Central Force Optimization Approach for Data Clustering. *IETE Journal of Research*. 1-9.

Javadian, N., Alikhani, M. G. and Tavakkoli-moghaddam, R. (2008). A Discrete Binary Version of the Electromagnetism-Like Heuristic for Solving Traveling Salesman Problem, *International Conference on Intelligent Computing* (pp. 123-130). Springer, Berlin, Heidelberg.

Javidy, B., Hatamlou, A. and Mirjalili, S. (2015). Ions Motion Algorithm for Solving Optimization Problems. *Applied Soft Computing Journal*. 32: 72–79.

Jeet, K., Dhir, R. and Singh, P. (2016). Hybrid Black Hole Algorithm for Bi-Criteria Job Scheduling on Parallel Machines. *International Journal of Intelligent Systems and Applications*. 8(4): 1–17.

Jensen, J. L.W.V. (1906). Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta Mathematica*. 30: 175–193.

Jhang, J.-Y. and Lee, K.-C. (2009). Array Pattern Optimization Using Electromagnetism-Like Algorithm. *International Journal of Electronics and Communications*. 63: 491–496.

Jin, G. G. and Tran, T. Do. (2010). A Nature-Inspired Evolutionary Algorithm Based on Spiral Movements. *Proceedings of the SICE Annual Conference* (pp. 1643–1647). IEEE.

Jolai, F., Tavakkoli-Moghaddam, R., Golmohammadi, A. and Javadi, B. (2012). An Electromagnetism-Like Algorithm for Cell Formation and Layout Problem. *Expert Systems With Applications*. 39(2): 2172–2182.

Kang, Y., Wang, C., Li, H. and Dai, L. (2016). A Job Shop Scheduling Algorithm Using Big Bang-Big Crunch Strategy. *2016 12th International Conference on Computational Intelligence and Security* (pp. 411–414). IEEE.

Kantorovich, L.V. (1939). Mathematical Methods of Organizing and Planning

Production. *Management Science*. 6(4): 366–422.

Kar, A. K. (2016). Bio Inspired Computing - A Review Of Algorithms and Scope of Applications. *Expert Systems with Applications*. 59: 20–32.

Karaboga, D. and Basturk, B. (2007). A Powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm. *Journal of Global Optimization*. 39: 459–471.

Karmakar, S., Dey, A. and Saha, I. (2018). Use of Quantum-Inspired Metaheuristics During Last Two Decades. *Proceedings - 7th International Conference on Communication Systems and Network Technologies, CSNT 2017* (pp. 272–278). IEEE.

Karp, R. M. (1972). Complexity of Computer Computations. *Complexity of Computer Computations* (pp. 85–103). Editors R. E. Miller and J. W. Thatcher, Plenum Press.

Kashan, A. H. (2009). League Championship Algorithm: A New Algorithm for Numerical Function Optimization. In *2009 International Conference of Soft Computing and Pattern Recognition* (pp. 43–48). IEEE.

Kashan, A. H. (2015a). A New Metaheuristic For Optimization : Optics Inspired Optimization. *Computers & Operations Research*. *55*: 99–125.

Kashan, A. H. (2015b). An Effective Algorithm for Constrained Optimization Based On Optics Inspired Optimization. *Computer-Aided Design*. 63: 52–71.

Kaveh, A. (2017). *Applications of metaheuristic optimization algorithms in civil engineering*. Switzerland: Springer International Publishing.

Kaveh, A. and Ahmadi, B. (2013). Simultaneous Analysis, Design and Optimization of Structures Using The Force Method and Supervised Charged System Search Algorithm. *Scientia Iranica*. 20(1): 65–76.

Kaveh, A. and Behnam, A. F. (2013). Charged System Search Algorithm for the Optimum Cost Design of Reinforced Concrete Cantilever Retaining Walls. *Arabian Journal for Science and Engineering*. 38(3): 563–570.

Kaveh, A. and Bijari, S. (2014). Optimum Cost Design of Reinforced Concrete One-Way Ribbed Slabs Using CBO, PSO and Democratic PSO Algorithms. *Asian Journal of Civil Engineering*. 15(6): 788–802.

Kaveh, A. and Dadras, A. (2017). A Novel Meta-Heuristic Optimization Algorithm: Thermal Exchange Optimization. *Advances in Engineering Software*. 110: 69–84.

Kaveh, A. and Dadras, A. (2018a). Structural Damage Identification Using an Enhanced Thermal Exchange Optimization Algorithm. *Engineering Optimization*. 50(3): 430–451.

Kaveh, A., Dadras, A., and Bakhshpoori, T. (2018b). Improved Thermal Exchange Optimization Algorithm for Optimal Design of Skeletal Structures. *Smart Structures and Systems*. 21(3): 263-278.

Kaveh, A. and Farhoudi, N. (2013). A New Optimization Method: Dolphin Echolocation. *Advances in Engineering Software*. 59: 53–70.

Kaveh, A. and Ghazaan, M. I. (2015). Layout and Size Optimization of Trusses with Natural Frequency Constraints Using İmproved Ray Optimization Algorithm. *Iranian Journal of Science and Technology - Transactions of Civil Engineering*. *39*(C2+): 395–408.

Kaveh, A. and Ghazaan, M.I. (2014). Enhanced Colliding Bodies Optimization for Design Problems with Continuous and Discrete Variables. *Advances in Engineering Software*, *77*, 66–75.

Kaveh, A., Javadi, S. M. and Maniat, M. (2014b). Damage Assessment via Modal Data

with a Mixed Particle Swarm Strategy, Ray Optimizer, and Harmony Search. *Asian Journal of Civil Engineering*. 15(1): 95–106.

Kaveh, A. and Khayatazad, M. (2012). A New Meta-Heuristic Method: Ray Optimization. *Computers and Structures*. 112–113: 283–294.

Kaveh, A. and Laknejadi, K. (2011). A Novel Hybrid Charge System Search and Particle Swarm Optimization Method for Multi-Objective Optimization. *Expert Systems with Applications*. 38(12): 15475–15488.

Kaveh, A. and Mahdavi, V.R. (2014). Colliding Bodies Optimization: A Novel Meta-Heuristic Method. *Computers and Structures.* 139: 18–27.

Kaveh, A. and Mahdavi, V.R. (2016). Colliding Bodies Optimization Extensions and Applications. *Advances in Metaheuristic Algorithms for Optimal Design of Structures* (pp. 11-38). Springer.

Kaveh, A. and Mahjoubi, S. (2019). Hypotrochoid Spiral Optimization Approach for Sizing and Layout Optimization of Truss Structures with Multiple Frequency Constraints. *Engineering with Computers*. 35(4): 1443–1462.

Kaveh, A., Mirzaei, B. and Jafarvand, A. (2015). An Improved Magnetic Charged System Search for Optimization of Truss Structures with Continuous and Discrete Variables. *Applied Soft Computing Journal*. 28: 400–410.

Kaveh, A., Mizaei, B. and Javarvand, A. (2014a). Optimal Design of Double Layer Barrel Vaults Using İmproved Magnetic Charged System Search. *Asian Journal of Civil Engineering*. 15(1): 135–154.

Kaveh, A., Motie Share, M.A. and Moslehi, M. (2013a). Magnetic Charged System Search: A New Meta-Heuristic Algorithm for Optimization. *Acta Mechanica*. 224(1): 85–107.

Kaveh, A., Rastegar Moghaddam, M. and Khanzadi, M. (2018b). Efficient Multi-Objective Optimization Algorithms For Construction Site Layout Problem. *Scientia Iranica*. 25: 2051–2062.

Kaveh, A. and Talatahari, S. (2010). A Novel Heuristic Optimization Method: Charged System Search. *Acta Mechanica*. 213(3–4): 267–289.

Kaveh, A. and Talatahari, S. (2011). A General Model for Meta-Heuristic Algorithms Using The Concept of Fields of Forces. *Acta Mechanica*. 221(1–2): 99–118.

Kaveh, A. and Talatahari, S. (2012). Charged System Search for Optimal Design of Frame Structures. *Applied Soft Computing*. 12(1): 382–393.

Kaveh, A. and Talatahari, S. (2014). Hybrid Charged System Search and Particle Swarm Optimization for Engineering Design Problems. *International Journal for Computer-Aided Engineering and Software*. 28(4): 423–440.

Kaveh, A. and Zolghadr, A. (2015). An Improved CSS for Damage Detection of Truss Structures Using Changes in Natural Frequencies and Mode Shapes. *Advances in Engineering Software*. 80(C): 93–100.

Kaveh, A. and Talatahari, S. (2009). Size Optimization of Space Trusses Using Big Bang – Big Crunch Algorithm. *Computers and Structures*. 87(17–18): 1129–1140.

Kaveh, A. and Talatahari, S. (2010). A Discrete Big Bang - Big Crunch Algorithm for Optimal Design of Skeletal Structures. *Asian Journal of Civil Engineering*. 11(1): 103–122.

Kaveh, A. (2014). *Advances in Metaheuristic Algorithms for Optimal Design of Structures*. Switzerland: Springer International Publishing.

Kaveh, A. and Hosseini, O.K. (2012). A Hybrid HS-CSS Algorithm for Simultaneous

Analysis, Design and Optimization of Trusses via Force Method. *Periodica Polytechnica Civil Engineering*. 56(2): 197–212.

Kaveh, A., Ghazaan, M.I. and Bakhshpoori, T. (2013b). An improved ray optimization algorithm for design of truss structures. *Periodica Polytechnica Civil Engineering*. 57(2): 97–112.

Kaveh, A., Khanzadi, M., Alipour, M. and Naraky, M. R. (2015). CBO and CSS Algorithms for Resource Allocation and Time-Cost Trade-Off. *Periodica Polytechnica Civil Engineering*. 59(3): 361–371.

Kaveh, A., Khanzadi, M., Moghaddam, M. R. and Rezazadeh, M. (2018a). Charged System Search and Magnetic Charged System Search Algorithms for Construction Site Layout Planning Optimization. *Periodica Polytechnica Civil Engineering*. 62(4): 841–850.

Kaveh, A. and Mahdavi, V.R. (2019). Multi-Objective Colliding Bodies Optimization Algorithm For Design Of Trusses. *Journal of Computational Design and Engineering*. 6(1): 49–59.

Kaveh, A. and Maniat, M. (2015). Damage Detection Based on MCSS and PSO Using Modal Data. *Smart Structures and Systems*. 15(5): 1253–1270.

Kaveh, A. and Talatahari, S. (2010). Optimal Design of Skeletal Structures via The Charged System Search Algorithm. *Structural and Multidisciplinary Optimization*. 41(6): 893–911.

Khan, T.A., Ling, S.H. and Mohan, A.S. (2019). Advanced Gravitational Search Algorithm with Modified Exploitation Strategy. In *IEEE International Conference on Systems, Man and Cybernetics* (pp. 1056–1061). IEEE.

Khanzadi, M., Kaveh, A., Alipour, M. and Aghmiuni, H. K. (2016). Application of

CBO and CSS for Resource Allocation and Resource Leveling Problem. *Iranian Journal of Science and Technology - Transactions of Civil Engineering*. 40(1): 1–10.

Kicinger, R., Arciszewski, T., De Jong, K. (2005). Evolutionary computation and structural design: A survey of the state-of-the-art. *Journal of Computers and Structures*, 83: 1943-1978.

Kirkpatrick, C.D., Gelatt, M. and Vecchi, S. (1983). Optimization by Simulated Annealing. *Science*. 220(4598): 671–680.

Kripka, M. and Kripka, R.M.L. (2008). Big Crunch Optimization Method. In *International Conference on Engineering Optimization* (pp. 1-5). Brazil.

Kumar, A., Wu, G., Ali, M. Z., Mallipeddi, R., Suganthan, P. N. and Das, S. (2020b). A Test-Suite of Non-Convex Constrained Optimization Problems from The Real-World and Some Baseline Results. *Swarm and Evolutionary Computation*. 56: 100693.

Kumar, A., Srungavarapu, G., Beiranvand, H. and Rokrok, E. (2017). A novel approach for automatic generation control of multi area power systems with nonlinearity using general relativity search algorithm. *2016 IEEE Annual India Conference.* (pp. 1-6). IEEE.

Kumar, B.S., Suryakalavathi, M. and Kumar, G. V. N. (2016). Optimal Power Flow with Static VAR Compensator Using Galaxy Based Search Algorithm to Minimize Real Power Losses. *Procedia Computer Science* (pp. 42–47).

Kumar, S., Singh, A. and Walia, S. (2018). Parallel Big Bang – Big Crunch Global Optimization Algorithm : Performance and its Applications to routing in WMNs. *Wireless Personal Communications*. 100(4): 1601–1618.

Kumar, S., Tejani, G.G., Pholdee, N. and Bureerat, S. (2020a). Multi-Objective

Modified Heat Transfer Search for Truss Optimization. *Engineering with Computers*. 1-16.

Kumar, Y. and Sahoo, G. (2014). A Charged System Search Approach for Data Clustering. *Progress in Artificial Intelligence*. 2: 249–261.

Kushwaha, N. and Pant, M. (2018). Fuzzy Magnetic Optimization Clustering Algorithm with Its Application to Health Care. *Journal of Ambient Intelligence and Humanized Computing*. 1-10.

Kushwaha, N., Pant, M., Kant, S. and Jain, V. K. (2018). Magnetic Optimization Algorithm for Data Clustering. *Pattern Recognition Letters*. 115: 59–65.

Labbi, Y. and Attous, D. (2010). Big Bang-Big Crunch Optimization Algorithm for Economic Dispatch with Valve-Point Effect. *Journal of Theoretical and Applied Information Technology*. 16.

Labbi, Y. and Attous, D.B. (2017). A Hybrid Big Bang–Big Crunch Optimization Algorithm for Solving The Different Economic Load Dispatch Problems. *International Journal of System Assurance Engineering and Management*. 8(2): 275–286.

Lalwani, P., Banka, H. and Kumar, C. (2017). CRWO: Clustering and Routing in Wireless Sensor Networks Using Optics Inspired Optimization. *Peer-to-Peer Networking and Applications*. 10(3): 453–471.

Lam, A.Y.S. and Li, V.O.K. (2010). Chemical-Reaction-Inspired Metaheuristic for Optimization. *IEEE Transactions on Evolutionary Computation*. 14(3): 381–399.

Langdon W.B., McKay R.I. and Spector L. (2010) Genetic Programming. *Handbook of Metaheuristics. International Series in Operations Research & Management Science*, vol 146. Editors Gendreau M., Potvin JY. Boston, MA: Springer.

Layeb, A. (2011). A Novel Quantum Inspired Cuckoo Search for Knapsack Problems. *International Journal of Bio-Inspired Computation*. 3(5): 297–305.

Layeb, A. (2013). A Hybrid Quantum Inspired Harmony Search Algorithm for 0-1 Optimization Problems. *Journal of Computational and Applied Mathematics*. 253: 14–25.

Le, D. T., Bui, D., Ngo, T. D., Nguyen, Q. and Nguyen-Xuan, H. (2019). A Novel Hybrid Method Combining Electromagnetism-Like Mechanism and Firefly Algorithms for Constrained Design Optimization of Discrete Truss Structures. *Computers and Structures*. 212: 20–42.

Lee, C.-H., Chang, F.-K., Kuo, C.-T. and Chang, H.-H. (2012). A Hybrid of Electromagnetism-Like Mechanism and Back-Propagation Algorithms for Recurrent Neural Fuzzy Systems Design. *International Journal of Systems Science*. 43(2): 231–247.

Lee, C.-H. and Lee, Y.-C. (2012). Nonlinear Systems Design by a Novel Fuzzy Neural System via Hybridization Of Electromagnetism-Like Mechanism and Particle Swarm Optimisation Algorithms. *Information Sciences*. 186: 59–72.

Lee, J.-W., Lee, J.-Y. and Lee, J.-J. (2013). Jenga-Inspired Optimization Algorithm for Energy-Efficient Coverage of Unstructured WSNs. *IEEE Wireless Communications Letters*. 2(1): 34–37.

Li, J., Hu, G., Zhou, Y., Zou, C., Peng, W. and Jahangir Alam, S. M. (2016). A Temperature Compensation Method for Piezo-Resistive Pressure Sensor Utilizing Chaotic Ions Motion Algorithm Optimized Hybrid Kernel LSSVM. *Sensors (Switzerland)*. 16(10): 1707.

Li, S., Chen, H., Wang, M., Heidari, A. A. and Mirjalili, S. (2020). Slime Mould Algorithm: A New Method for Stochastic Optimization. *Future Generation Computer*

Systems. 111: 300–323.

Li, Y., Wang, H. and Chai, Z. (2019). Multi-Objective Optimization of Spectrum Detection in Cognitive IoT Using Artificial Physics Physics. *Journal of the Chinese Institute of Engineers*. 42(3): 219–224.

Li, Y.Y. and Jiao, L. C. (2005). Quantum-Inspired Immune Clonal Algorithm. *Artificial Immune Systems ICARIS 2005. Lecture Notes in Computer Science* (pp. 304–317). Editors C. Jacob, M. L. Pilat, P. J. Bentley, and J.I. Timmis. Berlin, Heidelberg: Springer.

Liu, F., Li, F. and Jing, X. (2019). Navigability Analysis of Local Gravity Map with Projection Pursuit-Based Selection Method by Using Gravitation Field Algorithm. *IEEE Access*. 7: 75873–75889.

Lones, M.A. (2020). Mitigating Metaphors: A Comprehensible Guide to Recent Nature-Inspired Algorithms. *SN Computer Science*. 1(1): 1–12.

Lu, C., Gao, L., Li, X., Hu, C., Yan, X. and Gong, W. (2020). Chaotic-Based Grey Wolf Optimizer for Numerical and Engineering Optimization Problems. *Memetic Computing*. 12(4): 371–398.

Luke, S. (2011). *Essentials of Metaheuristics: A Set of Undergraduate Lecture Notes*. *Optimization*. http://cs.gmu.edu/~sean/book/metaheuristics/, (04.05.2020).

Maier, H. R., Razavi, S., Kapelan, Z., Matott, L. S., Kasprzyk, J. and Tolson, B. A. (2019). Introductory Overview: Optimization Using Evolutionary Algorithms and Other Metaheuristics. *Environmental modelling & software*, 114, 195-213.

Mahdavi, A. and Ghaffari, A. (2019). Embedding Virtual Machines in Cloud Computing Based on Big Bang-Big Crunch Algorithm. *Journal of Information Systems and Telecommunication*. 7(4): 305–315.

Mahdavi, M., Fesanghary, M. and Damangir, E. (2007). An Improved Harmony Search Algorithm for Solving Optimization Problems. *Applied Mathematics and Computation*. 188(2): 1567–1579.

Mahmood, M. and Al-Khateeb, B. (2019). The Blue Monkey: A New Nature Inspired Metaheuristic Optimization Algorithm. *Periodicals of Engineering and Natural Sciences*. 7(3): 1054–1066.

Man, L., Li, S., Wang, X., Yu, Y. and Lu, S. (2014). A Novel Method of Image Segmentation Based On PCNN with Spiral Optimization. *International Conference on Signal Processing Proceedings* (pp. 703–708).

Manju, A. and Nigam, M. J. (2012). Firefly Algorithm with Fireflies Having Quantum Behavior. *2012 International Conference on Radar, Communication and Computing, ICRCC 2012* (pp. 117–119). IEEE.

McPhee, N. F. and Hopper, N. J. (1999). Analysis of genetic diversity through population history. *Proceedings of the genetic and evolutionary computation conference* (pp. 1112-1120).

Menesy, A. S., Sultan, H. M. and Kamel, S. (2020). Extracting Model Parameters of Proton Exchange Membrane Fuel Cell Using Equilibrium Optimizer Algorithm. *Proceedings of the 2nd 2020 International Youth Conference on Radio Electronics, Electrical and Power Engineering, REEPE 2020*. IEEE.

Meshkat, M. and Parhizgar, M. (2017). Stud Multi-Verse Algorithm. *2nd Conference on Swarm Intelligence and Evolutionary Computation* (pp. 42–47). IEEE.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. and Teller, E. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*. 21(6): 1087–1092.

Mezura-Montes, E. and Coello, C.A.C. (2008). An Empirical Study About The Usefulness Of Evolution Strategies to Solve Constrained Optimization Problems. *International Journal of General Systems*. 37(4): 443–473.

Michalewicz, Z., (2000). Repair Algorithms. *Evolutionary Computation Advanced Algorithms and Operators 2*, (pp. 56-61). Editors T. Bäck, D. B. Fogel, Z. Michalewicz. U.K.: IOP Publishing.

Mirjalili, S. (2015). The Ant Lion Optimizer. *Advances in Engineering Software*. 83: 80–98.

Mirjalili, S. (2016). SCA: A Sine Cosine Algorithm for solving optimization problems. *Knowledge-Based Systems*. 96: 120–133.

Mirjalili, S., Jangir, P., Mirjalili, S. Z., Saremi, S. and Trivedi, I. N. (2017). Optimization of Problems with Multiple Objectives Using The Multi-Verse Optimization Algorithm. *Knowledge-Based Systems*. 134: 50–71.

Mirjalili, S. and Lewis, A. (2014). Adaptive Gbest-Guided Gravitational Search Algorithm. *Neural Computing and Applications*. 25(7–8): 1569–1584.

Mirjalili, S., Mirjalili, S. M. and Hatamlou, A. (2016). Multi-Verse Optimizer: A Nature-Inspired Algorithm for Global Optimization. *Neural Computing and Applications*. 27(2): 495–513.

Mirjalili, S., Mirjalili, S. M. and Lewis, A. (2014). Grey Wolf Optimizer. *Advances in Engineering Software*. 69: 46–61.

Moein, S. and Logeswaran, R. (2014). KGMO: A Swarm Optimization Algorithm Based On The Kinetic Energy of Gas Molecules. *Information Sciences*. 275: 127–144.

Moein, S., Logeswaran, R. and Faizal bin Ahmad Fauzi, M. (2016). Detection of Heart

Disorders Using an Advanced Intelligent Swarm Algorithm. *Intelligent Automation and Soft Computing*. 23(3): 419–424.

Moghaddam, F.F., Moghaddam, R.F. and Cheriet, M. (2012). *Curved Space Optimization: A Random Search based on General Relativity Theory*. https://arxiv.org/abs/1208.2214, (15.01.2020).

Mohamadi, M.R., Abedini, M. and Rashidi, B. (2020). An Adaptive Multi-Objective Optimization Method for Optimum Design of Distribution Networks. *Engineering Optimization*. 194-217.

Mohammadi-Ivatloo, B., Moradi-Dalvand, M. and Rabiee, A. (2013). Combined Heat and Power Economic Dispatch Problem Solution Using Particle Swarm Optimization with Time Varying Acceleration Coefficients. *Electric Power Systems Research*. 95: 9–18.

Molina, D., Poyatos, J., Ser, J. Del, García, S., Hussain, A. and Herrera, F. (2020). Comprehensive Taxonomies of Nature- and Bio-inspired Optimization: Inspiration Versus Algorithmic Behavior, Critical Analysis Recommendations. *Cognitive Computation*. 1–76.

Montes, E.M., Aguirre A.H., Coello C.A.C., (2005). *Using Evolutionary Strategies to Solve Constrained Optimization Problems*. Evolutionary Algorithms and Intelligent Tools in Engineering Optimization. (pp. 1-25) Editors W. Annicchiarico, J. Périaux, M. Cerrolaza and G. Winter. Barcelona: CIMNE.

Moore, M.P. and Narayanan, A. (1995). *Quantum-Inspired Computing*, https://www.researchgate.net/profile/Ajit-Narayanan/publication/2274710_Quantum-Inspired_Computing/links/5a36e4cf0f7e9b10d8484a98/Quantum-Inspired-Computing.pdf, (10.05.2019).

Morales-Castañeda, B., Zaldívar, D., Cuevas, E., Fausto, F. and Rodríguez, A. (2020).

A Better Balance in Metaheuristic Algorithms: Does it Exist?. *Swarm and Evolutionary Computation*. 54: 100671.

Mosa, M.A. (2020). A Novel Hybrid Particle Swarm Optimization and Gravitational Search Algorithm for Multi-Objective Optimization of Text Mining. *Applied Soft Computing Journal*. 90: 106189.

Mozaffari, A., Emami, M., Azad, N.L. and Fathi, A. (2017). Comparisons of Several Variants of Continuous Quantum-İnspired Evolutionary Algorithms. *Journal of Experimental and Theoretical Artificial Intelligence*. 29(4): 869–909.

Mozaffari, A., Fathi, A. and Behzadipour, S. (2012). The Great Salmon Run: A Novel Bio-Inspired Algorithm for Artificial System Design and Optimisation. *International Journal of Bio-Inspired Computation*. 4(5): 286.

Mucherino, A. and Seref, O. (2007). Monkey search: A novel metaheuristic search for global optimization. In *AIP Conference Proceedings* (pp. 162–173). American Institute of Physics.

Muhsen, D. H., Ghazali, A. B., Khatib, T. and Abed, I. A. (2015). Extraction of Photovoltaic Module Model's Parameters Using an Improved Hybrid Differential Evolution/ Electromagnetism-Like Algorithm. *Solar Energy*. 119: 286–297.

Naderi, B., Tavakkoli-Moghaddam, R. and Khalili, M. (2010). Knowledge-Based Systems Electromagnetism-like Mechanism and Simulated Annealing Algorithms for Flowshop Scheduling Problems Minimizing The Total Weighted Tardiness and Makespan. *Knowledge-Based Systems*. 23(2): 77–85.

Nadimi-Shahraki, M. H., Taghian, S. and Mirjalili, S. (2021). An Improved Grey Wolf Optimizer for Solving Engineering Problems. *Expert Systems with Applications*. 166: 113917.

Nanda, S.J. and Panda, G. (2015). A Clustering Model Based on Colliding Bodies Optimization for Analysis of Seismic Catalog. *2015 International Conference on Microwave, Optical and Communication Engineering, ICMOCE 2015* (pp. 68–71). IEEE.

Narayanan, A. and Moore, M. (1996). Quantum-Inspired Genetic Algorithms. *Proceedings of the IEEE Conference on Evolutionary Computation* (pp. 61–66). IEEE.
Nash, J. C. (2000). The (Dantzig) Simplex Method for Linear Programming. *Computing in Science & Engineering*. 2(1): 29–31.

Nayak, C., Saha, S. K., Kar, R. and Mandal, D. (2019). An Efficient QRS Complex Detection Using Optimally Designed Digital Differentiator. *Circuits, Systems, and Signal Processing*. 38(2): 716–749.

Neggaz, N., Houssein, E. H. and Hussain, K. (2020). An Efficient Henry Gas Solubility Optimization for Feature Selection. *Expert Systems with Applications*. *152*: 113364.

Neshat, M., Sepidnam, G. and Sargolzaei, M. (2013). Swallow Swarm Optimization Algorithm: A New Method to Optimization. *Neural Computing and Applications*. 23(2): 429–454.

Nesmachnow, S. (2014). An Overview of Metaheuristics: Accurate and Efficient Methods for Optimisation. *International Journal of Metaheuristics*. 3(4): 320-347.

Newton, I. (1999). *The principia: Mathematical Principles of Natural Philosophy*. University of California Press.

Nguyen, T.T., Vo, D.N. and Dinh, B. H. (2016). Cuckoo Search Algorithm for Combined Heat and Power Economic Dispatch. *International Journal of Electrical Power and Energy Systems*. 81: 204–214.

Niknam, T., Bavafa, F. and Jabbari, M. (2014). A Novel Self-Adaptive Learning Charged System Search Algorithm for Unit Commitment Problem. *Journal of Intelligent and Fuzzy Systems*. 26(1): 439–449.

Odili, J. B., Noraziah, A., Ambar, R., and Wahab, M. H. A. (2018). A Critical Review of Major Nature-Inspired Optimization Algorithms. *The Eurasia Proceedings of Science, Technology, Engineering & Mathematics*. 2: 376-394.

Osaba, E., Diaz, F. and Onieva, E. (2014). Golden Ball: A Novel Meta-Heuristic to Solve Combinatorial Optimization Problems Based on Soccer Concepts. *Applied Intelligence*. 41(1): 145–166.

Osman, I. H. and Kelly, J. P. (1996). Meta-Heuristics: An Overview. *Meta-Heuristics* (pp. 1–21). Editors Osman I.H., Kelly J.P. Boston, MA: Springer US.

Oyama, A., Shimoyama, K. and Fujii, K. (2005). New Constraint-Handling Method for Multi-Objective Multi-Constraint Evolutionary Optimization and Its Application to Space Plane Design. *Evolutionary and Deterministic Methods for Design, Optimization and Control with Applications to Industrial and Societal Problems (EUROGEN 2005)*, Editors Schilling, R., Haase, W., Periaux, J., Baier, H., Bugeda, G. Munich, Germany: FLM.

Oztas, G. Z. and Erdem, S. (2021). Framework Selection for Developing Optimization Algorithms: Assessing Preferences by Conjoint Analysis and Best–Worst Method. *Soft Computing*. 25: 3831–3848.

Özyön, S., Temurtaş, H., Durmuş, B. and Kuvat, G. (2012). Charged System Search Algorithm for Emission Constrained Economic Power Dispatch Problem. *Energy*. 46(1): 420–430.

Pál, K.F. (2003). Hysteretic Optimization for The Traveling Salesman Problem. *Physica A: Statistical Mechanics and Its Applications*. 329(1–2): 287–297.

Pál, K.F. (2006). Hysteretic Optimization, Faster and Simpler. *Physica A: Statistical Mechanics and Its Applications*, *360*(2), 525–533.

Pan, W.-T. (2012). A new Fruit Fly Optimization Algorithm: Taking the Financial Distress Model as an Example. *Knowledge-Based Systems*. 26: 69–74.

Panda, A. and Pani, S. (2016). Multi-Objective Colliding Bodies Optimization. *Proceedings of Fifth International Conference on Soft Computing for Problem Solving* (pp. 651-664). Springer: Singapore.

Panthagani, P. and Rao, R. S. (2020). Pareto-Based Allocations of Multi-Type Flexible AC Transmission System Devices for Optimal Reactive Power Dispatch Using Kinetic Gas Molecule Optimization Algorithm. *Measurement and Control*. 53(1–2): 239–249.

Panthagani, P. and Rao, R.S. (2017). KGMO for Multi-Objective Optimal Allocation of SVC and Reactive Power Dispatch. *International Conference on Power and Embedded Drive Control, ICPEDC 2017* (pp. 365–369). IEEE.

Pashaei, E. and Aydin, N. (2017). Binary Black Hole Algorithm for Feature Selection and Classification on Biological Data. *Applied Soft Computing Journal*. 56: 94–106.

Pashaei, E., Pashaei, E. and Aydin, N. (2019). Gene Selection Using Hybrid Binary Black Hole Algorithm and Modified Binary Particle Swarm Optimization. *Genomics*. 111(4): 669–686.

Patel, V. K. and Savsani, V. J. (2015). Heat Transfer Search (HTS): A Novel Optimization Algorithm. *Information Sciences*. 324: 217–246.

Pattanaik, J. K., Basu, M. and Dash, D. P. (2020). Heat Transfer Search Algorithm for Combined Heat and Power Economic Dispatch. *Iranian Journal of Science and Technology - Transactions of Electrical Engineering*. 44(2): 963–978.

Petrowski, A. and Hamida, S. B. (2016). Evolutionary algorithms. *Metaheuristics* (pp. 115-178) Editor P. Siarry. Springer, Cham.

Phu-Ang, A. (2018). The New Technique Based On The Galaxy Based Search Algorithm for Solving The Symmetric Travelling Salesman Problem. *1st International ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering, ECTI-NCON 2018* (pp. 131–134). IEEE.

Piotrowski, A. P., Napiorkowski, J. J. and Rowinski, P. M. (2014). How Novel Is The "Novel" Black Hole Optimization Approach? *Information Sciences*. 267: 191–200.

Purnomo H. D. and Wee H. M. (2013). Soccer Game Optimization: An Innovative Integration Of Evolutionary Algorithm and Swarm Intelligence Algorithm. *Metaheuristics Optimization Algorithms in Engineering, Business, Economics, and Finance* (pp.386–420). Editor P.Vasant. IGI Global.

Prasad, S. and Vinod Kumar, D. M. (2017). Hybrid Fuzzy Charged System Search Algorithm Based State Estimation in Distribution Networks. *Engineering Science and Technology, an International Journal*. 20(3): 922–933.

Precup, R.E., David, R.C., Petriu, E.M., Preitl, S. and Rădac, M.B. (2014). Novel Adaptive Charged System Search Algorithm for Optimal Tuning of Fuzzy Controllers. *Expert Systems with Applications*. 41: 1168–1175.

Precup, R. E., Petriu, E. M., Radac, M. B., Voisan, E. I. and Dragan, F. (2015). Adaptive Charged System Search Approach to Path Planning for Multiple Mobile Robots. In *IFAC-PapersOnLine* (pp. 294–299).

Priya, S., Selvakumar, S. and Leela Velusamy, R. (2020). Gravitational Search Based Feature Selection for Enhanced Phishing Websites Detection. *Proceedings of the Second International Conference on Innovative Mechanisms for Industry Applications (ICIMIA* (pp. 453–458).

Pulluri, H., Naresh, R., Sharma, V. and Preeti. (2016). A New Colliding Bodies Optimization for Solving Optimal Power Flow Problem in Power System. *2016 IEEE 6th International Conference on Power Systems, ICPS 2016*. IEEE.

PyCharm, "The Python IDE", https://www.jetbrains.com/pycharm/, (12.04.2018)

Qasim, O. S., Al-Thanoon, N. A. and Algamal, Z. Y. (2020). Feature Selection Based On Chaotic Binary Black Hole Algorithm for Data Classification. *Chemometrics and Intelligent Laboratory Systems*. 204: 104104.

Qiao, W. and Yang, Z. (2019). Solving Large-Scale Function OptimizationProblem by Using a New Metaheuristic Algorithm Based on Quantum DolphinSwarm Algorithm. *IEEE Access*. 7: 138972–138989.

Qubati, G. M., Formato, R. A. and Dib, N. I. (2010). Antenna Benchmark Performance and Array Synthesis Using Central Force Optimisation. *IET Microwaves, Antennas & Propagation*. 4(5): 583–592.

Ragsdell, K. M. and Phillips, D. T. (1976). Optimal Design of a Class of Welded Structures Using Geometric Programming. *Journal of Manufacturing Science and Engineering, Transactions of the ASME*. 98(3): 1021–1025.

Rahchamani, G., Movahedifar, S. M. and Honarbakhsh, A. (2019). A Hybrid Optimized Learning-Based Compressive Performance of Concrete Prediction Using GBMO-ANFIS classifier and genetic algorithm reduction. *Structural Concrete*. 1–21.

Rao, C. V. G. K. and Yesuratnam, G. (2015). Multi-Objective Optimization Using BigBang Big-Crunch based Optimization: Application to Optimal Economic Environmental Dispatch. *Journal of Electrical Systems*. 11(4): 476–492.

Rao, R.V., Savsani, V.J. and Vakharia, D.P. (2011). Teaching–Learning-Based Optimization: A Novel Method for Constrained Mechanical Design Optimization Problems. *Computer-Aided Design*. 43(3): 303–315.

Rashedi, E., Nezamabadi-pour, H. and Saryazdi, S. (2009). GSA: A Gravitational Search Algorithm. *Information Sciences*. 179(13): 2232–2248.

Rashedi, E., Nezamabadi-Pour, H. and Saryazdi, S. (2010). BGSA: Binary Gravitational Search Algorithm. *Natural Computing*. 9(3): 727–745.

Rathore, C. and Roy, R. (2014). A Novel Modified GBMO Algorithm Based Static Transmission Network Expansion Planning. *International Journal of Electrical Power and Energy Systems*. 62: 519–531.

Ray, T. and Liew, K. M. (2003). Society and civilization: an optimization algorithm based on the simulation of social behavior. *IEEE Transactions on Evolutionary Computation*, 7(4), 386-396.

Recioui, A. (2016). Application of a Galaxy-Based Search Algorithm to MIMO System Capacity Optimization. *Arabian Journal for Science and Engineering*. 41(9): 3407–3414.

Reddy, T. S. and Reddy, T. V. S. (2019). Optimization of Shell and Tube Heat Exchanger Design in Organic Rankine Cycle System Using Kinetic Gas Molecule Optimization. *International Journal of Intelligent Engineering and Systems*. 12(2): 297–304.

Reynolds, R. G., Peng, B. and Whallon, R. (2005). Emergent social structures in cultural algorithms. *Annual Conference of the North American Association for Computational Social and Organizational Science* (NAACSOS 2005) (pp. 26-28).

Robic, F., Micallef, D., Paul, S. and Ellul, B. (2020). Implementation and Fine-Tuning of The Big Bang-Big Crunch Optimisation Method for Use in Passive Building Design. *Building and Environment*. 173.

Rong, G., Liu, G., Zheng, M., Sun, A., Tian, Y., and Wang, H. (2013). Parallel

Gravitation Field Algorithm Based on The CUDA Platform. *Journal of Information and Computational Science*, 10(12): 3635-3644.

Ross, O.H.M. (2020). A Review of Quantum-Inspired Metaheuristics: Going from Classical Computers to Real Quantum Computers. *IEEE Access*. 8: 814–838.

Sabri, N. M., Puteh, M. and Mahmood, M. R. (2013). A Review of Gravitational Search Algorithm. *International Journal of Advances in Soft Computing and Its Applications*. 5(3): 1–39.

Sacco, W.F. and De Oliveira, C. (2005). A new stochastic optimization algorithm based on a particle collision metaheuristic. *Proceedings of the 6th World Congress of Structural and Multidisciplinary Optimization, WCSMO, Rio de Janeiro*, papers://b6c7d293-c492-48a4-91d5-8fae456be1fa/Paper/p1889%5Cnfile:///C:/Users/Serguei/OneDrive/Documents/Papers/A new stochastic optimization algorithm.pdf, (04.03.2019).

Sacco, W.F. and Rios-Coelho, A.C. (2016). A New Metropolis Optimisation Method, The Cross-Section Particle Collision Algorithm: Some Preliminary Results. *International Journal of Nuclear Energy Science and Technology*, *10*(1), 59–71.

Sacco, W.F., Filho, H.A. and De Oliveira, C.R. (2007). A Populational Particle Collision Algorithm Applied to A Nuclear Reactor Core Design Optimization. *Joint International Topical Meeting on Mathematics and Computations for Supercomputing in Nuclear Applications*. (pp. 15-19). California: American Nuclear Society.

Sadiq, A.S., Alkazemi, B., Mirjalili, S., Ahmed, N., Khan, S., Ali, I., … Ghafoor, Z. K. (2018). An Efficient IDS Using Hybrid Magnetic Swarm Optimization in WANETs. *IEEE Access*. 6: 29041–29053.

Sahab, M.G., Toropov, V.V. and Gandomi, A.H. (2013). A Review on Traditional and Modern Structural Optimization: Problems and Techniques. *Metaheuristic*

Applications in Structures and Infrastructures (pp. 25–47).

Şahin, M. and Kellegöz, T. (2019). Balancing Multi-Manned Assembly Lines with Walking Workers: Problem Definition, Mathematical Formulation, and an Electromagnetic Field Optimisation Algorithm. *International Journal of Production Research*. 57(20): 6487–6505.

Sahoo, B. P. and Panda, S. (2020). Chaotic Multi Verse Optimizer Based Fuzzy Logic Controller for Frequency Control Of Microgrids. *Evolutionary Intelligence*, 1-22.

Salcedo-Sanz, S. (2016). Modern Meta-Heuristics Based On Nonlinear Physics Processes: A Review Of Models and Design Procedures. *Physics Reports*. 655: 1–70.

Salem, S.A. (2012). BOA: A Novel Optimization Algorithm. *2012 International Conference on Engineering and Technology (ICET)* (pp. 1–5). IEEE.

Salmani, M.H. and Eshghi, K. (2017). A Metaheuristic Algorithm Based on Chemotherapy Science: CSA. *Journal of Optimization*. 1–13.

Sanders, R. (1987). The Pareto Principle: Its Use And Abuse. *Journal of Services Marketing*. 1(2): 37–40.

Sandgren, E. (1990). Nonlinear integer and discrete programming in mechanical design optimization. *Journal of Mechanical Design*. 112(2): 223–229.

Sarafrazi, S., Nezamabadi-Pour, H. and Saryazdi, S. (2011). Disruption : A New Operator in Gravitational Search Algorithm. *Scientia Iranica*. 18(3): 539–548.

Saranya, S. and Saravanan, B. (2020). Effect of Emission in SMES Based Unit Commitment Using Modified Henry Gas Solubility Optimization. *Journal of Energy Storage*. 29: 101380.

Sardari, F. and Moghaddam, M. E. (2016). An Object Tracking Method Using Modified Galaxy-Based Search Algorithm. *Swarm and Evolutionary Computation*. 30: 27–38.

Savsani, V., Patel, V., Gadhvi, B. and Tawhid, M. (2017). Pareto Optimization of a Half Car Passive Suspension Model Using a Novel Multiobjective Heat Transfer Search Algorithm. *Modelling and Simulation in Engineering*. 2017: 2034907.

Scopus Database. *Analyze Search Results*, https://www.scopus.com/term/analyzer.uri?sid=5c482f4076a97e037c834eddb5130f3b&origin=resultslist&src=s&s=%28TITLE-ABS-KEY%28Optimization%29%29+AND+%28%28%28%28%28%22Bio-inspired%22%29%29+OR+%28%22Nature-inspired%22%29%29+OR+%28%22Swarm+intelligence%22%29%29+OR%28%22Physics-based%22%29%29+OR%28%22Chemistry-based%22%29&sort=plf-f&sdt=a&sot=a&sl=149&count=61913&analyzeResults=Analyze+results&txGid=9008ae79476449b0091855f4848b91d4, (01.04.2021).

Sedighizadeh, M. and Ghalambor, M. (2014). Reconfiguration of Radial Distribution Systems with Fuzzy Multi-Objective Approach Using Modified Big Bang-Big Crunch Algorithm. *Arabian Journal for Science and Engineering*. 6287–6296.

Sels, V. and Vanhoucke, M. (2014). A hybrid Electromagnetism-like Mechanism / Tabu Search Procedure for The Single Machine Scheduling Problem with a Maximum Lateness Objective. *Computers & Industrial Engineering*. 67: 44–55.

Shabani, A., Asgarian, B., Salido, M. and Asil Gharebaghi, S. (2020). Search and Rescue Optimization Algorithm: A New Optimization Method For Solving Constrained Engineering Optimization Problems. *Expert Systems with Applications*. 161: 113698.

Shah-Hosseini, H. (2011a). Otsu's Criterion-Based Multilevel Thresholding by a

Nature-Inspired Metaheuristic Called Galaxy-Based Search Algorithm. *Proceedings of the 2011 3rd World Congress on Nature and Biologically Inspired Computing, NaBIC 2011* (pp. 383–388). IEEE.

Shah-Hosseini, H. (2011b). Principal Components Analysis by The Galaxy-Based Search Algorithm: A Novel Metaheuristic for Continuous Optimisation. *International Journal of Computational Science and Engineering*. 6(1/2): 132.

Sharma, S. and Kumar, S. (2017). Discrete Gravitational Search Algorithm for Virtual Machine Placement in Cloud Computing. *International Journal of Pure and Applied Mathematics*. 117(19): 337–342.

Shehabeldeen, T. A., Elaziz, M. A., Elsheikh, A. H., Hassan, O. F., Yin, Y., Ji, X., … Zhou, J. (2020). A Novel Method for Predicting Tensile Strength of Friction Stir Welded AA6061 Aluminium Alloy Joints Based on Hybrid Random Vector Functional Link and Henry Gas Solubility Optimization. *IEEE Access*. 8: 79896–79907.

Shen, J. and Li, J. (2010). The Principle Analysis of Light Ray Optimization Algorithm. In *2010 Second International Conference on Computational Intelligence and Natural Computing (CINC) The* (pp. 154–157). IEEE.

Shen, J. and Li, J. (2012). Light Ray Optimization Algorithm Based on Annealing Strategy. *Advanced Materials Research*. 461: 435–439.

Shen, J. and Li, Y. (2009). Light Ray Optimization and Its Parameter Analysis. *Proceedings of the 2009 International Joint Conference on Computational Sciences and Optimization*, (pp. 918–922). IEEE.

Shi, Y. (2011a). Brain Storm Optimization Algorithm. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 6728 LNCS*(PART 1), (pp. 303–309). Berlin, Heidelberg: Springer.

Shi, Y. (2011b). Brain Storm Optimization Algorithm. *Advances in Swarm Intelligence* (pp. 303–309). Editors Y. Tan, Y. Shi, Y. Chai, and G. Wang. Berlin, Heidelberg: Springer.

Shirgir, S., Azar, B. F. and Hadidi, A. (2020). Reliability-Based Simplification of Bouc-Wen Model and Parameter Identification Using a New Hybrid Algorithm. *Structures*. 27: 297–308.

Shukla, A. K., Singh, P. and Vardhan, M. (2020). Gene Selection for Cancer Types Classification Using Novel Hybrid Metaheuristics Approach. *Swarm and Evolutionary Computation*. 54: 100661.

Siarry, P. (2016). Introduction, *Metaheuristics*, (pp. 1-18) Editor P. Siarry. Springer, Cham.

Siddique, N. and Adeli, H. (2015b). Central Force Metaheuristic Optimisation. *Scientia Iranica*. 22(6): 1941–1953.

Siddique, N. and Adeli, H. (2015a). Nature Inspired Computing: An Overview and Some Future Directions. *Cognitive Computation*. 7(6): 706–714.

Siddique, N. and Adeli, H. (2016). Physics-Based Search and Optimization: Inspirations From Nature. *Expert Systems*. 33(6): 607–623.

Smit, S. K. and Eiben, A. E. (2010). Using entropy for parameter analysis of evolutionary algorithms. *Experimental Methods for the Analysis of Optimization Algorithms* (pp. 287-310). Editors Bartz-Beielstein, Thomas Chiarandini, Marco Paquete, Luís Preuss, Mike. Springer, Berlin, Heidelberg.

Simon, D. (2013). *Evolutionary Optimization Algorithms.* USA: John Wiley & Sons.

Song, S., Jia, H. and Ma, J. (2019). A Chaotic Electromagnetic Field Optimization Algorithm Based on Fuzzy Entropy for Multilevel Thresholding Color Image

Segmentation. *Entropy*. 21(4): 398.

Sörensen. K. (2015). Metaheuristics—The Metaphor Exposed. *International Transactions in Operational Research*. 22(1): 3-18.

Sörensen, K., Sevaux, M. and Glover, F. (2018). A History of Metaheuristics. *Handbook of Heuristics* (pp. 2–16). Editors R. Martí, P. Pardalos, and M. Resende. http://arxiv.org/abs/1704.00853, (6.07.2020).

Soto, R., Crawford, B., Olivares, R., Taramasco, C., Figueroa, I., Gómez, Á., … Paredes, F. (2018). Adaptive Black Hole Algorithm for Solving The Set Covering Problem. *Mathematical Problems in Engineering*. 2018: 2183214.

Sotoudeh-Anvari, A. and Hafezalkotob, A. (2018). A Bibliography of Metaheuristics-Review from 2009 to 2015. *International Journal of Knowledge-Based and Intelligent Engineering Systems*. *22*(1): 83–95.

Srinivasa Rao, G., Vijaya Kumar, V. and Suresh Varma, P. (2015). A New Optimized Data Clustering Technique using Cellular Automata and Adaptive Central Force Optimization. *Research Journal of Applied Sciences, Engineering and Technology*. 10(5): 522–531.

Srivastava, A. and Das, D. K. (2020). A New Kho-Kho Optimization Algorithm: An Application To Solve Combined Emission Economic Dispatch and Combined Heat and Power Economic Dispatch Problem. *Engineering Applications of Artificial Intelligence*. 94: 103763.

Su, H. and Yang, Y. (2008). Quantum-Inspired Differential Evolution for Binary Optimization. *Proceedings - 4th International Conference on Natural Computation, ICNC 2008* (pp. 341–346). IEEE.

Su, C. and Lin, H. (2011). Applying Electromagnetism-Like Mechanism for Feature

Selection. *Information Sciences*. 181(5): 972–986.

Sulaiman, M., Salhi, A., Selamoglu, B. I. and Kirikchi, O. B. (2014). A Plant Propagation Algorithm for Constrained Engineering Optimisation Problems. *Mathematical Problems in Engineering*. 2014: 627416.

Sun, J. and Li, Y. (2018). Social Cognitive Optimization with Tent Map for Combined Heat and Power Economic Dispatch. *International Transactions on Electrical Energy Systems*. 29(1): 1–15.

Sun, J., Feng, B. and Xu, W. (2004). Particle Swarm Optimization with Particles Having Quantum Behavior. *Proceedings of the 2004 Congress on Evolutionary Computation, CEC2004* (pp. 325–331). IEEE.

Tabrizian, Z., Afshari, E., Ghodrati, G., Hossein, M. and Beigy, A. (2013). A New Damage Detection Method: Big Bang-Big Crunch Algorithm. *Shock and Vibration*. 20: 633–648.

Tahani, M., Yousefi, H., Noorollahi, Y. and Fahimi, R. (2019). Application of Nature İnspired Optimization Algorithms in Optimum Positioning of Pump-As-Turbines in Water Distribution Networks. *Neural Computing and Applications*. 31(11): 7489–7499.

Talatahari, S., Sheikholeslami, R., Farahmand Azar, B. and Daneshpajouh, H. (2013). Optimal Parameter Estimation for Muskingum Model Using a CSS-PSO Method. *Advances in Mechanical Engineering*. 5: 480954.

Talatahari, S., Aalami, M. T., and Parsiavash, R. (2018). Risk-Based Arch Dam Optimization Using Hybrid Charged System Search. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering*. 4(2): 04018008.

Talatahari, S. and Azizi, M. (2020). Optimization of Constrained Mathematical and

Engineering Design Problems Using Chaos Game Optimization. *Computers and Industrial Engineering*. 145: 106560.

Talbi, E.-G. (2009). *Metaheuristıcs from Design to Implementation*. John Wiley & Sons.

Talebi, B. and Dehkordi, M. N. (2018). Sensitive Association Rules Hiding Using Electromagnetic Field Optimization Algorithm. *Expert Systems with Applications*. 114: 155–172.

Tamura, K. and Yasuda, K. (2011a). Primary Study of Spiral Dynamics İnspired Optimization. *IEEJ Transactions on Electrical and Electronic Engineering*. 6(S1): *S98-S100.*

Tamura, K. and Yasuda, K. (2011b). Spiral Multipoint Search for Global Optimization. *Proceedings - 10th International Conference on Machine Learning and Applications, ICMLA 2011* (pp. 470–475). IEEE.

Tamura, K. and Yasuda, K. (2011c). Spiral Optimization. *IEEE International Conference on Systems Man and Cybernetics* (pp. 1759–1764). IEEE.

Tamura, K. and Yasuda, K. (2013). A Stability Analysis Based Parameter Setting Method for Spiral Optimization. *Proceedings - 2013 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2013* (pp. 3909–3914). IEEE.

Tang, R., Fong, S., Yang, X.S. and Deb, S. (2012). Wolf Search Algorithm with Ephemeral Memory. *7th International Conference on Digital Information Management, ICDIM 2012* (pp. 165–172).

Tanyildizi, E. and Demir, G. (2017). Golden sine algorithm: a novel math-inspired algorithm. *Advances in Electrical and Computer Engineering*. 17(2): 71–79.

Taramsco, C., Crawford, B., Soto, R., Cortés-Toro, E. M. and Olivares, R. (2020). A

New Metaheuristic Based on Vapor-Liquid Equilibrium for Solving a New Patient Bed Assignment Problem. *Expert Systems With Applications*. 158: 113506.

Tatsumura, K., Dixon, A. R. and Goto, H. (2019). FPGA-based simulated bifurcation machine. *2019 29th International Conference on Field Programmable Logic and Applications (FPL)* (pp. 59-66). IEEE.

Tavakkoli-Moghaddam, R., Khalili, M. and Naderi, B. (2009). A Hybridization of Simulated Annealing and Electromagnetic-Like Mechanism for Job Shop Problems with Machine Availability and Sequence-Dependent Setup Times to Minimize Total Weighted Tardiness. *Soft Computing*. 13: 995–1006.

Tavakolan, M. and Share, M. A. M. (2013). Applying Magnetic Charged System Search Algorithm to Construction Project Planning Problems. *Computing in Civil Engineering - Proceedings of the 2013 ASCE International Workshop on Computing in Civil Engineering* (pp. 733–740) Editors L. Angeles, I. Technology, I. Brilakis, S. Lee, & B. Becerik-gerber. Los Angeles, California: ASCE.

Tayarani-N., M. H. and Akbarzadeh-T., M. R. (2014). Magnetic-Inspired Optimization Algorithms: Operators and Structures. *Swarm and Evolutionary Computation*, *19*, 82–101.

Tayarani, M. H. and Akbarzadeh. T., N. M. R. (2008). Magnetic Optimization Algorithms A New Synthesis. *2008 IEEE Congress on Evolutionary Computation, CEC 2008*, (pp. 2659–2664). IEEE.

Teeparthi, K. and Kumar, D.M.V. (2018). Security-Constrained Optimal Power Flow with Wind and Thermal Power Generators Using Fuzzy Adaptive Artificial Physics Optimization Algorithm. *Neural Computing and Applications*. 29(3): 855–871.

Tejani, G. G., Kumar, S. and Gandomi, A. H. (2019b). Multi-Objective Heat Transfer Search Algorithm for Truss Optimization. *Engineering with Computers*. 1-22.

Tejani, G. G., Savsani, V. J., Patel, V. K. and Mirjalili, S. (2019a). An Improved Heat Transfer Search Algorithm for Unconstrained Optimization Problems. *Journal of Computational Design and Engineering*. 6(1): 13–32.

Tian, Y., Liu, D., Ma, X. and Zhang, C. (2009). A Hybrid PSO with EM for Global Optimisation. *Internnational Journal of Modelling, Identification and Control*. 8(4): 327–334.

Tolabi, H. B., Shakarami, M. R., Hosseini, R. and Ayob, S. B. M. (2016). Novel FGbSA: Fuzzy-Galaxy-Based Search Algorithm for Multi-Objective Reconfiguration of Distribution Systems. *Russian Electrical Engineering*. 87(10): 588–595.

Torres, R. H., Velho, H. F. de C. and Chiwiacowsky, L. D. (2018). Rotation-Based Multi-Particle Collision Algorithm with Hooke–Jeeves Approach Applied to the Structural Damage Identification. *Computational Intelligence, Optimization and Inverse Problems with Applications in Engineering* (pp. 87–110). Editors G. M. Platt, X. S. Yang, and A. J. Silva Neto. Cham: Springer.

Toshiba Corporation (2019). *Corporate Research & Development Center*, https://www.global.toshiba/ww/technology/corporate/rdc/rd/topics/19/1910-02.html, (25.04.2021).

Tovey, C. A. (2018). Nature-Inspired Heuristics : Overview and Critique. *Recent Advances in Optimization and Modeling of Contemporary Problems.* 158-192.

Tsai, C. W., Huang, B. C. and Chiang, M. C. (2014). A Novel Spiral Optimization for Clustering. *Mobile, Ubiquitous, and Intelligent Computing Lecture Notes in Electrical Engineering* (pp. 529–534). Editors J. Park, H. Adeli, N. Park, and I. Woungang. Berlin, Heidelberg: Springer.

Tsou, C.-S. and Kao, C.-H. (2008). Multi-Objective İnventory Control Using Electromagnetism-Like Meta-Heuristic. *International Journal of Production*

Research. 46(14): 3859–3874.

Valenzuela, M., Pena, A., Lopez, L. and Pinto, H. (2017). A Binary Multi-Verse Optimizer Algorithm Applied to The Set Covering Problem. *2017 4th International Conference on Systems and Informatics, ICSAI 2017* (pp. 513–518).

Vasebi, A., Fesanghary, M. and Bathaee, S. M. T. (2007). Combined Heat and Power Economic Dispatch by Harmony Search Algorithm. *International Journal of Electrical Power and Energy Systems*. 29(10): 713–719.

Veres, P., Bányai, T. and Illés, B. (2017). Optimization of In-Plant Production Supply with Black Hole Algorithm. *Solid State Phenomena*. 261(1): 503–508.

Vinay Kumar, V., Kusumavathi, S. and Sharma, K. S. (2019). Kinetic Gas Molecule Optimization for MRI Brain Segmentation Using The Fuzzy C-Means Clustering. *International Journal of Recent Technology and Engineering*. 8(2 Special Issue 8): 900–907.

Wang, T., Liu, W. F., and Liu, C. F. (2016). Optimization Algorithm of Black-Hole Based on Euclidean Distance. *Journal of Shenyang University of Technology*, 38(2): 201-205.

Wang, B., Wang, C., Wang, L. U., Xie, N. and Wei, W. E. I. (2019b). Recognition of Semg Hand Actions Based on Cloud Adaptive Quantum Chaos Ions Motion Algorithm Optimized SVM. *Journal of Mechanics in Medicine and Biology*. 19(6): 1950047.

Wang, G. G. (2018). Moth Search Algorithm: A Bio-Inspired Metaheuristic Algorithm for Global Optimization Problems. *Memetic Computing*. 10(2): 151–164.

Wang Y.-J, and Ma C.-L (2018) Opposition-Based Learning Differential Ion Motion Algorithm, *Journal of Information Hiding and Multimedia Signal Processing*, 9(4): 987-996.

Wang, H., Huang, M. and Wang, J. (2019a). An Effective Metaheuristic Algorithm for Flowshop Scheduling with Deteriorating Jobs. *Journal of Intelligent Manufacturing*. 30(7): 2733–2742.

Wang, K. and Shen, J. H. (2012). Multi-Objective Light Ray Optimization. *Proceedings of the 2012 5th International Joint Conference on Computational Sciences and Optimization*, (pp. 822–825). IEEE.

Wang, L., Niu, Q. and Fei, M. (2007). A Novel Quantum Ant Colony Optimization Algorithm. *Bio-Inspired Computational Intelligence and Applications. LSMS 2007. Lecture Notes in Computer Science* (pp. 277–286). Editors K. Li, M. Fei, G. W. Irwin, and S. Ma. Berlin, Heidelberg: Springer.

Wang, X., Pan, J. S. and Chu, S. C. (2020a). A Parallel Multi-Verse Optimizer for Application in Multilevel Image Segmentation. *IEEE Access*. 8: 32018–32030.

Wang, H., Zhang, C. and Zeng, J. (2020b). Precipitation Estimation by Multi-Time Scale Support Vector Machine with Quantum Optics İnspired Optimization Algorithm. *Second Target Recognition and Artificial Intelligence Summit Forum* (pp. 822-825). International Society for Optics and Photonics.

Wang J., Ma L., Liu Y., and Yang W. (2017). Self-Adaptive Optics Inspired Optimization for Real-Time Pricing of Smart Grid. *Power System Protection and Control*, 45(24), 29-35.

Wang, L. G., and Wei, F. J. (2013). Artificial Physics Optimization Algorithm Combined Band Selection for Hyperspectral İmagery. *Journal of Harbin Institute of Technology*. 45(9): 100-106.

Wegener, I. (2005). *Complexity Theory, Exploring the Limits of Efficient Algorithms*. Springer.

Whitley, D. and Watson, J.P. (2005). Complexity Theory And The No Free Lunch Theorem. *Search Methodologies* (pp. 317–339). Editors E. K. Burke & G. Kendall. Boston, MA: Springer.

Wolpert, D.H. and Macready, W.G. (1995). *No free lunch theorems for search*. https://www.researchgate.net/profile/David-Wolpert/publication/221997149_No_Free_Lunch_Theorems_for_Search/links/0c960529e2b49c4dce000000/No-Free-Lunch-Theorems-for-Search.pdf, (05.04.2019).

Wu W. (2016). *[Algorithm] P, NP, NPC, NPH*, https://williamswu.wordpress.com/2016/04/17/algorithm-np-p-nph-npc/, (10.05.2019)

Wu, C., Wu, T., Fu, K., Zhu, Y., Li, Y., He, W. and Tang, S. (2017). AMOBH: Adaptive Multiobjective Black Hole Algorithm. *Computational Intelligence and Neuroscience*. 2017: 6153951.

Wunnava, A., Naik, M. K., Panda, R., Jena, B. and Abraham, A. (2020). A Novel İnterdependence Based Multilevel Thresholding Technique Using Adaptive Equilibrium Optimizer. *Engineering Applications of Artificial Intelligence*. *94*: 103836.

Xie, J. and Ma, H. (2018). Application of Improved APO Algorithm in Vulnerability Assessment and Reconstruction of Microgrid. *IOP Conference Series: Earth and Environmental Science*. 108: 1–9.

Xie, L. and Zeng, J. (2010). The Performance Analysis of Artificial Physics Optimization Algorithm Driven by Different Virtual Forces. *ICIC Express Letters*. 4(1): 239–244.

Xie, L., Zeng, J. and Cui, Z. (2009a). General Framework of Artificial Physics Optimization Algorithm. *2009 World Congress on Nature and Biologically Inspired*

Computing, NABIC 2009 - Proceedings (pp. 1321–1326). IEEE.

Xie, L., Zeng, J. and Cui, Z. (2009b). On Mass Effects to Artificial Physics Optimisation Algorithm for Global Optimisation Problems. *International Journal of Innovative Computing and Applications*. *2*(2): 69–76.

Xie, W., Wang, J. S. and Tao, Y. (2019). Improved Black Hole Algorithm Based on Golden Sine Operator and Levy Flight Operator. *IEEE Access*. 7: 161459–161486.

Xing, B. and Gao, W.-J. (2014). *Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms*. Cham: Springer International Publishing.

Xing, Z. and Jia, H. (2020). Modified Thermal Exchange Optimization Based Multilevel Thresholding for Color Image Segmentation. *Multimedia Tools and Applications*. 79(1–2): 1137–1168.

Xiong, N., Molina, D., Ortiz, M. L. and Herrera, F. (2015). A Walk into Metaheuristics for Engineering Optimization: Principles, Methods and Recent Trends. *International Journal of Computational Intelligence Systems*. 8(4): 606–636.

Xiong, R., Lu, Y.-Z., Zhou, X. and Xu, W. (2012). Hysteretic Optimization for the 3D Protein Folding Based on the Lattice Model. *Applied Mechanics and Materials*. 198–199: 40–47.

Yaghoobi, S. and Mojallali, H. (2016). Modified Black Hole Algorithm with Genetic Operators. *International Journal of Computational Intelligence Systems*. *9*(4): 652–665.

Yalcin, Y. and Pekcan, O. (2020). Nuclear Fission–Nuclear Fusion Algorithm for Global Optimization: A Modified Big Bang–Big Crunch Algorithm. *Neural Computing and Applications*. 32(7): 2751-2783.

Yan, H.-S., Wan, X.-Q. and Xiong, F.-L. (2014). A Hybrid Electromagnetism-Like Algorithm for Two-Stage Assembly Flow Shop Scheduling Problem. *International Journal of Production Research*. 52(19): 5627–5640.

Yan, X. and Wu, W. (2012). Hysteretic Optimization for the Capacitated Vehicle Routing Problem. In *Proceedings of 2012 9th IEEE International Conference on Networking, Sensing and Control* (pp. 18–21). IEEE.

Yang, X.S. (2018). Mathematical Analysis of Nature-Inspired Algorithms. *Nature-Inspired Algorithms and Applied Optimization* (pp. 1–25). Editor X.S. Yang. Cham: Springer.

Yang, X.S. (2014). *Nature-Inspired Optimization Algorithms*. Academic Press.

Yang, X.S. and Deb, S. (2010). Eagle strategy using Lévy walk and firefly algorithms for stochastic optimization. *Nature Inspired Cooperative Strategies for Optimization* (pp. 101–111). Editors J. R. González, D. A. Pelta, C. Cruz, G. Terrazas, and N. Krasnogor. Berlin, Heidelberg: Springer.

Yang, X.S., Deb, S. and Fong, S. (2014). Metaheuristic Algorithms: Optimal Balance of Intensification and Diversification. *Applied Mathematics and Information Sciences*. 8(3): 977–983.

Yang, X.S., Deb, S., Hanne, T. and He, X. (2015). Attraction and Diffusion in Nature-Inspired Optimization Algorithms. *Neural Computing and Applications*. 31(7): 1987-1994.

Yang, X.S.. (2009). Firefly Algorithms for Multimodal Optimization. *Stochastic Algorithms: Foundations and Applications* (pp. 169–178). Editors O. Watanabe and T. Zeugmann. Berlin, Heidelberg: Springer.

Yang, X.S.. (2010a). *Engineering Optimization: An Introduction with Metaheuristic*

Applications. John Wiley & Sons.

Yang, X.S.. (2010b). A New Metaheuristic Bat-Inspired Algorithm. *Nature Inspired Cooperative Strategies for Optimization* (pp. 65–74). Editors J. R. González, D. A. Pelta, C. Cruz, G. Terrazas, and N. Krasnogor. Berlin, Heidelberg: Springer.

Yang, X.S. (2012). Flower Pollination Algorithm for Global Optimization. *Unconventional Computation and Natural Computation* (pp. 240–249). Editors J. Durand-Lose and N. Jonoska. Berlin, Heidelberg: Springer.

Yang, X.S., and Deb, S. (2009). Cuckoo Search via Lévy Flights. *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)* (pp. 210–214). IEEE.

Yapici, H. and Cetinkaya, N. (2019). A New Meta-Heuristic Optimizer: Pathfinder algorithm. *Applied Soft Computing Journal*. 78: 545–568.

Yao X., Liu, Y. and Lin. G. (1999). Evolutionary Programming Made Faster. *IEEE Transactions on Evolutionary Computation*. 3(2): 82–102.

Yazdani, A., Jayabarathi, T., Ramesh, V. and Raghunathan, T. (2013). Combined Heat and Power Economic Dispatch Problem Using Firefly Algorithm. *Frontiers in Energy*. 7(2): 133–139.

Yin, Z., Liu, J., Luo, W., and Lu, Z. (2018). An Improved Big Bang-Big Crunch Algorithm for Structural Damage Detection. *Structural Engineering and Mechanics*. 68(6): 735-745.

Yıldız, B. S., Yıldız, A. R., Pholdee, N., Bureerat, S., Sait, S. M., and Patel, V. (2020). The Henry Gas Solubility Optimization Algorithm for Optimum Structural Design of Automobile Brake Components. *Materials Testing*. 62(3): 261-264.

Yılmaz, S. and Gökaşan, M. (2015). Optimization Based Path Planning via Big Bang-Big Crunch with Local Search. In *2015 IEEE International Conference on Control System, Computing and Engineering* (pp. 27–29).

Yokota, T., Gen, M., Ida, K. and Taguchi, T. (1995). Optimal Design of System Reliability by an Improved Genetic Algorithm. *Transactions of Institute of Electronics, Information and Computer Engineering*. J78-A(6): 702-709.

Yurtkuran, A. (2019). An Improved Electromagnetic Field Optimization for The Global Optimization Problems. *Computational Intelligence and Neuroscience*. 2019: 6759106.

Yurtkuran, A. and Emel, E. (2010). Expert Systems with Applications A New Hybrid Electromagnetism-like Algorithm for Capacitated Vehicle Routing Problems. *Expert Systems With Applications*. 37(4): 3427–3433.

Zamani, A., Barakati, S. M. and Yousofi-Darmian, S. (2016). Design of a Fractional Order PID Controller Using GBMO Algorithm for Load–Frequency Control with Governor Saturation Consideration. *ISA Transactions*. 64: 56–66.

Zandevakili, H., Rashedi, E. and Mahani, A. (2019). Gravitational Search Algorithm with Both Attractive and Repulsive Forces. *Soft Computing*. 23(3): 783–825.

Zarand, G., Pazmandi, F., Pál, K.F. and Zimanyi, G.T. (2002). Hysteretic Optimization. *Physical Review Letters*. 89(15): 1–4.

Zerigat, H., Benasla, L., Belmadani, A. and Rahli, M. (2013). Solution of Combined Economic and Emission Dispatch Problems Using Galaxy-Based Search Algorithm. *Journal of Electrical Systems*. 9(4): 468–480.

Zerigat, D. H., Benasla, L., Belmadani, A. and Rahli, M. (2014). Galaxy-Based Search Algorithm to Solve Combined Economic and Emission Dispatch. *UPB Scientific*

Bulletin, Series C: Electrical Engineering. 76(1): 209–220.

Zhang, Y. and Jin, Z. (2020). Group Teaching Optimization Algorithm: A Novel Metaheuristic Method for Solving Global Optimization Problems. *Expert Systems with Applications*. 148: 113246.

Zhang, Yu, Wu, L., Zhang, Y. and Wang, J. (2011). Immune Gravitation Inspired Optimization Algorithm. *Advanced Intelligent Computing* (pp. 178–185). Editors D.-S. Huang, Y. Gan, V. Bevilacqua, and J. C. Figueroa. Berlin, Heidelberg: Springer-Verlag.

Zhao, Z., Cui, Z., Zeng, J. and Yue, X. (2011). Artificial Plant Optimization Algorithm for Constrained Optimization Problems. In *2011 Second International Conference on Innovations in Bio-inspired Computing and Applications* (pp. 120–123). IEEE.

Zheng, M., Liu, G. X., Zhou, C. G., Liang, Y. C. and Wang, Y. (2010). Gravitation Field Algorithm and Its Application in Gene Cluster. *Algorithms for Molecular Biology*. 5(1): 1–11.

Zheng, M., Sun, Y., Liu, G.X., Zhou, Y. and Zhou, C.G. (2012a). Improved Gravitation Field Algorithm and Its Application in Hierarchical Clustering. *PLoS One*. 7(11).

Zheng, M., Wu, J.N, Huang, Y.X, Liu, G.X, Zhou, Y. and Zhou, C.G. (2012b). Inferring Gene Regulatory Networks by Singular Value Decomposition and Gravitation Field Algorithm. *PLoS One*. 7(12): 1–6.

Zheng, C., Li, H., and Wang, L. (2019). An Improvement of Gravitational Search Algorithm. *Proceedings of 2019 Chinese Intelligent Systems Conference. CISC 2019. Lecture Notes in Electrical Engineering*, (pp. 490-503). Editors Jia Y., Du J., Zhang W. Singapore: Springer.

Zhong, K., Luo, Q., Zhou, Y. and Jiang, M. (2020). TLMPA: Teaching-Learning-Based Marine Predators Algorithm. *AIMS Mathematics*. 6(2): 1395–1442.

APPENDICES

APPENDIX 1: Unconstrained/Bounded Benchmark Problems

Name	Test Function	Range	Objective
De JongF1 /Sphere	$\sum_{i=1}^{n} x_i^2$	[-100,100]	Min: 0 at (0, 0, …,0)
AckleyF1	$20 + e - 20\exp(-0.20\sqrt{\frac{1}{5}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i))$	[-32,32]	Min: 0 at (0, 0…, 0)
Rastrigin	$100x + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i)$	[-5.12,5.12]	Min: 0 at (0, 0…, 0)
Cosine Mixture	$\sum_{i=1}^{n} x_i^2 - \frac{1}{10}\sum_{i=1}^{n} cos(5\pi x_i)$	[-1,1]	Min -0.1*n at (0,0,..,0)
Exponential	$-exp(-0.5\sum_{i=1}^{n} x_i^2)$	[-1,1]	Min: -1 at (0, 0,…,0)
Cb3 (Three Hump Camel)	$2x_1^2 - 1.05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$	[-5,5]	Min: 0 at (0, 0)
Bohachevsky 2	$x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1)\,0.4\cos(4\pi x_2) + 0.3$	[-50,50]	Min: 0 at (0, 0)
Griewank	$\frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\frac{x_i}{\sqrt{i}} + 1$	[-100,100]	Min: 0 at (0,…,0)
Alpine 1	$\sum_{i=1}^{n} \lvert x_i \sin(x_i) + 0.1x_i \rvert$	[-10,10]	Min: 0 at (0,…,0)
Egg Crate	$\sum_{i=1}^{n} x_i^2 + 25sin^2(x_i)$	[-5,5]	Min: 0 at (0,…,0)
3-D Paraboloid	$2x^2 + 10y^2 + 5z^2 + 6xy - 2xz + 4yz - 6x - 14y - 2z + 6$	[-10,10]	Min: 0 at (1.4, 0.2, 0.4)
Price 2	$1 + \sum_{i=1}^{n} sin^2(x_i) - 0.1e^{-\sum_{i=1}^{n} x_i^2}$	[-10,10]	Min: 0.9 at (0,…,0)
Schaffer 1	$0.5 + \frac{sin^2(\sum_{i=1}^{n} x_i^2)^2 - 0.5}{1 + 0.001(\sum_{i=1}^{n} x_i^2)^2}$	[-100,100]	Min: 0 at (0,…,0)
Schwefel 1.2	$\sum_{i=1}^{n}(\sum_{j=1}^{i} x_j)^2$	[-100,100]	Min: 0 at (0,…,0)
Xin-She Yang Function 2	$\sum_{i=1}^{n} \lvert x_i \rvert e^{(-\sum_{i=1}^{n}\sin(x_i^2))}$	[-2π,2π]	Min: 0 at (0,…,0)
Himmelblau	$(x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$	[-5,5]	Min: 0 at (3, 2)
Guinta	$0.6 + \sum_{i=1}^{2}[\sin\left(\frac{16}{15}x_i - 1\right) + sin^2(\frac{16}{15}x_i - 1) + \frac{1}{50}\sin(4(\frac{16}{15}x_i - 1))]$	[-1,1]	Min: 0.060447 at (0.45834282, 0.45834282)
Adjiman	$\cos(x_1)\sin(x_2) - \frac{x_1}{(x_2^2 + 1)}$	[-1,2] [-1,1]	Min: -2.02181 at (2, 0.10578)
Branin	$\frac{1}{51.95}(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) - 44.81$	[-5,10] [0,15]	Min: 0.397887 at (-π, 12.275;π, 2.275;9.424 78,2.475)
Beale	$(1.5 - x_1 - x_1x_2)^2 + (2.25 - x_1 + x_1x_2^2)^2 + (2.625 - x_1 + x_1x_2^3)^2$	[-4.5,4.5]	Min: 0 at (3, 0.5)
Bird	$\sin(x_1)\,e^{(1-\cos(x_2))^2} + \cos(x_2)\,e^{(1-\sin(x_1))^2} + (x_1 - x_2)^2$	[-2π,2π]	Min: -106.764537 at (4.70104, 3.15294; -1.58214, -3.13024)
McCormick	$\sin(x_1 + x_2) + (x_1 - x_2)^2 - \left(\frac{3}{2}\right)x_1 + \left(\frac{5}{2}\right)x_2 + 1$	[-1.5,4] [-3,3]	Min: -1.9133 at (-0.547, -1.547)

APPENDIX 2: Pressure Vessel Model

Minimize $f(\vec{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$

Subject to

$$-x_1 + 0.0193x_3 \leq 0$$

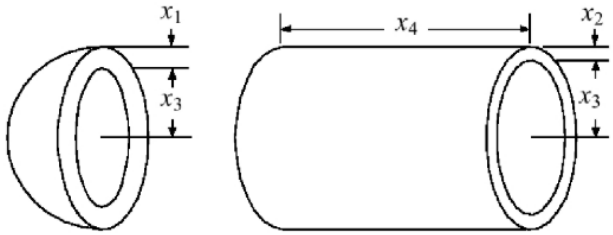
$$-x_2 + 0.00954x_3 \leq 0$$

$$-\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \leq 0$$

$$x_4 - 240 \leq 0$$

$$x_1, x_2 \in [0.0625, 10]; x_3 \in [0, 100]; x_4 \in [0, 240]$$

APPENDIX 3: Pressure Vessel Figure



Source: (Cagnina et al., 2008: 323)

APPENDIX 4: Himmelblau's Function Model

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

Subject to

$$g_1(\vec{x}) = 85.334407 + 0.0056858x_2x_5 + 0.00026x_1x_4 - 0.0022053x_3x_5$$

$$g_2(\vec{x}) = 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2$$

$$g_3(\vec{x}) = 9.300961 + 00.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4$$

$$0 \leq g_1(\vec{x}) \leq 92$$

$$90 \leq g_2(\vec{x}) \leq 110$$

$$20 \leq g_3(\vec{x}) \leq 25$$

$$x_1 \in [78,102]; x_2 \in [33,45]; x_3, x_4, x_5 \in [27,45]$$

APPENDIX 5: Welded Beam Model

$$\text{Minimize } f(\vec{x}) = (1 + c_1)x_1^2 x_2 + c_2 x_3 x_4 (L + x_2)$$

Subject to

$$g_1(\vec{x}) = \tau(\vec{x}) - \tau_{max} \leq 0$$

$$g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{max} \leq 0$$

$$g_3(\vec{x}) = x_1 - x_4 \leq 0$$

$$g_4(\vec{x}) = c_1 x_1^2 + c_2 x_3 x_4 (L + x_2) - 5 \leq 0$$

$$g_5(\vec{x}) = \delta(\vec{x}) - \delta_{max} \leq 0$$

$$g_6(\vec{x}) = P - P_c(\vec{x}) \leq 0$$

$$\tau(\vec{x}) = \sqrt{(\tau')^2 + 2\tau'\tau'' \frac{x_2}{2R} + (\tau'')^2}$$

$$\tau' = \frac{P}{\sqrt{2}x_1 x_2};\ \tau'' = \frac{MR}{J};\ M = P(L + \frac{X_2}{2});\ R = \sqrt{\frac{x_2^2}{4} + (\frac{x_1 + x_3}{2})^2}$$

$$J = 2\left\{\sqrt{2}x_1 x_2 \left[\frac{x_2^2}{12} + (\frac{x_1 + x_3}{2})^2\right]\right\}$$

$$\sigma(\vec{x}) = \frac{6PL}{x_4 x_3^2};\ \delta(\vec{x}) = \frac{4PL^3}{E x_3^3 x_4}$$

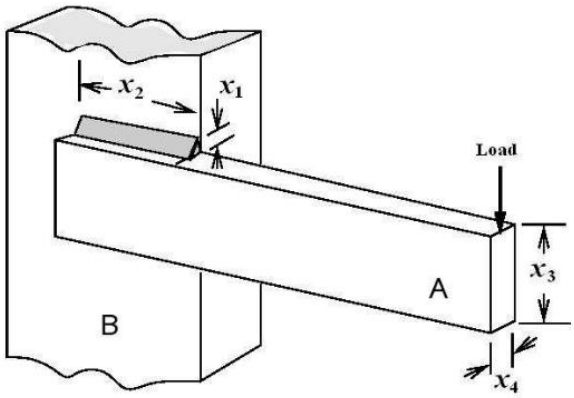
$$P_c(\vec{x}) = \frac{4.013E\sqrt{\frac{x_3^2 x_4^6}{36}}}{L^2}(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}})$$

$$x_1 \in [0.125, 5]; x_2, x_3 \in [0.1, 10]; x_4 \in [0.1, 5]$$

$c_1 = 0.10471$; $c_2 = 0.04811$; $P = 6000$; $L = 14$; $E = 30000000$; $G = 12000000$

$$\delta_{max} = 0.25;\ \tau_{max} = 13600;\ \sigma_{max} = 30000$$

APPENDIX 6: Welded Beam Figure



Source: (Cagnina et al., 2008: 323)

APPENDIX 7: Tension/Compression Spring Design Model

$$\text{Minimize } f(\vec{x}) = (x_3 + 2)x_2x_1^2$$

Subject to

$$g_1(\vec{x}) = 1 - \frac{x_2^3 x_3}{71785x_1^4} \leq 0$$

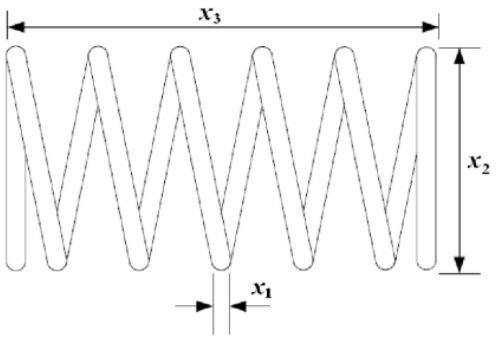
$$g_2(\vec{x}) = \frac{4x_2^2 - x_1x_2}{12566(x_2x_1^3x_1^4)} + \frac{1}{5108x_1^2} - 1 \leq 0$$

$$g_3(\vec{x}) = 1 - \frac{140.45x_1}{x_2^2x_3} \leq 0$$

$$g_4(\vec{x}) = \frac{x_1 + x_2}{1.5} - 1 \leq 0$$

$$x_1 \in [0.05, 1]; x_2 \in [0.25, 1.3]; x_3 \in [2, 15]$$

APPENDIX 8: Tension/Compression Spring Design Figure



Source: (Cagnina et al., 2008: 323)

APPENDIX 9: Combined Heat and Power Economic Dispatch Model

$$Min\ c_1(P_1) + \sum_{j=2}^{3} c_j\left(H_j, P_j\right) + c_4(H_4)$$

$$g_1(\vec{x}) = P_1 + P_2 + P_3 \leq 200$$

$$g_2(\vec{x}) = P_1 + P_2 + P_3 \geq 200$$

$$g_3(\vec{x}) = H_2 + H_3 + H_4 \leq 115$$

$$g_4(\vec{x}) = H_2 + H_3 + H_4 \geq 115$$

$$g_5(\vec{x}) = P_2 + 0.177778H_2 \leq 247$$

$$g_6(\vec{x}) = P_2 + 0.16985H_2 \geq 98.8$$

$$g_7(\vec{x}) = -P_2 + 1.781915H_2 \leq 105.74468$$

$$g_8(\vec{x}) = P_3 + 0.151163H_3 \leq 130.697$$

$$g_9(\vec{x}) = P_3 + 0.067682H_3 \geq 45.076142$$

$$g_{10}(\vec{x}) = -P_3 + 1.1584H_3 \leq 46.8812$$

$$P_1 \in [0,150], P_2 \in [81,274], P_3 \in [40,125.8]$$

$$H_2 \in [0,180], H_3 \in [0,135.6], H_4 \in [0,2695.2]$$

$$c_1(P_1) = 50P_1$$

$$c_2(H_2, P_2) = 2650 + 14.5P_2 + 0.0345P_2^2 + 4.2H_2 + 0.03H_2^2 + 0.031P_2H_2$$

$$c_3(H_3, P_3) = 1250 + 36P_3 + 0.0435P_3^2 + 0.6H_3 + 0.027H_3^2 + 0.011P_3H_3$$

$$c_4(P_4) = 23.4H_4$$