# Hybridized Probability Collectives: A Multi-Agent Approach for Global Optimization

**Zixiang Xu**

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Doctor of philosophy
in
Computer Engineering

Eastern Mediterranean University
October 2016
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Mustafa Tümer
Acting Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Doctor of Philosophy in Computer Engineering.

Prof. Dr. Işık Aybay
Chair, Department of Computer Engineering

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Doctor of Philosophy in Computer Engineering.

Asst. Prof. Dr. Ahmet Ünveren
Supervisor

Examining Committee

1. Prof. Dr. Atilla Elçi
2. Prof. Dr. Kemal Leblebicioğlu
3. Assoc. Prof. Dr. Önsen Toygar
4. Asst. Prof. Dr. Adnan Acan
5. Asst. Prof. Dr. Ahmet Ünveren

# ABSTRACT

Probability Collectives (PC) employ multiple agents to distribute sampling moves through using probability distributions over a solution space. This multi-agent systems (MAS) affords the advantage of parallel and distributed load to intelligent agents coordinated by PC for optimal search. This thesis addresses single and multi-objective hybrid learning algorithms based on probability collectives, which solve single and multi-objective global optimization problems. In the first hybrid learning model, search guided by adaptive heuristic method of Differential Evolution (DE) algorithm based on the modified PC is implemented to tackle large-scale continuous optimization problems consisting of classical and intractable single-objective functions. DE/rand/1 classical scheme maintains appropriate search directions and improve MAS's performance by adaptive vector mutation for different search regions. Two well-known benchmark problem sets, 23 classical benchmark problems and CEC2005 contest instances, were used and experimental results reveal that the presented approach is capable of integrating the collective learning methodology effectively and competitively in the proposed agent-based model.

In the second proposed approach, a PC-based multi-objective optimization algorithm is implemented using various efficient techniques and naturally promoted search operators to find the set of solutions of MOPs that achieve the best compromise with regard to the whole set of objectives. This method uses weighted aggregation technique to decompose multi-objective solutions into a single objective and created population evolves by evolutionary operators based on PC, with which the objectives are optimized in a collaborative manner. Multi-objective Evolutionary Algorithm

Based on Decomposition (MOEA/D) learns and samples probabilistic distribution from PC stochastic engine. In terms of the employment of useful information from neighbors, decomposition mechanism adopted in multi-objective optimization lumps various problems into single objective concept. PC approach is then provided with initial local search for enhancing the performance of MOEA/D framework. Additionally, a combined mutation operator of the framework is also proposed as the global optimizer to approximate the Pareto optimal set. This algorithm effectively explores the feasible search space and enhances the convergence for the true Pareto-optimal region. To validate the hybrid algorithm, the experimental study is conducted on the set of multi-objective unconstrained benchmark problems provided for CEC2009 contest, and its performance is compared with some state-of-the-art metaheuristic algorithms. In addition, the simulation results demonstrated that the proposed approach performs competitively with state-of-the-art multi-objective algorithms.

**Keywords:** Probability Collectives, Multi-agent systems, Differential Evolution, Single-objective Problem, Multi-objective Problem, MOEA/D.

# ÖZ

Olasılık Kolektifleri (OK) çözüm uzayı üzerinde olasılık dağılımları kullanarak örnek hareket için birden fazla ajan kullanır. Çoklu-ajan sistemleri (ÇAS) eniyi arama için OK tarafından koordine edilen akıllı ajanları kullanmaktadır. Bu tez, tek ve çok amaçlı genel eniyileme problemleri için OK'ye dayalı iki hibrid algoritması içermektedir. İlk hibrid öğrenme modelinde, modifiye edilmiş OK tabanlı adaptif buluşsal yöntem olan Diferansiyel Evrim (DE) algoritması tarafından yönlendirilen arama, klasik ve inatçı tek amaçlı fonksiyonlardan oluşan büyük ölçekli sürekli en iyileme problemlerini çözmek için kullanıldı. DE/rand/1 klasik yöntemi uygun arama yönlerini belirliyerek farklı arama bölgeleri için ÇAS'ın performansını adaptif vektör mütasyonu ile artırmaktadır. Önerilen ilk yöntem iyi bilinen 23 klasik problem ve CEC2005'de kullanılan problemler ile test edilmiş ve deneysel sonuçlar önerilen yöntemin mevcut yöntemler ile kıyaslanabilecek seviyede olduğunu göstermiştir.

İkinci önerilen yöntem, PC tabanlı çok amaçlı eniyileme algoritması olup çeşitli etkili teknikler ve arama operatörleri yardımıyle çok amaçlı eniyileme problemleri için çözüm setleri bulmaktadır. Bu yöntem, ağırlıklı toplama tekniğini kullanarak çok amaçlı çözümleri tek amaçlı çözüm haline getirip elde edilen nüfusu OK tabanlı evrimsel operatörler yardımıyle geliştirir ve ortak bir şekilde eniyilenmiş olur. Çok amaçlı ayrıştırmaya dayalı Evrimsel Algoritma  (MOEA/D) OK'dan olasılık dağılımını öğrenir ve örnekler. Komşulardan yararlı bilgilerin elde edilmesi amacıyle çok amaçlı en iyileme içine uyarlanan ayrışma mekanizması bazı problemleri tek amaçlı sistemde toplar. PC, MOEA/D için yerel bir algoritma olarak çalışır ve ilk sonuçları oluşturur. Ayrıca birleştirilmiş mutasyon operatörü MOEA/D'nin genel eniyi sonuçlara ulaşmasında yardımcı olur. Bu algoritma etkili bir şekilde uygulanabilir arama alanını araştırır ve gerçek en iyi bölge için yakınsamayı artırır. Önerilen metot, CEC2009 için verilen kısıtsız çok amaçlı problemler ile test edildi ve sonuçlar iyi bilinen metasezgisel algoritmalar ile karşılaştırıldı. Elde edilen sonuçlar mevcut sonuçlar ile rekabet edebilecek seviyede olduğu gösterilmiştir.

**Anahtar Kelimeler:** Olasılık Kollektifleri, Çoklu-ajan sistemleri, Diferansiyel Evrim, Tek amaçlı Problemler, Çok amaçlı Problemler, MOEA/D.

# ACKNOWLEDGMENT

First and foremost, I would like to express my special appreciation and thanks to my supervisor Asst. Prof. Dr. Ahmet Ünveren for his support and inspiration in the preparation of the thesis. He has been a tremendous mentor for me. I would like to thank him for continuous advice and for encouraging throughout the course of this thesis. His ideas on both researches as well as on my career have been invaluable.

I would also like to thank my committee members for serving as my committee members even at hardship. I owe a debt of gratitude to Asst. Prof. Dr. Adnan Acan for his time and careful attention to detail. I thank him for his untiring support and guidance throughout my work. I would also like to express my gratitude to Assoc. Prof. Dr. Önsen Toygar for her support, guidance and insightful comments on my thesis during the long process of evaluating the program in the field.

Most importantly, I take this opportunity to express the profound gratitude from my deep heart to my beloved parents, who have always been a ray of hope and a fountain of new experiences and insights for the thesis, for the research part, and for me.

# TABLE OF CONTENTS

ABSTRACT ........ iii

ÖZ……… ........ v

ACKNOWLEDGEMENT ........ vii

LIST OF TABLES ........ x

LIST OF FIGURES ........ xii

LIST OF SYMBOLS/ABBREVIATIONS ........ xiii

1 INTRODUCTION ........ 1

  1.1 General Description of PC ........ 2

  1.2 Literature Review ........ 3

  1.3 Summary of the Proposed Works ........ 5

  1.4 Thesis Overview ........ 6

2 PROBLEM DEFINITIONS ........ 8

  2.1 Single Objective Optimization Problem ........ 8

  2.2 Multi-objective Optimization Problem ........ 9

3 PROBABILITY COLLECTIVES ........ 13

  3.1 Single Objective PC ........ 13

    3.1.1 Detailed PC algorithm ........ 13

  3.2 Multi-Objective PC ........ 23

4 PROBABILITY COLLECTIVES FOR SINGLE OBJECTIVE OPTIMIZATION PROBLEMS ........ 25

  4.1 Introduction ........ 25

  4.2 Differential Evolution ........ 26

    4.2.1 Description of the DE Algorithm ........ 26

4.2.2 Operation of the DE Algorithm ..... 28
4.3 Cross Entropy Method ..... 31
4.4 A Hybrid Proposed Approach for SOP (MPCDE) ..... 33
4.5 Experimental Studies ..... 37
4.5.1 Experimental Setting ..... 37
4.5.2 Problem Categories ..... 38
4.5.3 First Set of Benchmark Problems ..... 39
4.5.4 CEC2005 Benchmark Problems ..... 52
5 PROBABILITY COLLECTIVES FOR MULTI-OBJECTIVE OPTIMIZATION PROBLEMS ..... 59
5.1 Introduction ..... 59
5.2 MOEAs ..... 60
5.3 MOEA/D ..... 63
5.3.1 Decomposition of Multiobjective Optimization ..... 65
5.3.2 Tournament Selection ..... 66
5.4 The Proposed Approach for MOP (MOEA/D-PC) ..... 67
5.4.1 Decomposition of Multi-objective Optimization ..... 70
5.4.2 MOEA/D-PC Framework ..... 71
5.4.3 PC Algorithm in MOEA/D Framework ..... 75
5.4.4 Combined Mutation Operator ..... 76
5.5 Experimental Studies ..... 77
6 CONCLUSIONS AND FUTURE PERSPECTIVES ..... 86
REFERENCES ..... 91

# LIST OF TABLES

Table 4.1: Features of 23 classical benchmarks .................................................... 40

Table 4.2: Description (definition, search domain and functional value of minima) of 23 classical benchmark test functions.................................................................. 41

Table 4.3: Maximum number of function evaluations as introduced by Yao et al. .. 42

Table 4.4: Comparative results of mean (in first row for every function) and standard deviation (in next row) derived from algorithm of PC, MPCDE, TSM-GDA and FEP for the set of classical benchmark problems over 50 runs ...................................... 44

Table 4.5: Comparative results of the *mean* (in first row for each function), and the *standard deviation* (in new row) derived from algorithm of ABC, DE, TSM-GDA, SEA, PSO, and MPCDE within 5.0e+5 FEs over 30 runs...................................... 48

Table 4.6: Results of statistical tests between MPCDE and its adversaries for each function created by the Wilcoxon tool population: (p,h)........................................ 50

Table 4.7: The Description (Name, Domain, Optimum, Type, Rotated (R), Separable (Se), and Scalable (Sc)) of the CEC 2005 benchmark functions (Y: Yes; N: No), U: unimodal, BM: Basic multimodal, HC: Hybrid Composition................................. 53

Table 4.8: Comparative results of *mean* values among 14 and MPCDE Algorithms for CEC2005 test benchmarks with 10-dimension of 1.0e+5 FEs over 25 runs....... 55

Table 4.9: Comparative results of *mean* values among 11and MPCDE Algorithm for CEC2005 test benchmarks with 30-dimension of 3.0e+5 FEs over 25 runs ............ 57

Table 4.10: Comparative results of *mean* values between L-CMA-ES, G-CMA-ES, MPCDE and TSM-GDA Algorithm for CEC2005 test benchmarks with 50-dimension of 5.0e+5 FEs over 25 runs ................................................................. 58

Table 5.1: Description (Name, number of Objective, Dimension, Domain, Geometry, Modality and Separability) of CEC2009 Benchmark Multi-objective Test Problems with M number of objective, D number of decision variables, scalable S, separable SP, nonseparable NS ........................................................................................ 78

Table 5.2: IGD values obtained with MOEA/D-PC and MOEA/D for 10 unconstrained test problems (UF1−UF10) in 30 independent runs.......................... 81

Table 5.3: Comparison of mean IGD values and the standard deviation (Std) between 10 algorithms and MOEA/D-PC on unconstrained CEC2009 benchmark test functions (UF1−UF10) in 30 independent runs with 300,000 FEs .......................... 82

# LIST OF FIGURES

Figure 2.1: Global vs local minima ........................................................................ 9

Figure 2.2: A basic notation for a multi-objective optimization ............................. 10

Figure 2.3: A graphical interpretation of the Pareto dominance ............................. 12

Figure 3.1: Unconstrained PC Flowchart............................................................... 14

Figure 3.2: Optimization space conversion with Homotopy function...................... 17

Figure 3.3: Probability distribution of agents......................................................... 20

Figure 4.1: Main phases of the DE algorithm cycle ............................................... 27

Figure 4.2: Mutation using DE/rand/1 .................................................................... 29

Figure 4.3: Flowchart of the proposed hybrid MPCDE.......................................... 34

Figure 4.4: Evolution plots in comparisons with PC, FEP, and MPCDE, PC on (a) Rosenbrock and (b) Ackley Function ................................................................... 46

Figure 4.5: Evolutionary convergence of averaging optimal results yielded in ABC, DE, PSO, SEA and MPCDE with FEs for 6 problems............................................ 51

Figure 5.1: Convex and concave problems ............................................................ 66

Figure 5.2: The general structure of connections between MOP and SOP.............. 68

Figure 5.3(a): The final approximation Pareto front with lowest IGD value in the objective space for CEC2009 ($UF1 - UF6$) test problem....................................... 84

Figure 5.3(b): The final approximation Pareto front with lowest IGD value in the objective space for CEC2009 ($UF7 - UF10$) test problem..................................... 85

# LIST OF SYMBOLS/ABBREVIATIONS

| | |
|---|---|
| ABC | Artificial Bee Colony |
| AMGA | Archive-based Micro Genetic Algorithm |
| BFGS | Broyden-Fletcher-Goldfarb-Shanno |
| CE | Cross Entropy |
| COIN | Collective Intelligence |
| CR | Crossover |
| DE | Differential Evolution |
| DERL | Differential Evolution with random localization |
| DM | Decision maker |
| DRA | Dynamical Resource Allocation |
| EAs | Evolutionary Algorithms |
| EP | Evolutionary Programming |
| EP | External Population |
| FEP | Fast Evolutionary Programming |
| FEs | Function Evaluations |
| GA | Genetic Algorithms |
| GDE | Generalized Differential Evolution |
| HV | Hypervolume |
| HypE | Hypervolume Estimation |
| IBEA | Indicator-based evolutionary algorithm |
| IGD | Inverted Generational Distance |
| MAS | Multi-Agent Systems |
| Max_FEs | Maximum number of Function Evaluations |

| | |
|---|---|
| MO-ABC/DE | Multi-Objective ABC with DE |
| MODE | Multi-Objective Differential Evolution |
| MOEA/D | Multi-Objective Evolutionary Algorithm Based on Decomposition |
| MOEA/D-DE | MOEA/D with Differential Evolution |
| MOEA/D-PC | Probability Collectives MOEA/D |
| MOEADGM | MOEA/D Guided Mutation |
| MOEAs | Multi-Objective Evolutionary Algorithms |
| MOP | Multiple-Objective Problem |
| MOPC | Multi-Objective Probability Collectives |
| MOPC/D | MOPC based on decomposition |
| MOSaDE | Multi-objective Self-adaptive Differential Evolution |
| MPCDE | Modified Probability Collectives and Differential Evolution |
| MSOPS | Multiple Single Objective Pareto Sampling |
| NSGA-II | Non-Dominated Sorting Genetic Algorithm II |
| OWMOSaDE | Objective-Wise MOSaDE |
| PC | Probability Collectives |
| PDF | Probability Density Function |
| PF | Pareto Front |
| P-MOEA/D | Parallel MOEA/D |
| PS | Pareto Set |
| PSO | Particle Swarm Optimization |
| RCGA | Real-Coded Genetic Algorithm |
| SaDE | Self adaptive Differential Evolution |
| SaDESA | Self-Adaptive Differential Evolution with Simulated Annealing |
| SaGPDE | Self-Adaptive Genetically Programmed Differential Evolution |

| | |
|---|---|
| SBX | Simulated Binary Crossover |
| SEA | Simple Evolutionary Algorithms |
| SMPSO | Speed-constrained Multi-objective PSO |
| SOP | Single Objective Problem |
| SPEA2 | Strength Pareto evolutionary algorithm 2 |
| TSM-GDA | Two-Stage Memory of Great Deluge Algorithm |
| VEGA | Vector evaluated genetic algorithm |

# Chapter 1

# INTRODUCTION

A large number of real-world engineering problems in operations research and engineering field are defined as optimization problems. Multi-agent systems (MAS) involve a number of agents and their environment in which the agents are used to perform potential tasks for achieving the possible goals and satisfying inter-agent constraints. Any action by an agent may affect the further decisions by other agents and so on.

MAS and the Evolutionary algorithms (EAs) [1] are two independent, well-known problem-solving paradigms with different characteristics that are usually applied to slightly different problem domains. Recently, these two paradigms have been combined, resulting in agent-based evolutionary algorithms (AEA) that have improved problem-solving in both domains. Evolutionary algorithms such as Differential Evolution (DE) [2], and Genetic Algorithms (GA) [3], are successful stochastic optimziation methods for the solution of global optimization problems in faster, reliable and easier way. The aim of these algorithms is to locate a superior solution, which explores the minimum/maximum value of a single objective problem (SOP). However, in the real-world optimization problems, there exists more than one objective, each of which may have distinct optimal solutions. Thus they are called multi-objective problems (MOPs). These objectives are generally conflicting and competing with each other, it is no longer possible to find the single solution which is superior to all others in every objective function. This thesis attempts to utilize and combine distinct techniques within the probability collective (PC) [4] framework to cope with the classical and recently published benchmark MOPs.

## 1.1 General Description of PC

The framework of Collective Intelligence (COIN) involves a great deal of approaches to construct a collective that consists of adaptive intelligent agents according to system-level acceptance rules [5]. Probability collectives extends structure of COIN paradigm to model, progress and control of distributed approaches, using inspirations from closely connected fields of mathematics, engineering, and optimization [6]. It uses the distributed MAS as a tool for estimating the joint probability space and updating the probability distributions for strategies, which leads the optimization of system objectives across the evolving distributions. In particular, PC method deals with the assigned strategies in the design space as individual agents being acted as intelligent components iteratively [7].

## 1.2 Literature Review

Under the framework of COIN, Dr. David Wolpert firstly proposed PC in a technical report presented to NASA [5], and used distributions to update individuals for the final aim. Except for local utilities of individual agents, world utility is introduced by Lee and Wolpert to decrease the sample sizes [8]. PC algorithms are able to solve different problems such as unconstrained [9, 10] and constrained optimization problems [11, 12]. Sequentially updated PC, proposed by Smyrnakis et al. [13], applies approximate regression to estimate the expected utility from sampling areas of continuous actions by Sequential Monte Carlo algorithm. Bieniawski used the data aging method to effectively diminish the sample sizes [14]. Afterwards, the sampled strategies from PC have been updated with a set of variables in the predefined limits by Kulkarni et al. [15, 16]. For modifying the probability distributions, Wolpert et al. discussed several mathematical equations, for example, the Nearest Newton Descent method [17].

PC has also been shown to be effective for the solution of various complex problems, for example, the aircraft assignment problem [7], Truss Structure Problems [12], single-depot multiple traveling salesmen problem [15, 18], vehicle routing problems [15, 18], and aircraft weapon delivery trajectory problem [19]. A majority of PC approach has emerged to improve a number of single objective problems with discrete, continuous and mixed variables [12, 15, 18, 20, 21].

Recently, PC algorithms are extended for solving MOPs. Waldock et al. first successfully addressed a multi-objective PC framework (MOPC) [22], which is adopted a max-min function and Pareto-based ranking strategy to carry out a number of single objectives for the PC strategy in multi-objective optimisation. Also the probability distributions is guided towards non-dominated solutions. Morgan et al. then developed a MOPC based on decomposition (MOPC/D) [23] to exploit the search operators within a probabilistic Gaussian mixture model.

Additionally, this thesis mainly involves the hybridization of PC and DE algorithms since it has been proven that DE was efficiently integrated with other evolutionary methods [24, 25]. By its evolving operators, a number of DE variants have been proposed for the puprpose of adapting the DE parameters dynamically during its execution. For instance, DE mutation by Cauchy [26], alternative memory of adaptive DE [27], DE with global and local neighborhood [28], Mixed mutation strategy based DE [29], Self adaptive DE (SaDE) [30] and DE with self-adaptive control parameter [31], and DE with random localization (DERL) [32] are some of the widely referred studies in literature.

Over the past decades, extension of DE algorithm for multi-objective optimization has received grown interest in applications of real-world problems. Babu et al. proposed a multi-objective differential evolution algorithm (MODE) [33], and solved two problems by using a penalty function and a weighting factor. Then, Kukkonen and Lampinen addressed the selection criterion for the first version of the Generalized Differential Evolution (GDE) [34]. And the third version of GDE (GDE3) [35] was developed with constrained non-dominance sorting and crowdedness to choose the best solution candidates for the purpose of reducing the population size. Besides, Huang et al. proposed a Multi-objective Self-adaptive Differential Evolution (MOSaDE) [36], which was adaptively controlled by the parameter settings and associated objective-wise learning techniques to improve its performance.

## 1.3 Summary of the Proposed Works

In this thesis, hybrid optimization methods based on PC approach were developed for large scale single and multi-objective optimization problems. In the first proposed algorithm, an adaptive technique was introduced which combines a global search engine with PC optimizer. In fact, PC approach conducts preliminary random search in the solution space by updating its probability distributions. Then obtained favorable solutions are reproduced by using Cross Entropy (CE) method [37]. New population obtained from CE is submitted to modified DE algorithm for a further search in the global space to extract the best solution. In effect, the proposed approach iteratively updates its learning parameters for achieving the best possible performance. The experimental studies validated the efficiency of the proposed hybrid algorithm over a set of classical benchmark problems.

In this thesis, the second proposed algorithm hybridize MOEA/D algorithm with conventional PC for the solution of MOPs. Also, for the purpose of speeding up the convergence, two mutation operators are designed and used with the CE method.

## 1.4 Thesis Overview

Chapter 1 starts with an overview of single and multiple objective optimization approaches, which are supported by presenting a comprehensive review of scientific literatures and publications related to the PC and DE algorithms for single and multiple objective optimization problems. Following the short introduction and general PC concept, two hybrid algorithms are summarized for the large scale single objective problems and multi-objective unconstrained optimization problems. Chapter 2 generally defines the single and multi-objective optimization problems in the form of the mathematical formulations. The fundamental concepts of multi-objective optimization are also described with the Pareto dominance concept.

A detailed review of the classical PC algorithm is presented in Chapter 3. The concepts and techniques used in PC are highlighted with its flowchart. The first novel hybrid algorithm for single objective problems are presented in Chapter 4, where PC finds favorable solutions and the CE, DE algorithms then update promising points with corresponding search operators for two different sets of problems. Meanwhile, the associated background and the differential evolution algorithm are shown with its flowchart. Three remarkable operators of mutation, recombination, and selection are described in DE algorithm. Experimental evaluations and comparisons with a state-of-the-art methods are carried out using two classical sets of benchmark instances. In addition, experimental results show that that the use adaptive mutation in DE significantly improved the search capability of the proposed method.

Chapter 5 describes the second hybrid approach for multi-objective problems and depicts the population-based search of EAs that can be used to converge to the set of best trade-off solutions. The subsections presents three categories of multi-objective evolutionary algorithms (MOEAs) [38] with corresponding published literatures and provides description of multi-objective evolutionary algorithm based on decomposition (MOEA/D) [39] framework. Techniques shown in this chapter aggregate all objectives to a set of individual solutions that provide the way for the PC. This chapter introduces the new mutation scheme in the MOEA/D framework that converges to the Pareto Front (PF) in a single optimization run. The statistical results and the corresponding convergence figures are shown at the end of this chapter.

Finally, Chapter 6 presents the conclusions associated with proposed hybrid approaches for single and multi-objective optimization problems based on PC method. This chapter also covers an outlook on the future perspectives of research work.

# Chapter 2

# PROBLEM DEFINITIONS

## 2.1 Single Objective Optimization Problems

A generic single objective optimization problem can be defined as

$$\text{minimize } f(x) \tag{2.1}$$

$$\text{subject to } g_i(x) \leq 0, \quad i = \{1, 2, \ldots, m\}, \tag{2.2}$$

$$h_j(x) = 0, \quad j = \{1, 2, \ldots, p\}. \tag{2.3}$$

Extraction of a solution that minimizes the scalar function $f(x)$, where $x$ is a $D$-dimensional decision variable vector $x = (x_1, \ldots, x_D)$ from some universe $\Omega \in \Re^D$, is the fundamental task of any optimization algorithm designed to solve this problem [21]. Functional expressions $g_i(x)$ and $h_j(x)$ represent constraints that must be fulfilled while optimizing $f(x)$ and $\Omega$ contains all possible $x$'s that can be used for the evaluation of $f(x)$ and its constraints. The function $f : \Omega \subseteq \Re^D \rightarrow \Re$, $\Omega \neq \emptyset$, is a real valued objective function. The variable vectors $x$ that result in the smallest objective function value is referred to as the minimizers, which are further classified as:

**Local minimizer**: A point $x^* \in \Omega$ is a local minimizer of $f$ if there exist some $\varepsilon > 0$ such that, $f(x) \geq f(x^*)$, $\forall x \in \Omega \backslash \{x^*\}$ and $\|x - x^*\| < \varepsilon$, where $f(x^*)$ is a local minimum.

**Global minimizer**: A point $x^* \in \Omega$ is a global minimizer of $f$ if $f(x) \geq f(x^*)$, $\forall x \in \Omega \backslash \{x^*\}$, where $f(x^*)$ is known as a global minimum. The difference between a local and a global minimum is presented in Figure 2.1 [21].

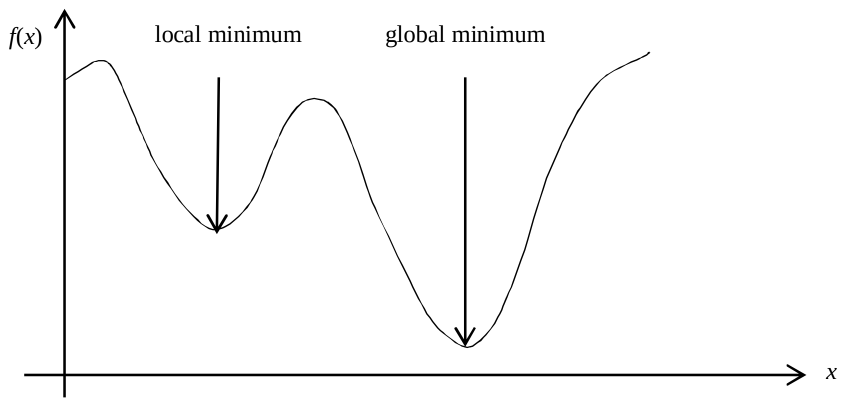


Figure 2.1: Global vs local minima

In general, global minimizers are difficult to locate and verify, especially when the search algorithm gets trapped in local minima. The task of locating the global optimum is referred to as global optimization. In some cases, global optimization may extract many local minima during the course of its execution.

## 2.2 Multi-objective Optimization Problems

In multi-objective optimization, one attempts to simultaneously maximize or minimize $M$ objectives, $F(x)$ while satisfying $J$ inequality constraints, $g_i$ and $P$ equality constraints, $h_j$, which are functions of decision variable vector, $x = (x_1,\ldots,x_D)^T \in X$: $F_i = f_i(x)$, $i = 1,\cdots, M$. While problems exist for which the decision vectors are discrete, this study of MOPs is concerned with problems for which each decision variable, $x$, is continuous, between a lower bound $x_i^L$ and an upper bound $x_i^H$, where $\Re^D$ is the $D$-dimensional decision space, $F$: $x \rightarrow \Re^D$ is $M$-dimensional objective space and $X$ is called decision space. Both decision and objective spaces are real spaces, as they connect to continuous variables and objectives for the proposed approach. The mapping process from decision space $X$ to objective space $F$ is shown as Figure 2.2.

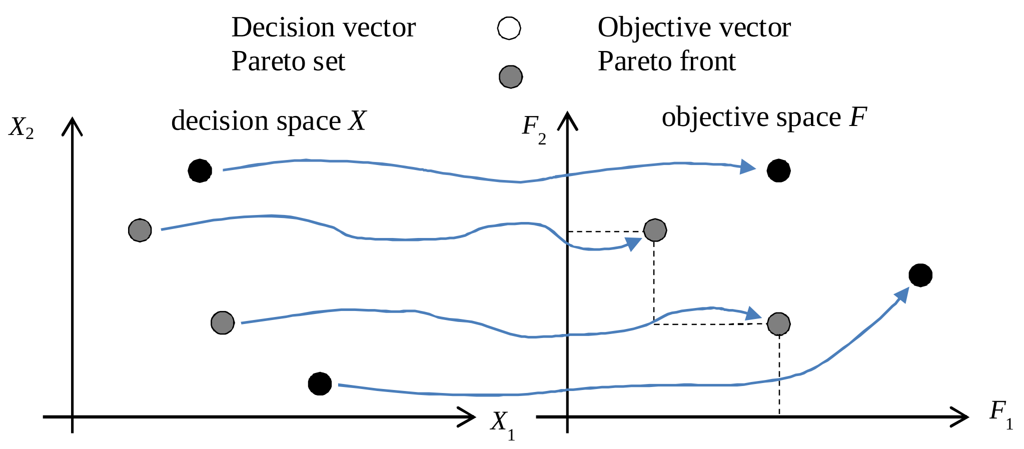


Figure 2.2: An example of mapping between decision space and objective space for a 2-objective MOP

Without loss of generality, it can be assumed that the objectives are to be minimized, as maximization of $F_i$ is equivalent to the minimization of $1/F_i$ or $-F_i$ and that constraints are of the ‘greater than or equal to’ form; Such multi-objective optimization problem can be formally expressed as follows:

$$\text{Minimize } F(x) = (f_1(x), f_2(x), \ldots, f_M(x))^D, \tag{2.4}$$

$$\text{Subject to } g_i(x) \geq 0; \quad i = 1, 2, \cdots, b \tag{2.5}$$

$$h_j(x) = 0; \; j = 1, 2, \cdots, p \tag{2.6}$$

Note that $p$, the number of equality constraints, must be less than dimension $D$ to provide sufficient degrees of freedom left for optimization. The constraints given in Equation (2.5) and (2.6) define the solution space which contains the set of all feasible solutions. In the interest of simplicity, those constraints are not considered in this thesis.

**Pareto Dominance**: For any two decision vectors $a$ and $b$,

- $a \succ b$ ($a$ dominates $b$), *iff* $\forall i : f_i(a) \leq f_i(b) \wedge \exists\, i : f_i(a) < f_i(b)$;
- $a \succcurlyeq b$ ($a$ weakly dominates $b$), *iff* $\forall i : f_i(a) \leq f_i(b)$;

- $a \sim b$ ($a$ is indifferent to $b$), *iff* $\exists\, i : f_i(a) < f_i(b) \wedge \exists\, j : f_j(a) > f_j(b)$.
- The definitions of the opposite binary relations ($\prec$, $\preccurlyeq$, $\sim$) are analogical [40].

Following the Pareto concept of optimality, there exists a feasible point $x^1 \in X$ that dominates a point $x^2 \in X$ if $f(x^1) \leq f(x^2)$. If strict inequality holds for all $M$ objectives, i.e. $f(x^1) < f(x^2)$, then $x^1$ strongly dominates $x^2$. If there does not exist any feasible point that dominates $x \in X$, we say that $x$ is an efficient solution. For the case that there exists no feasible point that strongly dominates $x \in X$, we say that $x$ is weakly-efficient. If there is no $x^2 \in X$, $x^2 \neq x^1$, such that $f(x^2) \leqq f(x^1)$, $x^1$ is called strictly efficient. Accordingly, the efficient set $X_E$ and the weakly efficient set $X_{wE}$ are defined as,

$$X_E \coloneqq \{\, x \in X : \text{there is no } \bar{x} \in X \text{ with } f(\bar{x}) \leq f(x)\},$$

$$X_{wE} \coloneqq \{\, x \in X : \text{there is no } \bar{x} \in X \text{ with } f(\bar{x}) < f(x)\}.$$

For the given set of points in the decision space, the points located on the non-domination front do not get dominated by any other point in the objective space, hence those points are called Pareto-optimal solutions (non-dominated solutions).

A solution is clearly better than (dominating) another solution, if it is better or equal in all objectives, but at least better in one objective. By discarding all solutions that are dominated by at least one other solution, the set of best compromise solutions can approximate the set of Pareto optimal solutions. The remaining solutions are all of equal quality, where the set of indifferent solutions is called as the Pareto set (PS). The Pareto dominance relation is illustrated for the two-objective case in Figure 2.3 [40].

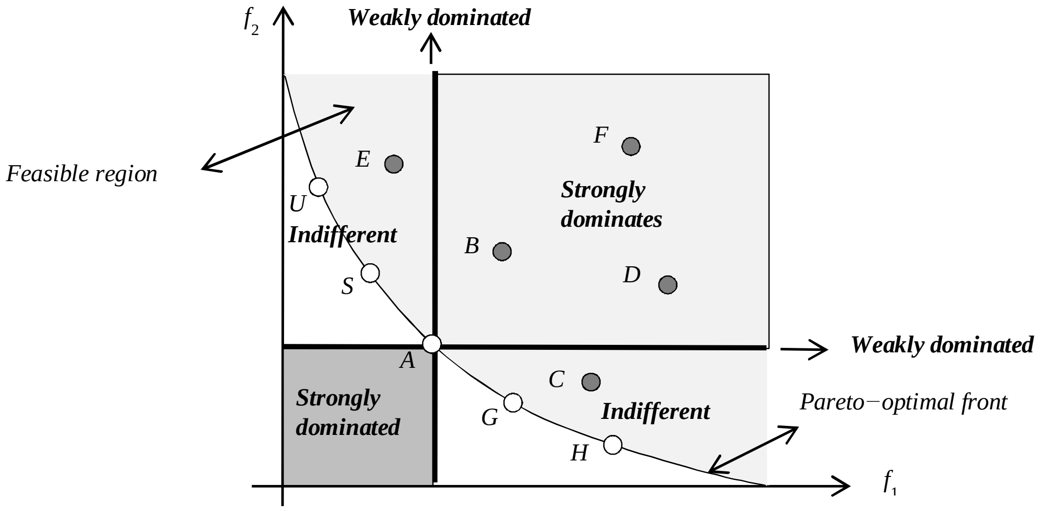


Figure 2.3: A graphical interpretation of the Pareto dominance

The Pareto dominance relations between solutions in a two-objective example are further represented in Figure 2.3, where five nondominated solutions at points *A*, *G*, *H*, *S*, and *U* are on Pareto front. This figure depicts a partial ordering among different solutions based on the dominance criterion. Those solutions selected may yield the possible trade-offs among competing objectives. From this figure, the objective solutions space is divided into four main blocks (light grey, dark grey, and two others) based on the dominance relations. The reference point *A* is better in both objectives, this solution thus strongly dominates solutions lying in the light grey block. On the contrary, point *A* is strongly dominated by solutions of the dark block because those solutions have better objective values than point *A*. For solutions that lying in the boundaries of the shaded blocks, they share the equal objective values in one of the objectives as point *A*, however, point *A* has a better objective value in another objective. Hence, those solutions are weakly dominated by point *A*. For solutions located in the rest of two blocks, they are inferior in one of the objective functions; meanwhile, they are superior in another objective function compared to point *A*. Consequently, these solutions are indifferent to point *A*.

# Chapter 3

# PROBABILITY COLLECTIVES

## 3.1 PC approach

Probability collectives is a single objective optimization solver across the probability distributions. Using the ability of MASs, each agent performs random sampling of individuals along with its probability distributions, which can be iteratively updated to make optimization decisions on their alternative actions [11]. The decisions are done by a set of PC strategies (variables) over updating probability distributions within a number of iterations that is different with other metaheuristic random optimization approaches. Consequently, the procedure indirectly provides a fitness landscape with the promising strategies from the highest probability. Hence, the solutions have tightly relationship between search indicators and probability distributions.

### 3.1.1 Detailed PC Algorithm

As mentioned before, Chapter 2 defines a single objective minimization problem. Let's consider an unconstrained optimization problem $G$: $\Omega \subset \Re^D \rightarrow \Re$ that is a real-valued objective function to be minimized in solution space $\Omega$ with $D$-dimensional variable (agent) vector $x^*$=[ $x_1, \ldots, x_D$ ] $\in \Re^D$. The detailed PC procedure is explained below over the flowchart given in Figure 3.1.

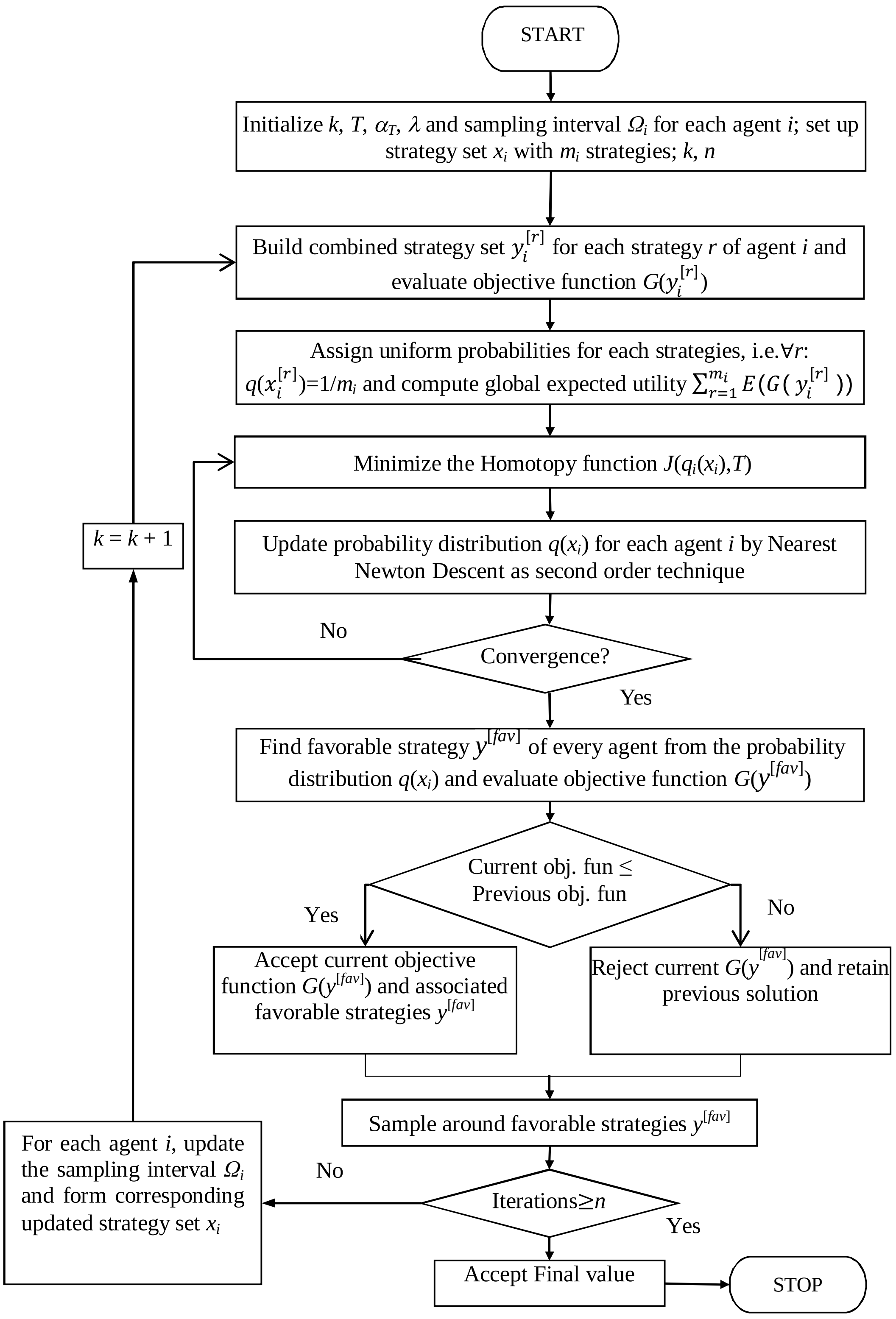

START
Initialize $k$, $T$, $\alpha_T$, $\lambda$ and sampling interval $\Omega_i$ for each agent $i$; set up strategy set $x_i$ with $m_i$ strategies; $k$, $n$
Build combined strategy set $y_i^{[r]}$ for each strategy $r$ of agent $i$ and evaluate objective function $G(y_i^{[r]})$
Assign uniform probabilities for each strategies, i.e.$\forall r$: $q(x_i^{[r]})=1/m_i$ and compute global expected utility $\sum_{r=1}^{m_i} E(G(y_i^{[r]}))$
Minimize the Homotopy function $J(q_i(x_i),T)$
Update probability distribution $q(x_i)$ for each agent $i$ by Nearest Newton Descent as second order technique
Convergence?
No
Yes
$k = k + 1$
Find favorable strategy $y^{[fav]}$ of every agent from the probability distribution $q(x_i)$ and evaluate objective function $G(y^{[fav]})$
Current obj. fun ≤ Previous obj. fun
Yes
No
Accept current objective function $G(y^{[fav]})$ and associated favorable strategies $y^{[fav]}$
Reject current $G(y^{[fav]})$ and retain previous solution
Sample around favorable strategies $y^{[fav]}$
For each agent $i$, update the sampling interval $\Omega_i$ and form corresponding updated strategy set $x_i$
No
Iterations≥$n$
Yes
Accept Final value
STOP

Figure 3.1: PC Flowchart for unconstrained optimization

Each agent $i$ performs sampling of its variables iteratively within its predefined range denoted as $\Omega_i \in [\Omega_i^L, \Omega_i^H]$ and it may modify its lower limit $\Omega_i^L$ and its upper limit $\Omega_i^H$ of the interval $\Omega_i$ iteratively as the procedure runs. The procedural explanations in the notations described in this part can be found similar to the ones in [21]. Strategy updates, also called as moves, are carried out by every agent $i$ resulting a set of strategies $x_i$ representing its variables from its associated probability distributions by agent $i$: $x_i = \{x_i^{[1]}, x_i^{[2]}, x_i^{[3]}, ..., x_i^{[m_i]}\}$, $i$ = 1, 2,…,$D$ and every agent is supposed to have a same number of strategies, i.e. $m_1 = m_2 = \dots = m_i = \dots = m_{D\text{-}1} = m_D$. Each sample $x_i^{[r]}$, $1 \leq r \leq m_i$, is a random value created from $\Omega_i \in [x_i^L, x_i^H]$ over the probability distribution $q(x_i)$ corresponding to agent $i$. Thus, the parameters of probability distributions have a large impact on the efficiency of the strategy set of each agent $i$. Initially, each agent $i$ forms $m_i$ set of solution in combination with the other strategies as $y_i^{[j]} = \{x_1^{[?]}, x_2^{[?]}, ..., x_i^{[j]}, ..., x_{D-1}^{[?]}, x_D^{[?]}\}$, $1 \leq j \leq m$. The superscript [?] of $x_r^{[?]}$, $r \neq i$, means that agent $i$ randomly samples from another agent $r$. These newly built variables form a set of solutions including $m_i$ random strategy values for $N$ agents. The agent $i$ is used to change its strategy set with the solution $j$ by a selection rule to discard old value or accept new strategy within its domains. Hence, each agent $i$ establishes the $m_i$ strategy set shown as:

$$\begin{aligned}
y_i^{[1]} &= \{x_1^{[?]}, x_2^{[?]}, ..., x_i^{[1]}, ..., x_{D-1}^{[?]}, x_D^{[?]}\}, \\
y_i^{[2]} &= \{x_1^{[?]}, x_2^{[?]}, ..., x_i^{[2]}, ..., x_{D-1}^{[?]}, x_D^{[?]}\}, \\
y_i^{[3]} &= \{x_1^{[?]}, x_2^{[?]}, ..., x_i^{[3]}, ..., x_{D-1}^{[?]}, x_D^{[?]}\}, \\
&\vdots \\
y_i^{[r]} &= \{x_1^{[?]}, x_2^{[?]}, ..., x_i^{[r]}, ..., x_{D-1}^{[?]}, x_D^{[?]}\}, \\
&\vdots \\
y_i^{[m_i]} &= \{x_1^{[?]}, x_2^{[?]}, ..., x_i^{[m_i]}, ..., x_{D-1}^{[?]}, x_D^{[?]}\}.
\end{aligned} \tag{3.1}$$

Similarly, all the remaining agents form their combined strategy sets. Since all agents generated a number of random strategy values, they can calculate their fitness values to be optimized. In other words, $i$-th agent evaluates $m_i$ objective functions for its solutions as $\{G(y_i^{[1]}), G(y_i^{[2]}), \dots G(y_i^{[r]}), \dots, G(y_i^{[m_i]})\}$. To minimize the problem, every agent attempt to find the best possible solution from these object functions. Each agent $i$ accumulates its fitness across its strategy set to be optimized as $\sum_{r=1}^{m_i} G(y_i^{[r]})$. The combined strategy sets, involving the associated objective functions and the collection of system objectives for $D$ number of agents, are presented in Equation (3.2) as follows.

$$
\begin{array}{l}
\left.\begin{array}{l}
y_1^{[1]} = \{x_1^{[1]}, x_2^{[?]}, \dots, x_i^{[?]}, \dots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_1^{[1]}) \\
y_1^{[2]} = \{x_1^{[2]}, x_2^{[?]}, \dots, x_i^{[?]}, \dots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_1^{[2]}) \\
\qquad\qquad\qquad\qquad \vdots \\
y_1^{[r]} = \{x_1^{[r]}, x_2^{[?]}, \dots, x_i^{[?]}, \dots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_1^{[r]}) \\
\qquad\qquad\qquad\qquad \vdots \\
y_i^{[m_i]} = \{x_1^{[m_i]}, x_2^{[?]}, \dots, x_i^{[?]}, \dots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_1^{[m_i]})
\end{array}\right\} \Rightarrow \sum_{r=1}^{m_i} G(y_1^{[r]}) \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
\left.\begin{array}{l}
y_i^{[1]} = \{x_1^{[?]}, x_2^{[?]}, \dots, x_i^{[1]}, \dots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_i^{[1]}) \\
y_i^{[2]} = \{x_1^{[?]}, x_2^{[?]}, \dots, x_i^{[2]}, \dots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_i^{[2]}) \\
\qquad\qquad\qquad\qquad \vdots \\
y_i^{[r]} = \{x_1^{[?]}, x_2^{[?]}, \dots, x_i^{[r]}, \dots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_i^{[r]}) \\
\qquad\qquad\qquad\qquad \vdots \\
y_i^{[m_i]} = \{x_1^{[?]}, x_2^{[?]}, \dots, x_i^{[m_i]}, \dots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_1^{[m_i]})
\end{array}\right\} \Rightarrow \sum_{r=1}^{m_i} G(y_i^{[r]}) \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
\left.\begin{array}{l}
y_D^{[1]} = \{x_1^{[?]}, x_2^{[?]}, \dots, x_i^{[?]}, \dots, x_{D-1}^{[?]}, x_D^{[1]}\} \Rightarrow G(y_D^{[1]}) \\
y_D^{[2]} = \{x_1^{[?]}, x_2^{[?]}, \dots, x_i^{[?]}, \dots, x_{D-1}^{[?]}, x_D^{[2]}\} \Rightarrow G(y_D^{[2]}) \\
\qquad\qquad\qquad\qquad \vdots \\
y_D^{[r]} = \{x_1^{[?]}, x_2^{[?]}, \dots, x_i^{[?]}, \dots, x_{D-1}^{[?]}, x_D^{[r]}\} \Rightarrow G(y_D^{[r]}) \\
\qquad\qquad\qquad\qquad \vdots \\
y_D^{[m_i]} = \{x_1^{[?]}, x_2^{[?]}, \dots, x_i^{[?]}, \dots, x_{D-1}^{[?]}, x_D^{[m_i]}\} \Rightarrow G(y_D^{[m_i]})
\end{array}\right\} \Rightarrow \sum_{r=1}^{m_i} G(y_i^{[r]})
\end{array}
\qquad (3.2)
$$

Consequently, the set of objective functions of minimization problem becomes $\{\sum_{r=1}^{m_1} G(y_1^{[r]}), \sum_{r=1}^{m_2} G(y_2^{[r]}), \ldots, \sum_{r=1}^{m_D} G(y_D^{[r]})\}$, which forms the collection of system objectives.

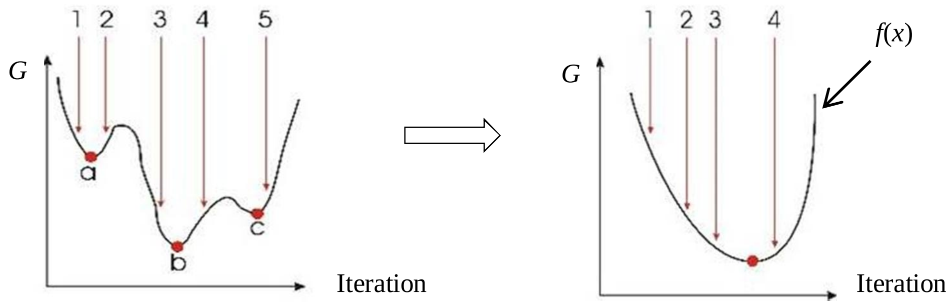


Figure 3.2: Optimization space conversion with Homotopy function

It is often difficult to get optimal solutions to this computationally difficult task for multimodal function optimization. In order to address this difficulty, the above discussed fitness is converted into easier one by use of a *Homotopy function* as illustrated in Figure 3.2. Definition of the new form of the Homotopy function is given in Equation (3.3).

$$J_i(q(x_i), T) = \sum_{r=1}^{m_i} G(y_i^{[r]}) - T * Es, \tag{3.3}$$

where $q(x_i)$ indicates probability distribution of agent $i$ and the temperature is denoted by $T \in [0,\infty)$. At the beginning, the uniform probability distribution sets $q(x_i^{[r]}) = 1/m_i$, $r$ = 1, 2,…, $m_i$. Each agent $i$ then calculates the expectation by aggregating its fitness as $\sum_{r=1}^{m_i} G(y_i^{[r]})$. From the joint probability distribution of all agents, the equation for the accumulated fitness is expressed as

$$E(\sum_{r=1}^{m_i} G(y_i^{[r]})) = \sum_{r=1}^{m_i} E(G(y_i^{[r]})) = \sum_{r=1}^{m_i} G(y_i^{[r]}) q(x_i^{[r]}) \prod_{(i)} q(x_{(i)}^{[?]}),$$

where ($i$) denotes every agent other than $i$. Consequently, the expected system objectives and the associated expected collections for $D$ agents are shown in Equation (3.4) as follows:

$$
\begin{array}{l}
\left.\begin{array}{l}
y_1^{[1]} = \{x_1^{[1]}, x_2^{[?]}, \ldots, x_i^{[?]}, \ldots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_1^{[1]}) \\
y_1^{[2]} = \{x_1^{[2]}, x_2^{[?]}, \ldots, x_i^{[?]}, \ldots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_1^{[2]}) \\
\qquad\qquad\qquad \vdots \\
y_1^{[r]} = \{x_1^{[r]}, x_2^{[?]}, \ldots, x_i^{[?]}, \ldots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_1^{[r]}) \\
\qquad\qquad\qquad \vdots \\
y_i^{[m_i]} = \{x_1^{[m_i]}, x_2^{[?]}, \ldots, x_i^{[?]}, \ldots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_1^{[m_i]})
\end{array}\right\} \Rightarrow \sum_{r=1}^{m_i} G(y_1^{[r]}) \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
\left.\begin{array}{l}
y_i^{[1]} = \{x_1^{[?]}, x_2^{[?]}, \ldots, x_i^{[1]}, \ldots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_i^{[1]}) \\
y_i^{[2]} = \{x_1^{[?]}, x_2^{[?]}, \ldots, x_i^{[2]}, \ldots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_i^{[2]}) \\
\qquad\qquad\qquad \vdots \\
y_i^{[r]} = \{x_1^{[?]}, x_2^{[?]}, \ldots, x_i^{[r]}, \ldots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_i^{[r]}) \\
\qquad\qquad\qquad \vdots \\
y_i^{[m_i]} = \{x_1^{[?]}, x_2^{[?]}, \ldots, x_i^{[m_i]}, \ldots, x_{D-1}^{[?]}, x_D^{[?]}\} \Rightarrow G(y_1^{[m_i]})
\end{array}\right\} \Rightarrow \sum_{r=1}^{m_i} G(y_i^{[r]}) \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
\left.\begin{array}{l}
y_D^{[1]} = \{x_1^{[?]}, x_2^{[?]}, \ldots, x_i^{[?]}, \ldots, x_{D-1}^{[?]}, x_D^{[1]}\} \Rightarrow G(y_D^{[1]}) \\
y_D^{[2]} = \{x_1^{[?]}, x_2^{[?]}, \ldots, x_i^{[?]}, \ldots, x_{D-1}^{[?]}, x_D^{[2]}\} \Rightarrow G(y_D^{[2]}) \\
\qquad\qquad\qquad \vdots \\
y_D^{[r]} = \{x_1^{[?]}, x_2^{[?]}, \ldots, x_i^{[?]}, \ldots, x_{D-1}^{[?]}, x_D^{[r]}\} \Rightarrow G(y_D^{[r]}) \\
\qquad\qquad\qquad \vdots \\
y_D^{[m_i]} = \{x_1^{[?]}, x_2^{[?]}, \ldots, x_i^{[?]}, \ldots, x_{D-1}^{[?]}, x_D^{[m_i]}\} \Rightarrow G(y_D^{[m_i]})
\end{array}\right\} \Rightarrow \sum_{r=1}^{m_i} G(y_i^{[r]})
\end{array}
\tag{3.4}
$$

Thus the expectation with respect to fitness values for $D$ agents is formed with utilities: $\{\sum_{r=1}^{m_1} E(G(y_1^{[r]})), \sum_{r=1}^{m_2} E(G(y_2^{[r]})), \ldots, \sum_{r=1}^{m_D} E(G(y_D^{[r]}))\}$. It also means that the PC algorithm can convert discrete variables into continuous variable vectors in the form of probabilities corresponding to these discrete variables. An ordinary choice for the *Es* function of Equation (3.3) is the entropy function given in Equation (3.5).

$$S_i = -\sum\nolimits_r^{m_i} q(x_i^{[r]}) \log_2(x_i^{[r]}). \tag{3.5}$$

Hence, minimizing the Homotopy function for each agent *i* can now be redescribed in Equation (3.6) as:

$$
\begin{aligned}
J_i(q(x_i),T) &= \sum_{r=1}^{m_i} E(G(y_i^{[r]}) - T * S_i \\
&= G(y_i^{[1]})q(x_i^{[1]})\prod\nolimits_{(i)} q(x_{(i)}^{[?]}) + G(y_i^{[2]})q(x_i^{[2]})\prod\nolimits_{(i)} q(x_{(i)}^{[?]}) + \cdots \\
&\quad \cdots G(y_i^{[m_i-1]})q(x_i^{[m_i-1]})\prod\nolimits_{(i)} q(x_{(i)}^{[?]}) + G(y_i^{[m_i]})q(x_i^{[m_i]})\prod\nolimits_{(i)} q(x_{(i)}^{[?]}) \\
&\quad - T(-\sum_{r=1}^{m_i} q(x_i^{[r]})\log_2(x_i^{[r]}))
\end{aligned}
\tag{3.6}
$$

where $T \in [0,\infty)$ is the temperature parameter. The Homotopy function is to be optimized using any admissible optimization tool. In literature, several approaches are considered to find the minimal value of $J(q_i(x_i),T)$; widely used approaches are Nearest Newton Descent Scheme, Broyden Fletcher Goldfarb Shanno (BFGS) and the Deterministic Annealing [21]. This thesis presents a further acceleration method by use of DE algorithm according to the probability distributions produced by Nearest Newton Descent Scheme. The Homotopy function, as the procedure of minimization for every agent, leads to new probability vectors $q(x_i)$, $i$ =1 ,…, $D$. In this regard, agent *i* has the distribution of its probability $q(x_i)$, that yields the strategy vector $y_i^{[r]}$, $r$ = 1,…,$m_i$, for the minimization of the aggregate expectation of the fitness, $\sum_{r=1}^{m_i} E(G(y_i^{[r]}))$. The rule of updated probability $q(x_i^{[r]})_k$, where *k* indicates number of algorithmic iterations, relies on the adopted optimization approach. For the core PC algorithm, the Nearest Newton Descent method is shown for agent *i* in Equation (3.7-3.8):

$$
\begin{aligned}
& q(x_i^{[r]})_{k+1} \leftarrow q(x_i^{[r]})_k - \Delta q_{i,k} \\
& \Delta q_{i,k} = \alpha_s * q(x_i^{[r]})_k * \\
& \left\{ \frac{(Contribution\ of\ x_i^{[r]})_k}{T_k} + S_{i,k} + \log_2(q(x_i^{[r]})_k) \right\}
\end{aligned}
\tag{3.7}
$$

$$(Contribution\ of\ x_i^{[r]})_k = (E(G(y_i^{[r]})))_k - (\sum_{r=1}^{m_i} E(G(y_i^{[r]})))_k \quad (3.8)$$

where $\alpha_s \in (0,\ 1]$ denotes the descent step size. In order to ensure non-negative probabilities, values less than 0 are set to $10^{-6}$ and then all probability values are re-normalized accordingly.

There is one special strategy $x_i^{[r]}, 1 \le r \le m_i$, in the $m_i$ strategy set that takes the largest effert in the minimization of the expectation of utilities. This distinguished variable is called the favorable strategy, $x_i^{fav}$. As shown in Figure 3.3, for a case where there are 10 strategies, i.e. $m_i = 10$, the convergence of the highest probability value among the probability distributions of agent $i$ is illustrated. Consequently, the favorable strategy vectors for the $D$ agents are rearranged as: $y^{[fav]} = \{x_1^{[fav]}, x_2^{[fav]}, ..., x_{D-1}^{[fav]}, x_D^{[fav]}\}$.

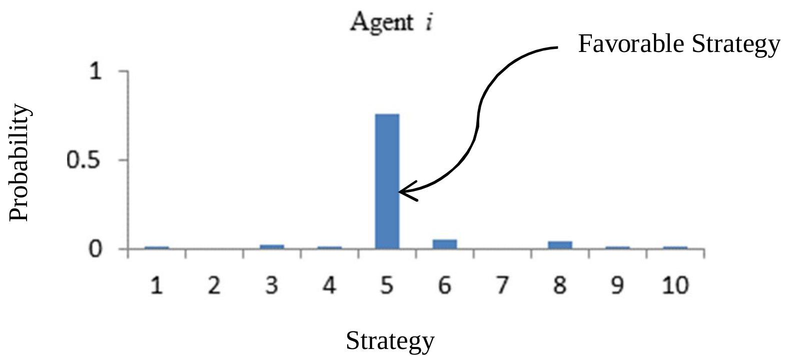


Figure 3.3: Probability distribution of agents

Then the objective function $G(y^{[fav]})$ can be evaluated by $y^{[fav]}$. If the current system objective $G(y^{[fav]})$ is better than that from the previous iteration solution, accept the current system objective $G(y^{[fav]})$ and corresponding $y^{[fav]}$ as current solution and continue to the next iteration, else reject current system objective $G(y^{[fav]})$ and the corresponding $y^{[fav]}$, and retain the previous iteration solution to next run.

In order to the convergence of the optimization process, Nash equilibrium is achieved at termination as follows: either temperature $T = T_{final}$ or $G(y^{fav})_k - G(y^{fav})_{k-1} \leq \varepsilon$, for a predefined $\varepsilon > 0$. If any termination criterion is not met, then the PC algorithm modifies the strategy limits of interval $\Omega_i$ and temperature parameter in Equation (3.9-3.11) as follows.

$$x_i^{\mathrm{L}}(k+1) = (1-\lambda)*x_i^{\mathrm{fav}},\ i=1,\ldots,D \tag{3.9}$$

$$x_i^{\mathrm{H}}(k+1) = (1+\lambda)*x_i^{\mathrm{fav}},\ i=1,\ldots,D \tag{3.10}$$

$$T_{k+1} = (1-\alpha_T)*T_k \tag{3.11}$$

where $0 < \lambda < 1.0$ represents the limit factor, while $0< \alpha_T < 1.0$ denotes temperature ratio. If strategy values exceed their corresponding upper and lower bounds, those values should be repaired with random and uniform sampling within the predefined range. Each agent $i$ then samples $m_i$ strategies within the updated sampling interval $\Omega_i$ and forms new strategy set $x_i$ represented as $x_i = \{x_i^1, x_i^2, \ldots, x_i^{m_i}\}$, and $i = 1, 2, \ldots, D$. Finally, PC increases inner iteration $k$ and the process repeats until termination criteria are reached. The detailed procedure of PC algorithm is summarized in Algorithm 3.1.

While PC approach is applied to a given problem, each agent coordinates all distributed strategy vectors with its probability distribution. According to the independent probabilities assumption, all agents build a set of solutions to minimize a Homotopy function with customized expected utilities in the joint probability space. Then each agent updates the probability distributions to determine the favorable strategy vectors from the peaky probability in its distribution. Thus PC approach uses random sampling search mechanism to explore the solution space around the favorable strategies based on the agent's knowledge during the iterations. This procedure thus prevents premature convergence due to the updated search space. Eventually, the algorithm enhances search ability and converges to a global optimal solution.

**Algorithm 3.1:** Procedure of Probability Collectives

```
 1: Specify the number of samples for each iteration;
 2: Set the parameters: learning range factor $\lambda$, probability update step size $\alpha_s$, cooling rate
    $\alpha_T$ and the termination criteria $\varepsilon$;
 3: Assign agents to the variables in the problem, with their actions representing choices for
      values. Set the starting probabilities for each agent to uniform over its possible actions;
 4: Initialize $T$ and iteration step $k \leftarrow 0$;
 5: **Repeat**
 6:   Form $m_i$ number of strategies $y_i^{[r]}$ of agent $i$;
 7:   Evaluate $m_i$ objective functions $G(y_i^{[r]})$;
 8:   Calculate local expectation $E(G(y_i^{[r]}))$ over joint actions;
 9:   Compute every agent's global expected utility $\sum_{r=1}^{m_i} E(G(y_i^{[r]}))$ for the agents for each
      of their possible moves;
10:   **Repeat**
11:     Minimize the Homotopy function $J(q_i(x_i),T)$;
12:     Compute contributions for each agent;
13:     Compute new probability value to update its distribution $q(x_i)$ using an second order
        technique such as Nearest Newton Descent Scheme;
14:     Update the temperature $T$;
15:   **Until** the termination condition are met
16:   Determine favorable strategy $y^{\text{fav}}$ by the highest probability action for each agent;
17:   Update variable ranges along with the most favorable strategy vector in $\Omega_i$;
18:   $k = k + 1$;
19: **Until** Termination Criterion are met
20: Return the best solution found so far.
```

## 3.2 Multi-Objective PC

From empirical findings, researchers adapted the PC framework for multi-objective optimization problems. Decomposition technique in multi-objective optimization is known to be superior to other methods on a wide variety of problems since it provides optimal portfolio for converting a MOP into a number of individual single objective optimization problems. Hence, PC optimization can easily conduct the single objective optimization for the solution of the multi-objective problem. The detailed information about decomposition with its technical expressions will be discussed in Chapter 5.

In this thesis, the potential schemes explore the nondominated points along the Pareto optimal front during the PC procedure are investigated. Through sufficient search of the sampling distributions of the decision variables, promising regions of *Ω* in the Pareto set are exploited for fitness values. It is observed that multi-objective PC-based optimization involving the proposed methods exhibit that PC works successfully for MOPs. The first multi-objective algorithm based on PC approach is MOPC algorithm [22], which is shown in Algorithm 3.2. In this algorithm, the max-min fitness function [41] was introduced to approximate the Pareto set with the expression shown in Equation (3.12).

$$f_{\max i \min}(x) = \max_{x \neq x^j}(\min_{1 \leq i \leq m}(f_i(x) - f_i(x^j))) \tag{3.12}$$

where $m$ is the number of objective functions, $x^j$ is the $j$th sample vector in the set and $f_{maximin}(x)$ is the fitness value for the decision vector $x$. This method keeps an updating archive for maintenance of better solutions and allows a greater diversity of distributions. A more detailed description of this algorithm can be found in [22]. In this concern, most multi-objective PC algorithms with constructed objective functions must deal with the behavior of various objectives in a single run.

**Algorithm 3.2:** MOPC Optimization

```
1: Initialize the archive set $A$ to empty, $T$ to $T_{start}$ and calculate $T_{decay}$ and the number of
   evaluations to 0
2: Initialize the set of MOPC Particles $P$
3: **Repeat**
4:   **For all** MOPC Particles **Do**
5:     **If** first run **Then**
6:       Draw and evaluate a set of samples $D$ from $X$ using a uniform distribution for the first
         run and $q_\theta$ thereafter
7:     **End If**
8:     Add the samples taken in $D$ to a local cache $L$
9:     Calculate maximin for the members of $L \cup A$
10:    Find the new $q_\theta$ (using $L$) by minimizing the KL Divergence
11:    Add the samples from $D$ that are not dominated to the archive $A$
12:    evaluations←evaluations + 1
13:  **End For**
14:  **If** ($T > T_{end}$) **Then**
15:    Decrement $T$ by $T_{decay} = \frac{T_{end}}{T_{start}}^{\frac{|P|*|D|}{|E|}}$ where maximum number of evaluations allows $E$
16:  **End If**
15: until (evaluations > maximum evaluations)
16: Output the non-dominated set from archive set $A$
```

# Chapter 4

# PROBABILITY COLLECTIVES FOR SINGLE OBJECTIVE OPTIMIZATION PROBLEMS

## 4.1 Introduction

This chapter provides descriptions of DE and CE algorithms that are proposed for unconstrained global optimization. Through a great deal of empirical investigation and observation, PC and DE algorithms together with the CE method are the powerful methods to resolve difficult problems. As discussed in the second chapter, PC progresses with different techniques such as gradient technique and Nearest Newton Descent scheme to update its probability distributions for the favorable solutions. CE refines solution distributions using two smoothing operators. On the other hand, DE algorithm performs as one of the high-speed and robust global heuristic search to update parameter vectors of population in progress.

Using a harmonious combination of the three algorithms, an innovative hybrid algorithm is proposed, named MPCDE, i.e., modified PC and DE [42]. This hybrid model uses conventional PC optimization updated by CE method and adaptive DE mutation scheme to improve search ability for large scale single-level global optimization problems. In the context of hybridization, the PC algorithm initially seeks a set of feasible strategy vectors to find the most favorable strategies by the update of probability distributions with its corresponding parameters while DE takes experience of PC into existing population to perform a further search using adaptive operators for achieving appropriate indicators. Meanwhile, CE improves solutions related to knowledge of PC by refining distributions with two smoothing operators and improves the population of DE algorithm. At the end of this chapter, experimental results presented that the introduced algorithm has competitive performance for two classical sets of single objective functions.

## 4.2 Differential Evolution

Differential Evolution algorithm is first introduced by Storn, R. [2]. It is a typical representative of EAs for solving real-parameter optimization problems. Indeed, DE has many advantages including its robustness, reliablility, and ease of use for many aspects. Due to these properties, DE is considered to be an effective global optimization algorithm [43]. DE has also been used to optimize a mixture of integer, discrete, and continuous variables optimization problems [43].

### 4.2.1 Description of the DE Algorithm

Differential Evolution is one of the most powerful population based stochastic search methods used to deal with global optimization problems. DE is usually affected and adjusted by two primary factors: control factor $F$ and crossover ratio $CR$ [43]. Even though DE can capture the better solution by exploring existing population and improve the convergence speed for the global solutions, it possibly gets stuck into a local minimum for some cases. In order to tackle problems such as stagnation, the most researchers have developed variants of DE algorithm, which resulted in five different mutation operators such as basic DE/rand/1, DE/current-to-best/1 and so on [30]. Meanwhile, two different significant crossover types: exponential and binomial [30] were also developed to keep coordination with other search operators. Detailed descriptions of exponential crossover operator are presented in [30]. Binomial (uniform) crossover will be introduced in the following section.

DE operates using a set or population vectors $S = \{x_1, x_2, \cdots, x_N\}$ of potential solutions or points. The population size $N$ remains constant throughout. The population vectors evolve into a final individual solution by simple techniques of combing simple arithmetic operators with a cycle of events of mutation, crossover and selection. Mutation and crossover operators are used to produce new trial vectors, and selection operator then determines whether the trial vector survives for next generation or not. At each generation $g$, the procedure aims to generate a new population by replacing points in the current population $S$ with better ones. The population is simply a set of parameter vectors $X_{i,g'}$ , where $i$ indicates the index of the population member. The main operators of DE perform along with the cycle of phases, as shown in Figure 4.1.

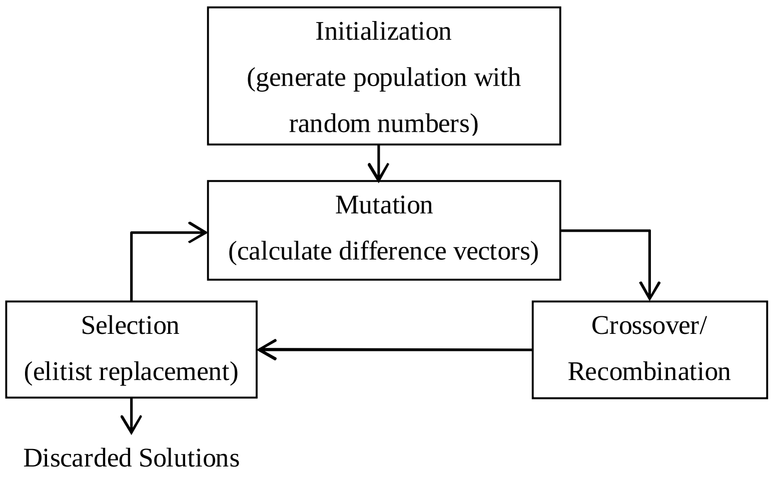


Figure 4.1: Main phases of the DE algorithm cycle

### 4.2.2 Operation of the DE Algorithm

***Initialization***. The first step of DE algorithm is to initialize the population. DE generates a population of $N$ individuals of $D$-dimensional parameter vectors representing the candidate solutions, i.e., $X_{i,g'} = \{x^1_{i,g'}, x^2_{i,g'} \ldots, x^D_{i,g'}\}, i = 1, 2, \ldots, N$ and every component in the original population is randomly sampled with seeding uniformly and subject to boundary constraints. The parameter (target) vectors give the following simple initialization formula for each component:

$$x_{i,0}^{j} = x_i^L + \text{rand} * (x_i^U - x_i^L), \ \ j = 1, 2, ..., D, \forall i, \tag{4.1}$$

where rand$\in$[0, 1] is a uniformly distributed random value generated for each *j* and $x_i^U$ and $x_i^L$ are the upper and lower bounds respectively.

***Mutation***. This procedure of the DE algorithm produces new trial vectors. At every generation *g*, each member of *S* is targeted to be replaced with a better trial vector. For the simplest case of DE/rand/1, a mutated point is created by adding the weighted difference of two population members to a third vector. By this scheme, the mutation procedure generates an associated mutant (donor) vector $V_{i,g'} = \{v_{i,g'}^1, v_{i,g'}^2 \ldots, v_{i,g'}^D\}$, $i = 1,\ldots,N$ as follows: for each target $X_i$, the procedure operates a uniform selection of three random elements $X_{r_1}$, $X_{r_2}$, and $X_{r_3}$ and $r_1 \neq r_2 \neq r_3 \neq i$, i.e. all vectors are unique and none of these vectors corresponds to the target vector $X_{i,g'}$. Generate the donor vector $V_j$ as,

$$V_{j,g'} = X_{r_1,g'} + F * (X_{r_2,g'} - X_{r_3,g'}) \tag{4.2}$$

where the factor $F\in$[0,2] refers to a remarkable scaling rate for adjusting subtracted variations from two components. Figure 4.2 illustrates the location of $V_{j,g'}$ as would be given in Equation (4.2).

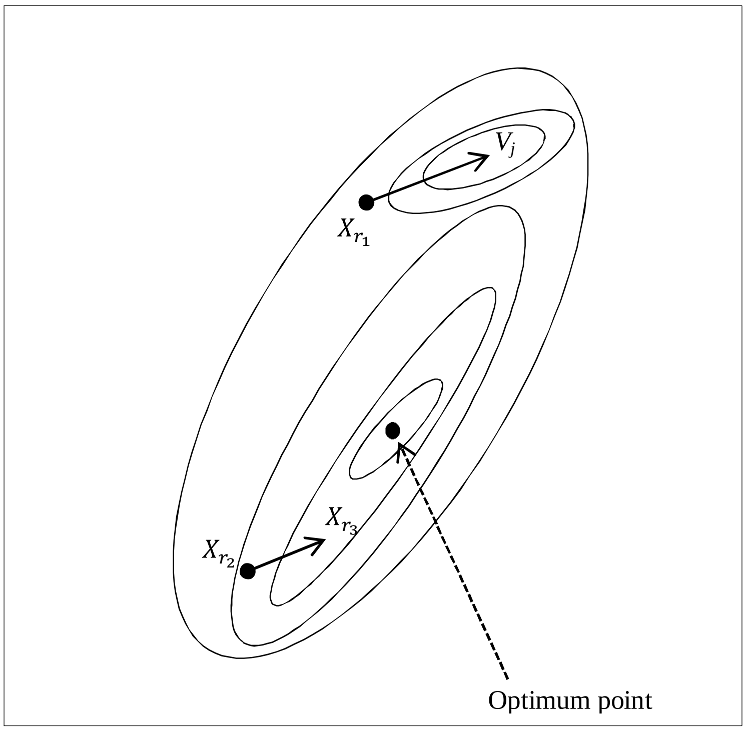


Figure 4.2: Mutation operation using DE/rand/1

***Crossover.*** The target or parent point $X_{i,g'}$ together with the new mutated point $V_{j,g'}$ are recombined to create the trial point $U_{i,g'} = (u^1_{i,g'}, u^2_{i,g'}, \ldots, u^D_{i,g'})$. The hybrid algorithm applies the binomial method for DE. Binomial recombination [30] generates a random number rand$_j \in [0,1]$ for each component. If rand$_j \leq CR$ then the binominal crossover operator copies the $j^{th}$ component of mutant vector $V_{i,g'}$ to the corresponding element in the trial vector $U_{i,g'}$, otherwise, it is copied from the associated target vector $X_{i,g'}$. This process continues until all parameter vectors from $X_{i,g'}$ have been considered. The condition of a random integer $I_{\text{rand}} \in [1, 2, ...., D]$ is introduced to ensure that the trial vector $U_{i,g}$ will differ from its corresponding target vector $X_{i,g'}$ by at least one parameter. Binomial recombination can be formulated as in Equation (4.3) with the crossover constant $CR \in (0,1]$.

$$u_{i,g'}^{j} = \begin{cases} v_{i,g'}^{j} & \text{if } (rand_j \le CR) \vee (j = I_{rand}) \\ x_{i,g'}^{j} & Otherwise \end{cases} \tag{4.3}$$
$$\forall\ i = 1,2,...,N \text{ and } j = 1,2,...,D.$$

***Selection.*** So far, $N$ competitions are maintained to identify the members of $S$ for the next iteration. The $i^{th}$ competition is reserved to replace the target vector $X_{i,g'}$ in $S$. This is done by comparing the trial vector $U_{i,g'}$ to those of $X_{i,g'}$. The better $U_{i,g'}$ and $X_{i,g'}$, based on the objective function values, is greedily accepted by the selection operation with its expression in Equation (4.4).

$$X_{i,g'+1} = \begin{cases} U_{i,g'} & \text{if } f(U_{i,g'}) \le f(X_{i,g'}) \\ X_{i,g'} & \text{otherwise} \end{cases} \tag{4.4}$$

The trial vector with better fitness value will thus serve for the next generation. Accordingly, all the selected individuals of the next generation are better than others in the current generation. The DE operation loop repeats until the total maximum number of function evaluations, or any other predefined termination criterion is satisfied. The basic DE procedure with mutation scheme DE/rand/1 is given in Algorithm 4.1.

**Algorithm 4.1:** The DE algorithm for unconstrained optimization

```
 1: Set control parameters $N$, $CR$, $F$ and $g \leftarrow 0$;
 2: Initialize Population, $S = \{x_{1,0}, x_{2,0},\dots, x_{N,0}\}$ using Equation (4.1);
 3: Evaluate objective function $f$ for each member in the population;
 4: Repeat
 5:   For $i = 1$ To $N$,
 6:     generate trial vector $U_{i,g'}$ via:
 7:     Mutation using Equation (4.2);
 8:     Crossover using Equation (4.3);
 9:     Evaluate $f(U_{i,g'})$;
10:   End For
11:   Update population using Equation (4.4);
12:   $g' = g' + 1$;
13: Until Termination Criterion are met.
```

## 4.3 Cross Entropy Method

The cross entropy method is an optimization algorithm which manipulates multiple distributions of possible solutions under relational rules in a parallel way for a stochastic optimization problem. Thus, a general Monte Carlo approach using the importance sampling technique is used to solve rare event probability estimation problems [44]. In optimization problems, an optimal solution can be considered as a rare event. The CE method consists of the following two main phases: generation of $N$ samples of random data or vectors according to a random mechanism; updating the parameters of the random mechanism, typically parameters of probability density function (PDF), to produce better samples from population for the next iteration.

Let $X$ be a random sample taking its value in some discrete space $\chi$ with a pdf $f(\cdot)$, $S'(\cdot)$ be a real-valued function defined on $\chi$ and $\gamma$ be a real number. In the rare-event simulation context, one needs to estimate the probability of occurrence $l$ of an event $\{S'(\cdot) \geq \gamma\}$, i.e. to estimate the expression $E_{X\sim f(\cdot)}\, [I_{\{S'(\cdot)\geq\gamma\}}]$.

The CE method requires specifying the sampling distribution and the updating rules for its parameters. The choice of the sampling distribution is quite arbitrary. The $s$-dimensional normal distribution with independent components, mean vector $\mu$ = $(\mu_1, . . . , \mu_s)$ and variance vector $\sigma^2 = (\sigma_1^2,\ldots, \sigma_s^2)$ is denoted by $N(\mu, \sigma^2)$. It is recalled that a multivariate Gaussian distribution can be used to describe the distribution of vectors in $\mathfrak{R}^s$ with the corresponding probability density [45].

$$f(X, v = \{\mu, \sigma\}) = \prod_i^s \frac{1}{\sqrt{2\pi}\sigma_i} \exp(-\frac{(X_i - \mu_i)^2}{2\sigma_i^2}) \tag{4.5}$$

where $v$ is the set of $2s$ parameters used to define the probability density and $\mu_i$ indicates the mean of the $i$-th component of $X$. Similarly, $\sigma_i$ denotes the standard deviation of the $i$-th component of $X$. Using this parameterized distribution, it can be shown that the updating rules of Equation (4.6-4.7) will be as follows:

$$\tilde{\mu}_t = \frac{\sum_{i=1}^{N} I_{\{S(X_i)\geq\gamma\}} X_i}{\sum_{i=1}^{N} I_{\{S(X_i)\geq\gamma\}}} \tag{4.6}$$

$$\tilde{\sigma}_t^2 = \frac{\sum_{i=1}^{N} I_{\{S(X_i)\geq\gamma\}} (X_i - \tilde{\mu}_t)^2}{\sum_{i=1}^{N} I_{\{S(X_i)\geq\gamma\}}} \tag{4.7}$$

In the current iteration $t$ of the algorithm, the distribution is updated in a step-wise mode, using a smoothing parameter $\alpha$ to modify the mean:

$$\mu_t = \alpha\tilde{\mu}_t + (1-\alpha)\tilde{\mu}_{t-1} \tag{4.8}$$

This smooth updating criterion can make the CE method escape from being trapped at local optimums. Empirically a value of $\alpha$ is given in [0.6, 0.9] for the best results. To prevent the sampling PDF from getting stuck in a suboptimal solution, Rubinstein and Kroese [45] proposed the use of dynamic smoothing rule where, at each iteration $t$, the variance is updated using a smoothing parameter $\beta_t$ by Equation (4.9-4.10) as:

$$\beta_t = \beta_0 - \beta_0(1-\frac{1}{t})^c \tag{4.9}$$

$$\sigma_t^2 = \beta_t\tilde{\sigma}_t^2 + (1-\beta_t)\sigma_{t-1}^2 \tag{4.10}$$

where $\beta_0$ is smoothing constant (typically between 0.8 and 0.99) and $c$ is a fixed integer (typically between 5 and 10). The proposed algorithms in this thesis apply the Equation (4.6-4.10) to update the population distribution and its operators for achieving the better performance.

## 4.4 A Hybrid Proposed Approach for SOP (MPCDE)

As mentioned before, the PC approach is a random search optimizer that uses iteratively the joint probability distributions and gradient descent technique to update the sampling probability distributions for the favorable solutions. However, the descent-based technique is likely to trap into local optimum. Such case could deteriorate the quality of solutions especially for some hardest instances. Also, the DE algorithm may unusually suffer from locally optimum and premature convergence. As stated in [46], the cause of stagnations of DE is that the recreation method gives only a finite set of possible trial vectors and if none of them modifies a component of the current population during comparison, then stagnation occurs. Most researchers pay more attentions to study the dimension $D$ and two control factors $F$, $CR$. From empirically scientific articles, a high dimension of decision variables can result in decreasing stagnation situation. By a great deal of simulation results, the control factors were usually selected nearly 1.0. Thus, our studies need to identify suitable parameter settings that are obtained from promising fitness values during the experiments.

In order to address the typical issues from the PC and DE algorithms, this thesis introduces a learning model MPCDE to explore the optimal solutions, enhancing convergence speed and removing from weaknesses of the two approaches. Based on the search operators of two main algorithms, the evolutionary framework combines these mechanisms in a coordinated way in which they exchange their search experience iteratively. Obviously, PC approach optimizes the decision strategy vectors along with their probability distributions simultaneously to accomplish the purpose of improving solutions. The solution distributions are produced and refined by the dynamic operators from the CE approach. The cross entropy method is used to update its components of distribution of a Gaussian density by using mean $\mu_i$ and standard deviation $\sigma_i$, $i$ = 1,…,$D$. Then, DE obtains the knowledge from CE method for updating the population and modifies and selects all components of the parameter vectors to guide the global minimum by the repeated process. Each individual is thus randomly sampled to construct new population of solutions. Also, PC utilizes random search with various techniques over probability distributions to determine the favorable strategy vectors from the highest probability values for a number of runs $M_1$. Figure 4.3 describes a general flowchart of the proposed hybrid algorithm MPCDE.

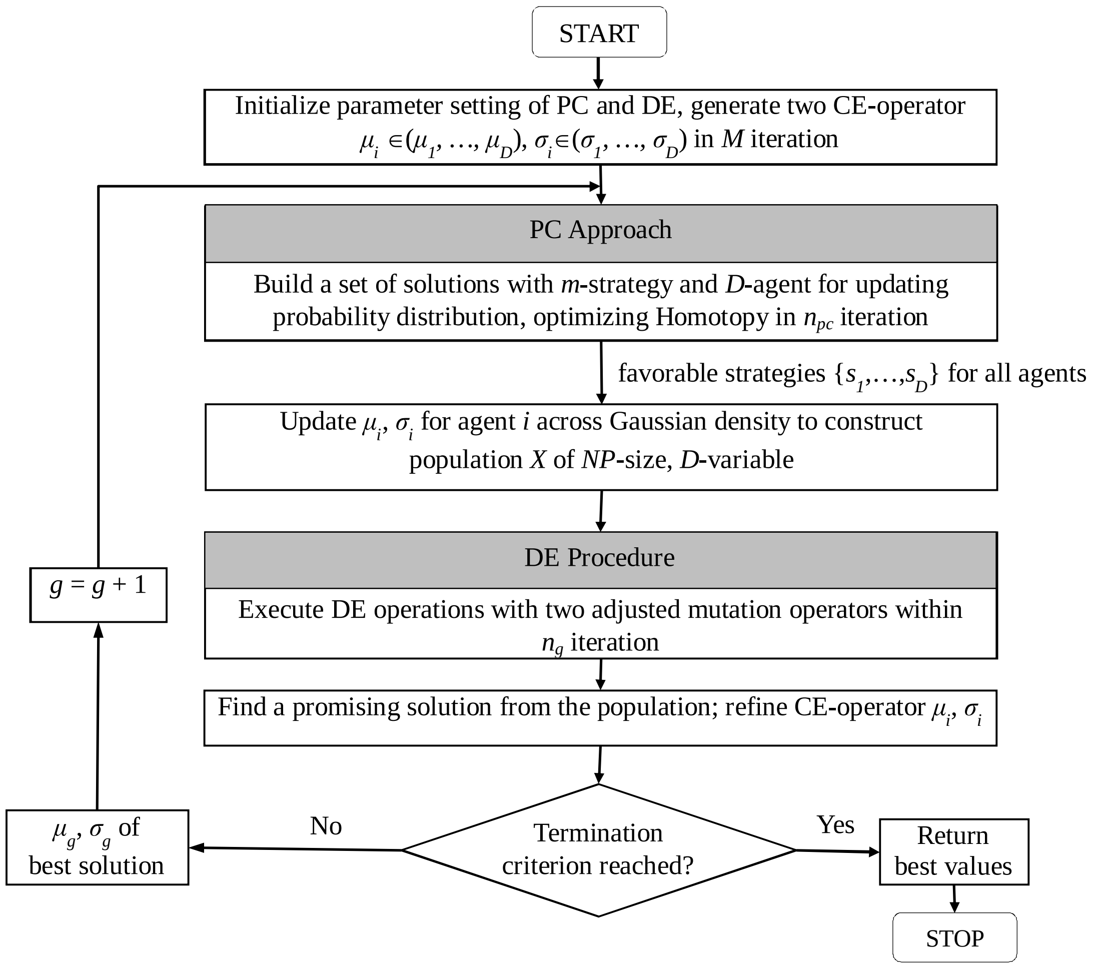

START
Initialize parameter setting of PC and DE, generate two CE-operator $\mu_i \in(\mu_1, \ldots, \mu_D)$, $\sigma_i\in(\sigma_1, \ldots, \sigma_D)$ in $M$ iteration
PC Approach
Build a set of solutions with $m$-strategy and $D$-agent for updating probability distribution, optimizing Homotopy in $n_{pc}$ iteration
favorable strategies $\{s_1,\ldots,s_D\}$ for all agents
Update $\mu_i$, $\sigma_i$ for agent $i$ across Gaussian density to construct population $X$ of $NP$-size, $D$-variable
DE Procedure
Execute DE operations with two adjusted mutation operators within $n_g$ iteration
Find a promising solution from the population; refine CE-operator $\mu_i$, $\sigma_i$
Termination criterion reached?
No
Yes
$\mu_g$, $\sigma_g$ of best solution
$g = g + 1$
Return best values
STOP

Figure 4.3: Flowchart of the hybrid algorithm MPCDE

The proposed MPCDE step conducts DE with existing population that is randomly constructed by the updated parameter vectors. A new population of DE is contributed by CE, where its operators $\mu_i$ and $\sigma_i$ are estimated from the solutions obtained by PC approach. Therefore, the Gaussian distribution is able to initialize each component of a population and is made up of the values of $\mu_i$ and $\sigma_i$ that are computed as;

$$\mu_i = \sum_{S'} \Big/ \sum_{S'} q_i \tag{4.11}$$

$$\sigma_i = \sum_{S'} q_i (x_i - \mu_i) \cdot (x_i - \mu_i)^T \Big/ \sum_{S'} q_i \tag{4.12}$$

where $q_i$ indicates the optimized strategy of PC for agent $i$, $S'$ denotes the sample set of solution obtained from the objective function in PC. Let $g$ be the current step of iteration for the designed model, two search operators are generated and updated by use of cross-entropy concept as shown in Equation (4.13−4.15):

$$\mu_{i,g} = \beta\mu_{i,g-1} + (1-\beta)\mu_{i,g} \tag{4.13}$$

$$\sigma_{i,g} = \beta\sigma_{i,g-1} + (1-\beta)\sigma_{i,g} \tag{4.14}$$

$$\beta = \beta - \beta(1 - 1/g)^{ri} \tag{4.15}$$

where $\beta$ represents smoothing factor, $ri$ is an integer and denotes exponential growth rate. In particular, the introduced hybrid algorithm dynamically refines search schemes by the distribution factors of CE and explores favorable indicators around local neighborhood for convergence of optimal solutions in DE. In other words, DE learns from the existing population that is maintained by the distribution of PC across a set of favorable strategy vectors and the population solutions are updated by use of advanced CE tool for reinforcing localization of the strategy vectors.

The DE procedure launches a new population explained before and executes for a number of generations $M_2$. The hybrid algorithm presents a modified DE mechanism, where adopted mutation scheme measures the distance between the candidate parameter vectors randomly selected from the population and adaptable calculated learning size. In this respect, adaptive mutation scheme efficiently explores and perturbs two operators in the search space along with the promising direction of optimal solution. The procedure of adaptive scheme for DE mutation is outlined in Algorithm 4.2 as follows:

**Algorithm 4.2:** The modified mutation operator of DE

```
$r_1$ = randi(1,|$P_{DE}$|); $r_2$ = randi(1,|$P_{DE}$|); $r_3$ = randi(1,|$P_{DE}$|);  /* $r_1 \neq r_2 \neq r_3$ * /
$X_{r1}$=$P_{DE}$($r_1$,:);  $X_{r2}$=$P_{DE}$($r2$,:);  $X_{r3}$=$P_{DE}$($r3$,:);
For $i$ =1 to $D$ Do,
   If $| x_{r2,i} | \geq \gamma_2$
      $coeff = \frac{x_{r2,i}}{x_i^H - x_i^L}$
   Else
      $coeff = \frac{x_{r2,i}}{2\gamma_2}$
   End if
   If $d_i = |x_{r2,i} - x_{r3,i}| \leq \gamma_1$,
      $u_i = x_{r1,i} + F * (x_{r2,i} - x_{r3,i})$
   Else
      $u_i = x_{r1,i} + coeff$ * rand * $x_{r2,i}$
   End if
End for
```

The parameters $\gamma_1$ and $\gamma_2$ of Algorithm 4.2 are two control coefficients. Values of these parameters are defined in the experimental evaluations given below. The global search of the hybrid algorithm involves two schemes, the classical DE/rand/1 and modified mutation, determined by an adaptive factor according to the length of vectors in population solutions. The other parameter *coeff* is defined as a dynamical factor to measure a scaling scheme for DE according to the search length of vectors. When the search length is greater than the limited factor $\gamma_1$ then *coeff* is considered as the rate of change for one component $x_{r2,i}$ to the other limited factor $\gamma_2$. Otherwise, *coeff* is evaluated on the proportion of $x_{r2,i}$ in the problem domain. Two proper perturbations adjust the algorithm with well-defined parameters to explore the solution space towards the global optimum.

## 4.5 Experimental Studies

In the interest of verifying efficiency of the modified evolutionary algorithm based on PC and present its successful comparison with state-of-the-art methods, two classical sets of unconstrained problems were applied: one involves 23 classical benchmarks selected in [47] and the second part comprises 25 competition benchmarks problems for CEC2005 Special Session. Definition, categorization, fitness landscape characteristics and other specification of these complicated functions can be found in [48]. Various difficulties for these problems make them appropriate for comparison of relative success of continuous global optimization methods.

### 4.5.1 Experimental Setting

For the purpose of providing fair comparisons of evaluations for each run, the dimension $D$ and the maximum number of function evaluations (Max_FEs) are defined the same as the parameters' set in the associated articles. The same number of independent runs is carried out for each problem and statistical analyses are conducted using results of all the trials. The termination criterion for MPCDE is defined with Max_FEs. The aim of the experimental evaluations is to validate the optimization performance of MPCDE for all problems by comparing mean values with some metaheuristics in terms of the number of function evaluations. Every agent of PC approach is allocated to occupy 15 strategy values so that it forms a population size of $D$*15 individuals, while DE algorithm uses a population size of 50 sizes individuals. When the proposed algorithm starts to initiate population of two main approaches for global search, the procedure needs to use CE method for completing this task. In the iterative step, values of iterations for PC and DE are given to $M_1$ = 10, and $M_2$ = 100, respectively. Temperature parameter $\alpha_T$ is updated by a cooling factor of 0.9 and step size for probabilities is set as $\alpha_s = 0.098$. The inner termination threshold for PC sets as $\varepsilon = 0.0001$ and the step size of modifying domain ranges is set to $\lambda = 0.5$. The parameters for CE method for updating the mean and standard deviation of distributions are defined with $\beta = 0.9$ and $ri = 9$, respectively. In DE operation, two controlling factors are set as $F = 0.5$ and $CR = 0.8$. Moreover, the two learning factors of the combined schemes for DE are given to $\gamma_1 =$ 2 and $\gamma_2 = 1$, respectively. If the introduced hybrid algorithm runs and reaches a limit of 500,000 FEs totally − then it terminates and yields the optimal solution so far.

The hybrid algorithm runs in Matlab®10a programming language on Windows 7 environment and a personal computer (Intel (tm) i5-2540 Dual Core Processor 2540 2.60 GHz, 4GB RAM) is used for program execurions. The precision for the floating-point operations is set to 15 fractional digits. In all tables illustrating experimental results, scores of the best performing algorithms are typed in boldface.

### 4.5.2 Problem Categories

Many potential features such as modality, separability, linearity, noises, and rotations determine problem difficulties. Generally, different problems are divided into four characteristics as follows: unimodal functions, basic multimodal functions, highly multimodal functions and hybrid composition functions. Unimodal (convex) functions are easier to solve and have one globally optimum while multimodal functions have several optimal solutions that offer require algorithms to jump out of locally optimal solutions. The hybrid composition functions are constructed by mixing different components of basic problems and get harder to solve according to dimension of problems, due to a larger relationship between the fitness solution and its variable values.

### 4.5.3 First Set of Benchmark Problems

Twenty-three classical test benchmarks selected from [47] for real-valued single objective optimization are commonly considered for evaluation and exhibit the convergence speed of MPCDE framework. The test functions that are categorized into a set of classes in terms of various features are presented in the Table 4.1. This table lists the names of functions, dimensions ($D$) of decision variables, modality and associated properties in each category of the functions. Besides, the test functions are also defined in Table 4.2 with their corresponding domain of search and global minima in the fitness landscapes. A more detailed description of these test functions can be found in [47]. Separable functions as those that can be decomposed in terms of single-dimensional functions (one for each dimension) and nonseparable functions are those for which this decomposition is not possible. In other words, separable functions are quite easy to solve, when compared with their nonseparable counterpart, because each variable of a function is independent of the others [49]. In Table 4.1, functions $f_1 - f_{13}$ are high-dimensional problems. In particular, functions $f_1 - f_5$ are unimodal problems while $f_7$ is a noisy problem and illustrates a simple convex shape occupied noisy variables in [0, 1) for its equation. Additionally classic Rosenbrock function $f_5$ defined in Table 4.2 belongs to unimodal class and has a tight optimum that is located in a narrow and bent shaped valley. For these unimodal function, $f_2$, $f_3$ and $f_5$ are nonseparable and relatively difficult to solve because their variables display inter-relation among themselves or are not independent. Moreover, $f_{10}$, $f_{11}$, $f_{12}$

and $f_{13}$ are nonseparable functions which are the hardest instances. Functions $f_8 - f_{13}$ are denoted as highly multimodal functions which are non-convex with many local optima, but all have strong symmetries around the global optimum, located at either 0 or others. Except that, functions $f_{14} - f_{23}$ have lower dimensions of decision variables for the multimodal problems. The convergence behavior of the hybrid algorithm highly affects the final results of convex problems. The best solutions obtained for these multimodal problems is important because they impact a method's capability of jumping from poorly locally optima and getting stuck in sub-optimal solution.

Table 4.1: Features of 23 classical benchmarks

| *Function* | *Name* | *Dim* | *Characteristics* |
|---|---|---|---|
| $f_1$ | Sphere | 30 | Unimodal, separable |
| $f_2$ | Schwefel 2.22 | 30 | Unimodal, nonseparable |
| $f_3$ | Schwefel 1.2 | 30 | Unimodal, nonseparable |
| $f_4$ | Schwefel 2.21 | 30 | Unimodal, separable |
| $f_5$ | Rosenbrock | 30 | Unimodal, nonseparable |
| $f_6$ | Step | 30 | Unimodal, separable, discontinuous |
| $f_7$ | Quartic | 30 | Unimodal, separable |
| $f_8$ | Schwefel 2.26 | 30 | Highly Multimodal, separable |
| $f_9$ | Rastrigin | 30 | Highly Multimodal, separable |
| $f_{10}$ | Ackley | 30 | Highly Multimodal, nonseparable |
| $f_{11}$ | Griewank | 30 | Highly Multimodal, nonseparable |
| $f_{12}$ | Levy | 30 | Highly Multimodal, nonseparable |
| $f_{13}$ | Levy 8 | 30 | Highly Multimodal, nonseparable |
| $f_{14}$ | Shekel Foxholes | 2 | Basic Multimodal, separable |
| $f_{15}$ | Kowalik | 4 | Basic Multimodal, nonseparable |
| $f_{16}$ | Six-Hump Camel | 2 | Basic Multimodal, nonseparable |
| $f_{17}$ | Branin | 2 | Basic Multimodal, separable |
| $f_{18}$ | Goldstein-Price | 2 | Basic Multimodal, nonseparable |
| $f_{19}$ | Hartman 4 | 4 | Basic Multimodal, nonseparable |
| $f_{20}$ | Hartman 6 | 6 | Basic Multimodal, nonseparable |
| $f_{21}$ | Shekel 5 | 4 | Basic Multimodal, nonseparable |
| $f_{22}$ | Shekel 7 | 4 | Basic Multimodal, nonseparable |
| $f_{23}$ | Shekel 10 | 4 | Basic Multimodal, nonseparable |

Table 4.2: Description (definition, search domain and functional value of minima) of 23 classical benchmark test functions

| *Algebraic Equation of Test Function* | *Search Domain* | *Optimum* |
|---|---|---|
| $f_1(x)=\sum_{i=1}^{n}x_i^2$ | [-100, 100]$^D$ | 0 |
| $f_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$ | [-10, 10]$^D$ | 0 |
| $f_3(x)=\sum_{i=1}^{n}(\sum_{j=1}^{i}x_j)^2$ | [-100, 100]$^D$ | 0 |
| $f_4(x) = \max_i\{ \lvert x_i\rvert, 1 \le i \le \text{n} \}$ | [-100, 100]$^D$ | 0 |
| $f_5(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$ | [-30, 30]$^D$ | 0 |
| $f_6(x)=\sum_{i=1}^{n}(\lfloor x_i+0.5\rfloor)^2$ | [-100, 100]$^D$ | 0 |
| $f_7(x)=\sum_{i=1}^{n}ix_i^4+random[0,1)$ | [-1.28, 1.28]$^D$ | 0 |
| $f_8(x)=\sum_{i=1}^{n}-x_i\sin(\sqrt{\lvert x_i\rvert})$ | [-500, 500]$^D$ | -12569.5 |
| $f_9(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$ | [-5.12, 5.12]$^D$ | 0 |
| $f_{10}(x)=-20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2})$ $-\exp(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i)+20+e$ | [-32, 32]$^D$ | 0 |
| $f_{11}(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})+1$ | [-600, 600]$^D$ | 0 |
| $f_{12}(x)=\frac{\pi}{n}\{10\sin^2(\pi Y_1)$ $+\sum_{i=1}^{n-1}(Y_i-1)^2[1+10\sin^2(\pi Y_{i+1})]+(Y_n-1)^2\}$ $+\sum_{i=1}^{n}u(x_i,10,100,4),$ $Y_i=1+\frac{1}{4}(x_i+1)$ $u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m, & \text{if } x_i>a,\\ 0, & \text{if } -a\le x_i\le a,\\ k(-x_i-a)^m, & \text{if } x_i<-a.\end{cases}$ | [-50, 50]$^D$ | 0 |
| $f_{13}(x)=0.1\{\sin^2(3\pi x_1)$ $+\sum_{i=1}^{n-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]$ $+(x_n-1)[1+\sin^2(2\pi x_n)]$ $+\sum_{i=1}^{n}u(x_i,5,100,4)$ | [-50, 50]$^D$ | 0 |
| $f_{14}(x)=\left[\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right]^{-1}$ | [-65,536, 65.536]$^D$ | 0.998004 |

| | | |
|---|---|---|
| $f_{15}(x)=\sum_{i=1}^{11}\left[a_i-\frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}\right]^2$ | $[-5, 5]^D$ | 0.0003075 |
| $f_{16}(x)=((4+\frac{1}{3}x_1^4)-2.1x_1^2)x_1^2+x_1x_2+(-4+4x_2^2)x_2^2$ | $[-5, 5]^D$ | -1.03163 |
| $f_{17}(x)=(x_2-\frac{5.1}{4\pi^2}x_1+\frac{5}{\pi}x_1-6)^2$ $+10(1-\frac{1}{8\pi})\cos(x_1)+10$ | [-5, 10] × [0, 15] | 0.398 |
| $f_{18}(x)=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2$ $+6x_1x_2+3x_2^2)]\times[30+(2x_1-3x_2)^2(18-32x_1$ $+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$ | $[-2, 2]^D$ | 3 |
| $f_{19}(x)=-\sum_{i=1}^{4}c_i\exp\left[-\sum_{j=1}^{4}(a_{ij}(x_j-p_{ij})^2\right]$ | $[0, 1]^D$ | -3.86 |
| $f_{20}(x)=-\sum_{i=1}^{4}c_i\exp\left[-\sum_{j=1}^{6}(a_{ij}(x_j-p_{ij})^2\right]$ | $[0, 1]^D$ | -3.32 |
| $f_{21}(x)=-\sum_{i=1}^{5}[(x_j-a_{ij})(x_j-a_{ij})^T+c_i]^{-1}$ | $[0, 10]^D$ | -10.1532 |
| $f_{22}(x)=-\sum_{i=1}^{7}[(x_j-a_{ij})(x_j-a_{ij})^T+c_i]^{-1}$ | $[0, 10]^D$ | -10.4029 |
| $f_{23}(x)=-\sum_{i=1}^{10}[(x_j-a_{ij})(x_j-a_{ij})^T+c_i]^{-1}$ | $[0, 10]^D$ | -10.5364 |

This experiment is conducted in 50 independent runs for every algorithm and every test problem, leading a total of 50 × 23 = 1150 independent runs. The experimental formalities are identical to the ones introduced in [47]. The maximum number of function evaluations for all problems are presented in Table 4.3.

Table 4.3: Maximum number of function evaluations as introduced by Yao et al.

| *Func.* | *#FEs* | *Func.* | *#FEs* | *Func.* | *#FEs* |
|---|---|---|---|---|---|
| $f_1$ | 150,000 | $f_9$ | 500,000 | $f_{17}$ | 10,000 |
| $f_2$ | 200,000 | $f_{10}$ | 150,000 | $f_{18}$ | 10,000 |
| $f_3$ | 500,000 | $f_{11}$ | 200,000 | $f_{19}$ | 10,000 |
| $f_4$ | 500,000 | $f_{12}$ | 150,000 | $f_{20}$ | 20,000 |
| $f_5$ | $2\times10^6$ | $f_{13}$ | 150,000 | $f_{21}$ | 10,000 |
| $f_6$ | 150,000 | $f_{14}$ | 10,000 | $f_{22}$ | 10,000 |
| $f_7$ | 300,000 | $f_{15}$ | 400,000 | $f_{23}$ | 10,000 |
| $F_8$ | 900,000 | $f_{16}$ | 10,000 | | |

According to successful implementations in literature, some heuristic algorithms are considered for comparison and evaluation, such as fast evolutionary programming (FEP) [47], differential evolution [50], particle swarm optimization (PSO) [50, 51], artificial bee colony optimization (ABC) [52, 53], simple evolutionary algorithms (SEA) [50, 54] and two-staged memory Great Deluge Algorithms (TSM_GDA) [55]. In those algorithms, TSM_GDA is recent variant of GDA which used two different referenced templates of memories to operate searching schemes and reserve the best possible solution under the structure of a Great Deluge Algorithm (GDA) [55]. By contrast, FEP shows a distinct probability of evolution model, which proposed a mutation scheme related to Cauchy distribution for creating random sampling and enhanced the classical evolution program (EP) algorithm during its operation.

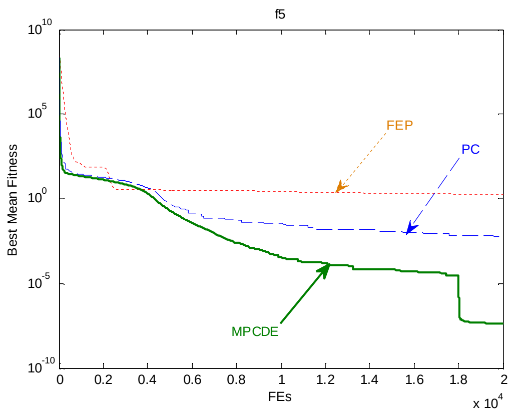
All simulation outputs of means and standard deviations obtained from TSM-GDA, FEP, PC, and MPCDE methods are shown in Table 4.4. It can apparently be observed that the proposed MPCDE algorithm performs better than other methods for the test functions according to their values of mean and standard deviation. In particular, achievements yielded from DE in terms of the adaptive mutation operator provide alternative valuable ways with perturbed parameter schemes for global evolution search. It is obviously shown that MPCDE algorithm is better than other participated algorithms from functions $f_8 - f_{13}$. Considering the other functions $f_{14} - f_{23}$, the TSM-GDA algorithm and MPCDE have similarity on their achievements and beated the rest of two approaches with their mean values and standard deviations because they have lesser dimensions of decision variables and get stuck in a limited set of locally minimum solutions. In all these instances from Table 4.4, the MPCDE algorithm generally has excellent performances for all problems through all methods.

Table 4.4: Comparative results of *mean* (in first row for every function) and *standard deviation* (in next row) derived from algorithm of PC, MPCDE, TSM-GDA and FEP for the set of classical benchmark problems over 50 runs

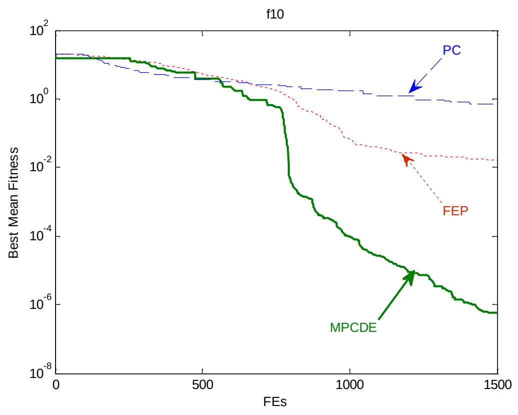
| *Function* | *PC* | *MPCDE* | *TSM-GDA* | *FEP* |
|---|---|---|---|---|
| $f_1$ | 6.591e-5 | **6.889e-8** | 7.513e-6 | 5.7e−4 |
| | 1.301e-5 | **1.938e-7** | 1.340e-5 | 1.3e−4 |
| $f_2$ | 4.304e-3 | **4.606e-6** | 2.569e-5 | 8.1e−3 |
| | 1.252e-3 | **8.856e-6** | 3.575e-5 | 7.7e−4 |
| $f_3$ | 9.938e-3 | **3.351e-7** | 2.038e-6 | 1.6e−2 |
| | 3.590e-3 | **3.343e-7** | 6.300e-6 | 1.6e−2 |
| $f_4$ | 3.715 | **3.908e-5** | 2.891e-3 | 0.3 |
| | 2.353e-1 | **2.795e-5** | 2.606e-3 | 0.5 |
| $f_5$ | 8.476e-3 | **4.684e-10** | 7.021e-6 | 5.06 |
| | 1.784e-3 | **5.697e-10** | 1.652e-5 | 5.87 |
| $f_6$ | **0.00** | **0.0** | **0.0** | **0.0** |
| | **0.00** | **0.0** | **0.0** | **0.0** |
| $f_7$ | 7.911e-3 | **1.813e-4** | 1.604e−3 | 7.6e−3 |
| | 1.751e-3 | **1.110e-4** | 1.255e−3 | 2.6e−3 |
| $f_8$ | -1.143e+4 | **-1.256e+4** | -12,542.88 | −12,554.5 |
| | 7.168e+2 | **14.01** | 43.15 | 52.6 |
| $f_9$ | 2.666 | **4.832e-13** | 3.208e−8 | 4.6e−2 |
| | 1.156 | **1.095e-12** | 5.214e−8 | 1.2e−2 |
| $f_{10}$ | 8.734e-1 | **1.008e-6** | 5.527e−5 | 1.8e−2 |
| | 9.337e-1 | **4.763e-7** | 6.177e−5 | 2.1e−3 |
| $f_{11}$ | 3.221e-1 | **5.115e-5** | 1.743e−3 | 1.6e−2 |
| | 5.642e-1 | **9.683e-6** | 5.444e−3 | 2.2e−2 |
| $f_{12}$ | 1.823e-1 | **1.325e-19** | 2.437e−10 | 9.2e−6 |
| | 9.563e-1 | **6.382e-20** | 4.027e−10 | 3.6e−6 |
| $f_{13}$ | 1.932e-1 | **8.370e-19** | 9.166e−8 | 1.6e−4 |
| | 2.785e-1 | **8.053e-19** | 2.626e−7 | 7.3e−5 |
| $f_{14}$ | 1.016 | **0.998** | **0.998** | 1.22 |
| | 1.634e-1 | 1.051e-3 | **1.215e−11** | 0.56 |
| $f_{15}$ | 1.160e-3 | **3.075e-4** | **3.075e−4** | 5.0 e−4 |
| | 1.354e-4 | **8.439e-11** | 1.188e−10 | 3.2e−4 |
| $f_{16}$ | **-1.032** | **-1.032** | **−1.032** | −1.031 |
| | 5.342e-5 | 5.274e-9 | **5.776e−12** | 4.9e−7 |
| $f_{17}$ | **0.398** | **0.398** | **0.398** | **0.398** |
| | 7.700e-4 | **2.364e-10** | 9.600e−9 | 1.5e−7 |
| $f_{18}$ | **3.000** | **3.000** | **3.000** | 3.02 |
| | 7.777e-6 | 1.296e-6 | **3.877e−7** | 0.11 |
| $f_{19}$ | **-3.863** | **-3.863** | **−3.863** | 3.86 |
| | 2.129e-6 | 9.737e-7 | **5.053e−14** | 1.4e−5 |
| $f_{20}$ | -3.298 | **-3.312** | −3.297 | −3.27 |
| | 4.827e-2 | **3.264e-2** | 1.758e−2 | 5.9e−2 |
| $f_{21}$ | **-10.153** | **-10.153** | **−10.153** | −5.52 |
| | 3.078e-5 | **1.713e-10** | 3.775e−5 | 1.59 |
| $f_{22}$ | -10.059 | **-10.402** | **−10.402** | −5.52 |
| | 1.375 | **9.528e-11** | 3.641e−5 | 2.12 |
| $f_{23}$ | -10.198 | **-10.536** | **−10.536** | −6.57 |
| | 1.385 | **2.046e-10** | 1.695e−5 | 3.14 |

In order to gain better results as a reward, PC random search exchange the knowledge with CE method and DE operation coordinated with multi-agent approach for the communication of adaptive techniques to accomplish given optimization tasks more efficiently and robustly. Therefore, the proposed method presents superior mean results compared to other three competitors for all the 23 test problems.

The convergence speeds of the MPCDE algorithm is illustrated for two classical multimodal problems: Rosenbrock and Ackley, that are described in Figure 4.4. Horizontal axis shows the FEs and vertical axis captures the mean fitness extracted solutions. Obviously, it is evident from Figure 4.4(a) that MPCDE algorithm yields significantly promising performance in comparison on Rosenbrock problem. Three algorithms starts with nearly similar values, but the MPCDE algorithm performs faster towards the global optima at 6000 FEs and obtains a approximate value of $10^{-5}$ at about 18000 FEs, whereas the PC approach can only yield about value of 0.01 and gets stuck into a locally optimal solution. The MPCDE algorithm is advanced PC approach in the evolutionary runs and this is mainly because the MPCDE algorithm emphasizes the smoothing factors of distribution operators of CE concept to update strategy vector’s search operator. Later, the MPCDE algorithm improves convergence speeds since it attempts to combine with the updated DE operation for exploring the global optimal solution by use of search distances based on scaling mutated schemes in promising solution of search spaces. On the contrary, the FEP and PC approaches maintain almost identical convergence speeds. It can be seen from Figure 4.4(b) that, MPCDE obviously has better performance than the FEP and PC approaches such that all selected algorithms display similarly converging speeds at 600 FEs or earlier. Since the proposed algorithm explores a quite narrow localized neighbor problem space for earlier time, it applies dynamical smoothing updates of distribution operators to improve the speed of convergences and arrives at about 1 over 750 FEs.



(a) *Rosenbrock* *f*5



(b) *Ackley* *f*10

Figure 4.4: Evolution plots in comparisons with PC, FEP, and MPCDE, PC on (a) Rosenbrock and (b) Ackley Function

Table 4.5 demonstrates comparative results among the algorithms DE, PSO, ABC, SEA, TSM-GDA, and MPCDE for the 23 test benchmarks for 30-trials and 5.0e+5 FEs. ABC is corporately developed by Karaboga and Basturk [52], Karaboga and Akay [53], while the PSO, SEA, and DE algorithm's outputs are available in [50]. The TSM-GDA algorithm's outputs are available in [55]. Given a set of the single objective functions, first analytical comparison of their evaluations was undertaken with MPCDE and DE to reveal the hybridized strengths of the proposed approach against the classical DE algorithm. From this table, The DE method has better performance than MPCDE for functions $f_{10}$, $f_{12}$, and $f_{13}$ respectively. In contrast, TSM-GDA yields the best result for function $f_{23}$ only. Note that the algorithm of DE, PSO and SEA produce the poor results for test functions $f_{19}$ and $f_{20}$ in comparison with other algorithms. Simulation evaluations show that MPCDE outperforms 6 various algorithms on entire problems except for $f_3$, $f_{10}$, $f_{12}$, $f_{13}$, $f_{23}$, respectively. For function $f_3$ PSO obtains optimal result in comparison to the MPCDE algorithm.

Table 4.5: Comparative results of the *mean* (in first row for each function), and the *standard deviation* (in new row) derived from algorithm of ABC, DE, TSM-GDA, SEA, PSO, and MPCDE within 5.0e+5 FEs over 30 runs

| *Func.* | *MPCDE* | *DE* | *PSO* | *ABC* | *SEA* | *TSM-GDA* |
|---|---|---|---|---|---|---|
| $f_1$ | **0.0** | **0.0** | **0.0** | 5.02e−47 | 1.79e−3 | 8.087e−10 |
| | **0.0** | **0.0** | **0.0** | 4.10e−47 | 2.77e−4 | 1.959e−9 |
| $f_2$ | **0.0** | **0.0** | **0.0** | 2.87e−31 | 1.72e−2 | 3.644e−6 |
| | **0.0** | **0.0** | **0.0** | 1.62e−31 | 1.7e−3 | 3.592e−6 |
| $f_3$ | 5.955e-14 | 2.020e−9 | **0.0** | 3.04e+3 | 1.59e−2 | 3.870e−8 |
| | 1.730e-14 | 8.26e−10 | **0.0** | 4.55e+2 | 4.25e−3 | 5.835e−8 |
| $f_4$ | **3.645e-17** | 3.850e−8 | 2.11e−16 | 39.19 | 1.98e−2 | 3.602e−4 |
| | **3.986e-17** | 9.17e−9 | 8.01e−16 | 3.42 | 2.07e−3 | 2.375e−4 |
| $f_5$ | **1.099e-13** | 29.0 | 4.026 | 1.417 | 31.32 | 2.462e−3 |
| | **5.835e-14** | 0.0 | 4.99 | 2.23 | 17.4 | 3.609e−3 |
| $f_6$ | **0.0** | **0.0** | 4.0e−2 | **0.0** | **0.0** | **0.0** |
| | **0.0** | **0.0** | 1.98e−1 | **0.0** | **0.0** | **0.0** |
| $f_7$ | **1.193e-4** | 3.939e−3 | 1.91e−3 | 0.10 | 7.11e−4 | 4.980e−3 |
| | **1.173e-4** | 1.13e−3 | 1.14e−3 | 0.02 | 3.27e−4 | 2.691e−3 |
| $f_8$ | **−1.257e+4** | **−1.257e+4** | −7.187e+3 | **−1.257e+4** | −1.167e+4 | **−1.257e+4** |
| | **2.933e-8** | 2.30e−4 | 6.72e+2 | 5.87e−8 | 2.34e+2 | 1.39e−4 |
| $f_9$ | **0.0** | **0.0** | 49.17 | **0.0** | 7.18e−1 | 2.869e−12 |
| | **0.0** | **0.0** | 16.2 | **0.0** | 9.22e−1 | 8.289e−12 |
| $f_{10}$ | 5.587e-15 | **8.881e−16** | 1.4 | 3.18e−14 | 1.05e−2 | 1.370e−5 |
| | 1.688e-15 | **7.03e−16** | 7.91e−1 | 3.76e−15 | 9.08e−4 | 2.737e−5 |
| $f_{11}$ | **0.0** | **0.0** | 2.35e−2 | **0.0** | 4.64e−3 | 1.997e−6 |
| | **0.0** | **0.0** | 3.54e−2 | **0.0** | 3.96e−3 | 3.811e−6 |
| $f_{12}$ | 1.571e-32 | **0.0** | 3.819e−1 | 2.59e−4 | 4.56e−6 | 3.009e−12 |
| | 1.113e-47 | **0.0** | 8.4e−1 | 9.36e−5 | 8.11e−7 | 6.549e−12 |
| $f_{13}$ | 1.350e-32 | **0.0** | −5.969e−1 | 1.10e−3 | −1.143 | 1.896e−9 |
| | 2.782e-48 | **0.0** | 5.17e−1 | 3.64e−4 | 1.34e−5 | 4.689e−9 |
| $f_{14}$ | **9.980e-1** | **9.98e−1** | 1.157 | **9.980e−1** | **9.98e−1** | **9.98e−1** |
| | 9.178e-6 | 3.75e−8 | 3.68e−1 | 3.21e−14 | 4.33e−8 | **2.024e−14** |
| $f_{15}$ | **3.075e-4** | 4.173e−4 | 1.338e−3 | 3.90e−4 | 3.704e−4 | **3.075e−4** |
| | **5.511e-20** | 3.01e−4 | 3.94e−3 | 8.33e−5 | 8.78e−5 | **1.425e−9** |
| $f_{16}$ | **−1.032** | **−1.032** | **−1.032** | **−1.032** | **−1.032** | **−1.032** |
| | **0.0** | 1.92e−8 | 3.84e−8 | **0.0** | 3.16e−8 | **0.0** |
| $f_{17}$ | **3.979e-1** | **3.979e−1** | 3.98e−1 | 0.3979 | 3.98e−1 | **3.979e−1** |
| | **1.129e-16** | 1.17e−8 | 5.01e−9 | 3.27e−10 | 2.20e−8 | 1.521e−10 |
| $f_{18}$ | **3.00** | **3.0** | **3.0** | **3.00** | **3.0** | **3.0** |
| | 9.029e-16 | 0.0 | 0.0 | **6.28e−16** | 0.0 | 3.078e−10 |
| $f_{19}$ | **-3.862** | – | – | **−3.862** | – | **−3.862** |
| | **3.160e-15** | – | – | 2.77e−11 | – | 2.657e−13 |
| $f_{20}$ | **-3.322** | – | – | **−3.322** | – | **−3.322** |
| | **1.037e-8** | – | – | 3.35e-8 | – | 3.103e−8 |
| $f_{21}$ | **−10.15** | **−10.15** | −5.4 | **−10.15** | −8.41 | **−10.15** |
| | 4.941e-6 | 4.60e−7 | 3.40 | 4.60e−7 | 3.16 | **1.768e−8** |
| $f_{22}$ | **-10.40** | **−10.40** | −6.946 | **−10.40** | −8.912 | **−10.40** |
| | **0.0** | 3.58e−7 | 3.70 | 3.11e−8 | 2.86 | 7.421e−9 |
| $f_{23}$ | -10.536 | −10.53 | −6.71 | −10.536 | −9.8 | **−10.541** |
| | **9.029e-15** | 2.09e−7 | 3.77 | 2.02e−8 | 2.24 | 9.501e−9 |

MPCDE is able to escape from poor locally optima of the PC approach and explores promising strategy vectors with corresponding parameters, which results in guiding the appropriate search direction. In this respect, for *Rosenbrock* function ($f_5$) where MPCDE found better results in comparison to the rest four algorithms. This is a popular problem that has multiple features involving nonseparable, differentiable, scalable, and unimodal, where a globally optimum point is held in a cramped, extended, curved valley with bending parabola. The search distance based adaptive mutation scheme for MPCDE enables the hybrid algorithm to determine the high quality solutions in the complex landscape. Even though DE shows better performance than MPCDE and other three algorithms for Ackley problem $f_{10}$, MPCDE still reveals the 2nd superior achievement over some test functions: $f_3$, $f_{10}$, $f_{12}$, $f_{13}$ and $f_{23}$. For lower dimensional functions $f_{14}$ – $f_{23}$, TSM-GDA shows better achievements over 6 competitors because of its two-stage external memory architecture. Nonetheless, the proposed hybrid algorithm generally has preferable achievements in comparison with DE, PSO, ABC, and SEA on the rest of functions.

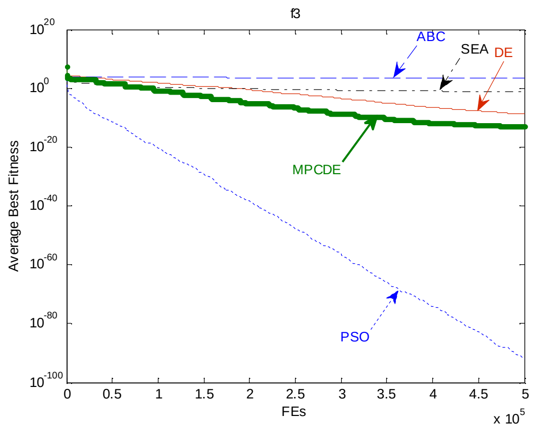
Wilcoxon signed ranks test is provided for comparison between MPCDE algorithm and other methods for the sake of assessing statistical similarities of the associated scores. In the interest of testifying the population distribution of each test functions for every method, procedural parameters of competitors were defined in [50] to evaluate the simulation scores presented in Table 4.6. The scores of ABC, DE, SEA, and PSO algorithms can be found in [52]. Based on revealed results, the test of null presumption should be addressed to the same populations of MPCDE and other methods. Thereby the approximate sums of the ranks are estimated. If the difference among sum of ranks is too large, the test tool rejects the null presumption that the mean of populations are the same.

*Signrank* method with *Matlab* examines the sets of populations by the use of Wilcoxon signed ranks test. A matched coupling values in statistical signed rank $p$ and $h$ can be computed based on a factor $\alpha = 0.05$. Paires of values $[p, h]$ measured with the *signrank* tool are illustrated in Table 4.6. The statistical test yields the following results: eliminated the case of void presumption ($p \leq \alpha$) for $h = 1$, otherwise unable to discard the situation of void presumption ($p > \alpha$) for $h = 0$. In this respect, the value $p$ principally makes decision whether the test information sustains the case of null hypothesis or not. According to the indication from Table 4.5, the statistical test deletes null hypothesis for most instances as $h = 1$. Besides, value $p$ is quite minor representing extremely accuracy of the null hypothesis for the most cases as $h = 0$. These outcomes of the statistical tests apparently show that the population of distribution of MPCDE is totally different from its other opponents.

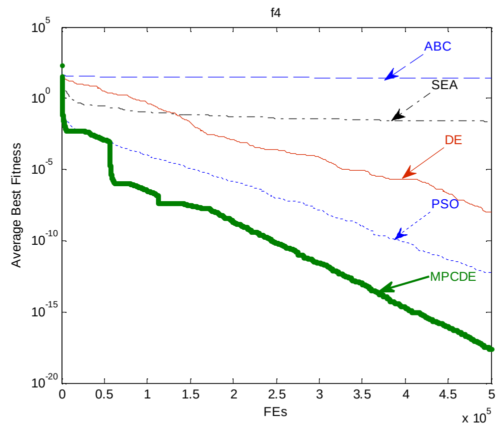
Table 4.6: Results of statistical tests between MPCDE and its adversaries for each function created by the *Wilcoxon tool* of population: ($p$,$h$)

| *Func.* | *MPCDE vs. DE* | *MPCDE vs. PSO* | *MPCDE vs. ABC* | *MPCDE vs. SEA* |
|---|---|---|---|---|
| $f_1$ | (1.0, 0.0) | (1.0, 0.0) | (1.73e-6, 1.0) | (1.73e-6, 1.0) |
| $f_2$ | (1.0, 0.0) | (1.0, 0.0) | (1.73e-6, 1.0) | (1.73e-6, 1.0) |
| $f_3$ | (1.73e-6, 1.0) | (1.73e-6, 1.0) | (1.73e-6, 1.0) | (1.73e-6, 1.0) |
| $f_4$ | (1.73e-6, 1.0) | (1.06e-4, 1.0) | (1.73e-6, 1.0) | (1.73e-6, 1.0) |
| $f_5$ | (1.67e-6, 1.0) | (5.79e-5, 1.0) | (3.88e-4, 1.0) | (1.73e-6, 1.0) |
| $f_6$ | (1.0, 0.0) | (1.50e-1, 0.0) | (1.0, 0.0) | (1.0, 0.0) |
| $f_7$ | (1.73e-6, 1.0) | (1.73e-6, 1.0) | (1.02e-5, 1.0) | (1.57e-2, 1.0) |
| $f_8$ | (1.73e-6, 1.0) | (1.73e-6, 1.0) | (1.73e-6, 1.0) | (1.73e-6, 1.0) |
| $f_9$ | (1.0, 0.0) | (2.84e-5, 1.0) | (1.0, 0.0) | (1.73e-6, 1.0) |
| $f_{10}$ | (1.73e-6, 1.0) | (1.73e-6, 1.0) | (1.73e-6, 1.0) | (1.73e-6, 1.0) |
| $f_{11}$ | (1.0, 0.0) | (9.63e-4, 1.0) | (1.0, 0.0) | (1.73e-6, 1.0) |
| $f_{12}$ | (1.31e-5, 1.0) | (1.73e-6, 1.0) | (1.73e-6, 1.0) | (1.73e-6, 1.0) |
| $f_{13}$ | (1.44e-5, 1.0) | (9.71e-5, 1.0) | (1.74e-4, 1.0) | (1.73e-6, 1.0) |
| $f_{14}$ | (1.72e-6, 1.0) | (3.11e-5, 1.0) | (1.72e-6, 1.0) | (1.73e-6, 1.0) |
| $f_{15}$ | (7.16e-4, 1.0) | (2.41e-4, 1.0) | (1.73e-6, 1.0) | (1.24e-5, 1.0) |
| $f_{16}$ | (3.59e-4, 1.0) | (3.58e-4, 1.0) | (4.32e-8, 1.0) | (3.59e-4, 1.0) |
| $f_{17}$ | (1.73e-6, 1.0) | (1.73e-6, 1.0) | (1.73e-6, 1.0) | (1.73e-6, 1.0) |
| $f_{18}$ | (1.73e-6, 1.0) | (1.71e-6, 1.0) | (6.15e-4, 1.0) | (9.62e-4, 1.0) |
| $f_{19}$ | (1.73e-6, 1.0) | (6.25e-2, 0.0) | (1.25e-1, 0.0) | (2.05e-4, 1.0) |
| $f_{20}$ | (3.11e-5, 1.0) | (1.5e-3, 1.0) | (1.73e-6, 1.0) | (1.73e-6, 1.0) |
| $f_{21}$ | (1.73e-6, 1.0) | (7.81e-1, 0.0) | (2.80e-1, 0.0) | (6.64e-4, 1.0) |
| $f_{22}$ | (4.45e-5, 1.0) | (5.79e-5, 1.0) | (1.73e-6, 1.0) | (4.17e-1, 0.0) |
| $f_{23}$ | (1.73e-6, 1.0) | (1.73e-6, 1.0) | (6.89e-5, 1.0) | (9.78e-2, 0.0) |

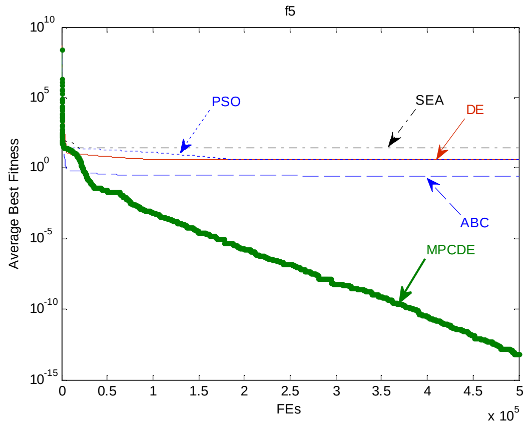
Different convergence plots of the mean optimal objective function of MPCDE and its other state-of-the-art metaheuristics are presented Figure 4.5 for some typical benchmark problems with various dimensions.



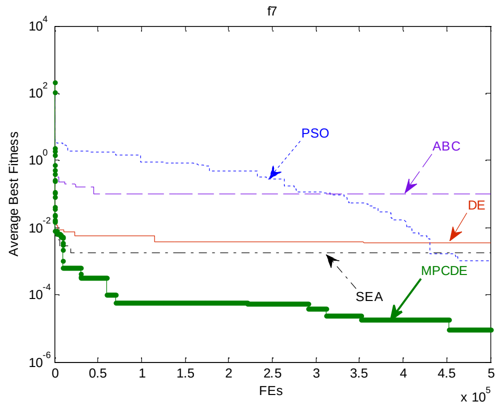
(a) *f*3



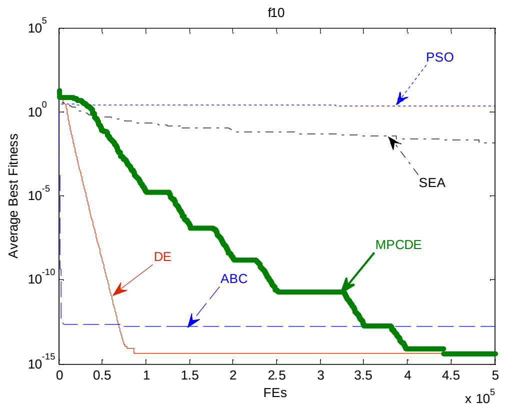
(b) *f*4



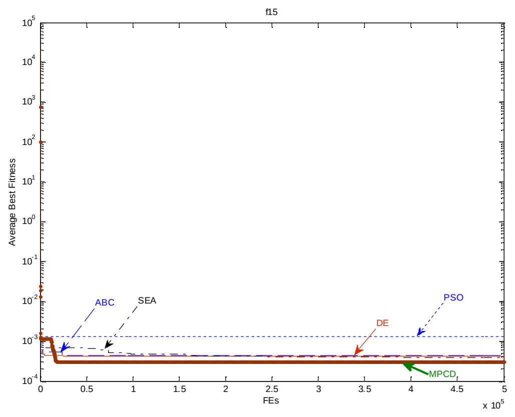
(c) *f*5



(d) *f*7



(e) *f*10



(f) *f*15

Figure 4.5: Evolutionary convergence of averaging optimal results yielded in ABC, DE, PSO, SEA and MPCDE with FEs for 6 problems

Most selected problems have various sizes and fitness landscape features in whole set of test benchmarks. Except for problem $f_3$ in Figure 4.5(a), the speed of convergence for the MPCDE algorithm has superior and quicker performance than its opponents for the problems $f_4$, $f_5$, $f_7$, $f_{10}$, and $f_{15}$ in Figure 4.5(b–f). This is due to the integrated strength of PC, CE and DE under the hybrid frameworks to improve global search ability for different types of selected functions. ABC illustrates a very fast convergence rate, but it may get trapped into local optimal solutions for case $f_5$ and $f_{10}$. MPCDE shows similar convergence results to the other methods and DE for $f_{10}$. Besides, it is observed from last figure for function $f_{15}$ that similar convergence rates has been obtained from MPCDE, DE, ABC and SEA because of lower dimensions. Observations have been done for the convergence results of the function $f_3$, $f_4$, $f_5$, $f_7$ and $f_{10}$, respectively. Obviously, it is evident from the Figure 4.5 that MPCDE showed faster speed of convergences on all problems but $f_3$. It is proved that MPCDE attempts to extract efficiently enhanced solutions due to its refining search schemes for parameters of procedures PC, CE, DE.

### 4.5.4 CEC2005 Benchmark Problems

In the interest of validating MPCDE for single objective optimization, the second set of experimental study is conducted on CEC2005 benchmark functions to compare its evaluations with state-of-the-art metaheuristics. These functions contain 25 different test problems. Most of the problems from this benchmark set are obtained using complex techniques of shifting, rotation, and scaling coordinate system based on the original equations that are given in the first set of benchmark functions. In terms of difficulty, most of the test functions are nonseparable, rotated and all of them are scalable as shown in Table 4.7. Except for two functions ($F_7$ and $F_{25}$), all other functions have limits of boundary constraints. Some of the benchmark functions contain their global minimum on the boundary. Similar with the first set of problems, these functions are also categoried into four classes which are presented in Table 4.7. This table lists name, domain of decision variable, optimum, modality, and related properties including rotated, separable, and scalable of the test functions.

Table 4.7: The Description (Name, Domain, Optimum, Type, Rotated (R), Separable (Se), and Scalable (Sc)) of the CEC 2005 benchmark functions (Y: Yes; N: No), U: unimodal, BM: Basic multimodal, HC: Hybrid Composition

| *#* | *Function Name* | *Initial Range* | *Opt.* | *Type* | *R* | *Se* | *Sc* |
|---|---|---|---|---|---|---|---|
| $F_1$ | Shifted Sphere Function | $[-100, 100]^D$ | -450 | U | N | Y | Y |
| $F_2$ | Shifted Schwefel's Problem 1.2 | $[-100, 100]^D$ | -450 | U | N | N | Y |
| $F_3$ | Shifted Rotated High Conditioned Elliptic Function | $[-100, 100]^D$ | -450 | U | Y | N | Y |
| $F_4$ | Shifted Schwefel's Problem 1.2 with Noise in Fitness | $[-100, 100]^D$ | -450 | U | N | N | Y |
| $F_5$ | Schwefel's Problem 2.6 with Global Optimum on Bounds | $[-100, 100]^D$ | -310 | U | N | N | Y |
| $F_6$ | Shifted Rosenbrock's Function | $[-100, 100]^D$ | -390 | BM | N | N | Y |
| $F_7$ | Shifted Rotated Griewank's Function without Bounds | $[0, 600]^D$ * | -180 | BM | Y | N | Y |
| $F_8$ | Shifted Rotated Ackley's Function with Global Optimum on Bounds | $[-32, 32]^D$ | -140 | BM | Y | N | Y |
| $F_9$ | Shifted Rastrigin's Function | $[-5, 5]^D$ | -330 | BM | N | Y | Y |
| $F_{10}$ | Shifted Rotated Rastrigin's Function | $[-5, 5]^D$ | -330 | BM | Y | N | Y |
| $F_{11}$ | Shifted Rotated Weierstrass Function | $[-0.5, 0.5]^D$ | 90 | BM | Y | N | Y |
| $F_{12}$ | Schwefel's Problem 2.13 | $[-\pi, \pi]^D$ | -460 | BM | N | N | Y |
| $F_{13}$ | Shifted Expanded Griewank's plus Rosenbrock's Function | $[-5, 5]^D$ | -130 | HC | N | N | Y |
| $F_{14}$ | Shifted Rotated Expanded Scaffer's $F_6$ | $[-100, 100]^D$ | -300 | HC | Y | N | Y |
| $F_{15}$ | Hybrid Composition Function 1 | $[-5, 5]^D$ | 120 | HC | N | Y | Y |
| $F_{16}$ | Rotated Version of Hybrid Composition Function 1 | $[-5, 5]^D$ | 120 | HC | Y | N | Y |
| $F_{17}$ | Rotated Hybrid Composition Function 1 with Noise in Fitness | $[-5, 5]^D$ | 120 | HC | Y | N | Y |
| $F_{18}$ | Rotated Hybrid Composition Function 2 | $[-5, 5]^D$ | 10 | HC | Y | N | Y |
| $F_{19}$ | Rotated Hybrid Composition Function 2 with Narrow Basin for Global Optimum | $[-5, 5]^D$ | 10 | HC | Y | N | Y |
| $F_{20}$ | Rotated Hybrid Composition Function 2 with the Global Optimum on the Bounds | $[-5, 5]^D$ | 10 | HC | Y | N | Y |
| $F_{21}$ | Rotated Hybrid Composition Function 3 | $[-5, 5]^D$ | 360 | HC | Y | N | Y |
| $F_{22}$ | Rotated Hybrid Composition Function 3 with High Condition Number Matrix | $[-5, 5]^D$ | 360 | HC | Y | N | Y |
| $F_{23}$ | Non-Continuous Rotated Hybrid Composition Function 3 | $[-5, 5]^D$ | 360 | HC | Y | N | Y |
| $F_{24}$ | Rotated Hybrid Composition Function 4 | $[-5, 5]^D$ | 260 | HC | Y | N | Y |
| $F_{25}$ | Rotated Hybrid Composition Function 4 without Bounds | $[2, 5]^D$ * | 260 | HC | Y | N | Y |

* Note: No boundary for search range.

Complete explanations of these functions consisting of the concepts, clarification and their associated function evaluations conditions are available in [48]. Each problem specifies the initialization space as well as the required accuracy level. These functions are solved by the proposed method for three dimensional sizes: $D$=10, $D$=30 and $D$=50 and a maximum number of function evaluations (Max_FEs) is set as $10000 \times D$ for these functions. All the investigated algorithms are repeatedly run in 25 independent trials to report all mean objective function values. All outcomes related to the revealed state-of-the-art metaheuristic methods are available in [55].

In order to measure the success of the proposed approach from its DE hybridization point of view, three additional DE hybridizations are selected from literature. These are self-adaptive hybrid differential evolution with simulated annealing algorithm (SaDESA) [56], self-adaptable genetically programming DE (SaGPDE) [57], and hybrid real-coded GA (RCGA) [58].

Considering problems with 10 dimensions, Table 4.8 shows the mean fitness values of the introduced algorithm compared to the scores of 14 well-known metaheuristics for twenty-five benchmarks of CEC2005 competition. It is very clear that MPCDE is better for fourteen out of twenty-five problems. In comparison to other methods, the MPCDE algorithm presented better results for 2 unimodal problems: $F_3$, $F_5$ and majority of the multimodal problems $F_6$, $F_7$, and $F_{12}$. In addition, it is as good as other methods for the rest of multimodal functions $F_8$, $F_{11}$, $F_{14}$. Hybrid composition functions are more difficult than the other methods because of the synthetic techniques for combination of several classical benchmark problems.

Table 4.8: Comparative results of *mean* values among 14 and MPCDE Algorithms for CEC2005 test benchmarks with 10-dimension of 1.0e+5 FEs over 25 runs

| *Alg.* | $F_1$ | $F_2$ | $F_3$ | $F_4$ | $F_5$ | $F_6$ | $F_7$ |
|---|---|---|---|---|---|---|---|
| BLX-GL50 | 1.00e-9 | 1.00e-9 | 5.71e+2 | 1.00e-9 | 1.00e-9 | 1.00e-9 | 1.17e-2 |
| BLX-MA | 1.00e-9 | 1.00e-9 | 4.77e+4 | 2.00e-8 | 2.12e-2 | 1.49 | 1.97e-1 |
| COEVO | 1.00e-9 | 1.00e-9 | 1.00e-9 | 1.00e-9 | 2.13 | 1.25e+1 | 3.71e-2 |
| DE | 1.00e-9 | 1.00e-9 | 1.94e-6 | 1.00e-9 | 1.00e-9 | 1.59e-1 | 1.46e-1 |
| DMS-L-PSO | 1.00e-9 | 1.00e-9 | 1.00e-9 | 1.89e-3 | 1.14e-6 | 6.89e-8 | 4.52e-2 |
| EDA | 1.00e-9 | 1.00e-9 | 2.12e+1 | 1.00e-9 | 1.00e-9 | 4.18e-2 | 4.20e-1 |
| G-CMA-ES | 1.00e-9 | 1.00e-9 | 1.00e-9 | 1.00e-9 | 1.00e-9 | 1.00e-9 | 1.00e-9 |
| K-PCX | 1.00e-9 | 1.00e-9 | 4.15e-1 | 7.94e-7 | 4.85e+1 | 4.78e-1 | 2.31e-1 |
| L-CMA-ES | 1.00e-9 | 1.00e-9 | 1.00e-9 | 1.76e+6 | 1.00e-9 | 1.00e-9 | 1.00e-9 |
| L-SADE | 1.00e-9 | 1.00e-9 | 1.67e-5 | 1.42e-5 | 1.23e-2 | 1.20e-8 | 1.99e-2 |
| SPC-PNX | 1.00e-9 | 1.00e-9 | 1.08e+5 | 1.00e-9 | 1.00e-9 | 1.89e+1 | 8.26e-2 |
| TSM-GDA | 1.69e−9 | 4.07e−6 | 3.42e+4 | 2.15e+1 | 3.11e−2 | 6.31e−2 | 1.28e−1 |
| SaDESA | **0.00e+0** | **0.00e+0** | 1.60e-6 | **0.00e+0** | 1.14e-3 | 1.59e-1 | 4.57e-2 |
| RCGA | 8.34e-9 | 8.21e-9 | 5.71e+2 | 8.32e-9 | 8.94e-9 | 8.87e-9 | 1.17e-2 |
| MPCDE | 7.05e-14 | 6.82e-14 | **7.28e-14** | 8.64e-14 | **4.51e-12** | **7.96e-14** | **8.30e-14** |
|  | $F_8$ | $F_9$ | $F_{10}$ | $F_{11}$ | $F_{12}$ | $F_{13}$ | $F_{14}$ |
| BLX-GL50 | 2.04e+1 | 1.15 | 4.97 | 2.33 | 4.07e+2 | 7.50e-1 | 2.17 |
| BLX-MA | 2.02e+1 | 4.38e-1 | 5.64 | 4.56 | 7.43e+1 | 7.74e-1 | 2.03 |
| COEVO | 2.03e+1 | 1.92e+1 | 2.68e+1 | 9.03 | 6.05e+2 | 1.14 | 3.71 |
| DE | 2.04e+1 | 9.55e-1 | 1.25e+1 | 8.47e-1 | 3.17e+1 | 9.77e-1 | 3.45 |
| DMS-L-PSO | **2.00e+1** | 1.00e-9 | 3.62 | 4.62 | 2.40 | 3.69e-1 | 2.36 |
| EDA | 2.03e+1 | 5.42 | 5.29 | 3.94 | 4.42e+2 | 1.84 | 2.63 |
| G-CMA-ES | **2.00e+1** | 2.39e-1 | **7.96e-2** | 9.34e-1 | 2.93e+1 | 6.96e-1 | 3.01 |
| K-PCX | **2.00e+1** | 1.19e-1 | 2.39e-1 | 6.65 | 1.49e+2 | 6.53e-1 | 2.35 |
| L-CMA-ES | **2.00e+1** | 4.49e+1 | 4.08e+1 | 3.65 | 2.09e+2 | 4.94e-1 | 4.01 |
| L-SADE | **2.00e+1** | 1.00e-9 | 4.97 | 4.89 | 4.50e-7 | 2.20e-1 | 2.92 |
| SPC-PNX | 2.10e+1 | 4.02 | 7.30 | 1.91 | 2.60e+2 | 8.38e-1 | 3.05 |
| TSM-GDA | **2.0e+1** | 6.69e-11 | 5.72 | 4.42 | 8.04e−1 | **7.27e-2** | 2.94 |
| SaDESA | 2.02e+1 | **0.00e+0** | 4.63 | 5.22 | 1.40e+2 | 4.87e-1 | 2.80 |
| RCGA | 2.04e+1 | 1.15 | 4.97 | 2.33 | 4.07e+2 | 7.50e-1 | 2.17 |
| MPCDE | **2.00e+1** | 5.68e-14 | 9.44e-1 | **1.33e-1** | **1.75E-9** | 5.87e-1 | **1.89** |
|  | $F_{15}$ | $F_{16}$ | $F_{17}$ | $F_{18}$ | $F_{19}$ | $F_{20}$ | $F_{21}$ |
| BLX-GL50 | 4.00e+2 | 9.35e+1 | 1.09e+2 | 4.20e+2 | 4.49e+2 | 4.46e+2 | 6.89e+2 |
| BLX-MA | 2.70e+2 | 1.02e+2 | 1.27e+2 | 8.03e+2 | 7.63e+2 | 8.00e+2 | 7.22e+2 |
| COEVO | 2.94e+2 | 1.77e+2 | 2.12e+2 | 9.02e+2 | 8.45e+2 | 8.63e+2 | 6.35e+2 |
| DE | 2.59e+2 | 1.13e+2 | 1.15e+2 | 4.00e+2 | 4.20e+2 | 4.60e+2 | 4.92e+2 |
| DMS-L-PSO | 4.85 | 9.48e+1 | 1.10e+2 | 7.61e+2 | 7.14e+2 | 8.22e+2 | 5.36e+2 |
| EDA | 3.65e+2 | 1.44e+2 | 1.57e+2 | 4.83e+2 | 5.64e+2 | 6.52e+2 | 4.84e+2 |
| G-CMA-ES | 2.28e+2 | **9.13e+1** | 1.23e+2 | 3.32e+2 | 3.26e+2 | **3.00e+2** | 5.00e+2 |
| K-PCX | 5.10e+2 | 9.59e+1 | 9.73e+1 | 7.52e+2 | 7.51e+2 | 8.13e+2 | 1.05e+3 |
| L-CMA-ES | 2.11e+2 | 1.05e+2 | 5.49e+2 | 4.97e+2 | 5.16e+2 | 4.42e+2 | 4.04e+2 |
| L-SADE | 3.20e+1 | 1.01e+2 | 1.14e+2 | 7.19e+2 | 7.05e+2 | 7.13e+2 | 4.64e+2 |
| SPC-PNX | 2.54e+2 | 1.10e+2 | 1.19e+2 | 4.40e+2 | 3.80e+2 | 4.40e+2 | 6.80e+2 |
| TSM-GDA | **7.43e−7** | 1.27e+2 | 1.42e+2 | 6.99e+2 | 4.96e+2 | 6.76e+2 | 4.72e+2 |
| SaDESA | 1.10e+2 | 1.03e+2 | 5.38e+2 | 8.35e+2 | 8.06e+2 | 8.09e+2 | 6.07e+2 |
| RCGA | 4.00e+2 | 9.35e+1 | 1.09e+2 | 4.20e+2 | 4.49e+2 | 4.46e+2 | 6.89e+2 |
| MPCDE | 1.33e+2 | 1.07e+2 | **1.07e+2** | **3.00e+2** | **3.00e+2** | **3.00e+2** | **3.96E+2** |
|  | $F_{22}$ | $F_{23}$ | $F_{24}$ | $F_{25}$ |  |  |  |
| BLX-GL50 | 7.59e+2 | 6.39e+2 | **2.00e+2** | 4.04e+2 |  |  |  |
| BLX-MA | 6.71e+2 | 9.27e+2 | 2.24e+2 | 3.96e+2 |  |  |  |
| COEVO | 7.79e+2 | 8.35e+2 | 3.14e+2 | **2.57e+2** |  |  |  |
| DE | 7.18e+2 | 5.72e+2 | **2.00e+2** | 9.23e+2 |  |  |  |
| DMS-L-PSO | 6.92e+2 | 7.30e+2 | 2.24e+2 | 3.66e+2 |  |  |  |
| EDA | 7.71e+2 | 6.41e+2 | **2.00e+2** | 3.73e+2 |  |  |  |

| | | | | |
|---|---|---|---|---|
| G-CMA-ES | 7.29e+2 | 5.59e+2 | **2.00e+2** | 3.74e+2 |
| K-PCX | **6.59e+2** | 1.06e+3 | 4.06e+2 | 4.06e+2 |
| L-CMA-ES | 7.40e+2 | 7.91e+2 | 8.65e+2 | 4.42e+2 |
| L-SADE | 7.35e+2 | 6.64e+2 | **2.00e+2** | 3.76e+2 |
| SPC-PNX | 7.49e+2 | 5.76e+2 | **2.00e+2** | 4.06e+2 |
| TSM-GDA | 7.05e+2 | **5.54e+2** | **2.00e+2** | 6.94e+2 |
| SaDESA | 7.24e+2 | 7.85e+2 | 2.01e+2 | 4.33e+2 |
| RCGA | 7.59e+2 | 6.39e+2 | **2.00e+2** | 4.04e+2 |
| MPCDE | 7.53e+2 | 5.59e+2 | **2.00e+2** | 3.68e+2 |

Interestingly, the TSM-GDA algorithm obtained better scores for problem $F_{15}$ only. In effect, the MPCDE algorithm outperformed its competitors on composition functions $F_{17} - F_{19}$, $F_{21}$, meanwhile it has similar better performance to the other algorithms for problems of $F_{20}$ and $F_{24}$. It was also observed from Table 4.8 that the other hybrid SaDESA algorithm has prominent performance with the other methods for four non-composition problems $F_1$, $F_2$, $F_4$, and $F_9$.

As dimensions are set to 30 in this experiment, Table 4.9 exhibits the mean fitness values obtained from 11 algorithms and MPCDE with 1.0e+5 FEs. Compared to the results of 10-dimension, similar performance of MPCDE can be observed for $F_4$ and $F_5$ due to exploration of relatively small local neighborhoods. MPCDE had similar results for $F_6 – F_{14}$, except $F_7$, $F_9$, $F_{10}$ and $F_{13}$, where MPCDE lost on those functions probably because of their multimodal landscape properties. For composition functions, the introduced algorithm had similar results and gave the same results compared to the other methods of $F_{23}$, however, it had better achievement than others for $F_{18} - F_{20}$, and $F_{22}$. Besides, it is quite clear that hybrid SaDESA algorithm has excellent performance with the other methods for six different types of problems of $F_1$, $F_2$, $F_7$, $F_9$, $F_{24}$, and $F_{25}$ respectively. It was clearly observed from Table 4.9 that, MPCDE generally outperformed the other methods on 12 out of 25 test functions.

Table 4.9: Comparative results of *mean* values among 11and MPCDE Algorithm for CEC2005 test benchmarks with 30-dimension of 3.0e+5 FEs over 25 runs

| Alg. | $F_1$ | $F_2$ | $F_3$ | $F_4$ | $F_5$ | $F_6$ | $F_7$ |
|---|---|---|---|---|---|---|---|
| BLX-GL50 | 1.00e-9 | 1.00e-9 | 3.11e+3 | 1.68e+1 | 3.33e+2 | 2.60e-7 | 1.00e-9 |
| BLX-MA | 1.00e-9 | 8.72e-6 | 8.77e+5 | 3.97e+1 | 2.18e+3 | 4.95e+1 | 1.33e-2 |
| COEVO | 7.97e-1 | 4.40e-1 | 3.67e+2 | 4.80e+3 | 8.34e+3 | 1.21e+3 | 1.41e-1 |
| DE | 1.00e-9 | 3.33e-2 | 6.92e+5 | 1.52e+1 | 1.70e+2 | 2.51e+1 | 2.96e-3 |
| G-CMA-ES | 1.00e-9 | 1.00e-9 | **1.00e-9** | 1.11e+4 | 1.00e-9 | 1.00e-9 | 1.00e-9 |
| K-PCX | 1.00e-9 | 1.00e-9 | 5.79e+1 | 1.11e+3 | 2.04e+3 | 1.75 | 1.50e-2 |
| L-CMA-ES | 1.00e-9 | 1.00e-9 | **1.00e-9** | 9.26e+7 | 1.00e-9 | 1.00e-9 | 1.00e-9 |
| SPC-PNX | 1.00e-9 | 6.95e-7 | 1.10e+6 | 8.13e-7 | 4.24e+3 | 1.52e+1 | 1.46e-2 |
| TSM-GDA | 2.80e−9 | 4.32e−2 | 4.68e+5 | 1.84e+4 | 5.25e+3 | 1.18e+1 | 3.62e−3 |
| SaGPDE | **0.00e+0** | **0.00e+0** | 3.52e+6 | 1.23e-3 | 6.20e+3 | 2.30e+1 | **0.00e+0** |
| RCGA | 8.88e-9 | 9.84e-9 | 3.11e+3 | 1.68e+1 | 3.33e+2 | 2.60e-7 | 9.07-9 |
| MPCDE | 9.09e-14 | 9.32e-14 | 2.62e+2 | **1.46e-13** | **1.36e-11** | **9.78e-14** | 1.19e-13 |
| | $F_8$ | $F_9$ | $F_{10}$ | $F_{11}$ | $F_{12}$ | $F_{13}$ | $F_{14}$ |
| BLX-GL50 | 2.09e+1 | 1.51e+1 | 3.52e+1 | 2.47e+1 | 9.52e+3 | 5.15 | 1.21e+1 |
| BLX-MA | 2.07e+1 | 6.81e-1 | 9.06e+1 | 3.11e+1 | 4.39e+3 | 3.96 | 1.26e+1 |
| COEVO | 2.09e+1 | 1.31e+2 | 2.32e+2 | 3.77e+1 | 1.01e+5 | 9.02 | 1.32e+1 |
| DE | 2.10e+1 | 1.85e+1 | 9.69e+1 | 3.42e+1 | 2.75e+3 | 3.23 | 1.34e+1 |
| G-CMA-ES | 2.01e+1 | 9.38e-1 | 1.65 | 5.48 | 4.43e+4 | 2.49 | 1.29e+1 |
| K-PCX | **2.00e+1** | 2.79e-1 | **5.17e-1** | 2.95e+1 | 1.68e+3 | 1.19e+1 | 1.38e+1 |
| L-CMA-ES | **2.00e+1** | 2.91e+2 | 5.63e+2 | 1.52e+1 | 1.32e+4 | 2.32 | 1.40e+1 |
| SPC-PNX | 2.09e+1 | 2.39e+1 | 6.03e+1 | 1.13e+1 | 1.31e+4 | 3.59 | 1.31e+1 |
| TSM-GDA | **2.00e+1** | 1.05e−9 | 1.66e+2 | 2.58e+1 | 6.90e+2 | **8.35e−1** | 1.29e+1 |
| SaGPDE | 2.01e+1 | **0.00e+0** | 1.25e+2 | 3.13e+1 | 1.72e+3 | 1.28 | 1.28e+1 |
| RCGA | 2.09e+1 | 1.51e+1 | 3.52e+1 | 2.47e+1 | 9.52e+3 | 5.15 | 1.21e+1 |
| MPCDE | **2.00e+1** | 1.02e-13 | 1.38e+1 | **1.12** | **3.76e+2** | 2.76 | **2.76** |
| | $F_{15}$ | $F_{16}$ | $F_{17}$ | $F_{18}$ | $F_{19}$ | $F_{20}$ | $F_{21}$ |
| BLX-GL50 | 3.04e+2 | 8.87e+1 | 1.35e+2 | 9.04e+2 | 9.04e+2 | 9.03e+2 | 5.00e+2 |
| BLX-MA | 3.56e+2 | 3.26e+2 | 2.79e+2 | 8.78e+2 | 8.80e+2 | 8.79e+2 | 5.00e+2 |
| COEVO | 4.11e+2 | 3.81e+2 | 4.54e+2 | 1.06e+3 | 1.05e+3 | 1.06e+3 | 6.04e+2 |
| DE | 3.60e+2 | 2.12e+2 | 2.37e+2 | 9.04e+2 | 9.04e+2 | 9.04e+2 | 5.00e+2 |
| G-CMA-ES | 2.08e+2 | **3.50e+1** | 2.91e+2 | 9.04e+2 | 9.04e+2 | 9.04e+2 | 5.00e+2 |
| K-PCX | 8.76e+2 | 7.15e+1 | 1.56e+2 | 8.30e+2 | 8.31e+2 | 8.31e+2 | 8.59e+2 |
| L-CMA-ES | 2.16e+2 | 5.84e+1 | 1.07e+3 | 8.90e+2 | 9.03e+2 | 8.89e+2 | **4.85e+2** |
| SPC-PNX | 3.68e+2 | 7.47e+1 | **8.54e+1** | 9.05e+2 | 9.05e+2 | 9.05e+2 | 5.00e+2 |
| TSM-GDA | **8.02e+1** | 2.09e+2 | 3.93e+2 | 9.10e+2 | 9.10e+2 | 9.09e+2 | 5.00e+2 |
| SaGPDE | 1.09e+2 | 2.80e+2 | 2.77e+2 | 9.00e+2 | 9.00e+2 | 9.00e+2 | 5.19e+2 |
| RCGA | 3.04e+2 | 8.87e+1 | 1.35e+2 | 9.04e+2 | 9.04e+2 | 9.03e+2 | 5.00e+2 |
| MPCDE | 2.11e+2 | 1.40e+2 | 1.51e+2 | **8.16e+2** | **8.16e+2** | **8.16e+2** | 5.00e+2 |
| | $F_{22}$ | $F_{23}$ | $F_{24}$ | $F_{25}$ | | | |
| BLX-GL50 | 8.74e+2 | 5.87e+2 | 8.77e+2 | 2.11e+2 | | | |
| BLX-MA | 9.08e+2 | 5.59e+2 | 2.00e+2 | 2.11e+2 | | | |
| COEVO | 1.16e+3 | 9.22e+2 | 1.10e+3 | 1.03e+3 | | | |
| DE | 8.97e+2 | **5.34e+2** | 2.00e+2 | 7.30e+2 | | | |
| G-CMA-ES | 8.03e+2 | **5.34e+2** | 9.10e+2 | 2.11e+2 | | | |
| K-PCX | 1.56e+3 | 8.66e+2 | 2.13e+2 | 2.13e+2 | | | |
| L-CMA-ES | 8.71e+2 | 5.35e+2 | 1.41e+3 | 6.91e+2 | | | |
| SPC-PNX | 8.81e+2 | **5.34e+2** | 2.00e+2 | 2.13e+2 | | | |
| TSM-GDA | 9.56e+2 | **5.34e+2** | 9.89e+2 | 9.84e+2 | | | |
| SaGPDE | 5.19e+2 | 5.51e+2 | **1.90e+2** | **1.91e+2** | | | |
| RCGA | 8.74e+2 | 5.87e+2 | 8.77e+2 | 2.11e+2 | | | |
| MPCDE | **5.14e+2** | **5.34e+2** | 2.00e+2 | 2.08e+2 | | | |

Table 4.10 demonstrated simulation outputs of three valuable methods and MPCDE algorithm for 25 classical benchmark functions with 50 dimensions and 5.0e+5 FEs. G-CMA-ES and L-CMA-ES are more advanced versions of CMA-ES that was introduced in [59]. Compared with its competitors, MPCDE obtained better mean values for 20 test functions. From the achievement of the proposed method, MPCDE was winner over the other competitors for all functions except for functions: $F_3$, $F_{10}$, $F_{13}$, $F_{15}$, $F_{16}$, $F_{17}$ and $F_{21}$, respectively. Because the proposed algorithm is able to jump from a locally minimum, obviously MPCDE has better performance compared to its competitors for more composition problems but functions $F_{15}$, $F_{16,}$ $F_{17}$ and $F_{21}$. In effect, Table 4.9 and 4.10 show useful references of the scalability's property for verifying the introduced hybridization's performance. As shown in Table 4.9−4.10, even though MPCDE failed in some cases because of the exploration in local search, MPCDE still presented better performance than the other state-of-the-art algorithms on most benchmark problems.

Table 4.10: Comparative results of *mean* values between L-CMA-ES, G-CMA-ES, MPCDE and TSM-GDA Algorithm for CEC2005 test benchmarks with 50-dimension of 5.0e+5 FEs over 25 runs

| *Alg.* | $F_1$ | $F_2$ | $F_3$ | $F_4$ | $F_5$ | $F_6$ | $F_7$ |
|---|---|---|---|---|---|---|---|
| G-CMA-ES | 1.00e-9 | 1.00e-9 | **1.00e-9** | 4.68e+5 | 2.85 | 1.00e-9 | 1.00e-9 |
| L-CMA-ES | 1.00e-9 | 1.00e-9 | **1.00e-9** | 4.46e+8 | 3.27 | 1.00e-9 | 1.00e-9 |
| TSM-GDA | 1.91e−9 | 1.37e−1 | 6.00e+5 | 1.05e+5 | 1.21e+4 | 1.48e+1 | 3.26e−4 |
| MPCDE | **1.11e-13** | **1.11e-13** | 2.82e+2 | **7.02e-10** | **8.66e-1** | **1.11e-13** | **1.42e-13** |
| | $F_8$ | $F_9$ | $F_{10}$ | $F_{11}$ | $F_{12}$ | $F_{13}$ | $F_{14}$ |
| G-CMA-ES | 2.01e+1 | 1.39 | **1.72** | 1.17e+1 | 2.27e+5 | 4.59 | 2.29e+1 |
| L-CMA-ES | **2.00e+1** | 5.67e+2 | 1.48e+3 | 3.41e+1 | 8.93e+4 | 4.70 | 2.39e+1 |
| TSM-GDA | 2.01e+1 | 1.23e−10 | 4.10e+2 | 4.77e+1 | 3.91e+3 | **1.26** | 2.25e+1 |
| MPCDE | **2.00e+1** | **1.18e-13** | 3.60e+1 | **3.50** | **9.45e+2** | 5.80 | **2.17e+1** |
| | $F_{15}$ | $F_{16}$ | $F_{17}$ | $F_{18}$ | $F_{19}$ | $F_{20}$ | $F_{21}$ |
| G-CMA-ES | 2.04e+2 | **3.09e+1** | **2.34e+2** | 9.13e+2 | 9.12e+2 | 9.12e+2 | 1.00e+3 |
| L-CMA-ES | 2.50e+2 | 7.09e+1 | 1.05e+3 | 9.06e+2 | 9.11e+2 | 9.01e+2 | 5.00e+2 |
| TSM-GDA | **2.00e+2** | 2.94e+2 | 3.99e+2 | 9.26e+2 | 9.23e+2 | 9.23e+2 | **1.02e+2** |
| MPCDE | 2.96e+2 | 6.29e+1 | 2.57e+2 | **8.39e+2** | **8.37e+2** | **8.36e+2** | 5.60e+2 |
| | $F_{22}$ | $F_{23}$ | $F_{24}$ | $F_{25}$ | | | |
| G-CMA-ES | 8.05e+2 | 1.01e+3 | 9.55e+2 | 2.15e+2 | | | |
| L-CMA-ES | 9.10e+2 | 6.37e+2 | 8.43e+2 | 4.77e+2 | | | |
| TSM-GDA | 1.04e+3 | 7.59e+2 | 1.22e+3 | 1.32e+3 | | | |
| MPCDE | **5.00e+2** | **5.80e+2** | **2.25e+2** | **2.12e+2** | | | |

# Chapter 5

# PROBABILITY COLLECTIVES FOR MULTI-OBJECTIVE OPTIMIZATION PROBLEMS

## 5.1 Introduction

Multi-objective optimization problems deal with optimization of two or more objective functions that must be simultaneously optimized. These objectives generally conflict with each other by their nature. Pareto optimal solutions involve a set of nondominated individuals as a compromise (or trade-off) between different objective functions. Because of the difficulties of real-world problems, the aim of the multi-objective optimization problem is to carry out an approximate set of solutions that is as close to the Pareto optimal front as possible. Thus, it is important to define two criteria to describe how good the generated set of solutions is. These criteria to be satisfied in MOPs are presented as follows:

1) Convergence: Determine how close the obtained solutions are to the Pareto optimal front.
2) Diversity: Determine how well the obtained solutions are distributed along the Pareto optimal front.

The convergence, which defines the distance between the obtained solutions and the optimal solutions, is the main criterion of all optimization algorithms. On the other hand, the diversity defines how well is the coverage of the obtained solutions in the optimal space while the spread defines how evenly distributed are the multiple solutions in the optimal space.

The decomposition technique is a major advantage that requires an aggregation of all objectives to a single figure of merit. The benefits are supported with single objective optimization algorithms such as CE, DE, PC and so on. Therefore, a single-objective problem can be solved more efficiently for a generalized multi-objective model of the problem. For multi-objective optimization, our research works covers the investigation of probability distributions of the PC approach together with the mechanisms of utility to efficiently determine fitness values based on the accumulated knowledge of communicating agents. Meanwhile a hybrid optimization algorithm, combined with PC random search optimization in the platform of evolutionary algorithms, is developed for solving MOPs. This algorithm involves most MOEA/D features and algorithmic refinements to improve solution's quality in the Pareto set. The performance of the proposed approach is compared with some state-of-the-art competitors including the MOEA/D algorithm for several unconstrained MOPs including two and three objectives functions. Experimental results empirically demonstrated that the proposed approach is highly competitive and can be considered as a powerful alternative for MOPs.

## 5.2 MOEAs

Over the past two decades, EAs and other population-based metaheuristics attempt to find the set of Pareto optimal solutions in a single run. MOEAs become increasingly popular for solving MOPs because they evolve simultaneously a population of potential solutions, which are particularly useful for approximating the trade-off between solutions extracted qualities in a single run. Most MOEAs perform the procedures of EAs to find a well-converged and well-diversified set of non-dominated solutions. Essentially, MOEAs utilize the concepts and development of evolutionary algorithms by means of their stochastic operators (crossover, mutation, and selection) to simply find a set of optimal trade-offs.

Various MOEAs have been proposed to solve MOPs [39, 60-64]. Depending on different strategies for selecting offspring solutions, the classical MOEAs can be mainly broken down into 3 categories: Pareto dominance-based methods, *e.g.* Non-Dominated Sorting Genetic Algorithm II (NSGA-II) [60] and Strength Pareto evolutionary algorithm 2 (SPEA2) [61]; Indicator (hypervolume) based algorithms, *e.g.* indicator-based evolutionary algorithm (IBEA) [62]; and Scalarizing Function-based approaches, *e.g.* multiple single objective Pareto sampling (MSOPS) [63] and MOEA/D [39, 64, 65].

Pareto dominance-based methods adopt the Pareto dominance relation along with the crowding distance in NSGA-II or clustering methods in SPEA2 to select offspring solutions for promoting the convergence and the diversity. Besides, with the prior information defined by the decision maker (DM), the earliest preference-based MOEAs were made to select the preferred solutions for well-distributed approximation of the PF [66]. Additionally, the preference information was hybridized with NSGA-II by perturbing the dominance concept and adapting weighted crowding distance to yield a more fine-grained selection operators [67].

The second class guides the selection operator by using scalar indicators, considered as the fitness. The most commonly recommended performance indicator is hypervolume (HV) [68], which can measure two multi-objective criteria simultaneously. However, indicator based method requires more search capability. For example, the high time complexity of hypervolume calculation exponentially increases as the number of objective functions raises [69]. To solve this problem, a general IBEA [62] takes different performance measures to compare differences of pairwise solutions by using an arbitrary indicator for approximating HV contributions.

In contrast to the above two classes of MOEAs, plain aggregation approaches calculate particular fitness assignment by predefined weight vectors and decompose various complicated problems into a set of subproblems for solution selection. The earliest application for non-Pareto-based approach of the MOEA, known as Vector Evaluated Genetic Algorithm (VEGA) [70] was proposed to extend the basic GA’s principle. Its linear aggregating feature transformed all objective functions into a single population without decreasing their solution qualities. Later, the term decomposition [39] has emerged as an effective way for tranforming a number of single objective subproblems based on population. Except that, Hybrid MOEAs usually lies in this group as well, so that different characteristics of scalability can be utilized as natural choices. For example, one hybrid EA-PSO was integrated with the search operators from two different algorithms of PSO and EAs for optimal solutions in permutation representation [71]. In hybridization, memetic MOEAs combined various local search methods for better offspring selection. As a typical representative EAs for this group of multi-objective optimization, MOEA/D has achieved great success in the field of evolutionary multi-objective optimization involving decomposition, scaling weight vectors and other nature inspired methods.

## 5.3 MOEA/D

Recently, a multi-objective evolutionary approach based on decomposition has attracted lots of researchers and has achieved enormous success in the field of evolutionary multi-objective optimization. Rather than other class of algorithms, it decomposes a MOP into a set of single objective scalar optimization subproblems based on aggregation method and then uses the evolutionary computation to optimize these subproblems in a concurrent manner. Each objective is handled by its optimizer with their corresponding weight vectors. Besides, neighborhood relations between these subproblems are established according to the distances among their aggregation weight vectors to provide useful information for each subproblem.

In the interest of the aggregation method, conventional MOEA/D mainly involves three decomposition approaches: weighted sum [41], Tchebycheff decomposition and boundary intersection [72]. According to their characteristics, MOEA/D discovers several merits over Pareto dominance concept such as computational adaptability, efficiency, scalability and feasibility to many types of problems. In spite of its advantages, MOEA/D has also been exposed two drawbacks: unable to create an arbitrary number of weight vectors when the number of objectives is more than two; unchanged weight vectors for different shapes of PF.

However, the MOEA/D paradigm has been studied and used with success for dealing with many MOPs and complex ones as well. The original MOEA/D [39] used the techniques of simulated binary crossover (SBX) [73] and polynomial mutation [60] to generate new offspring. A solution is allowed to couple only with its neighbors and a new solution could replace any old solution in its neighborhood if it is better than them. The evolutionary process flow of MOEA/D is illustrated with Algorithm 5.1. Meanwhile, MOEA/D has received increasing research attention and different follow-up studies can be classified into four aspects:

- Combination MOEA/D with other nature inspired metaheuristics;

- Incorporation of new decomposition strategies into the framework of MOEA/D;
- Creation of newly initial weight vectors and adaptively adjusting the distribution of problems in the process of evolution;
- Modification of offspring solution reproduction mechanisms in MOEA/D.

**Algorithm 5.1:** Pseudo-code of MOEA/D [38]

```
 Initialization:     // $N$, population size and $M$, the number of objective;
 1: Generate a set of weight vector $W \leftarrow \{w^1,\cdots,w^N\}$, form subproblem $B(i) = \{i_1,\ldots, i_Q\}$ from the
    $Q$ shortest Euclidean distance to $w^i$;
 2: Randomly initialize a population $P \leftarrow \{x^1,\cdots,x^N\}$, and evaluate its objective values $S \leftarrow$
    $\{F(x^1),\cdots,F(x^N)\}$;
 3: Determine reference point $z^* \leftarrow (z_1^*,\ldots,z_M^*)^T$ by $z_j^* = \min_{1\le i\le N} f_j(x^i)$;
 4: Set $g' \leftarrow 0$, $iter \leftarrow 0$;   // $g'$ is generation and $iter$ is iteration
 5: While $g' < FEs$ Do
 6:   For $i = 1,\ldots,N$, Do
 7:     $(x^a, x^b) \leftarrow$ Selection($B(i)$, $P$);
 8:     $(x^{a'}, x^{b'}) \leftarrow$ Crossover($x^a$, $x^b$) by simulated binary crossover;
 9:     $y \leftarrow$ Mutation($x^{a'}$, $x^{b'}$) by polynomial mutation;
10:     $F(y) \leftarrow$ Evaluate_Fitness($y$);
11:     Update of $z^*$: For $j = 1,\ldots,M$, If $z_j^* > F_j(y')$, Then $z_j^* = F_j(y')$;
12:     $g^{te} \leftarrow$ Assign_Fitness($W$, $F(y)$, $z^*$) by Tchebycheff;
13:     Replace_Solution: For $j \in B(i)$ Do If $g^{te}(y|\omega^j, z^*) \le g^{te}(x^j|\omega^j, z^*)$, Then
        $x^j = y$, and $S^j = F(y)$;
14:   End For
15: End While
```

A number of MOEA/D variants, like MOEA/D-DE [64] with DE operator was addressed and shown to outperform classical MOEA/D and NSGA-II, especially for complex problems. Another further improved version MOEA/D-DRA [65] allocates various amounts of computational effort to different subproblems according to their utility functions. This approach updates a set of descendant individuals by use of ranking and tournament selection [3] as shown in following section. A parallel version of MOEA/D-DE, named pMOEA/D [74], obtains linear speedup from running on different multi-core processors. Besides, the idea motivated from MOEA/D has been integrated with local search into a hybrid algorithm [75]. The

MOEA/D family usually addressed the straightforward decomposition techniques is described as follows.

### 5.3.1 Decomposition of Multiobjective Optimization

By way of simplicity, there are two main methods of decomposition. These techniques address composite functions, where a set of single objective functions can easily model a MOP.

1) Weighted Sum Approach

This method considers a convex association for the various objectives. Assume that $W = (w_1,\ldots,w_m)^T$ be a weight vector, i.e., $w_i \geq 0$ for all $i = 1,\ldots, m$. If $\sum_{i=1}^{m} w_i = 1$ and $w_i \leq 1$, the weighted sum is known as a convex combination of objectives. Then, the optimal solution can be defined as the following scalar optimization problem:

$$\text{maximize } g_1(x \mid w) = \sum_{i=1}^{m} w_i f_i(x)$$
$$\text{subject to } x \in \Omega. \tag{5.1}$$

where $\Omega \subset \Re^D$ is the feasible space of decision vector $x = (x_1, x_2, \ldots, x_D)^T$. The single objective function $g_1(x|w)$, is the combination of the $m$ objectives, given the weights $w_1,\ldots,w_m$ if the PF of the expression (2.4) is concave (convex in the case of minimization) as illustrated in Figure 5.1. By optimization, $g_1(x|w)$ function will be minimized for locating solutions to a set of Pareto optimal vectors. As changing $w_i$, each single objective optimizer determines distinct optimal solutions.

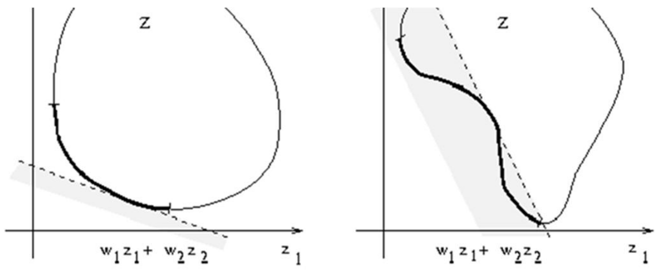


Figure 5.1: Convex and concave problems

2) Tchebycheff Approach

For $w \in \Re^M$, the scalar optimization problem can be defined in the following form

$$\text{minimize } g_2(x \mid w, z^*) = \max_{1 \le i \le m}\{w_i \mid f_i(x) - z_i^*\}$$
$$\text{subject to } x \in \Omega. \tag{5.2}$$

where $z^* = (z_1^*, \ldots, z_M^*)^T$ is the reference point, i.e., $z_i^* = \max\{f_i(x) | x \in \Omega)\}$ for each $i = 1, \ldots, m$. For each Pareto optimal point $x^*$, there exists a weight vector $w$ such that $x^*$ is the optimal solution of (5.2) and each optimal solution of (5.2) is a Pareto optimal solution of (5.1). Thus, this approach is considered to evaluate fitness functions using the weighted objectives (5.1) and is appealing since aggregation function $g_1(x|w)$ from (5.1) can obtain various Pareto optimal solutions by altering the weight vector. Even though such method is not smooth for a continuous MOP, the proposed method for MOPs modifies the scalar optimization technique to calculate the derivative of aggregation function in objective space.

**5.3.2 Tournament Selection**

Tournament [3] is one of the most popular selection operators in EAs because of its efficiency and simple implementation. Binary tournament selection is the most commonly used search operator. For this type of selection, two individual solutions are randomly selected from the larger population, and take the best one by comparing both solution into the intermediate population, and this process is repeated for a predefined number of times. The solution in the highest rank wins as a parent. The *n*-tournament selection is an implementation of the binary tournament selection where *n* is the tournament size that defines the number of individuals that compete for the tournament. Algorithm 5.2 shows the procedure for tournament selection [76]. In the natural process of evolution, the selection operator decides the probability of survival for the next generation.

**Algorithm 5.2:** Tournament selection [76]

```
1:  For individual $i$ = 1 to $pop\_size$
2:      Choose two individuals randomly from all population
3:      If fitness individual $p_1$ > fitness individual $p_2$
4:         Select individual $p_1$, otherwise, select individual $p_2$
5:      End If
6:  End For
```

## 5.4 The Proposed Approach for MOP (MOEA/D-PC)

From a different point of view, multiple objective problems are considered as a set of single objective optimization, because single objective approaches are frequently used to suggest new solution ideas for solving MOPs. This idea provides way to understand relationships between single objective problem and multi-objective problem, and demonstrates how solution concepts from single objective optimization can be used to embed a new perception into the structure of multiple objective optimization methods. Therefore, the proposed approach utilizes the idea of the decomposition in MOEA/D framework based on PC approach to deal with unconstrained hard MOPs.

The relationship between SOP and MOP are explained as follows. Firstly, a given MOP is decomposed into SOP by means of scaling single objective functions with their corresponding weight vectors. According to the multiple objective problem concepts, an associated single objective optimization problem is then formulated and there exists at least one optimal solution for each single objective problem, and their combination is also optimal for the original multiple objective problem. The general structure of relationship between MOP and SOP for the proposed approach is depicted in Figure 5.2. The figure states that a MOP is modeled by decomposition to form corresponding SOP. The optimal solution for the SOP is used to obtain the efficient set $X_E$ for the associated MOP as the nondominated solution. This computational step is also named as filtering.

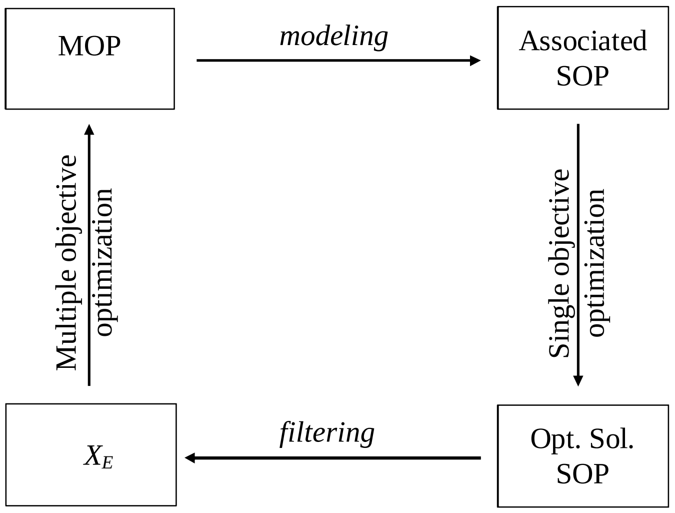

MOP
modeling
Associated SOP
Multiple objective optimization
Single objective optimization
$X_E$
filtering
Opt. Sol. SOP

Figure 5.2: The general structure of relationship between MOP and SOP

The proposed approach adopted the MOEA/D framework to transform the given MOP into a related SOP by means of an appropriate scalarization. By the decomposition, the individual solutions was initially handled by PC random search method. In terms of its randomness, the designed framework distributes principal tasks to PC agents, which estimates expected utilities of probabilities from extensive sampling and applies gradient descent technique for updating probability distributions to optimize them synchronously. The favorable solutions will be found from the existing population with respect to distinguishing probability by natural selection and have more opportunities to survive for following generation so that the algorithm eventually converges to the true optimal solution. According to the reviews, PC-based studies are naturally well-suited for solving MOPs and they apparently have more potential for multi-objective optimization. As mentioned earlier, two algorithms of MOPC [22] and MOPC/D [23] were proposed to place greater emphasis on highly promising exploration direction for the PC strategy vectors in multi-objective optimization.

Since PC is hybridized with classical MOEA/D framework, our proposed approach is called probability collectives MOEA/D (MOEA/D-PC) [77]. One combined mutation operator between polynomial and Gaussian schemes adjusted by CE method is introduced as the global search engine in the MOEA/D framework for the multi-objective optimization. The promising fitness values in the solution space are improved by MOEA/D, which then update population of points of the Pareto distribution. The proposed method involves random weight vectors, probability distributions search and combined evolutionary operators in order to accelerate the convergence to a near-optimal Pareto Front. Experimental evaluations of the proposed MOEA/D-PC approach exhibit the superiority of the proposed method over several state-of-the-art algorithms for the CEC2009 Special Session Competition problems. In the proposed framework, MOEA/D combines PC random search algorithm to deal with a number of single objective problems and optimized them in a cooperative manner. The proposed method is supported by the MOEA/D as general framework and emerges from three modifications to MOEA/D-PC:

(1) Constructing the aggregated objectives into a scalar function by using random weighted approach;

(2) Applying PC approach as a local search operator to approximate the PF in a single optimization run;

(3) Performing a new mutation scheme as a global search operator by combining two mutation operators, namely polynomial and Gaussian adjusted by the CE method.

These modifications generate new aggregated weight vectors by decomposition method to form composite problems, where PC improves the convergence rates towards nondominated solutions and then DE mutation scheme further updates the fitness values to maintain the diversity of the population and approximate the PF in the objective space.

**5.4.1 Decomposition of Multi-objective Optimization**

MOEA/D decomposes a MOP into a set of single objective optimization subproblems and optimizes them in a simultaneous manner using a set of random distributed weight vectors in the objective space. As discussed before, Tchebycheff approach has similar effects as Pareto dominance for selection. Here, the proposed algorithm constructs the fitness function which is similar with Tchebycheff approach in order to transform a MOP into a number of scalar SOPs. Each individual $i$ is related to a weight vector $\omega = (\omega_1^i, \ldots, \omega_M^i)^T$ consisting of $M$ real-valued objective functions and the set of all weight vector is $\{\omega^1, \ldots, \omega^N\}$, where $N$ is the population size.

Decomposition method indicates a convex distribution of the different objectives. Let $\omega = (\omega_1, \ldots \omega_M)^T$ be a weight vector, i.e., $\omega_j \geq 0$ is a uniformly random sampling chosen value from [0, 1] for each objective $j = 1, \ldots, M$ and satisfying the condition that $\sum_{j=1}^{M} \omega_j = 1$. To generate a set of different Pareto optimal vectors, each utilizing different weight vectors $\omega$, scalar single-optimization problem is defined as:

$$\begin{aligned} &\text{minimize} \quad g(x|\omega) = \max_{1 \leq j \leq M} [\omega_j^i \times h_j(x)], \\ &\text{subject to} \quad x \in \Omega \end{aligned} \tag{5.3}$$

where $x$ is the variable vector for Pareto optimal point with the convex PF of (2.4). The optimizing function $h_j(x)$ is formulated as

$$h_j(x) = \log(1 + f_j(x) - z_j^*) \qquad (5.4)$$

where $z^* = (z_1^*, \ldots, z_M^*)$ is the reference point, i.e., $z_i^* = \max\{f_j(x)|x \in \Omega)\}$ for each $j = 1, \ldots, M$. A weight vector $\omega$ guarantees to find its corresponding Pareto optimal point $x^*$, that is the optimal solution of (5.4) and each optimal solution of (5.4) is Pareto optimal of (5.3). The constructed fitness function ultimately approximates the objective points desired to lie on the PF by the use of logarithmic function in evolution.

### 5.4.2 MOEA/D-PC Framework

Motivated by decomposition, the MOEA/D-PC initially breaks down a MOP into a number of individual solutions with random weight vectors so that each optimizer has an equal probability to solve a problem. First, PC operates on probability distributions to find favorable solutions which can be stored in an archive as illustrated in Algorithm 5.4. Then offspring solutions are reproduced with the combined mutation scheme which is presented the following section. The fitter solutions are obtained from population by the selection operator in MOEA/D.

By $N$ randomly distributed weight vectors $\omega^i$ ($i$ = 1, 2, …, $N$), each individual solution $i$ is aggregated by the constructed fitness function $g(x|\omega^i)$ under consideration. Indeed, the hybrid algorithm conducts $N$ individual fitness solutions simultaneously for single objective subproblems. The uniformity of the aggregation of random weight vectors is contained within the uniformity of the Pareto optimal solutions, and the algorithm could explore the solution space starting from the initial generation.

Because $g(x|\omega)$ is a continuous function of $\omega$, any two subproblems are likely to have similar solutions if their weight vectors are close to each other. Following this observation, the neighborhood for each $i$, $B(i) = \{i_1,\ldots,i_{Tb}\}$ and $R(i) = \{i_1,\cdots,i_{Nr}\}$ are established according to the Euclidean distance between their weight vectors. $B(i)$ with $Tb$ weight vectors are the $Tb$ nearest to $\omega^i$ among all the $N$ weight vectors. $R(i)$ with $N_r$ weight vectors are the $N_r$ nearest to $\omega^i$. This relationship of neighboring for the problems is used for the selection of current solutions and the replacement of old solutions. $B(i)$ and $R(i)$ are generated and executed in initial and updated search steps. Therefore, the size of the neighborhood used for selection and replacement plays an essential role to exchange information for the solutions in a collaborative manner. For several investigations from other researchers, larger size may deteriorate effectiveness of the whole MOEA/D because of the furthest neighbor problems for selection. Smaller size may occupy inherent information to raise the delay due to limited neighbor problems for replacement.

MOEA/D-DRA [65] is a MOEA/D variant which adopts the dynamical resource allocation (DRA) technique. The proposed algorithm is derived from this version as an efficient tool to update individual solutions for all Pareto optimal solutions. During an iteration, a utility function $\pi^i$ is defined for each individual $p^i$ to estimate possible improvement in the aggregation function value according to their amount of computational efforts. Different computational efforts are thus distributed to a set of problems based on their utilities. The expression (5.5) of $\pi^i$ is:

$$\pi^i = \begin{cases} 1 & \text{if } \Delta^i > 0.001; \\ (0.95 + 0.05\dfrac{\Delta^i}{0.001})\pi^i & \text{otherwise.} \end{cases} \tag{5.5}$$

where $\Delta^i$ is the relative decrease in the aggregation function value of $p^i$, which can be evaluated in Equation (5.6) as follows,

$$\Delta^i = \frac{g(x^{old} \mid \omega^i, z^*) - g(x^{new} \mid \omega^i, z^*)}{g(x^{old} \mid \omega^i, z^*)} \tag{5.6}$$

where $x^{old}$ and $x^{new}$ are previous and current solutions of $p^i$.

The proposed hybrid approach MOEA/D-PC combines Gaussian mutation and polynomial mutation together with the CE method as global search operator to update offspring solutions. This technique efficiently improves the diversity of the population, thus escapes the local minima and approaches the near-global optimum. The hybrid algorithm contains PC approach in MOEA/D with DRA technique to form MOEA/D-PC, and the complete operation of proposed framework is summarized in Algorithm 5.3. More details of the base framework MOEA/D can be found in [65]. The polynomial mutation operator in line 21 of Algorithm 5.3 generates $y'$ based on $y_K^{new}$ based on the Equations (5.7-5.8):

$$y' = \begin{cases} y_K^{new} + \varphi_K \times (b_K - a_K) & \text{with probability } p_{m,} \\ y_K^{new} & \text{with probability } 1 - p_{m,} \end{cases} \tag{5.7}$$

$$\varphi_K = \begin{cases} (2 \times rand)^{\frac{1}{\eta+1}} - 1 & \text{if } rand < 0.5, \\ 1 - (2 - 2 \times rand)^{\frac{1}{\eta+1}} & \text{otherwise,} \end{cases} \tag{5.8}$$

**Algorithm 5.3:** MOEA/D-PC framework

```
 1: Step 1  Initialization
 2: Generate $N$ random spread weight vectors $\{\omega^1,\cdots,\omega^i,\cdots,\omega^N\}$, and $M$ objective function for each
    $\omega^i$;
 3: Randomly generate first population $Pop = \{x^1,\ldots, x^N\}$ by uniformly sampling for $D$
    dimensions $x^i$ from $\Omega$., evaluate their fitness function $F(x^i) = \{F_1(x^i),\ldots, F_M(x^i)\}$;
    Set an uniform probability of $x^i$ $q(x^i) = 1/N$ for PC;
 4: $EP \leftarrow$ non-dominated solutions from $Pop$;
 5: Calculate the Euclidean distances between any two weight vectors to form $Tb$ nearest weight
    vectors ($B(i) = \{i_1,\ldots, i_{Tb}\}$, $i \in [1, N]$) for each weight vector as neighboring solutions and
    determine the other weight vectors $R(i) = \{i_1,\cdots,i_{Nr}\}$ where $N_r$ is the maximal neighborhood
    size for replacement to new solution;
 6: Define the utility for update step  $\pi^i \leftarrow 1$;
 7: Initialize reference point of $z^* = (z_1,\ldots,z_M)^T$ by setting $z_j^* = \min_{1\le i\le N} f_j(x^i)$  $j = 1,\ldots, M$ to be the
    lowest objective values of the solutions;
 8: Set Temperature $Te \leftarrow 0$, evaluation $gen \leftarrow 0$, iteration $K \leftarrow 0$;
 9: Step 2  PC Search
10: Build PC for updating its distribution using $Pop$, $q$, $F(x^i)$ and $Te$ (see Algorithm 5.4);
11: Step 3  Update
12: Select $N/5$ individuals from population based on $\pi^i$ by using tournament to form set $I$;
13: For $i$ =1,…, $I$ Do
14:    Reproduction:  Uniformly Generate a random number $rand$ from (0,1);
15:    If $rand < \delta$ Then
16:       Randomly select two indexes $r1$ and $r2$ from $B(i)$;
17:    Else
18:       Randomly select two indexes $r1$ and $r2$ from $Pop$;
19:    End
20:    Create a new offspring $y^{K,new}$ from parents $x^{r1}$ and $x^{r2}$ by using DE operator as  $y^{K,new} = x^i$
          $+ F(x^{r1} - x^{r2})$;
21:    Do polynomial mutation to $y^{K,new}$ and generates $y'$ by combining two mutations;
22:    Evaluation: Evaluate the objective functions $F(y')$;
23:    $gen = gen + 1$;
24:    Update of $z^*$: For $j = 1,\ldots,M$, If $z_j^* > F_j(y')$ Then $z_j^* = F_j(y')$;
25:    Fitness assignment: Calculate fitness value to each solution $g(y'|\omega^*, z^*)$ using the
          constructed fitness solution;
26:    Update solution: set $c = 0$ and perform the following procedure;
27:    For each $j$ in $B(i)$ Do
28:       If $g(y'|\omega^j, z^*) \le g(x^j|\omega^j, z^*)$, Then set $x^j = y'$, and $c = c + 1$;
29:       If $c > Nr$, Then Go to Step 3;
30:    End
31:    Update $EP$ by accepting $y'$ if no solution dominates it, otherwise removing from $EP$, that the
          solution is dominated by $y'$.
32: End
33: $K \leftarrow K + 1$;
34: If $mod(K,20) = 0$ Then
35:    Update $\pi^i$ for each $i$ using previous and current solutions $x^{old}$ , $x^{new}$;
36: End
37: Step 4  Stopping Criteria
38: If $gen$ < MAX_FEs Then   Go to Step 2;
39:    Otherwise, stop the algorithm and pick $Pop$, output non-dominated solutions in $EP$.
40: End
```

### 5.4.3 PC Algorithm in MOEA/D Framework

As mentioned previously, probability distribution is updated by PC to become highly effective as near optimal strategy vectors. When updating the distribution, all samples would typically be used to find the favorable strategies simultaneously for each agent with respect to the highest probability value. This promising vector contributes to evaluate solution's quality of the Pareto set. During its iterations, PC repeatedly updates each of the agent's distributions as reducing *Te* until the predefined number of iterations or the quality of the best solution is reached. The final results of PC are stored in *EP*. Details of line 10 of Algorithm 5.3 are explained in Algorithm 5.4.

**Algorithm 5.4:** Update PC distribution

```
1: If $mod(K,20) = 0$ Then
2:   Select $Np = 10$ strategies from subproblems by using tournament selection;
3:   Find strategy vector $y^{tav}$ from updated $q$ by two functions of $E$ and $J$ for each agent (see
       Algorithm 1); Decrease $Te$ in Equation (3.11);
4:   Evaluation: Evaluate the objective functions $F(y^{tav})$;
5:   $gen = gen + 1$;
4:   For $i = 1, \ldots, Np$ Do
5:     Update of $z^*$: For $j = 1,\ldots,M$, If $z_j^* > F_j(y^{tav})$ Then $z_j^* = F_j(y^{tav})$;
6:     Fitness assignment: Calculate fitness value to each solution $g(y^{tav}|\omega^*, z^*)$ using the
           constructed fitness solution;
7:     If $g(y^{tav}|\omega^i, z^*) \le g(x^i|\omega^i, z^*)$, Then set $x^i = y^{tav}$;
8:   End
9:   Form a new strategy set $y_i$ within updated interval $[x_i^L, x_i^H] \in \Omega_i$ and evaluate the objective
       functions $G(y_i)$ for each agent $i \in [1, D]$;
10:  Update $z^*$ and assign fitness to estimate $g(y^{[j]}|w^*,z^*)$ for each $j = 1,\ldots, Np$;
11:  For each strategy $j$ Do
12:    If fitness $g(y^{[j]}|w^j,z^*) \le g(x^j|w^j,z^*)$ Then
13:      $g(x^j|w^j,z^*) = g(y^{[j]}|w^j,z^*)$, set $x^j = y^{[j]}$;
14:    End
15:  End
16:  Add $x^*$ for $Pop$ to update $EP$;
17: End
```

The proposed multi-objective algorithm employs the above described probability engine to represent its probability distributions for increasing flexibility in addressing difficult fitness landscapes, while it implements traditional gradient-based local search technique to form an integral part of MOEA/D framework without prejudicing search diversity. The nondominated solutions obtained from Algorithm 5.4 can be used to perform local search around the favorable solution space. The proposed approach based on the PC framework balances simultaneously the optimization criteria of convergence and diversity. Also, PC approach, using the cooling schedule, allows random sampling to be actively directed towards promising individuals. Therefore, MOEA/D-PC would generate high quality Pareto set approximations based on the Pareto dominance.

### 5.4.4 Combined Mutation Operator

The selection procedure determines the parent individuals based on pervious and current solutions generated by mutation operator, which is the most important procedure to produce offspring solutions. Instead of the original polynomial scheme in MOEA/D [65], the proposed approach combines the new Gaussian scheme and the original one together with CE method, which results in generating more efficient solutions closer to the global optimum. Consequently, the hybrid multi-objective algorithm needs to perturb individuals to specific points in fitness landscape and maintains the diversity of the population of solutions. For the implementation, Gaussian mutation is adopted to generate a new individual $y_g$' from a randomly selected vector $x^{K,new}$ in the population, which can be described as:

$$\sigma_K = (b_K - a_K)/20, \tag{5.9}$$

$$y_g' \sim N\,(x^{K,new}, \sigma_K), \tag{5.10}$$

After obtaining $y$' calculated from polynomial Equation (5.7) and (5.8), the proposed scheme combines two results $y$' and $y_g$' to yield a new mutated vector $y$' by the concept of CE method. Let $K$ represents the current iteration step, the vector $y$' is renewed by two distribution operators in the CE as shown in Equations (5.11-5.12) as follows,

$$y' = \beta y' + (1-\beta)y_g', \tag{5.11}$$

$$\beta = \beta - \beta(1-1/K)^{\tau}, \tag{5.12}$$

where $\beta$ denotes a constant of smoothing factor, $\tau$ is an integer of exponential growth rate. In this way, the introduced hybrid model reproduces the high-quality solutions by efficiently searching towards more promising regions in the search space. With the smoothing parameters in the CE concept, offspring population solutions moves faster convergence towards the true Pareto-optimal front and maintain the diversity in the discovered nondominated set of solutions.

## 5.5 Experimental Studies

In the interest of assessing the performance of the proposed MOEA/D-PC algorithm, it is compared to some state-of-the-art metaheuristics using the 10 benchmark problems from the CEC2009 contest on unconstrained multi-objective optimization. These test functions include seven 2-objective functions (i.e., *UF*1–*UF*7) and three 3-objective functions (i.e., *UF*8–*UF*10) with complicated Pareto Fronts [78]. Each problem has different characteristics and their descriptions are presented in Table 5.1. This table lists name, number of objectives, dimension, search domain, geometry, modality and separability of all problems. The detailed mathematical representations of these test functions are given in [78]. According to the PF geometry, these multi-objective test problems present different shapes of their Pareto-optimal sets (concave, convex, continuous, discrete, linear, and mixed). Also, the decision variables in most of the problems are dependent on each other due to nonseparable features. All test problems contain decision variable linkages and are tightly difficult to classical

MOEAs. Each problem is defined over 30 decision variables and a maximum of 300,000 function evaluations (FEs) is allowed for each test problem including 2-objective and 3-objective functions. Each algorithm is independently run 30 times for each test instance.

Table 5.1: Description (Name, number of Objective, Dimension, Domain, Geometry, Modality and Separability) of CEC2009 Benchmark Multi-objective Test Problems with $M$ number of objective, $D$ number of decision variables, scalable $S$, separable $SP$, nonseparable $NS$

| *Problem* | *M* | *D* | *Domain* | *Geometry* | *Modality* | *SP/NS* |
|---|---|---|---|---|---|---|
| *UF* 1 | 2 | 30(*S*) | $[0,1]\times[-1,1]^{D-1}$ | Convex | Multimodal | SP |
| *UF* 2 | 2 | 30(*S*) | $[0,1]\times[-1,1]^{D-1}$ | Convex | Multimodal | NS |
| *UF* 3 | 2 | 30(*S*) | $[0,1]^N$ | Convex | Multimodal | NS |
| *UF* 4 | 2 | 30(*S*) | $[0,1]\times[-2,2]^{D-1}$ | Concave | Multimodal | NS |
| *UF* 5 | 2 | 30(*S*) | $[0,1]\times[-2,2]^{D-1}$ | Linear | Multimodal | NS |
| *UF* 6 | 2 | 30(*S*) | $[0,1]\times[-1,1]^{D-1}$ | Linear, Disconnected | Multimodal | NS |
| *UF* 7 | 2 | 30(*S*) | $[0,1]\times[-1,1]^{D-1}$ | Linear | Multimodal | NS |
| *UF* 8 | 3 | 30(*S*) | $[0,1]^2\times[-2,2]^{D-2}$ | Concave | Multimodal | SP |
| *UF* 9 | 3 | 30(*S*) | $[0,1]^2\times[-2,2]^{D-2}$ | Linear, Disconnected | Multimodal | SP |
| *UF* 10 | 3 | 30(*S*) | $[0,1]^2\times[-2,2]^{D-2}$ | Concave | Multimodal | NS |

The hybrid multi-objective algorithm is implemented using Matlab®10a programming language environment and a personal computer with (Intel (tm) i5-2540 Dual Core Processor, 2.60 GHz clock speed, and 4GB RAM). The precision for the floating-point operations is set to 15 fractional digits. In all tables illustrating experimental results, scores of the best performing algorithms are typed in boldface.

The parameter settings in the experimental studies are kept in consistent in all trials. Details of parameter settings are as follows:

- Initial population $N$: 600 for 2-objectives, 1000 for 3-objectives problems; dimension of decision variables $D$ = 30;
- Neighborhood size and maximum number of replacement: $Tb$ = 0.1$N$, $n_r$ = 0.01$N$;

- For PC search, selection size, step size, cooling rate, limit factor and convergence parameter: $Np$ = 10, $\alpha_s = 0.098$, $\alpha_T = 0.9$, $\lambda = 0.9$ and $\varepsilon = 0.0001$;
- Selection rate from neighborhood: $\delta = 0.9$;
- For DE, crossover rate and mutation rate: $CR = 1.0$ and $F = 0.5$;
- For mutation operators, distribution index, mutation probability, smoothing constant and integer parameter: $\eta = 20$, $p_m = 1/N$, $\beta = 0.8$ and $r = 9$;
- Terminating condition: the algorithm stops after 300,000 function evaluations for all test instances.

Based on the CEC2009 setings for comparative evaluations, IGD scores of the proposed algorithm and its competitors are used to conclude on their mutual success. For the PC algorithm calculation of utility function is carried out in every 20 iterations. In order to verify the performance of all algorithnms under consideration, a popular quality indicator that takes into consideration both the accuracy of a solution set and its diversity is the inverse generational distance (IGD) [79]. IGD measures the average Euclidean distance between the true Pareto front and the approximation obtained by a multi-objective algorithm. Let $P^*$ be a set of uniformly distributed Pareto optimal solutions in the objective space along the PF. Let $A$ be an approximation to the PF, the average distance from $P^*$ to $A$ is defined as:

$$IGD(A, P^*) = \frac{\sum_{v \in P^*} d(v, A)}{|P^*|} \tag{5.12}$$

where $d(v, A)$ denotes the minimum Euclidean distance between the true PF and the approximate PF $A$. $|P^*|$ indicates the number of selected points used to represent the PF. IGD is normally used as an indication of the diversity and convergence of solutions. When the value of the IGD is lower, a better approximation is achieved. In our experiments, the numbers of nondominated solutions selected from final nondominated solutions to compute IGD are 100 for 2-objective functions and 150 for 3-objective functions.

The proposed algorithm is run under the standard configurations described in CEC2009 technical report [78]. The IGD scores for each benchmark function are shown in Table 5.2. This table includes the best score, the worst score, mean score and standard deviation of the IGD values obtained for each test instance over the 30 independent runs of the introduced approach MOEA/D-PC. Also the mean IGD values obtained by MOEA/D are also presented in the same table. The results associated with MOEA/D can be found in [65]. It is evident from the table that MOEA/D-PC outperforms MOEA/D for almost all the 10 test problems, loses in 1 problem only.

Table 5.2: IGD values obtained with MOEA/D-PC and MOEA/D for 10 unconstrained test problems ($UF1-UF10$) in 30 independent runs

| *Func.* | *MOEA/D-PC* | | | | *MOEA/D* | |
|---|---|---|---|---|---|---|
| | *Best* | *Worst* | *Mean* | *Std* | *Mean* | *Std* |
| *UF* 1 | 0.003436 | 0.004059 | **0.003885** | **0.000147** | 0.00435 | 0.00029 |
| *UF* 2 | 0.005226 | 0.007046 | **0.005678** | **0.000504** | 0.00679 | 0.00182 |
| *UF* 3 | 0.002478 | 0.007266 | **0.003195** | **0.000846** | 0.00742 | 0.00589 |
| *UF* 4 | 0.039551 | 0.040698 | **0.039756** | **0.000429** | 0.06385 | 0.00534 |
| *UF* 5 | 0.089697 | 0.112095 | **0.091366** | **0.005242** | 0.18071 | 0.06811 |
| *UF* 6 | 0.049578 | 0.060721 | 0.056016 | 0.004922 | **0.00587** | **0.00171** |
| *UF* 7 | 0.003169 | 0.005926 | **0.003622** | **0.000838** | 0.00444 | 0.00117 |
| *UF* 8 | 0.049672 | 0.053630 | **0.049893** | **0.000919** | 0.05840 | 0.00321 |
| *UF* 9 | 0.034935 | 0.042759 | **0.035534** | **0.001733** | 0.07896 | 0.05316 |
| *UF* 10 | 0.167260 | 0.471266 | **0.274512** | 0.110615 | 0.47415 | **0.0736** |

Performance of MOEA/D-PC algorithm is also evaluated and compared with a set of stochastic and deterministic methods. This experimental analysis is carried out over ten different multi-objective optimization algorithms as shown in Table 5.3(a, b). The algorithms selected for comparative evaluations are related to PC and DE as well as several MOEA/D versions. The first two methods are related to PC, i.e., MOPC and MOPC/D. GDE3 is one of popular multi-objective optimization algorithm because it is a simple but effective MODE with fixed parameter settings. MOEADGM and extension of the MOEA/D, which introduces two types of techniques of a guided mutation operator and a priority queue for updating population elements [80]. OW-MOSaDE adopts an objective-wise learning scheme by adaptive parameter settings [81]. AMGA (Archive-based Micro Genetic Algorithm) is a constrained hybrid multi-objective evolutionary optimization approach involving a classical gradient-based single objective optimization algorithm [82]. NSGA-II-ls performs a local search by using a scalarizing function [83].

Table 5.3(a): Comparison of *mean* IGD values and the standard deviation (*Std*) between 10 algorithms and MOEA/D-PC on unconstrained CEC2009 benchmark test functions ($UF1-UF5$) in 30 independent runs with 300,000 FEs

| *Alg.* | *Measure* | *UF* 1 | *UF* 2 | *UF* 3 | *UF* 4 | *UF* 5 |
|---|---|---|---|---|---|---|
| MOEA/D-PC | Mean IGD | **0.003885** | **0.005678** | **0.003195** | 0.039756 | 0.091366 |
| | Std | 0.000147 | 0.000504 | **0.000846** | **0.000429** | 0.005242 |
| MOPC | Mean IGD | 0.02417 | 0.03857 | 0.17403 | 0.11505 | 0.50165 |
| | Std | 0.00355 | 0.00147 | 0.01925 | 0.00605 | 0.02940 |
| MOPC/D | Mean IGD | 0.0097 | 0.0101 | 0.0128 | 0.0426 | 0.146 |
| | Std | 0.0029 | 0.0023 | 0.0087 | 0.0020 | 0.0340 |
| GDE3 | Mean IGD | 0.00534 | 0.01195 | 0.10639 | **0.0265** | 0.03928 |
| | Std | 0.000342 | 0.001541 | 0.0129 | 0.000372 | 0.003947 |
| MOEADGM | Mean IGD | 0.0062 | 0.0064 | 0.0429 | 0.0476 | 1.7919 |
| | Std | 0.00113 | 0.00043 | 0.0341 | 0.00222 | 0.512 |
| NSGA-II-ls | Mean IGD | 0.01153 | 0.01237 | 0.10637 | 0.0584 | 0.5657 |
| | Std | 0.0073 | 0.009108 | 0.06864 | 0.005116 | 0.1827 |
| OW-MOSaDE | Mean IGD | 0.0122 | 0.0081 | 0.103 | 0.0513 | 0.4303 |
| | Std | 0.0012 | 0.0023 | 0.019 | 0.0019 | 0.0174 |
| AMGA | Mean IGD | 0.035886 | 0.016236 | 0.06998 | 0.040621 | 0.094057 |
| | Std | 0.010252 | 0.003167 | 0.013954 | 0.00175 | 0.012055 |
| MO-ABC/DE | Mean IGD | 0.00632 | 0.00614 | 0.04552 | 0.02896 | **0.0250** |
| | Std | **0.000061** | **0.000082** | 0.00184 | 0.00134 | **0.00112** |
| P-MOEA/D | Mean IGD | 0.0046 | 0.0092 | 0.0060 | 0.0520 | 0.2362 |
| | Std | 0.0001 | 0.0018 | 0.0016 | 0.0021 | 0.0489 |
| SMPSO | Mean IGD | 0.06279 | 0.04393 | 0.12459 | 0.1083 | 0.74538 |
| | Std | 0.00663 | 0.00157 | 0.03697 | 0.00463 | 0.15191 |

Table 5.3(b): Comparison of *mean* IGD values and the standard deviation (*Std*) between 10 algorithms and MOEA/D-PC on unconstrained CEC2009 benchmark test functions (*UF*6−*UF*10) in 30 independent runs with 300,000 FEs

| *Alg.* | *Measure* | *UF* 6 | *UF* 7 | *UF* 8 | *UF* 9 | *UF* 10 |
|---|---|---|---|---|---|---|
| MOEA/D-PC | Mean IGD | **0.056016** | **0.003622** | **0.049893** | **0.035534** | **0.274512** |
| | Std | **0.004922** | 0.000838 | **0.000919** | **0.001733** | 0.110615 |
| MOPC | Mean IGD | 0.11150 | 0.05208 | 0.36911 | 0.15139 | 0.44892 |
| | Std | 0.01554 | 0.00297 | 0.01215 | 0.00443 | 0.01813 |
| MOPC/D | Mean IGD | 0.0724 | 0.0113 | 0.0771 | 0.0787 | 0.3412 |
| | Std | 0.0193 | 0.0011 | 0.0060 | 0.0230 | 0.0854 |
| GDE3 | Mean IGD | 0.25091 | 0.02522 | 0.24855 | 0.08248 | 0.43326 |
| | Std | 0.019573 | 0.008891 | 0.035521 | 0.022485 | **0.012323** |
| MOEADGM | Mean IGD | 0.5563 | 0.0076 | 0.2446 | 0.1878 | 0.5646 |
| | Std | 0.147 | 0.00094 | 0.0854 | 0.0287 | 0.102 |
| NSGA-II-ls | Mean IGD | 0.31032 | 0.02132 | 0.0863 | 0.0719 | 0.84468 |
| | Std | 0.19133 | 0.01946 | 0.01243 | 0.04504 | 0.1626 |
| OW-MOSaDE | Mean IGD | 0.1918 | 0.0585 | 0.0945 | 0.0983 | 0.743 |
| | Std | 0.029 | 0.0119 | 0.0244 | 0.0885 | 0.0384 |
| AMGA | Mean IGD | 0.129425 | 0.057076 | 0.171251 | 0.18861 | 0.324186 |
| | Std | 0.056588 | 0.065309 | 0.017224 | 0.042137 | 0.095718 |
| MO-ABC/DE | Mean IGD | 0.08659 | 0.05606 | 0.18658 | 0.27688 | 0.29283 |
| | Std | 0.00554 | 0.00183 | 0.00124 | 0.0209 | 0.0636 |
| P-MOEA/D | Mean IGD | 0.0721 | 0.0052 | 0.0725 | 0.0769 | 0.4112 |
| | Std | 0.0278 | **0.0005** | 0.0083 | 0.0514 | 0.0510 |
| SMPSO | Mean IGD | 0.33251 | 0.0481 | 0.23546 | 0.18704 | 0.28438 |
| | Std | 0.03695 | 0.00139 | 0.01406 | 0.02434 | 0.02143 |

MO-ABC/DE is a recent hybrid multi-objective optimization metaheuristic that combines natural behavior of ABC with the properties of the DE [84]. Speed-constrained Multi-objective PSO (SMPSO) utilizes the limited velocity of the particles to produce new particle positions [85]. Besides, a parallel version of MOEA/D is designed to integrate within the DREAM software package (P-MOEA/D) for test problems [86]. Table 5.3(a, b) lists mean IGD scores of the 10 algorithms, including MOEA/D-PC, for unconstrained functions in CEC2009 benchmarks set with 300,000 FEs through 30 independent runs.

As shown in the Table 5.2 and 5.3(a, b), it is apparent that MOEA/D-PC is the most competitive approach obtaining the best (lowest) IGD values for 7 of 10 problems (i.e., *UF*1, *UF*2, *UF*3, *UF*7, *UF*8, *UF*9, and *UF*10). GDE3 method wins on *UF*4 while the MO-ABC/DE algorithm has best fronts for *UF*5. Overall, the MOEA/D-PC approach obtains best approximation of high quality solutions for 7 test functions and outperforms the other competitors on all the test problems except for *UF*4, *UF*5, and *UF*6.

Also, the graphical representations of the PF's produced by the MOEA/D-PC algorithm are illustrated in Figure 5.3(a, b). These plots show the quality of the PF's produced by the proposed approach with the lowest IGD values for *UF*1−*UF*10. It is seen from Figure 5.3(a, b) that the MOEA/D-PC algorithm found high quality PF's on *UF*1, *UF*2, *UF*3, *UF*7, *UF*8, *UF*9 and *UF*10, but performs relatively poor performance on *UF*4, *UF*5, and *UF*6. *UF*5 and *UF*6 have discontinuous PF's while *UF*4 is a 2-objective nonseparable concave functions.

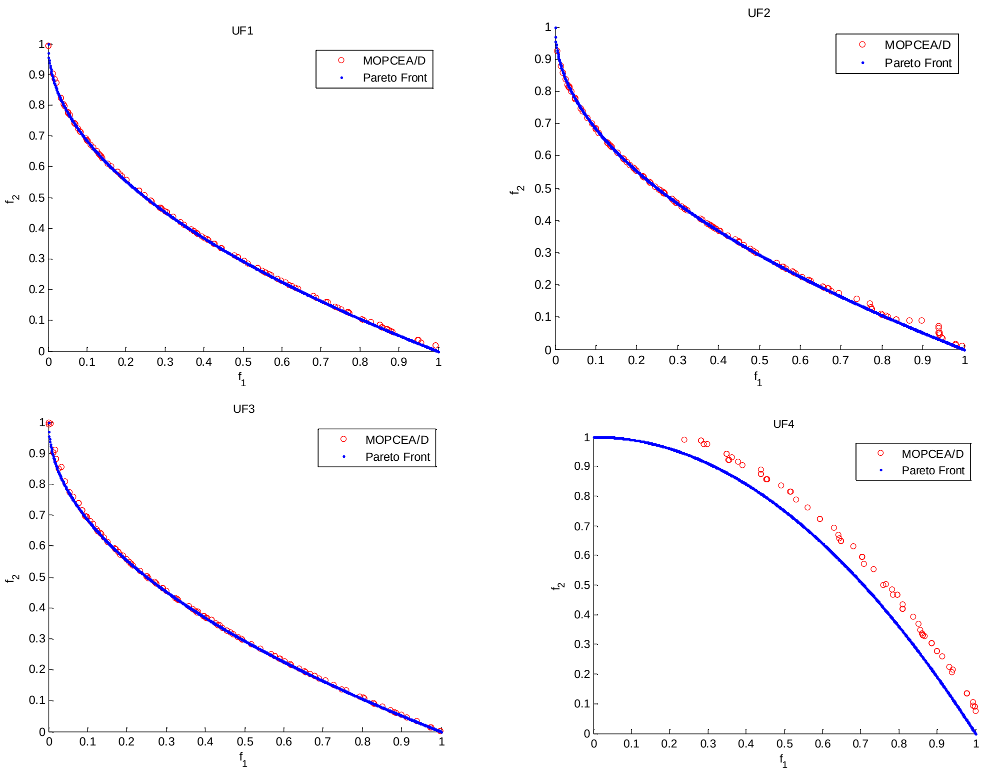


Figure 5.3(a): The final approximation Pareto front with lowest IGD value of the objective space for CEC2009 (*UF*1− *UF*4) test problem

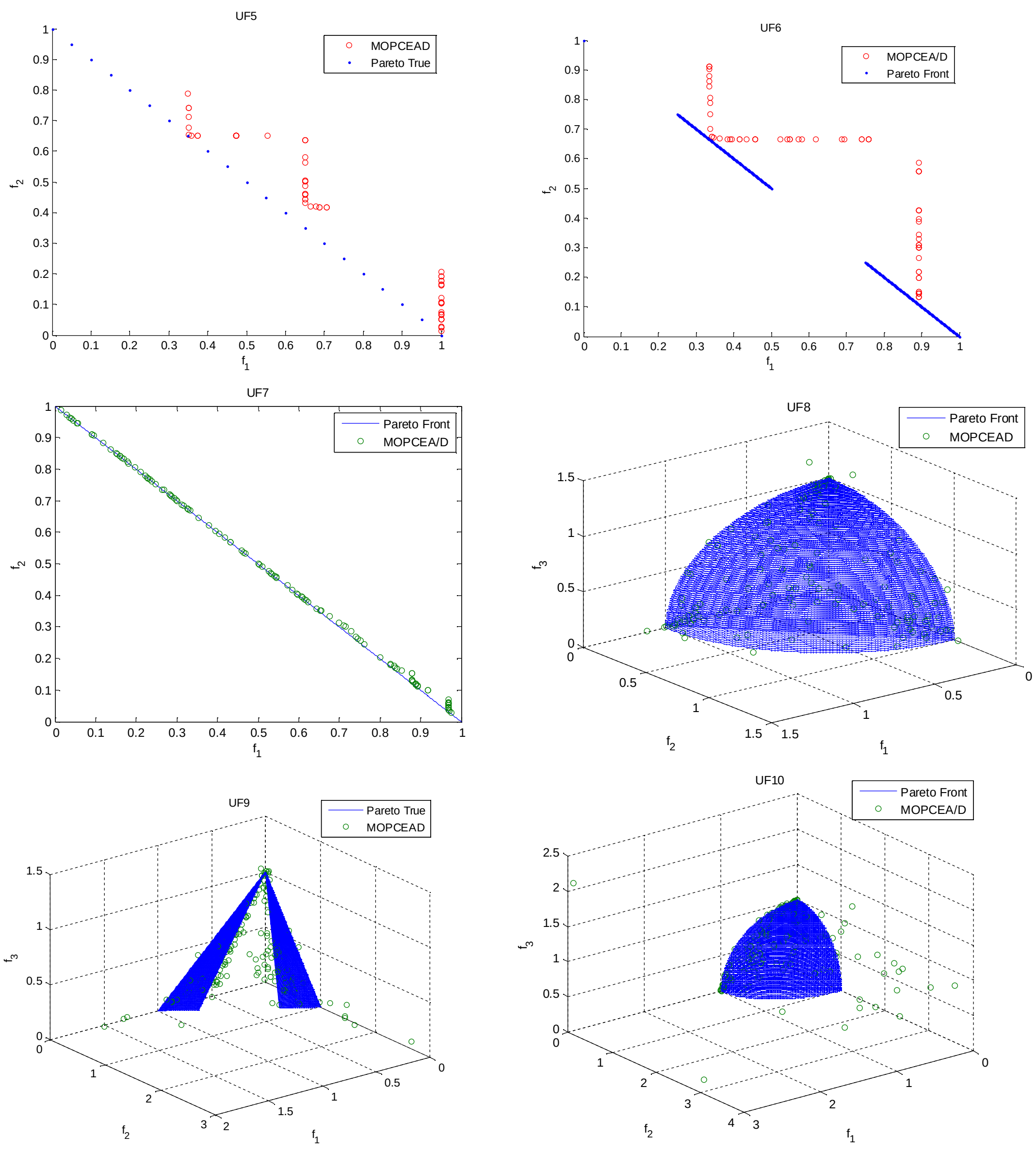


Figure 5.3(b): The final approximation Pareto front with lowest IGD value of the objective space for CEC2009 ($UF5- UF10$) test problem

As a result the above listed experimental results and the associated discussions, it can be condluded that the proposed MOEA/D-PC has competitive performance against the other competitors for the multi-objective test problems under consideration. This is due to the fact that the proposed algorithm utilizes PC approach and MOEA/D framework. MOEA/D provides several well-spread directions defined by the decomposition method and the PC approach, finds the favorable strategies as the promising indicators.

# Chapter 6

# CONCLUSIONS AND FUTURE PERSPECTIVES

A variety of real-world problems can be formulated into continuous optimization problems satisfied with several requirements. This thesis presents hybridizations using the PC and the DE algorithms. In this research work, several novel methods have been introduced by a great deal of heuristic techniques.

As the part of MAS, PC applies random sampling and selection mechanism to generate a set of individuals to form solution distributions. Each variable in the solutions entitles its associated learning agent to coordinate with distributed probabilities in the procedure of optimization at utility level. All agents employ gradient-based techniques to minimize a Homotopy function so that the approach can suit for a variety of problems.

Based on PC optimization, two promising hybridizations MPCDE and MOEA/D-PC were proposed for single and multiple objective problems. Meanwhile, all components of two proposed optimization methods are also discussed in the context of PC-based single and multiple objective algorithms. In single objective optimization architecture, collectives attempt to trace the promising directions for improvements in terms of evolving probability distributions and population of solutions as the overall optimization process proceeds. For two well-known test problem sets, experimental studies are conducted for comparisons between well-known metaheuristic methods. The experimental results show that the MPCDE algorithm achieved competitive performance compared to the other state-of-the-art algorithms for most of the benchmark functions.

PC-based optimization provides the appropriate search trend with the MOEA/D framework to inroduce a synergetic combination for multi-objective optimization. The proposed adaptive mechanism for multi-objective optimization is a generic approach, called MOEA/D-PC, which hybridizes PC and MOEA/D search algorithms. Initially, a set of single objective problems are established by decomposition approach with randomly weighted aggregation of corresponding objective functions. So the optimization tasks goes along a set of well-spread directions, which can be performed by PC-based single objective optimization. Furthermore, an extension of mutation operator is proposed as combining polynomial mutation with the Gaussian mutation. This promising perturbation is then guided by CE operators to explore solutions from local front to nondominated front of MOP, while the hybrid algorithm achieves greater diversity and converge to the PF in a global search. The performance of the proposed hybrid algorithm is assessed by conducting experiment on a range of unconstrained CEC2009 MOEA competition benchmarks. The experimental results have revealed satisfactory performance of the introduced approach for MOPs.

Based on the experimental results, PC approach may fail in local minima when the search space is highly biased, multimodal, and deceptive. In fact, the sampling strategies of PC adopted in this thesis noticeably depend on the lower and the upper bound of variable ranges. The boundaries essentially control the domain of the sampling mechanism, thus can shrink the search space in producing new solutions especially when the decision variables are highly interreleted for some problems. In this way, the updated domain is only defined according to the limit factor and the favorable strategy for the current iteration. However, for some special problems, the proposed algorithm may not prevent search processes from premature convergence since restricted range is possibly led away from the global optimal solution. In order to address this problem, it would be appropriate to adopt an admissible global supervised individual as a reference point to modify the search space for generating strategy set of PC. In this regard, the lower and the upper bound could be defined according to the favorable strategy with the reference point. From this idea, the updated domain may consider the adaptive exploration of existing solution space towards either the favorable strategy or the reference point relying on a predefined parameter.

Even though this research study has demonstrated the efficiency of MOEA/D-PC in solving MOPs, the constructed technique of MOEA/D-PC is recognized to be inefficient for complex problems with rigid properties. The unconstrained multi-objective test problems involve different types of characteristics such as continuous, linear, multimodality, separablity and so on. In fact, multimodality is always a challenging issue in optimization. CEC2009 problems may consist of several differences, which may arise in the solution representation and level of conflict between objective functions, which could be further studied for improving the performance of the proposed algorithm. In order to address slow convergence and poor performance for some problems, different operators can be designed for the purpose of improving the global search engine in the evolutionary optimization process. It is especially desirable to replace mutation scheme and add an evaluative recombination operator to handle MOPs. Future studies could utilize combined mutation scheme between polynomial and Gaussian adjusted by a reference parameter as a threshold instead of the CE method to modify the current population of solutions. Then, simulated binary crossover would be applied for selecting better offspring solution from mutated and existing populations to maintain population diversity of MOEA/D framework. This can make the algorithm capable to explore the solution space and avoid local Pareto optimal fronts.

# REFERENCES

[1] Storn, R., & Price, K. (1997). Differential Evolution − a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *Journal of Global Optimization*, Kluwer Academic Publishers, December 1997, 11(4), pp. 341–359.

[2] Holland, J. H. (1975). Adaptation in Natural and Artificial Systems. 1st ed., University of Michigan Press, Ann Arbor; 2nd ed., MIT Press, Cambridge.

[3] Deb, K. (2001). Multi-Objective Optimization using Evolutionary Algorithms. John Wiley & Sons, Chichester, UK, ISBN 0-471-87339-X.

[4] Wolpert, D. H., Strauss, C. E. M., & Rajnarayan, D. (2006). Advances in distributed optimization using probability collectives. *Advances in Complex Systems*, pp. 383-436.

[5] Wolpert, D. H., & Tumer, K. (1999). An introduction to collective intelligence. Technical Report NASA-ARC-IC-99-63, NASA Ames Research Center. *Handbook of Agent Technology*, Bradshaw, J. M. (ed.) AAAI/MIT Press.

[6] Wolpert, D. H. (2006). Information theory—the bridge connecting bounded rational game theory and statistical physics. *Complex Engineering Systems*, Braha, D., Minai, A. A., & Bar-Yam, Y. eds., Springer, Cambridge, 262–290.

[7] Wolpert, D. H., Antoine, N. E., Bieniawski, S. R., & Kroo, I. R. (2004). Fleet assignment using collective intelligence. Proceedings of 42nd AIAA *Aerospace*

*Science Meeting Exhibit*, AIAA-2004-0622.

[8] Wolpert, D. H. & Bieniawski, S. R. (2004). Distributed control by lagrangian steepest descent. In Proceedings of 43rd IEEE conference on decision and control, Washington, D.C, Vol. 2, pp. 1562–1567.

[9] Lee, C. F., & Wolpert, D. H. (2004). Product distribution theory for control of multi-agent systems. *Proceedings of* 3rd *international joint conference on autonomous agents and MAS*, Vol 2, IEEE Computer Society, Washington.

[10] Kulkarni, A. J., & Tai, K. (2008). Probability collectives for decentralized, distributed optimization: a collective intelligence approach. *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics* (*SMC*'08), pp.1271–1275.

[11] Kulkarni, A. J., & Tai, K. (2009). Probability collectives: a decentralized, distributed optimization for multi-agent systems. *Applications of Soft Computing*, pp. 441–450, Springer.

[12] Bieniawski, S. R., Kroo, I. M., & Wolpert, D. H. (2004). Discrete, continuous, and constrained optimization using collectives. *Proceedings of the* 10th *AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, vol. 5, pp.3079–3087.

[13] Kulkarni, A. J., Kale, I. R., & Tai, K. (2013). Probability Collectives for Solving Truss Structure Problems. *Proceedings of* 10th *World Congress on Structural and Multidisciplinary Optimization* (*WCSMO*), Orlando, Florida, USA, 19–24, PID:

5395.

[14] Bieniawski, S. R. (2005). Distributed optimization and flight control using collectives. PhD thesis, Stanford University.

[15] Kulkarni, A. J., & Tai, K. (2010). Probability Collectives: A multi-agent approach for solving combinatorial optimization problems. *Applied Soft Computing*, Vol. 10, No. 3, pp.759 –771.

[16] Kulkarni, A. J., & Tai, K. (2011). A probability collectives approach with a feasibility-based rule for constrained optimization. *Applied Computational Intelligence and Soft Computing*, Vol. 2011, Article ID 980216.

[17] Wolpert, D. H., Bieniawski, S. R., & Rajnarayan, D.G. (2013). Probability Collectives in Optimization. In Rao, C. R. & Govindaraju, V. (Eds.), *Handbook of Statistics – Machine Learning*: *Theory and Applications* (Vol. 31, pp. 61–99). Elsevier, Amsterdam.

[18] Kulkarni, A. J., & Tai, K. (2010). Probability collectives: a distributed optimization approach for constrained problems. *IEEE congress on Evolutionary computation* (*CEC*), Barcelona, Spain, pp.1–8.

[19] Wang, N., Shen, L. C., Liu, H. F., Chen, J., & Hu, T. J. (2013). Robust optimization of aircraft weapon delivery trajectory using probability collectives and meta-modeling. *Chinese Journal of Aeronautics,* Vol. 26, No. 2, pp.423–434.

[20] Kulkarni, A. J., Kale, I. R., & Tai, K. (2014). Probability Collectives for Solving Discrete and Mixed Variable Problems. *International Journal of Computer Aided Engineering and Technology*. DOI: 10.1504/IJCAET.2016.10000085

[21] Kulkarni, A. J., Tai, K., & Abraham, A. (2015). Probability Collectives: A Distributed Multi-Agent System Approach for Optimization. Intelligent Systems Reference Library 86, Springer International Publishing Switzerland.

[22] Waldock, A., & Corne, D. (2010). Multi-objective probability collectives. *Applications of Evolutionary Computation*, Istanbul, Turkey, pp. 461-470.

[23] Morgan, D., Waldock, A., & Corne, D. (2013). MOPC/D: A new Probability Collectives algorithm for Multiobjective Optimisation. *IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making* (*MCDM*), pp. 17-24.

[24] Jia, D. L., Zheng, G. X., & Khan, M. K. (2011). An effective memetic differential evolution algorithm based on chaotic local search. *Information Sciences,* Elsevier Science, August 2011, Vol. 181, No. 15, pp.3175–3187.

[25] Yildiz, A. R. (2013). A new hybrid differential evolution algorithm for the selection of optimal machining parameters in milling operations. *Applied Soft Computing*, Vol. 13, No. 3, March 2013, pp.1561–1566.

[26] Ali, M., & Pant, M., (2010). Improving the performance of differential evolution algorithm using cauchy mutation. *Soft Computing*, Vol. 15 (5), pp. 991–1007.

[27] Zhang, J., & Sanderson, A. (2009). JADE: Adaptive Differential Evolution with Optional External Archive. *IEEE Trans. Evol. Comput.*, Vol. 13(5), pp. 945–958.

[28] Das, S., Abraham, A., Chakraborty, U., & Konar, A. (2009). Differential evolution using a neighborhood-based mutation operator. *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 526–552, Jun. 2009.

[29] Pant, M., Ali, M., & Abraham, A. (2009). Mixed Mutation Strategy Embedded Differential Evolution. *IEEE Congress on Evolutionary Computation*, pp. 1240–1246.

[30] Qin, A. K., Huang, V. L. & Suganthan, P. N. (2009). Differential evolution algorithm with strategy adaptation of global numerical optimization. *IEEE Transactions on Evolutionary Computation*, Vol. 13, No. 2, April 2009, pp.398–417.

[31] Brest, J., Greiner, S., Boskovic, B., Mernik, M., & Zumer, V. (2006). Self Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems. *IEEE Trans. Evol. Comput.*, 10(6), 646–657.

[32] Kaelo, P., & Ali, M. M. (2006). A numerical study of some modified differential evolution algorithms. *European Journal of Operational Research*, vol. 169, no. 3, pp. 1176–1184.

[33] Babu, B.V., & Mathew Leenus Jehan., M. (2003). Differential Evolution for Multi-Objective Optimization. In *Proceedings of the* 2003 *Congress on*

*Evolutionary Computation* (*CEC*'2003), vol. 4, pp. 2696–2703, IEEE Press, Canberra, Australia.

[34] Lampinen, J. (2001). DE's selection rule for multiobjective optimization. Technical Report, Lappeenranta University of Technology, Dept. of Inf. Tech.

[35] Kukkonen, S., & Lampinen, J. (2005). GDE3: The third evolution step of Generalized Differential Evolution. In *Proceedings of the IEEE Congress on Evolutionary Computation* (*CEC*'2005), Edinburgh, UK, Vol.1, 5-5, pp.443-450.

[36] Huang, V. L., Qin, A. K., Suganthan, P. N., & Tasgetiren, M. F. (2007). Multi-objective Optimization based on Self-adaptive Differential Evolution Algorithm. *IEEE Congress on Evolutionary Computation,* 2007, Singapore , pp. 3601-3608.

[37] Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning. *Addison-Wesley*, Reading, MA.

[38] Zhang, Q., & Li, H. (2007). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6), pp. 712–731.

[39] Rubinstein, R. Y., & Kroese, D. P. (2004). The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization. *Monte-Carlo Simulation and Machine Learning*. Springer-Verlag, New York.

[40] Lepš, M. (2005). Single and Multi-Objective Optimization in Civil Engineering

with Applications. PhD thesis, CTU in Prague.

[41] Miettinen, K. (1999). Nonlinear Multiobjective Optimization. *Kluwer Academic Publishers*, Norwell, MA.

[42] Xu, Z., Unveren, A., & Acan, A. (2016). Probability Collectives Hybridized with Differential Evolution for Global Optimization. *International Journal of Bio-Inspired Computation*, Inderscience Publishers, Vol. 8, no. 3.

[43] Price, K., Storn, R., & Lampinen, J. (2005). Differential evolution – a practical approach to global optimization. *Springer*, Berlin.

[44] Boer, P., Kroese, D., Mannor, S., & Rubinstein, R. (2005). A tutorial on the cross-entropy method. *Ann. Oper. Res.*, vol. 134, no. 1, pp. 19–67, Jan. 2005.

[45] Rubinstein, R. Y., & Kroese, D. P. (2004). The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization. *Monte-Carlo Simulation and Machine Learning*. Springer-Verlag, New York.

[46] Lampinen, J., & Zelinka, I. (2000). On stagnation of the differential evolution algorithm. in Ošmera, P. (Eds.) Procs. of MENDEL 2000, 6th *International Mendel Conference on Soft Computing*, June 2000, Brno, Czech Republic, pp.76–83.

[47] Yao, X., Liu, Y., & Lin, G. (1999). Evolutionary Programming Made Faster. *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102.

[48] Suganthan, P. N., Hansen, N., Liang, J. J., Deb, K., Chen, Y., Auger, A. & Tiwari, S. (2005). Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. Technical Report, Nanyang Technological University, Singapore, May 2005, KanGAL Report #2005005, IIT Kanpur, India.

[49] Jamil, M., & Yang, X. (2013). A literature survey of benchmark functions for global optimization problems. *Int. Journal of Mathematical Modelling and Numerical Optimisation*, Vol. 4, No. 2, pp. 150-194.

[50] Vesterstrom, J., & Thomsen, R. (2004). A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. *Congress on Evol. Comput.*, June 2004, Vol. 2, pp.1980–1987.

[51] Shi, Y., & Eberhart, R. C. (1999). Empirical study of particle swarm optimization. In *Procs. of the Congress on Evolutionary Computation* (*CEC* ’99), Washington, DC, USA, July 1999, Vol. 3, pp.1945–1950.

[52] Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, Springer US, November 2007, Vol. 39, No. 3, pp.459–471.

[53] Karaboga, D., & Akay, B. (2009). A comparative study of artificial bee colony algorithm. *Applied Mathematics and Computation*, August 2009, 214(1), pp.108–132.

[54] Picek, S., Jakobovic, D., & Golub, M. (2013). On the recombination operator in the real-coded genetic algorithms. In *Procs. of Congress on Evolutionary computation* (*CEC* 2013), June 2013, pp.3103–3110.

[55] Acan, A., & Ünveren, A. (2015). A two-stage memory powered Great Deluge algorithm for global optimization. *Soft Computing*, Springer Berlin Heidelberg, September 2015, Vol. 19, No. 9, pp.2565–2585.

[56] Hu, Z., Su, Q., Xiong, S., & Hu, F. (2008). Self-adaptive Hybrid Differential Evolution with Simulated Annealing Algorithm for Numerical Optimization. In *Procs of the IEEE Congress on Evolutionary Computation* (*CEC* '08), China, pp.1189–1194.

[57] Roy, P., Islam, M. J., & Islam, M. M. (2012). Self-adaptive Genetically Programmed Differential Evolution. In *Procs. of* 7th *International Conference on Electrical and Computer Engineering* (*ICECE*), Dhaka, Bangladesh, pp.639–642.

[58] García-Martínez, C., & Lozano, M. (2005). Hybrid Real-Coded Genetic Algorithms with Female and Male Differentiation. In *Procs of the* 2005 *IEEE Congress on Evolutionary Computation*, Edinburgh, Scotland, Vol. 1, pp.896–903.

[59] Hansen, N., & Ostermeier, A. (2001). Completely Derandomized Self-Adaptation in Evolution Strategies. *Evolutionary Computation*, Vol. 9, No. 2, pp.159–195.

[60] Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. A. M. T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on*

*Evolutionary Computation*, 6(2), 182-197, Apr. 2002.

[61] Zitzler, E., Laumanns, M., & Thiele, L. (2001). SPEA2: Improving the strength Pareto evolutionary algorithm. Technical report 103, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Zurich, Switzerland, pp. 1-21.

[62] Zitzler, E. (2004). Indicator-based selection in multiobjective search. Parallel Problem Solving from Nature-PPSN VIII, 1-11.

[63] Hughes, E. J. (2003). Multiple single objective pareto sampling. *The* 2003 *Congress on Evolutionary Computation*. *CEC*'03. vol. 4, 2678-2684.

[64] Li, H., & Zhang, Q. (2009). Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 284-302.

[65] Zhang, Q., Liu, W., & Li, H. (2009). The Performance of a new version of MOEA/D on CEC'09 unconstrained MOP Test Instances. 2009 *IEEE Congress on Evolutionary Computation*, 18-21 May, 2009, Trondheim, Norway, pp. 203–208.

[66] Fonseca, C. M., & Fleming, P. J. (1993). Genetic algorithms for multiobjective optimization: formulation, discussion and generalization. In: 5th *International Conference Genetic Algorithms*, pp. 416–423.

[67] Branke, J., & Deb, K. (2004). Integrating user preferences into evolutionary multi-

objective optimization. *Tech. Rep. KanGAL* 2004, Indian Institute of Technology.

[68] Jiang, S., Zhang, Jie, Ong, Y. S., Zhang, A. N., & Tan, P. S. (2014). A Simple and Fast Hypervolume Indicator-Based Multiobjective Evolutionary Algorithm. *IEEE Transactions on Cybernetics*, 45(10), December 2014.

[69] Ishibuchi, H., Tsukamoto, N., & Nojima, Y. (2008). Evolutionary many-objective optimization: A short review. In Proc. of 2008 *IEEE Congress on Evolutionary Computation*, pp. 2424-2431, Hong Kong, Jun. 1-6, 2008.

[70] Schaffer, J. D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. 1st *International Conference on Genetic Algorithms* (*ICGA* 1985). Lawrence Erlbaum Associates, Pittsburgh, PA, USA, pp. 93–100.

[71] Elhossini, A., Areibi, S., & Dony, R. (2010). Strength Pareto particle swarm optimization and hybrid EA-PSO for multi-objective optimization. *Evolutionary Computation*, 18(1), 127–156.

[72] Das, I., & Dennis, J. E., (1998). Normal-Boundary Intersection: A New Method for Generating Pareto Optimal Points in Multicriteria Optimization Problems. *SIAM Journal on Optimization*, Vol. 8, No. 3, pp. 631-657.

[73] Deb, K., & Agrawal, R. (1995). Simulated binary crossover for continuous search space. *Complex Systems*, vol. 9, no. 2, pp. 115–148.

[74] Nebro, A. J., & Durillo, J. J. (2010). A Study of the Parallelization of the Multi-

Objective Metaheuristic MOEA/D. In: Blum, C., & Battiti, R. (eds.) LION 4. LNCS, vol. 6073, pp. 303-317. Springer, Heidelberg.

[75] Li, K., Zhang, Q., & Battiti, R. (2014). Hybridization of decomposition and local search for multiobjective optimization. IEEE Trans. Cybern., vol. 44, no. 10, pp. 1808–1820, Oct. 2014.

[76] Rahman, R. A., Ramli, R., Jamari, Z., & Ku-Mahamud, K. R. (2016). Evolutionary Algorithm with Roulette-Tournament Selection for Solving Aquaculture Diet Formulation. *Mathematical Problems in Engineering*, Vol. 2016, pp. 10.

[77] Xu, Z., Unveren, A., & Acan, A. (2016). MOEA/D Integrated with Probability Collectives for Unconstraint Multiobjective Optimization Problems. *International Conference on Machine Learning, Electrical and Electronics Engineering* (*MLEE*-16), pp. 56-63, April 20-21, 2016 Antalya, Turkey.

[78] Zhang, Q., Zhou, A., Zhao, S. Z., Suganthan, P. N., Liu, W., & Tiwari, S. (2008). Multiobjective Optimization Test Instances for the CEC2009 Special Session and Competition. *Technical Report* CES-887, University of Essex, Colchester, UK and Nanyang Technological University, Singapore.

[79] Jiang, S., Ong, Y.-S., Zhang, J., & Feng, L. (2014). Consistencies and Contradictions of Performance Metrics in Multiobjective Optimization. *IEEE Trans. Cybern.* 44, pp. 2391–404.

[80] Chen, C. M., Chen, Y., & Zhang, Q. (2009). Enhancing MOEA/D with guided mutation & priority update for multi-objective optimization. In *Proc*. 2009 *Congress on Evolutionary Computation*, 18-21 May, 2009, Trondheim, Norway, pp. 209-216.

[81] Huang, V. L., Zhao, S. Z., Mallipeddi, R., & Suganthan, P. N. (2009). Multi-objective optimization using self-adaptive differential evolution algorithm. In *Proceedings of Congress on Evolutionary Computation* (*CEC*'09), pp. 190-194.

[82] Tiwari, S., Fadel, G., Koch, P., & Deb, K. (2009). Performance Assessment of the Hybrid Archive-based Micro Genetic Algorithm (AMGA) on the CEC09 Test Problems. In *Proceeding of Congress on Evolutionary Computation*, pp. 1935-1942.

[83] Sindhya, K., Sinha, A., Deb, K., & Miettinen, K. (2009). Local Search Based Evolutionary Multiobjective Optimization Algorithm for Constrained and Unconstrained Problems. In *Proc*. 2009 *Congress on Evolutionary Computation*, Trondheim, 18-21 May 2009, pp. 2919-2926.

[84] Rubio-Largo,A., González-Álvarez, D. L., Vega-Rodríguez, M. A., Gómez-Pulido, J. A., & Sánchez-Pérez, J. M. (2012). MO-ABC/DE - multiobjective artificial bee colony with differential evolution for unconstrained multiobjective optimization. *IEEE International Symposium on Computational Intelligence and Informatics*, pp. 157-162.

[85] Nebro, A. J., Durillo, J. J., García, J., Coello C., C. A., Luna, F., & Alba, E.

(2009). SMPSO: A new pso-based metaheuristic for multi-objective optimization. *Symposium on Computational intelligence in miulti-criteria decision-making*, pp. 66-73.

[86] Laloy, E., & Vrugt, J. (2011). P-MOEA/D: A parallel decomposition-based multiple objective optimization scheme integrated within the DREAM software package. Environmental Modelling & Software, June 27, 2011.