Artificial Bee Colony Optimization for Multiobjective Quadratic Assignment Problem

Haytham Mohammed Eleyan

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

Master of Science
in
Computer Engineering

Eastern Mediterranean University
February 2015
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Serhan Çiftçioğlu
Acting Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Master of Science 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 Master of Science Computer Engineering.

Asst. Prof. Dr. Adnan Acan
Supervisor

Examining Committee

1. Assoc. Prof. Dr. Muhammad Salamah

2. Asst. Prof. Dr. Adnan Acan

3. Asst. Prof. Dr. Mehmet Bodur

ABSTRACT

Excellent ability of swarm intelligence can be used to solve multi-objective combinatorial optimization problems. Bee colony algorithms are new swarm intelligence techniques inspired from the smart behaviors of real honeybees in their foraging behavior. Artificial bee colony optimization algorithm has recently been applied for difficult real-valued and combinational optimization problems. Multiobjective quadratic assignment problem (mQAP) is a well-known and hard combinational optimization problem which is used in modeling of several assignment and scheduling problems. Benchmark mQAP instances are already solved near optimally using competitive metaheuristics and dedicated local search algorithms, but there is no absolute winner of these competitions in the sense that while a particular algorithm is quite successful for a kind of mQAP instance, it exhibits poor performance on the others. In this study, we test the performance of artificial bee colony optimization algorithm over multiobjective quadratic assignment problem. Experiments have shown that the new heuristic was effective and efficient to solve hard mQAP instances.

Keywords: Multi-objective optimization, Artificial Bee Colony, Bees Algorithm, Multiobjective Quadratic Assignment Problem.

ÖZ

Çok amaçlı bileşimsel en iyileme problemleri sürü zekasına dayalı yöntemlerle çözülebilir. Arı kolonisi algoritmaları son zamanlarda geliştirilen ve bal arılarının yiyecek ararken sergiledikleri zeki davranışlardan ilham alınan sürü zekası teknikleridir.

Çok amaçlı ikinci derece atama problemi iyi bilinen ve zor bir bilişimsel en iyileme problemidir.Bu problem bir çok atama ve listeleme probleminin modellenmesinde de yaygın olarak kullanılır. Çok amaçlı ikinci derece atama problemi için kıyas oluşturan örnekler metaheuristikler ve yerel araştırma yakın kalitede algoritmalar kullanılarak en iyi düzeyde çözülmüşlerdir.

Ancak bu çözüm yaklaşımlarının kesin bir kazananı yoktur, bir yöntem bazı problemleri başarıyla çözerken, başka bir problem kümesi için zayıf bir başarım gösterebilmektedir. Bu çalışmada yapay arı kolonisi en iyileme algonitmasının çok amaçlı ikinci derece atama probleminin çözümündeki başarımı test edilmiştir. Yapılan deneysel çalışmalar, bu yöntemin etkili ve hesaplama karmaşıklığı bakımından verimli olduğunu göstermiştir.

Anahtar Kelimeler: Çok Amaçlı Optimizasyon, Yapay Arı Kolonisi, Arı Algoritması, Çok amaçlı ikinci derece atama problemi.

To My Father

Prof.Dr. Mohammed Eleyan

ACKNOWLEDGMENT

When a person wants to achieve an important goal in his life, it cannot be achieved without the help and support of others. This achievement for me is not an exception. First, I want to extend my sincere thanks to the instructor I learned a lot from him, who stand out in my mind is the one who really inspired me to reach prospects and wide fields where I was not confident in the one day that I can reach it, but I achieved this goal because of him, he is Dr.Adnan Acan from Eastern Mediterranean University, I doubt I would be where I am today if it weren't for the valuable lessons he taught me. I want to thank also Dr. Ahmet Ünveren for wonderful times in lectures weekly seminar with the participation of Dr Adnan Acan, where I personally learned a lot from these lectures.

Finally I want to thank my family for their patience and support and would stand beside me in spite of the difficult conditions in which they live in Gaza, thank you dear dad, thank you my beloved mother, thank you to all my brothers and sisters.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATONS

SI	Swarm Intelligence
ABC	Artificial Bee Colony
ACO	Ant Colony Optimization
PSO	Particle Swarm Optimization
FA	Firefly Algorithm
BA	Bees Algorithm
VBA	Virtual Bee Algorithm
Gas	Genetic Algorithms
SA	Simulated Annealing
TS	Tabu Search
LS	Local Search
HBA	Honey Bee Algorithm
HBMO	Honey-Bee Mating Optimization
MOPs	Multiobjective Optimisation Problems
COPs	Combinatorial Optimization Problems
mCOPs	Multiobjective Combinatorial Optimization Problems
LAP	Linear Assignment Problem
QAP	Quadratic Assignment problem
mQAP	MultiobjectivevQuadratic Assignment Problem
TSP	Travelling Salesman Problem
mTSP	Multiobjective Travelling Salesman Problem
NPC	Non-polynomial-complete

Chapter 1

INTRODUCTION

As human being in a world of increasing complexity, we are faced with the problem of optimality in almost all daily activities. We often seek to solve these problems in a way that is efficient and effective from our point of view, it does not matter if we act consciously or not. Let's consider the simple task of commuting between our homes and workplaces as an optimization problem, with the objective of reducing the time it takes (objective value) for the implementation of this commute. This simple, ordinary, and sometimes seemingly trivial task for us, as humans to solve, can be very difficult for a computer.

In general optimization problems can be categorized into types of constrained and unconstrained problems. The first type contains the most practical applications in real-world, where the constraints of an optimization problem are usually handled by an independent (constraint-handling) mechanism [1, 2]. The second type of optimization problems are usually designed as benchmarks to test the performance of optimization algorithms over landscapes of various levels of complexity.

Multi-objective optimization problems (MOPs) deal with more than one objective function. Due to the lack of suitable solution methods in the past, a MOP has been transformed and solved as a single objective problem. In single objective optimization, the goal is to find one solution (or in special cases

multiple optimal solutions). In MOP_S the goal is not to find an optimal solution for each objective function, instead multiple points of comprises are searched for within the multiple objective solution space. Apart from obvious tradeoffs between cost and performance, the performance criteria required by some applications such as fast response time, small overshoot and good robustness, are also conflicting in nature [30, 31, 32, 33]. A general form of a MOP can be described as follows:

$$\begin{aligned} (\text{MOPs}) \quad & \min/\max \ \mathbf{f_k(x)} && k = 1, 2, \ldots, t, \\ & \text{s.t.} \mathbf{g_j(x)} \geq 0 && j = 1, 2, \ldots, l, \\ & \mathbf{h_j(x)} = 0 && j = l+1, \ldots, m, \\ & D_i = [x_i | x_{il} \leq x_i \leq x_{iu}] && i = 1, 2, \ldots, n, \end{aligned} \qquad (1.1)$$

where $\mathbf{x}$ is a vector of n decision variables: $x = (x_{i1}, x_{i2}, \ldots, x_{in})^T$. The decision variable space (decision space) X is bounded by sets of boundary constraints where $\mathbf{x}_{il}$ and $\mathbf{x}_{iu}$ are lower and upper bounds for the decision variable x_i. A solution that does not satisfy all constraints and variable bounds is called an infeasible solution all others are called feasible solutions. The set of all feasible solutions is called the feasible region $\tilde{S} \subseteq X$, where $X = D_1 \times D_2 \times \ldots\ldots\ldots \times D_n$.

The main difference between single and multiobjective optimizations is that, in multiobjective optimization the objective functions constitute a multi-dimensional space (objective space z). Each solution $\mathbf{x}$ maps to a point z in the objective space, where $f(x) = z = (z_1, z_2, \ldots, z_t)^T$. The mapping transfers an n-dimensional solution vector into an t-dimensional objective vector. This issue is illustrated in figure 1.1.

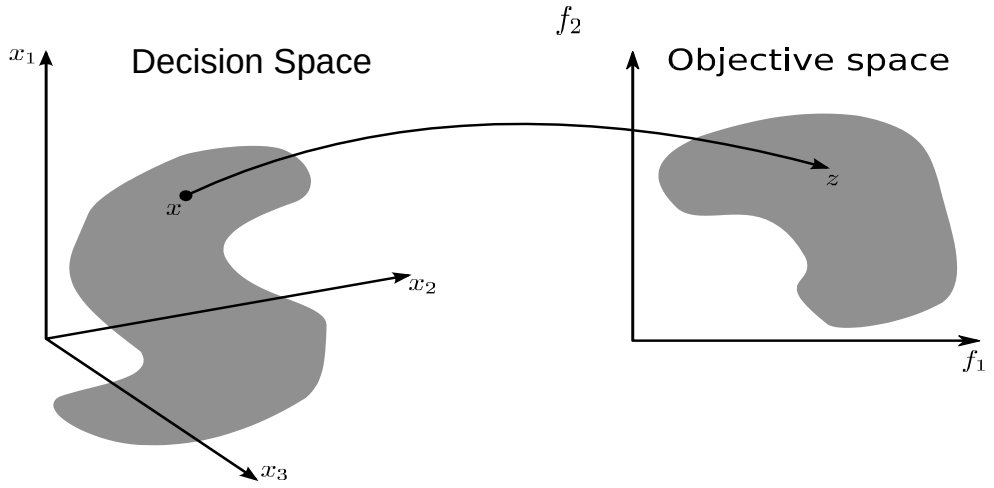

Figure 1.1: Mapping between decision and objective spaces in multiobjective optimization.

In multi-objective optimization problems, with objectives often conflicting with each other, no single solution can be found to optimize all the objectives simultaneously. This situation is different from the case in single objective optimization problems. Thus, for multi-objective optimization problems, the key issue is to find a set of well distributed solutions each of which is a point of compromise with respect to all objectives. The main aim is to discover a set of solutions in which each solution is better than the others in at least one objective value. This property is named as Pareto optimality for MOP_s.

The theory of Pareto optimality was presented by Vilfredo Pareto [3] who developed one of the most important results in the area of multiobjective optimization. The most important feature of Pareto optimality is that in a multi-objective optimization problem a Pareto optimal solution cannot be improved without worsening at least for one of its objectives [4, 38].

Problems with multiple objectives do not have a unique optimum solution, but a group of Pareto optimum solutions. A group of Pareto optimum solutions can be described by Pareto front. The general notion of domination used to compare two Pareto optimal solutions u and v is as followers: if u is not worse than **v** in all objectives, and is entirely better for one objective, then **u** is said to dominate **v**, written **u< v.** Thus **u< g iff:**

$$\begin{aligned} &\mathbf{u_i \leq v_i \ \forall i = 1, \ldots, t \ \ and} \\ &\mathbf{u_j < v_j \ \ for \ \ at \ \ least \ \ one \ \ j.} \end{aligned} \tag{1.2}$$

The set of all non-dominated solutions from the Pareto front. The set of solutions in the Pareto front represents the possible optimal trade-offs among competing objectives.

Within the context of multiobjective optimization, an algorithm is said to have converged when the set of solutions it produces is equal to the best Pareto front. As it is often the case that the Pareto front is continuous and algorithms produce a finite set, it is usually assumed that an algorithm have converged to the true Pareto front when its discovered boundary lies a very small distance apart from the best know one.

A nature-inspired metaheuristic optimization algorithm is a computational procedure that imitates several physical or biological processes such as the evolution of the species, the human immune system, and the social behavior of some animal groups. These processes are themselves optimization processes and so they naturally suggest computational algorithms applicable to mathematical optimization problems. e.g.,.[26,

34, 35, 36, 37].

The Quadratic Assignment Problem (QAP) is considered as one of the challenging and more interesting combinatorial problem in presence, Due to its NP-hard nature, the individual problem instances are usually complex and difficult to solve exactly. Multiobjective quadratic assignment problem (mQAP) is a well-known and hard combinational optimization problem which is used in modelling of several assignment and scheduling problems. Benchmark mQAP instances are already solved near optimally using competitive metaheuristics and dedicated local search algorithms but there is no absolute winner of these competitions in the sense that while a particular algorithm exhibits quite successful for a kind of mQAP instance, it exhibits poor performance on the others.

The remainder of this thesis is organized as follows: In Chapter 1 an introduction to the topic and sets out the objectives of the research work carried out. In Chapter 2 reviews the literature to give a formal introduction to the basic concepts to be dealt with in this thesis about Quadratic Assignment Problem, Multiobjective Quadratic Assignment Problem. In Chapter 3 we introduced the Artificial Bee Colony (ABC) techniques. It discusses the basics of swarm intelligence, the background of the ABC algorithm as well as the main control parameters of the algorithm .In Chapter 4 outlines the Artificial Bee Colony over Multiobjective Quadratic Assignment Problem, testing the algorithm on most instances in mQAP. And attempting tweaks on the algorithm to improve the runtime. In Chapter 5 draws the conclusions and summarises the work developed in this thesis.

Chapter 2

LITERATURE REVIEW

2.1 Quadratic Assignment Problem (QAP)

The Quadratic Assignment Problem (QAP) can be mathematically formulated as follows: let P (N) be set of all permutation of integers from 1 to N,

$$QAP = \min(\pi) = \sum_{i=1}^{N} \sum_{j=1}^{N} f_{i,j}\, d_{\pi(i),\pi(j)} \qquad (2.1)$$

where N is the number of facilities/locations, f_{ij} symbolizes the flow between facilities i and j, d_{ij} symbolizes the distance between locations i, j and π (i) represents the location of facility i in permutation $\pi \in$ P [5].

2.2 Multiobjective Quadratic Assignment Problem (mQAP)

When there are two or more flow matrices between facilities the problem will be a multi-objective quadratic assignment problem (mQAP). For example, a hospital can requests the simultaneous minimization of the costs associated with the flow of physicians, patients, nurses, guests, and tools between various facilities, remembering that the distance between the location where the facilities are set must also be minimized too.

The mQAP was formulated in [5] to broaden the QAP to encompass multiple objectives. QAP is a good model to solve planning for a single product in one place from a plant, whereas mQAP can take into consideration all the products and the whole plant design.

Mathematically, the last functions or objective functions mQAP can be represented as:

$$C^k(\pi) = \sum_{i=1}^{N} \sum_{j=1}^{N} f^k_{i,j} d_{\pi(i),\pi(j)} \quad , \quad k=1,\ldots..,m \qquad (2.2)$$

Where:

m is the number of objectives or flows matrices,

N is the number of facilities/locations,

$\pi(i)$ is the facility in the location i in permutation π,

$f^k_{i,j}$ is symbolize the flow from facility assigned to location i to facility assigned to location j within the k_{th} flow matrix.

2.3 Solved Method For mQAP

Since the mQAP inception in 2002 by Knowles and Corne[5], mQAP has studied by a few numbers of researchers, a little was done to study the characteristics of the landscape and its impact on the performance of the algorithm. As yet, the research for mQAP is still in its early stages. One of the proposals implemented mQAP to a group of heterogeneous UAVs, mQAP were used to improve communications for the formations of drones heterogeneous by using MOMGA-II [6]. This research spans the search using a parallel copy of MOMGA-II and compared the efficiency and speed up the results to the results of a parallel series.

Hui Li and Dario Landa- Silva they applied an elitist GRASP metaheuristic algorithm called mGRASP/MH to tackle the mQAP (multi-objective quadratic assignment problem). In the proposed approach, elitist-based greedy randomized construction, cooperation between solutions, and weight-vector adaptations are used to accelerate convergence and diversify the search [7]. Also another models of the flow of execution in the constellation of communications satellites [8]. In [5], they applied to

the problem of facilities planning, where there is a flow of over a single type of agent.

An execution of hybrid SPEA2 and ACO algorithms is tested with all of a next-ascent hill climber and number of various lengths of tabu search (TS) on the mQAP [9].

An evolutionary multiobjective optimization with a segment-based external memory support for the mQAP was proposed by Adnan Acan and Ahmet Ünveren[10]. In their comprehensive comparison, the performance is compared with a well-known multiobjective genetic algorithm MOGA.

In [11], they presented a fuzzy particle swarm algorithm over mQAP. In the fuzzy scheme, the representations of the position and velocity of the particles in the conventional PSO is extended from the real vectors to fuzzy matrices.

An eempirical comparison of memetic algorithm strategies on the multiobjective quadratic assignment problem [12]. They took the view that the two components were competing for the limited usage of search time, and compared several different strategies on a wide range of instances of the multiobjective quadratic assignment problem.

In[13]. A hybrid genetic/immune strategy to tackle the multiobjective quadratic assignment problem. They compared GISMOO algorithm with two well-known multiobjective algorithms (NSGAII, PMSMO) on benchmark mQAP problems. The largest difference among GISMOO and other two algorithms lies in the environmental selection and the way in which the immune metaphor is used in a Pareto GA to identify and emphasize the solutions located in less crowded regions found during the

iteration process of the algorithm.

In[14]. Deon Garrett and Dipankar Dasgupta they have shown a correlation between the advantage obtained through a hybrid MOEA versus a simpler iterated local search algorithm and the distribution offspring generated via recombination. As the correlation objectives creases, thus allowing the embedded local search procedure to exploit these good initial locations to find improved pareto optima.

In[15],they propose a multi-objective quadratic assignment instance generator that aggregates several small multi-objective QAP instances into a larger mQAP instance. Both the component mQAP instances and the cost of the elements outside these components have, by design, the identity permutation as the optimal solution. They give mild conditions under which the resulting composite mQAP instances have identity permutation as the optimum solution. They show that composite bQAP instances are more difficult than the uniform random bQAPs, and in addition, they have a known optimum solution.

Chapter 3

THE ARTIFICIAL BEE COLONY (ABC) ALGORITHM

3.1 The basics of Swarm Intelligence

In any biological or natural swarm system, there are two principal notions, namely division of labor and self-regulation. These are needed as adequate properties to get swarm intelligent behavior like distributed problem solving, self-organization and adaptation to a certain environment [16].

The definition of self-regulation could be as a set of dynamical techniques, which leads to structures at the global level for the system through the interactions between the components of the low-level. Those techniques create the basic bases of the interactions among the components of the system. The bases guarantee that the interactions are performed based on purely domestic information without any relationship to the universal pattern. Bonabeauet al. [17] marked by four fundamental characteristics on self-regulation: "Positive feedback, negative feedback, fluctuations and Multiple interactions".

a- The first component of self-regulation is the positive feedback into system; inflate a best solution that employees have discovered. Recruiting other agents to exploit these work a rounds are good.

b- The second component of self-regulation is the negative feedback; it operates on reducing the implications of positive feedback and help to find a parallel mechanism. An instance in negative feedback is the

numbers of propelled foragers.

c- The Third component of self-regulation is fluctuations or in other words randomness. This helps to add an uncertainty factor into the system and leads discovering new solutions to problems under consideration.

d- The last component of self-regulation is the multiple interactions: should there be a minimum number of individuals who are able to interact together to transform activities at the local level into one coherent organism.

Inside the swarm, there are various tasks, which are executed simultaneously by specialist individuals. This type of phenomenon is called division of labor. Simultaneous work on a task by individual's that are specialist in cooperating is thought to be more effective than the sequential task performance by non-specialist individuals.

3.2 Bees in Nature

Artificial bee colony algorithms are inspired from behaviors of natural bees and have emerged as a promising tool and a strong optimization instrument. Many groups of researchers have formulated different versions of this algorithm over a couple of years independently. In this respect, the behavioral paradigm of self-regulation was suggested for the colony of honeybees by Seeley in 1995 [18]. The foraging behavior in a bee colony remained a mystery for a lot of years until Von Frisch constructed a method as an integral part of the bees waggle dance [19]. In some of his experiences, he found out that bee truly have own color visibility and attracted through smell deposited via hive mates.

Honey bees dance, may be the most interesting in their own biology as well one of the coolest behaviors in the animal world. The worker bees that came back to the comb with pollen grains or nectars, use the waggle dance language to inform the other workers on the discovered nutrition. This information indicates the distance and direction with movements in particular.

Nakrani and Tovey[20], proposed for the first time Honey Bee Algorithm (HBA) to know the way to customize computers between different clientsand web-hosting servers. A Virtual Bee Algorithm (VBA) has been developed to solve numerical optimization problems by Yang[21].

Honey-Bee Mating Optimization (HBMO) algorithm who was later applied to reservoir modelling and clustering proposed by Afshar et al. [22]. Artificial Bee Colony (ABC) for numerical function optimization and a comparison was developed

by Karaboga [23]. These bee algorithms are nowadays becoming more and more popular [24].

3.3 ABC algorithm

The ABC algorithm is a meta heuristic algorithm that depends on swarm intelligence rather than evolutionary proceedings.

There are three fundamental elements of the ABC Model:

a- Food Sources: For determining the source of food, forager bees assesses many of the relevant properties with a food source such as proximity to the cell, the richness of energy, have a taste of nectar.

b- Unemployed foragers: Containing of two kinds, these are scouts and onlookers.

Their liability is to explore and exploit the food source. Two options are available for unemployed foragers:

- Becoming Scout, randomly find new dietary sources around the nest,
- Becoming onlooker, determining the amount of nectar food source after viewing waggle dances of bee employees, and in accordance to non-profit, determine the source of food

c- Employed foragers: the number of employed bees is equivalent to the number of food sources in all parts of the hive. The employed bees whose food sources were exhausted by bees become scouts [16].

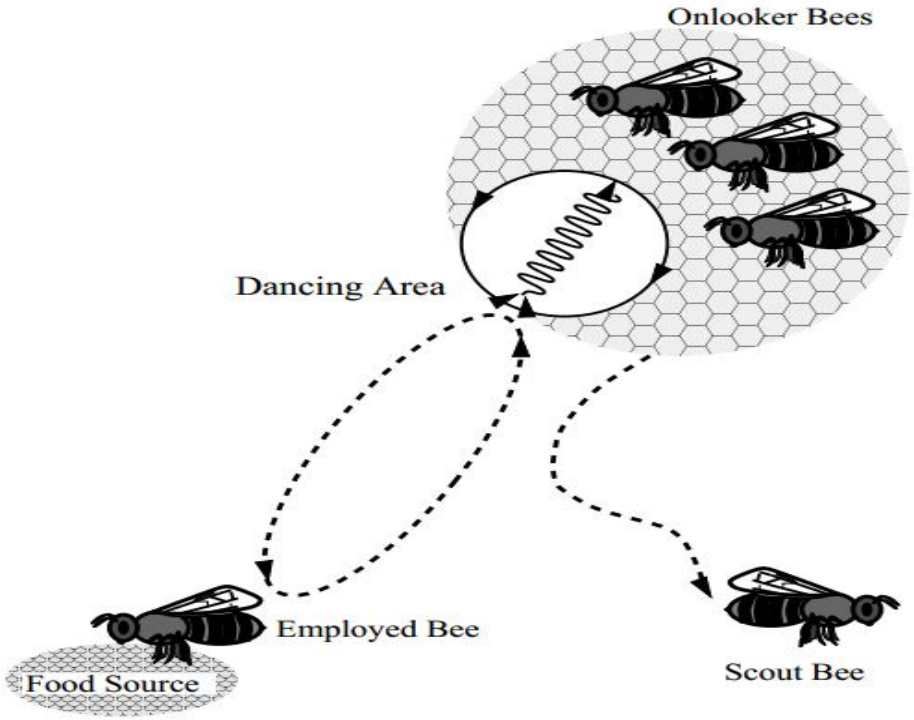

Figure 3.1: The three component of an ABC algorithm: employed, onlooker and scout bees.

Every cycle of the searching, consisting of three steps:

- Moving the employees and onlookers bees on the food sources.
- Calculate the amounts of nectar and determine scout bees and forwarded to potential food sources.
- A food sources positions represent a potential solution to the problem to be optimized. The amount of nectar food source complies with the quality of the solution posed by this food source [25].

Onlookers are placed on food sources using the selection process on the basis of probability [25].

Each bee colony has scouts that are explorers of the colony. Explorers do not have any direction whereas looking for food. They are primarily means to find any type of

food sources. As a result of such behavior, scouts are characterized by low average in quality food source and also lower search costs. Sometimes, Scouts can accidentally discovery rich food, completely unknown food sources. In status of artificial bees, the artificial scouts could be rapid discovery of a possible solution. In this work, only one of the employed bees were selected and categorized as the scout bee. The selection is controlled by a control parameter called "limit". If a solutions representing a food sources is not improved by a beforehand number of trials, then that food source is abandoned through its employed bees and the employed bees are converted to a scout. Number of trials for the release of the source of food is equal to the valuable of the "limit" which is significant control parameter of ABC [26]. Flowchart description of ABC algorithm is given in Figure 3.2.

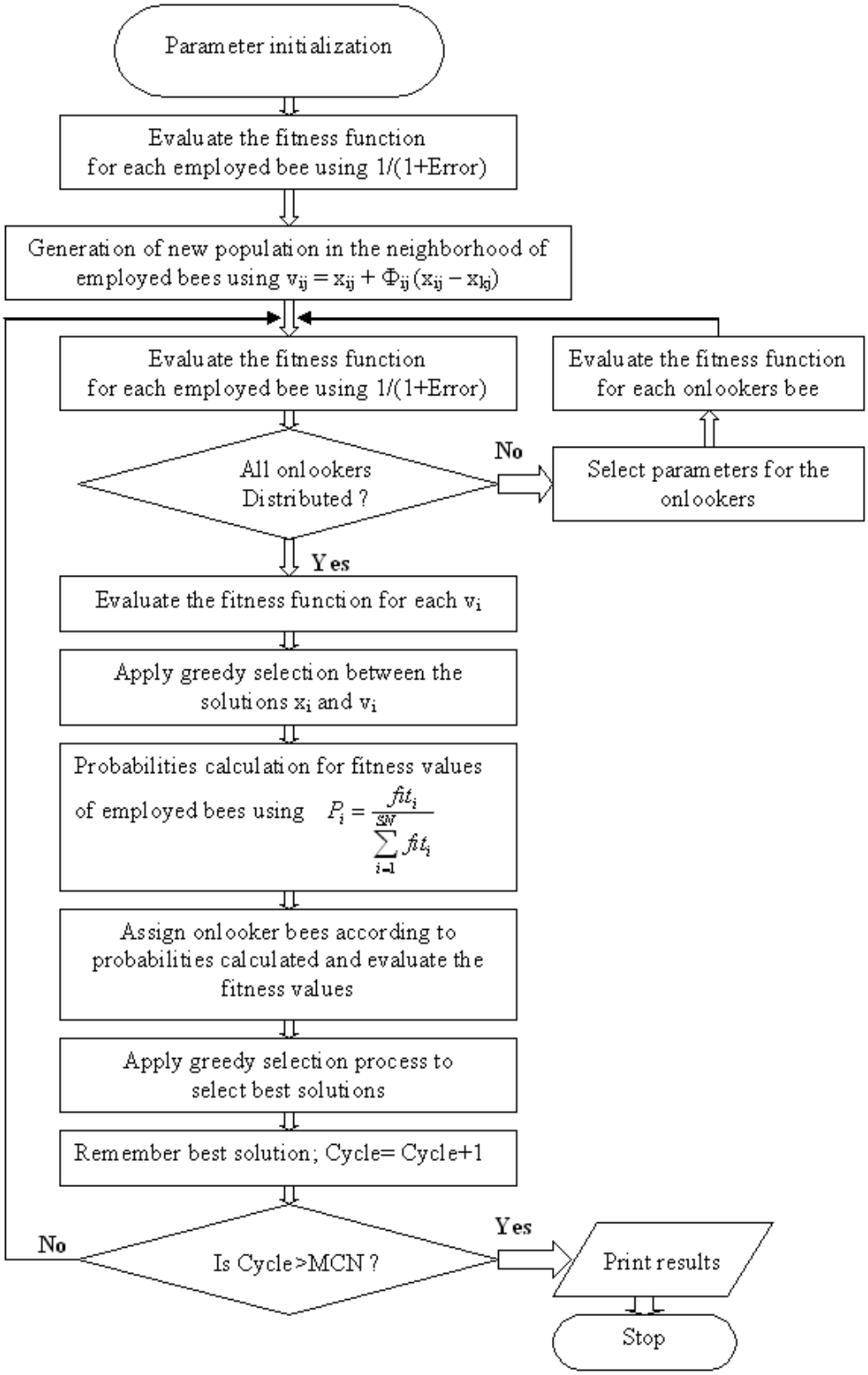

Figure 3.2: Flow chart of ABC algorithm

3.4 Control Parameters of ABC algorithm

- **Number of Bees in aColony (NP)**

Number of bees in Colony identifies the number of solutions being investigated at one time. Number of bees must be linked to the intricacy of the problem.

- **The Improvement Limit for a Solution (I_L)**

That is a hypersensitive parameter, affecting how “deep” a bee seeks to look close to reaching a specific solution and used when being stuck in domestic minimums. If the value of this parameter is high, it indicates that the number of tries a bee does in the vicinity of a given solution is high, which also means that the bee will be stuck for a long time before considering a certain solution as abandoned and trying a different search region.

- **Maximum Number of Iterations (I_{max})**

This parameter does not impact the optimization process immediately, but is utilized to set an higher limit for the period needed for optimization. The value should be sufficient to give the ABC optimization adequate period to converge.

- **Number of Independent Runs (r)**

The number of autonomous runs ought to be adequate in order to get representative results and examine the deviation of these results.

3.5 Steps of the Artificial Bee Colony (ABC) Algorithm

ABC algorithm is a repetitive process. On the assumption that the number of food sources, NS, is equivalent to the half of the gross number of bees in the colony(since the number of employed bees be equivalent to those of the onlooker bees), and D is the dimension of every solution vectors;

1) Random population of solution vectors equivalent for a number of food sources ($X_1, X_2,..., X_{NS}$) been initialized, where $Xi=\{x_{i1},x_{i2}, ...,x_{iD}\}$ and every solution vector is created using:

$$x_{ij}= x_{minj}+rand\ [0, 1]. (x_{maxj}- x_{minj})$$

for j =1, 2,…,D and i =1, 2, …, N_S, (3.1)

Where x_{maxj} and x_{minj} representing consecutive the upper and lower bounds to the dimension$j$. Afte rsetting of the population, validate every solution assigned to the corresponding step size. Thereafter, is to assess the fitness of each food source that conforms to its Punishable cost .

2) Every employed bee is looking in the neighborhood of the current food source to identify the new food source v_i by using:

$$vij= xij+\phi ij(xij- xkj) \quad (3.2)$$

Where: $k \in \{1,2,\ldots,N_S\}$ and $j \in \{1,2,\ldots,D\}$ are randomly selected indexes . $\phi_{i\ j}$is a random number between[-1,1], that helps to increase the hybridization process. Parameter values created by Eq.(3.2) which exceed their limit values are set to their limit values and validate each solution and set to the corresponding step size.

3) After generating the new food source, the nectar amount of it would be evaluated and a greedy selection would be performed. That is, if found the quality of the new food source is better than the present situation, the

employed bee leaves its present situation and move to a new food source. Further clarification, if the fitness of the new food sources equals or better than of X_i, the new food sources takes the location of X_i in the population, it will become a new member.

4) First an onlooker bee chooses a food sources via evaluating the data that accepted from all of the employees bee.

   The probability (p_i) of selecting the foods our cells presented by (Karboga, 2005):

$$p_i = 0.9 \frac{f_{min}}{f_i} + 0.1 \qquad (3.3)$$

   Where: the f_i is a fitness value of the source of food X_i. After choosing source of food, the onlooker generate an new source of foods by Eq.(3.2).

5) If a nominee solution, represented by a food source cannot be further improved through a predetermined number of trials, the food source is considered abandoned and the employed bee linked with a food source become a scouts. The scout randomly generating a new food sources v_i by using:

$$v_{ij} = x_{minj} + rand\ [0, 1].\ (x_{maxj} - x_{minj})$$

$$\text{for } {}_j = 1, 2, \ldots, D \qquad (3.4)$$

The abandoned food source is replaced by the randomly generated food source and validated to its corresponding step size. In the ABC algorithm, the predetermined number of trials for a abandoning a food source was discussed, also in this algorithm at most one employed bee at each cycle can become a scout.

Chapter 4

RESULTS

4.1 Test Data for the Multiobjective QAP

In experimental evaluations, artificial bee colony (ABC) algorithm framework is applied to solve a set of twenty-two multi-objective quadratic assignment problem (mQAP) instances created by Knowles and Corne [5] and are available at http://www.cs.man.ac.uk/~jknowles/mQAP/. Table 1 illustrates a detailed description of parameters of the test suite problems.

The name of every instance of mQAP is presented through naming conventions KCnn-mfl-itype

Where:

- nn represents to the number of facilities/locations,
- m is the number of objectives or flows, and
- itype shows the instance type where the uni symbolized to the uniformly random and rl is symbolized to the real-life distribution[5].

Table 1: Attributes of mQAP test suite problems used in experimental evaluations.

Test Name	Instance Category	#of Locations	#of Flows
KC10-2fl-1uni	Uniform	10	2
KC10-2fl-2uni	Uniform	10	2
KC10-2fl-3uni	Uniform	10	2
KC20-2fl-1uni	Uniform	20	2
KC20-2fl-2uni	Uniform	20	2
KC20-2fl-3uni	Uniform	20	2
KC30-3fl-1uni	Uniform	30	3
KC30-3fl-2uni	Uniform	30	3
KC30-3fl-3uni	Uniform	30	3
KC10-2fl-1rl	Real-like	10	2
KC10-2fl-2rl	Real-like	10	2
KC10-2fl-3rl	Real-like	10	2
KC10-2fl-4rl	Real-like	10	2
KC10-2fl-5rl	Real-like	10	2
KC20-2fl-1rl	Real-like	20	2
KC20-2fl-2rl	Real-like	20	2
KC20-2fl-3rl	Real-like	20	2
KC20-2fl-4rl	Real-like	20	2

KC20-2fl-5rl	Real-like	20	2
KC30-3fl-1rl	Real-like	30	3
KC30-3fl-2rl	Real-like	30	3
KC30-3fl-3rl	Real-like	30	3

4.2 Tabu Search of mQAP

In general, metaheuristics for permutation problems need hybridization with a problem specific local search method to intensify search process around potentially promoting solutions.

Tabu search was suggested by Fred Glover [28] as a meta-heuristic optimization method. One of the applications of tabu search to QAP was scheduled to Skorin-Kapov [29].

The most famous local search for QAP is the robust tabu search (RoTS) of Taillard [27]. RoTS is based on the 2-opt improvement strategy that swaps two elements of a permutation and accepts the new offspring if its fitness is better than that of its parent. Accordingly in the case of QAP, 2-opt neighborhood of a permutation is defined by a set of permutations that can be reached through an exchange two facilities. TS procedure starts from an initial feasible solution φ, and selects a best-quality solution between the neighbors of Υ obtained by permuting all points of (r, s) facilities $\in \Upsilon$. Assuming that the offspring generated is π, its elements are defined by,

$$\pi(k) = \varphi(k) \forall\ k \neq r, s$$

$$\pi(r) = \varphi(s)$$

$$\pi(s) = \varphi(r).$$

For symmetrical matrices with a null-diagonal, written the cost $\Delta(\varphi, r, s)$ of the value of a move is given by:

$$\Delta(\varphi, r, s) = \sum_{i=1}^{N} \sum_{j=1}^{N} \left(F_{i.j} D_{\varphi(I)\varphi(j)} - F_{i,j} D_{\pi(I)\pi(j)} \right)$$

$$= 2 . \sum_{k \neq r,s}^{N} \left(F_{s,k} - F_{r,k} \right) \left(D_{\varphi(s)\varphi(k)} - D_{\varphi(r)\varphi(k)} \right).$$

To compute the cost $\Delta(\pi, \mu, v)$ in a constant time that the moved units μ and v are different from r or s, we use the value $\Delta(\varphi, \mu, v)$ computed in the previous step and find:

$$\Delta(\pi, \mu, v) = \Delta(\varphi, \mu, v)$$

$$+ 2 \left(F_{r\mu} - F_{rv} + F_{s\mu} - F_{sv} \right) \left(D_{\pi(s)\pi(\mu)} - D_{\pi(s)\pi(v)} + D_{\pi(r)\pi(\mu)} - D_{\pi(r)\pi(v)} \right).$$

The value of moves which don't involve the nodes in the previous swap can be calculated in the time O (1). There are O (N^2) of these moves.

The value of moves which involve the nodes in the previous swap must be calculated from scratch. There are O (N) of these moves and the complexity of calculating each is O (N).

In the case of Artificial Bee Colony algorithm, 2-opt is applied as follows. A swap continues if it leads to a decrease cost or such other. One time the fixed swap is made, the 2-opt on the present solution π is stopped. If all potential swaps are checked and it brings any improvement, the 2-opt will cease and the ABC algorithm will resume their next step.

An alternative is the implementation of local search method for QAP so that structure is checked whole neighborhood and a swap that brings the maximum decrement in cost is determined. This swap would be made persistent if it is found to bring the maximum decrement in cost .The overhead for this option is high and hence this puts the option at a disadvantage.

The 2-opt best-improvement local search is integrated with a Tabu List .The main objective of TL is to make sure that the local search moves which have been performed before are not reiterated for a period of time.

The working mechanism of TL is as follows. At every particular time, a maximum of 20 local search moves can be stored in TL. These local search moves will be stored in TL for a certain period of time. Once a move is expired, it is discarded to make room for a new move. The addition of moves is done such that no duplication is permitted in TL. A move in TL has the structure [a, b, c, d, e]. a and b denote the facilities to be swapped and c and d denote the locus of that two facilities designates the staying duration of that move in TL.

When a local search move is to be performed, the local search algorithm will first traverse through TL to check if this move has been performed before and hence identical move is avoided. A strict checking mechanism is employed whereby a move candidate is compared symmetrically against all the moves in TL.

Local search begins from some of the first assignment and trying over and over again to improve the present assignment due to local changes. In case of finding the best assignment in the neighborhood of the present assignment, thus replaces the present assignment and the search continues.

Aspiration function calculates how good a new solution should be to accept it even though it is on a tabu list. The length of the tabu list, construction of the neighborhood, and the aspiration function are implementation-specific parameters of this method. Tabu search is not a probabilistic method. However, the search preformed from one initial solution x_0 may be not thoroughly enough. Various methods are applied to diversify the tabu search. Tahe simplest ay to do it is by restarting the search from random initial solutions.

4.3 Experimental Results

The results of 2-objective QAP's instances with 10 locations &facilities are compared with the true Pareto-optimal set, which is available from Knowles source [5].

The results of 2&3-objective QAP's instances with 20 and 30 locations & facilities are compared with the ABC by running the algorithm ten runs (ABC 10 data runs).

For all instances the Control Parameters of ABC algorithm as follows:

NP=600 the number of colony size (employed bees + onlooker bees).

FoodNumber=NP/2 the number of food sources equals the half of the colony size.

maxCycle=5000 the number of cycles for foraging {a stopping criteria}.

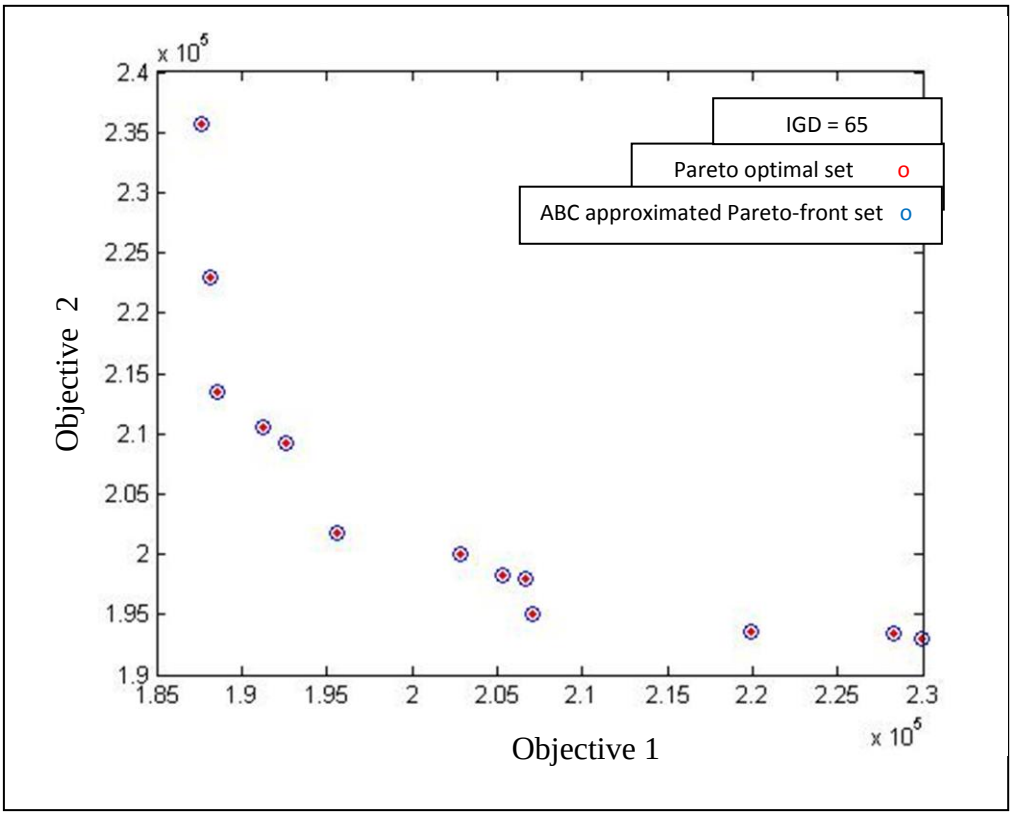

Figure 4.1: The ABC result for 2-objective QAP (KC10-2fl-1uni) test problem with uniformly random distribution.

In Figure 4.1 shows how the ABC approximated Pareto-front set performed to (KC10-2fl-1uni) compared with the true Pareto-optimal set, which is available from Knowles source. The non-dominated solutions found by ABC approximated Pareto-front set almost the same set of non-dominated to the ones on the true Pareto-optimal set. In this respect, the number of points on the ABC algorithm is 13 solutions while the number of non-dominated solutions found by Knowles is 13.

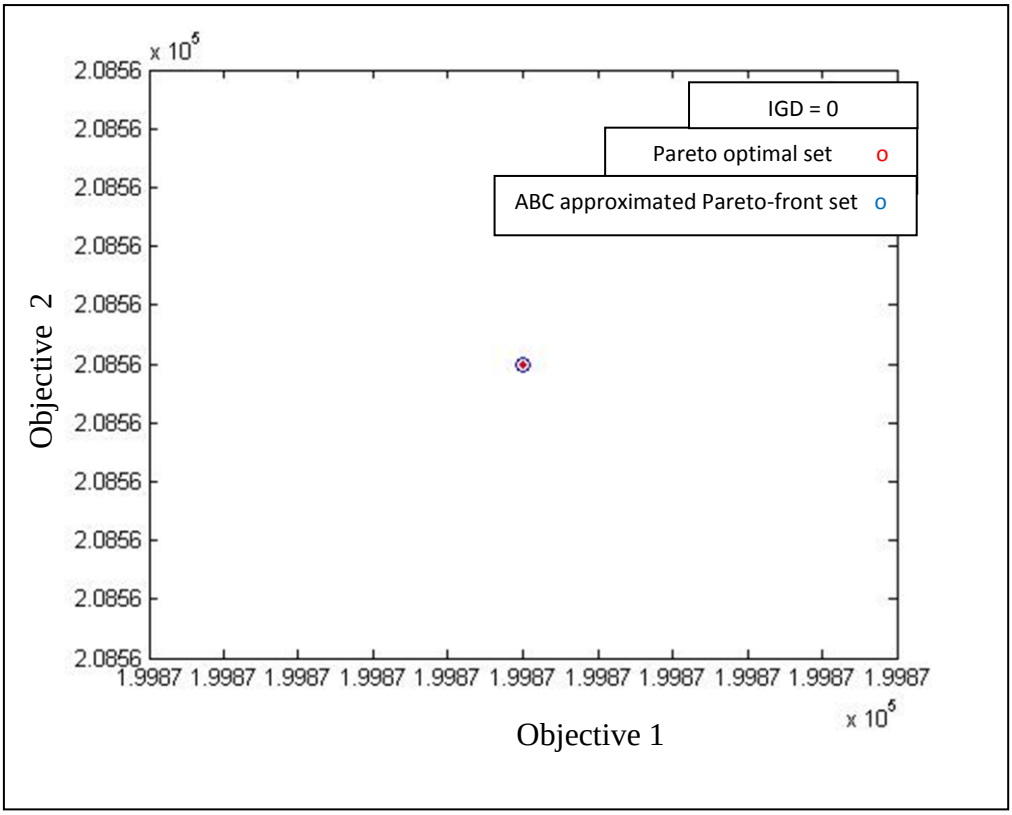

Figure 4.2: The ABC result for 2-objective QAP (KC10-2fl-2uni) test problem with uniformly random distribution.

In Figure 4.2 shows how the ABC approximated Pareto-front set performed to (KC10-2fl-2uni) compared with the true Pareto-optimal set, which is available from Knowles source. The non-dominated solutions found by ABC approximated Pareto-front set almost the same set of non-dominated to the ones on the true Pareto-optimal set. In this respect, the number of points on the ABC algorithm is 1 solutions while the number of non-dominated solutions found by Knowles is 1.

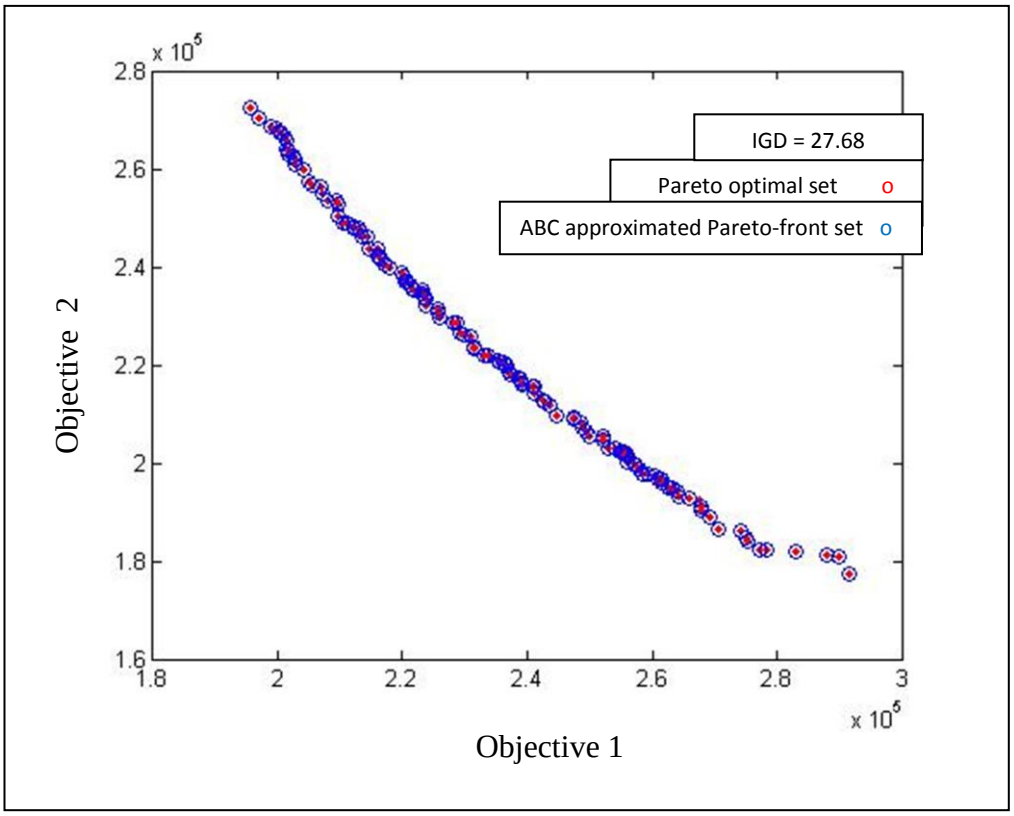

Figure 4.3: The ABC result for 2-objective QAP (KC10-2fl-3uni) test problem with uniformly random distribution.

In Figure 4.3 shows how the ABC approximated Pareto-front set performed to (KC10-2fl-3uni) compared with the true Pareto-optimal set , which is available from Knowles source . The non-dominated solutions found by ABC approximated Pareto-front set almost the same set of non-dominated to the ones on the true Pareto-optimal set. Considering the performances in terms of the number of Pareto points computed, the score of ABC approximated Pareto-front is 130 while that of Knowles is 130.

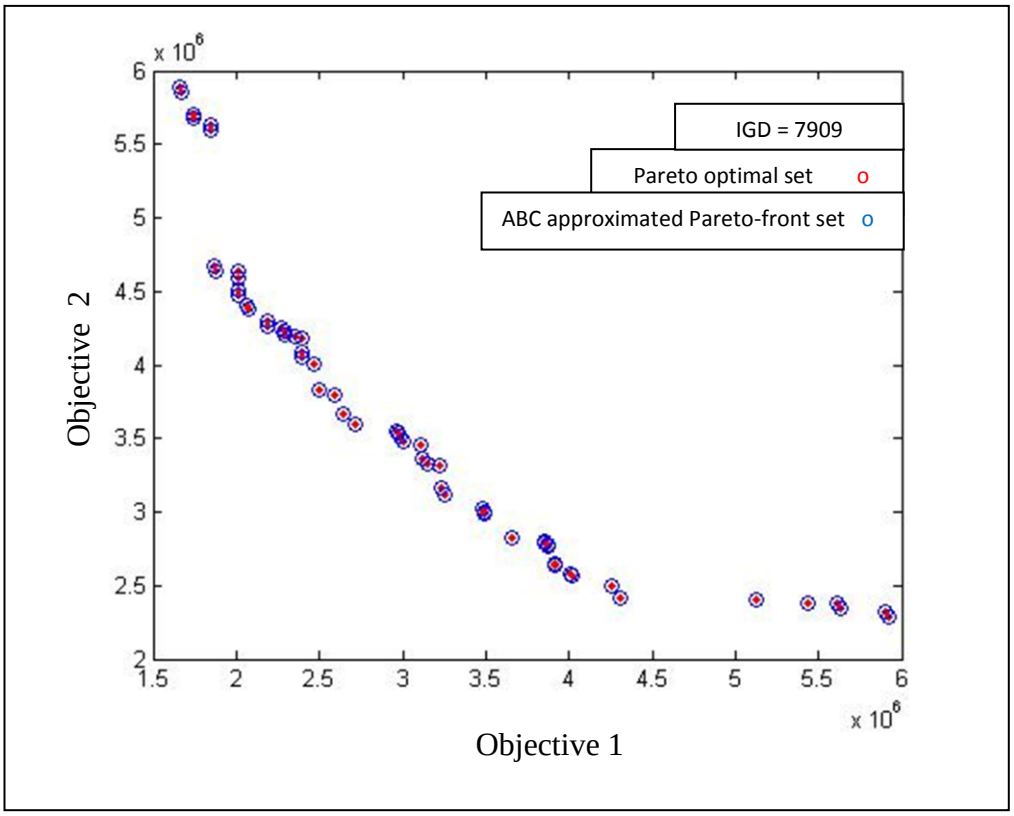

Figure 4.4: The ABC result for 2-objective QAP (KC10-2fl-1rl) test problem with real-life distribution.

In Figure 4.4 shows how the ABC approximated Pareto-front set performed to (KC10-2fl-1rl) compared with the true Pareto-optimal set , which is available from Knowles source . The non-dominated solutions found by ABC approximated Pareto-front set almost the same set of non-dominated to the ones on the true Pareto-optimal set. Considering the performances in terms of the number of Pareto points computed, the score of ABC approximated Pareto-front is 58 while that of Knowles is 58.

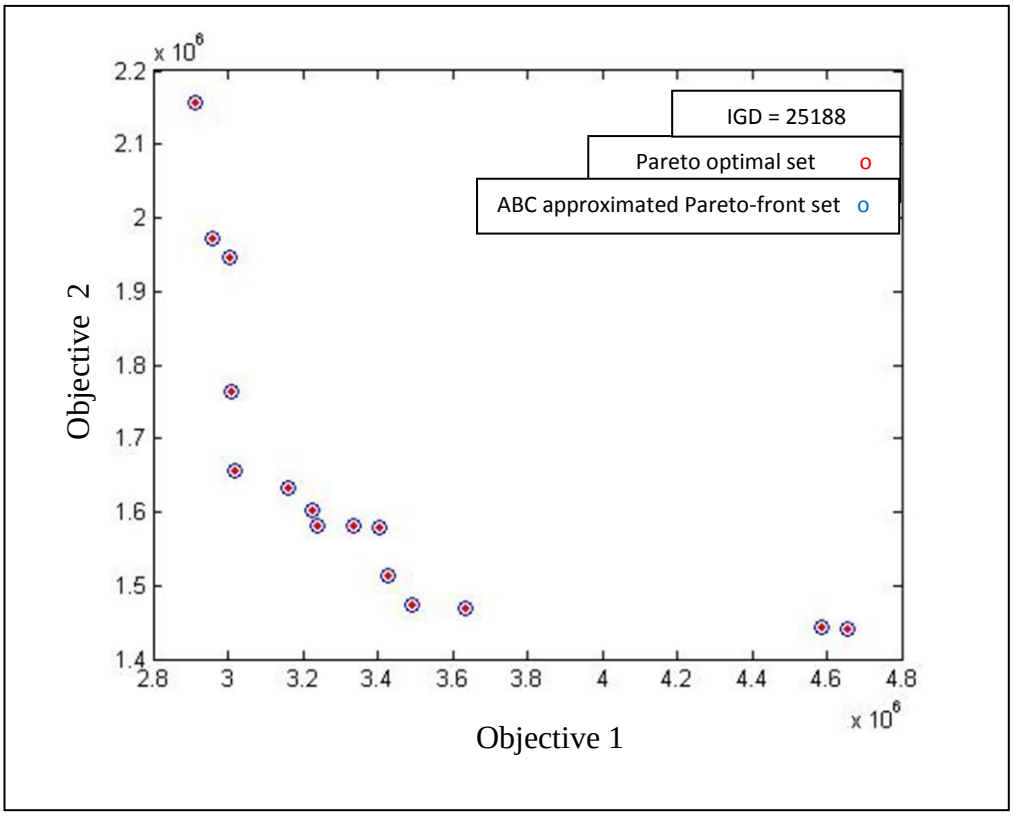

Figure 4.5: The ABC result for 2-objective QAP (KC10-2fl-2rl) test problem with real-life distribution.

In Figure 4.5 shows how the ABC approximated Pareto-front set performed to (KC10-2fl-2rl) compared with the true Pareto-optimal set, which is available from Knowles source . The non-dominated solutions found by ABC approximated Pareto-front set almost the same set of non-dominated to the ones on the true Pareto-optimal set the number of Pareto points computed by algorithm and Pareto optimal set of Knowles are 15 and 15, respectively.

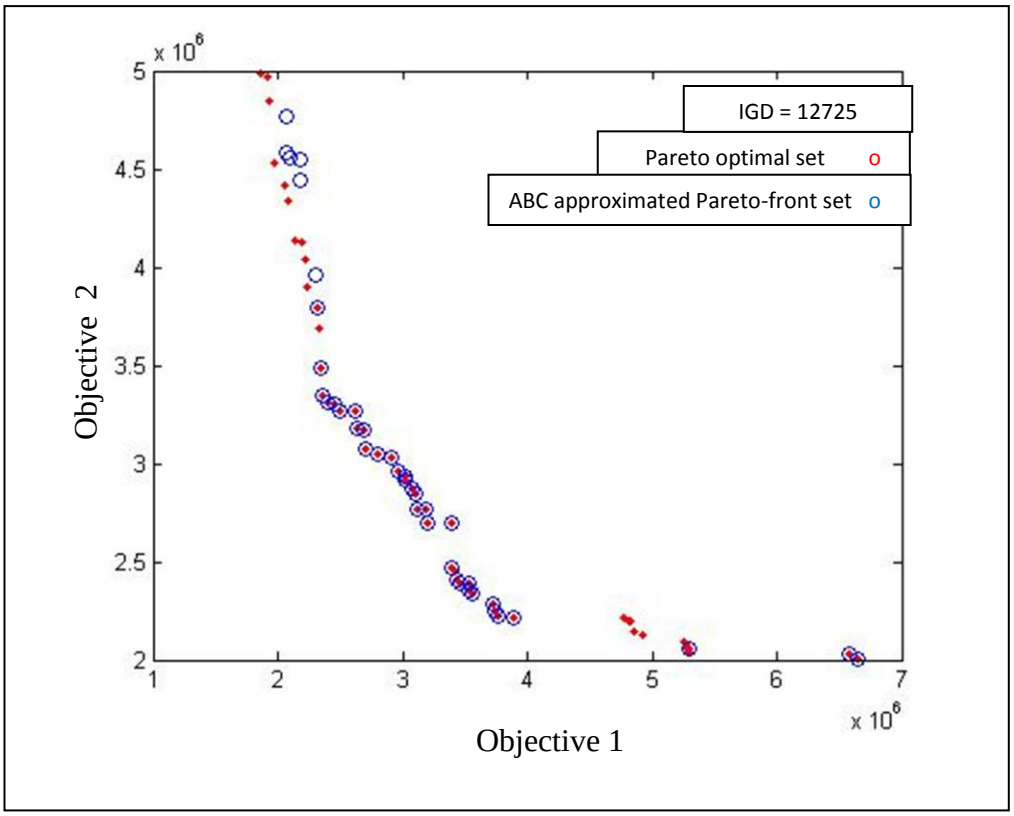

Figure 4.6 : The ABC result for 2-objective QAP (KC10-2fl-3rl) test problem with real-life distribution.

In Figure 4.6 shows how the ABC approximated Pareto-front set performed to (KC10-2fl-3rl) compared with the true Pareto-optimal set, which is available from Knowles source. The non-dominated solutions found by ABC approximated Pareto-front set are very close to the ones on the true Pareto-optimal set. In this respect, the number of points on the ABC algorithm is 40 solutions while the number of non-dominated solutions found by Knowles is 55.

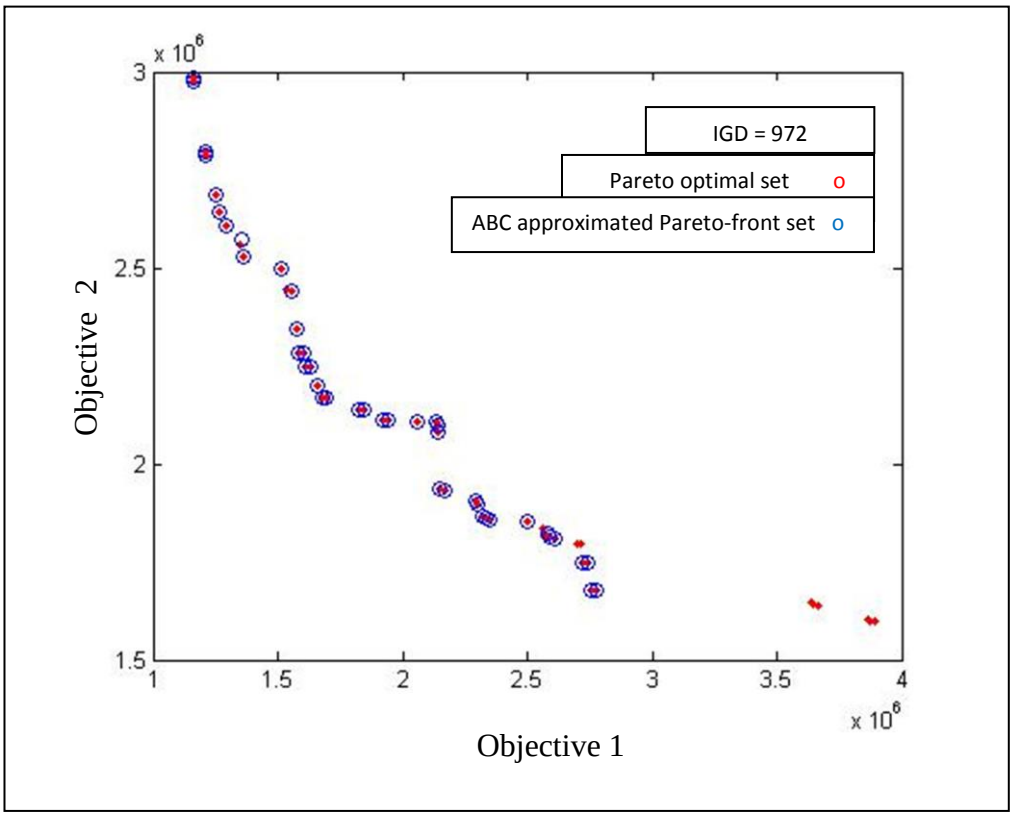

Figure 4.7 : The ABC result for 2-objective QAP (KC10-2fl-4rl) test problem with real-life distribution.

In Figure 4.7 shows how the ABC approximated Pareto-front set performed to (KC10-2fl-4rl) compared with the true Pareto-optimal set, which is available from Knowles source. The non-dominated solutions found by ABC approximated Pareto-front set are nearly to the ones on the true Pareto-optimal set. Considering the performances in terms of the number of Pareto points computed, the score of ABC approximated Pareto-front is 42 while that of Knowles is 53.

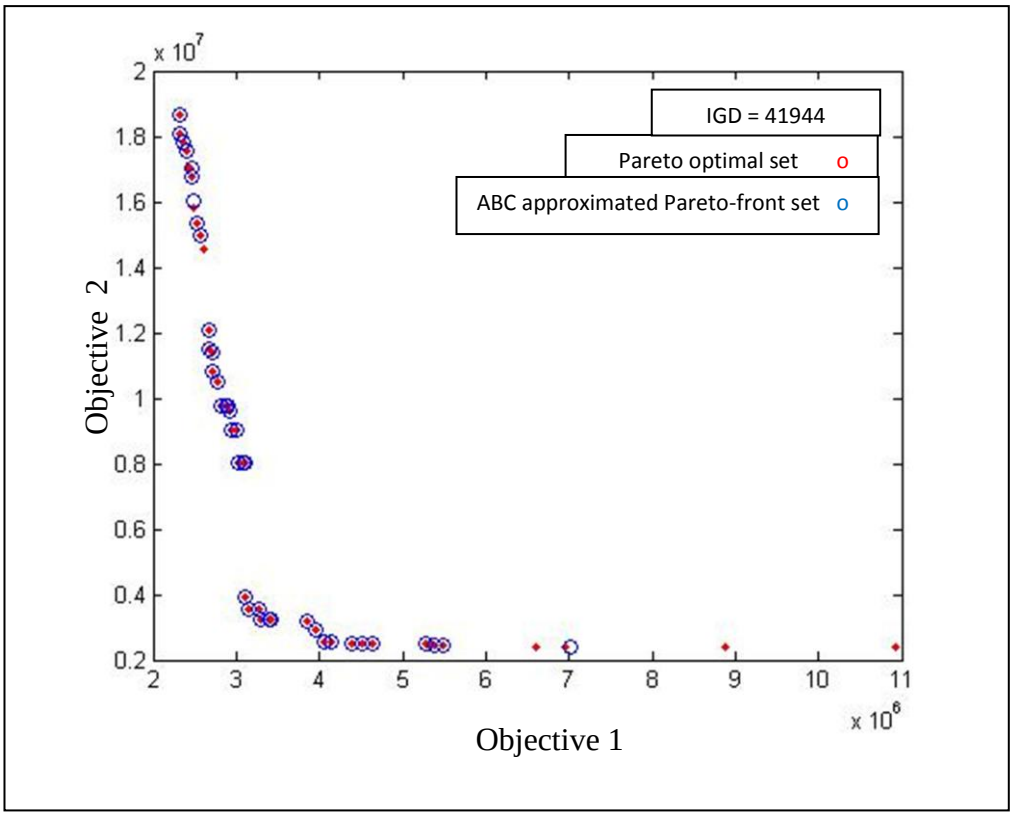

Figure 4.8: The ABC result for 2-objective QAP (KC10-2fl-5rl) test problem with real-life distribution.

In Figure 4.8 shows how the ABC approximated Pareto-front set performed to (KC10-2fl-5rl) compared with the true Pareto-optimal set, which is available from Knowles source. The non-dominated solutions found by ABC approximated Pareto-front set are very close to the ones on the true Pareto-optimal set. Considering the performances in terms of the number of Pareto points computed, the score of ABC approximated Pareto-front is 43 while that of Knowles is 49.

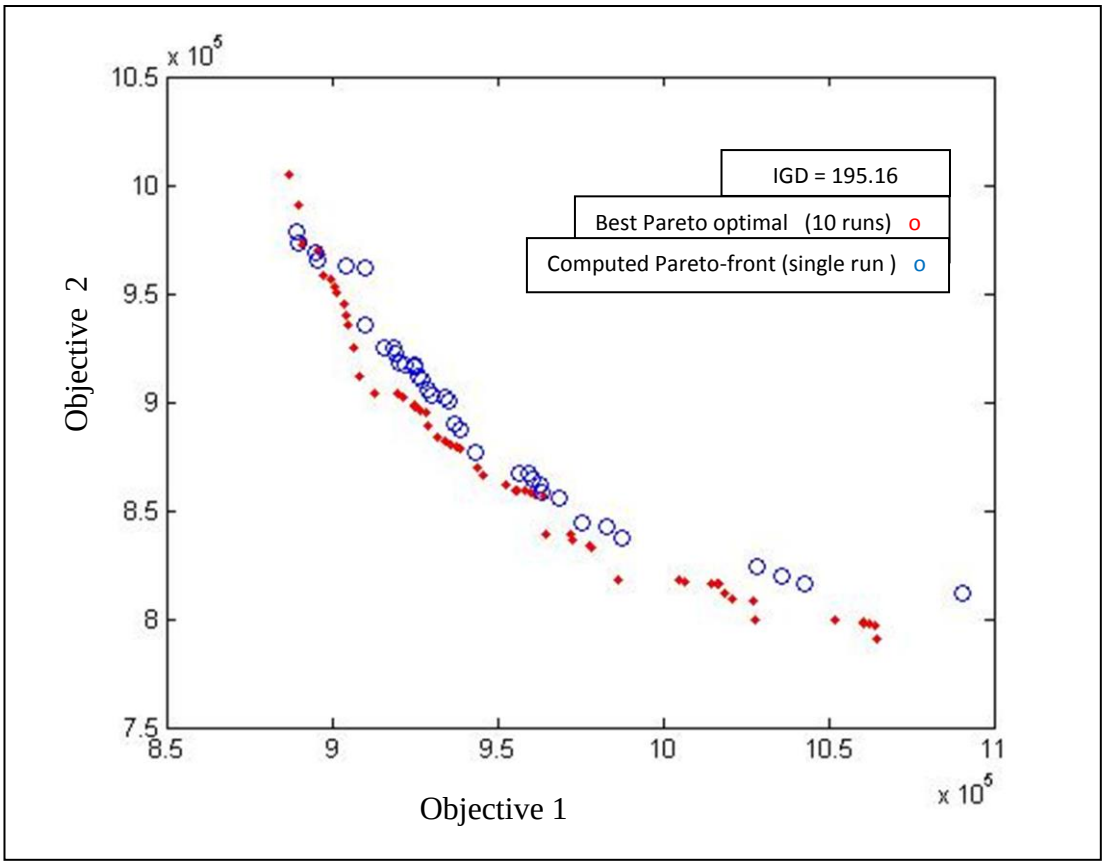

Figure 4.9 : The ABC result for 2-objective QAP (KC20-2fl-1uni) test problem with uniformly random distribution.

In Figure 4.9 shows how the Pareto-front (single run) performed to (KC20-2fl-1uni) compared to the data from computed Pareto optimal set (10 runs). The non-dominated solutions found by Pareto-front (single run) are close to the ones on the computed Pareto optimal (10 runs). In this respect, the number of points on the Pareto-front set (single run) is 36 solutions while the number of non-dominated solutions found by (10 runs) is 60.

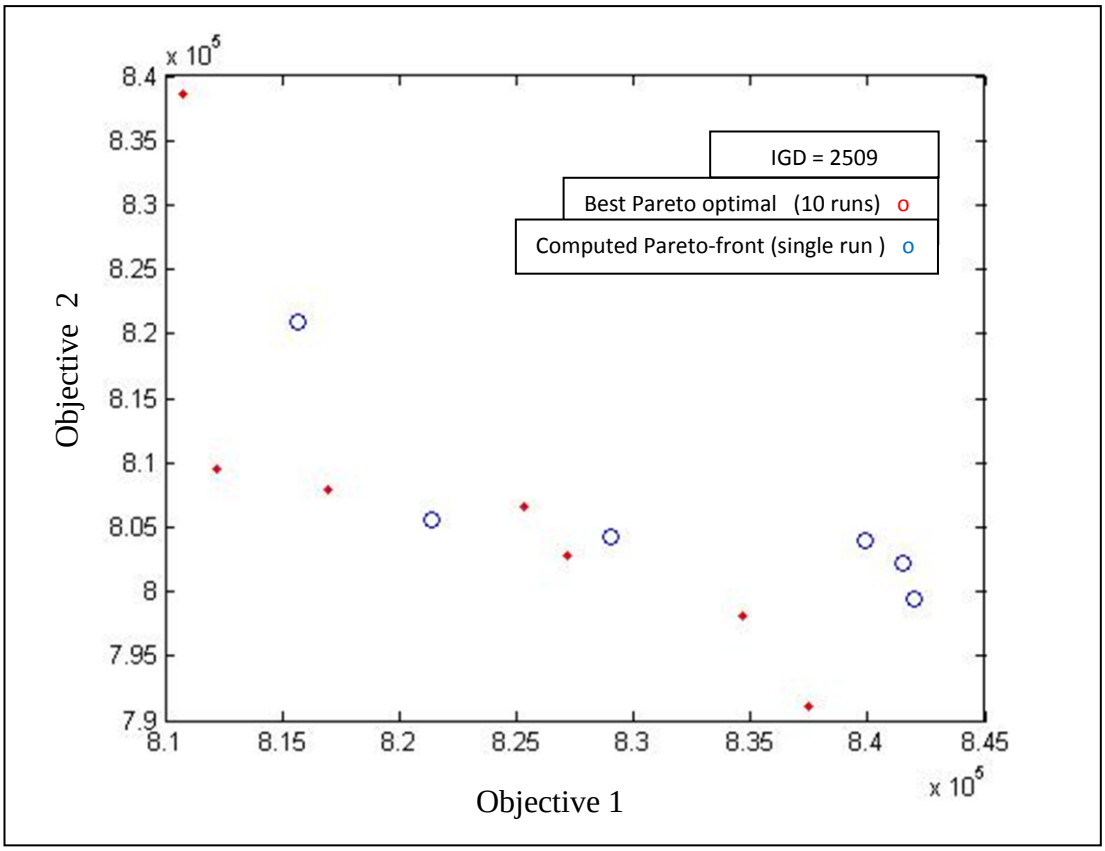

Figure 4.10: The ABC result for 2-objective QAP (KC20-2fl-2uni) test problem with uniformly random distribution.

In Figure 4.10 shows how the Pareto-front (single run) performed to (KC20-2fl-2uni) compared to the data from Pareto optimal (10 runs). The non-dominated solutions found by Pareto-front (single run) are outperforms to the ones on the computed Pareto optimal (10 runs). Considering the performances in terms of the number of Pareto points computed, the score of Pareto-front set (single run) is 6 while that of (10 runs) is 7.

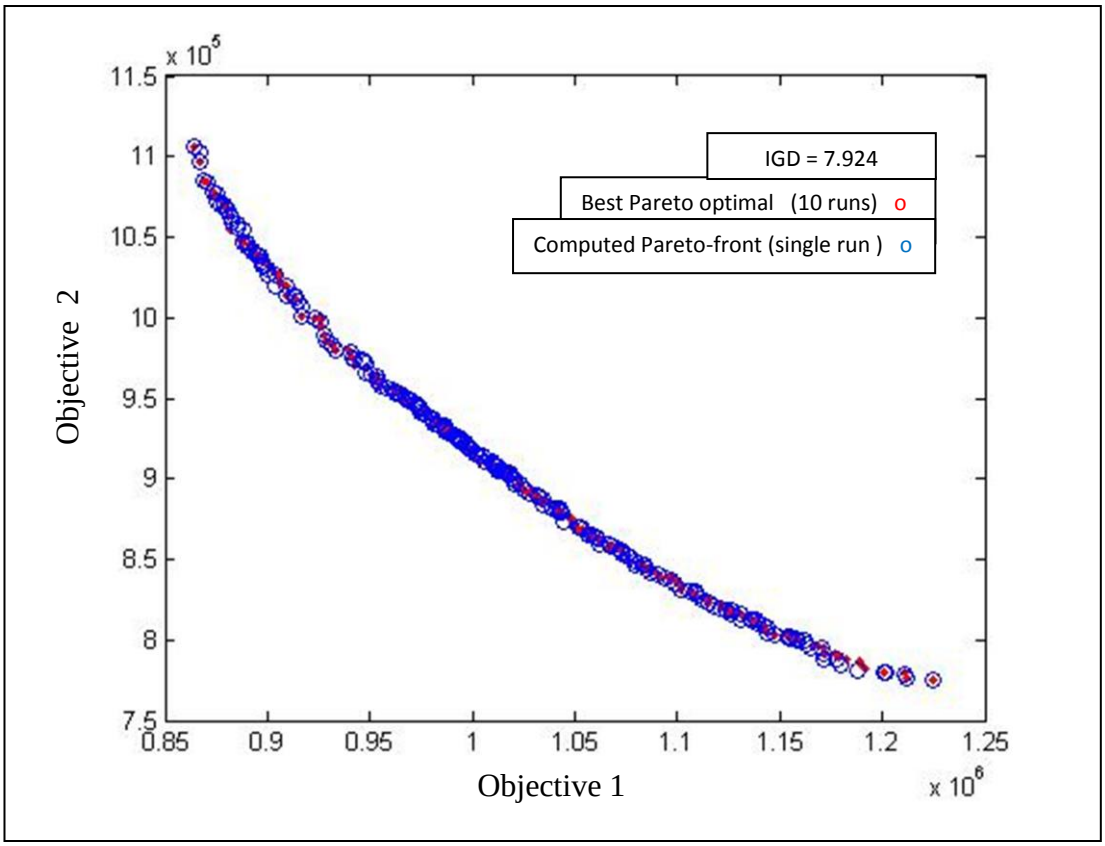

Figure 4.11 : The ABC result for 2-objective QAP (KC20-2fl-3uni) test problem with uniformly random distribution.

In Figure 4.11 shows how the Pareto-front set (single run) performed to (KC20-2fl-3uni) compared to the data from Pareto optimal set (10 runs). The non-dominated solutions found by Pareto-front set (single run) are very close to the ones on the Pareto optimal set (10 runs). The number of Pareto points computed by Pareto-front set (single run) and Pareto optimal set (10 runs) are 216 and 221, respectively.

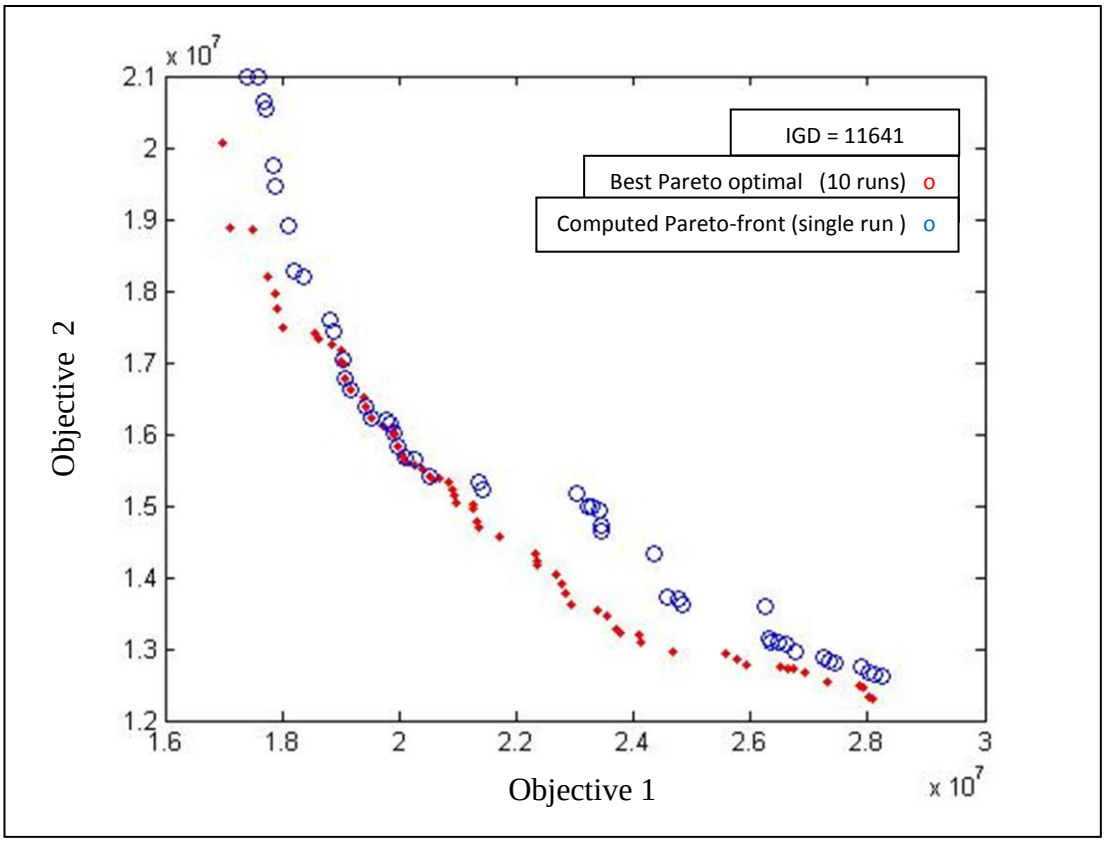

Figure 4.12 : The ABC result for 2-objective QAP (KC20-2fl-1rl) test problem with real-life distribution.

In Figure 4.12 shows how the Pareto-front set (single run) performed to (KC20-2fl-1rl) compared to the data from Pareto optimal set (10 runs). The non-dominated solutions found by Pareto-front set (single run) are nearly to the ones on the Pareto optimal set (10 runs). The number of Pareto solutions calculated by Pareto-front set (single run) and Pareto optimal set (10 runs) are 48 and 68, respectively.

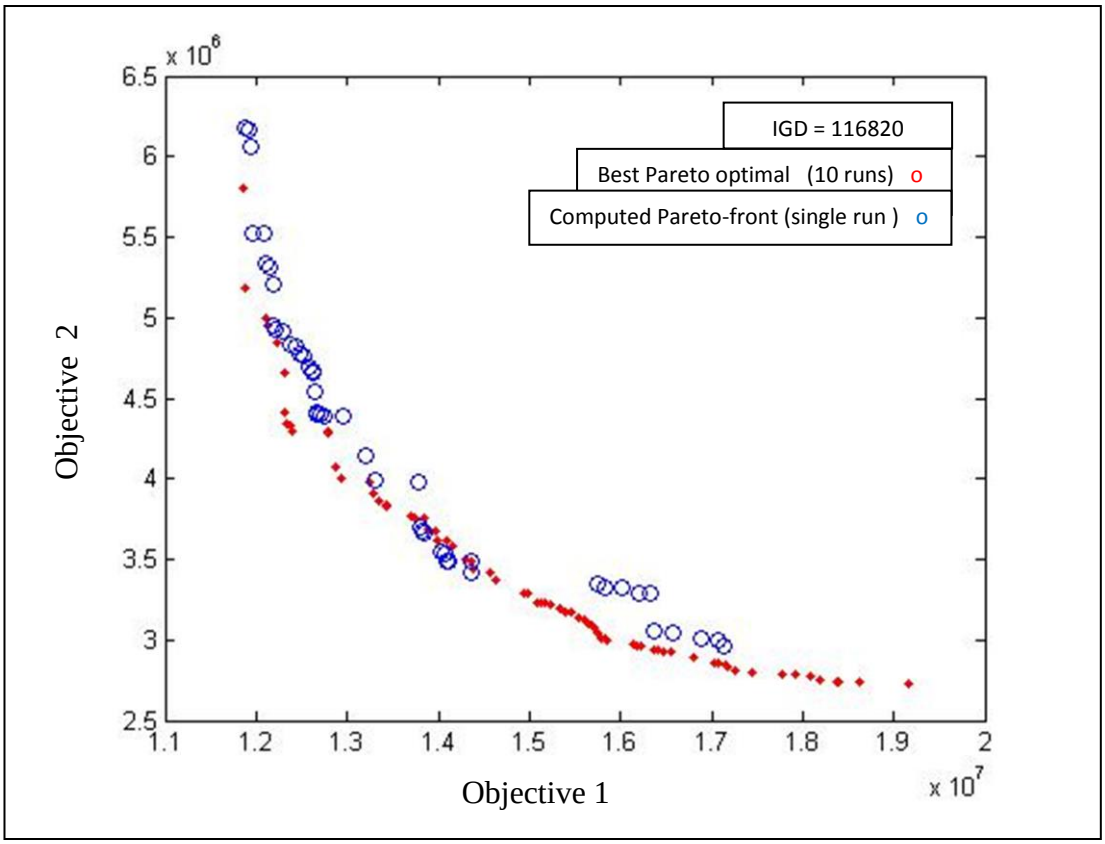

Figure 4.13: The ABC result for 2-objective QAP (KC20-2fl-2rl) test problem with real-life distribution.

In Figure 4.13 shows how the Pareto-front set (single run) performed to (KC20-2fl-2rl) compared to the data from Pareto optimal set (10 runs). The non-dominated sets found by Pareto-front set (single run) are very close to the ones on the Pareto optimal set (10 runs). Considering the performances in terms of the number of Pareto points computed, the score of Pareto-front set (single run) is 48 while that of (10 runs) is 74.

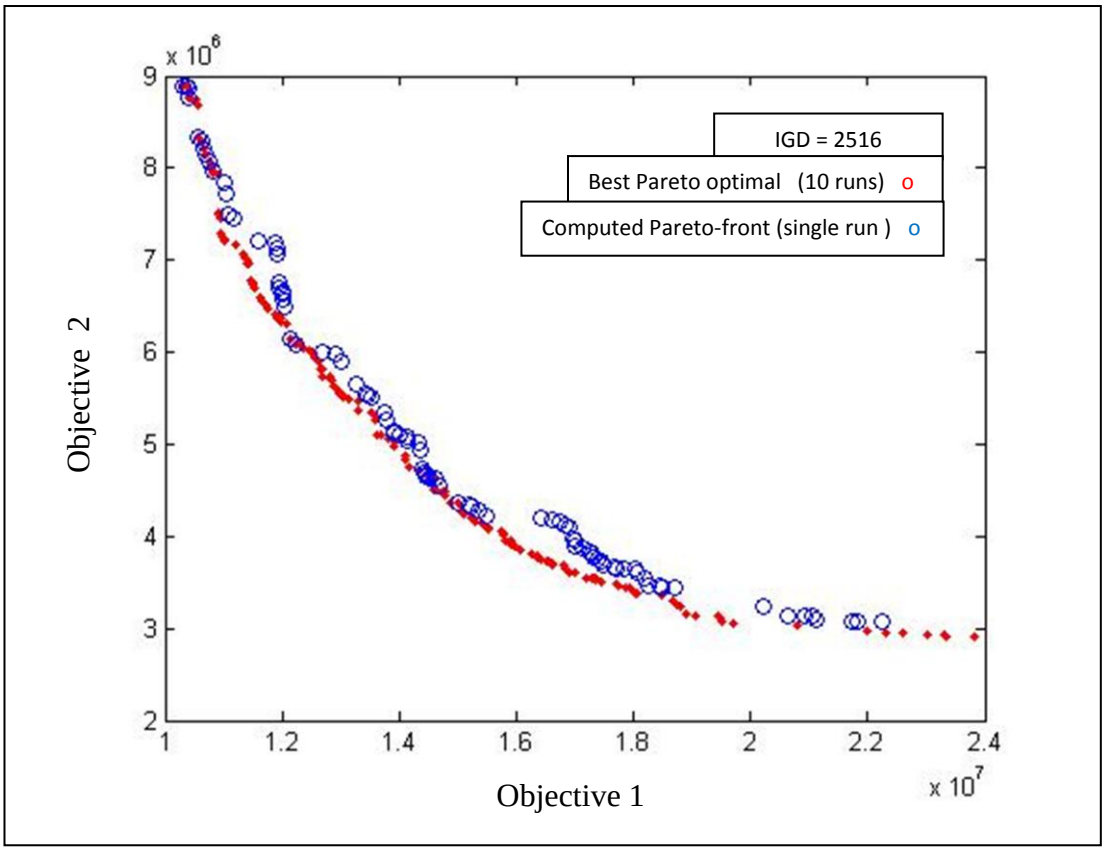

Figure 4.14 : The ABC result for 2-objective QAP (KC20-2fl-3rl) test problem with real-life distribution.

In Figure 4.14 shows how the Pareto-front set (single run) performed to (KC20-2fl-3rl) compared to the data from Pareto optimal set (10 runs). The non-dominated sets found by (single run) are near to the ones on the (10 runs). In this respect, the number of solutions on the Pareto-front set (single run) is 92 solutions while the number of non-dominated solutions found by (10 runs) is 168.

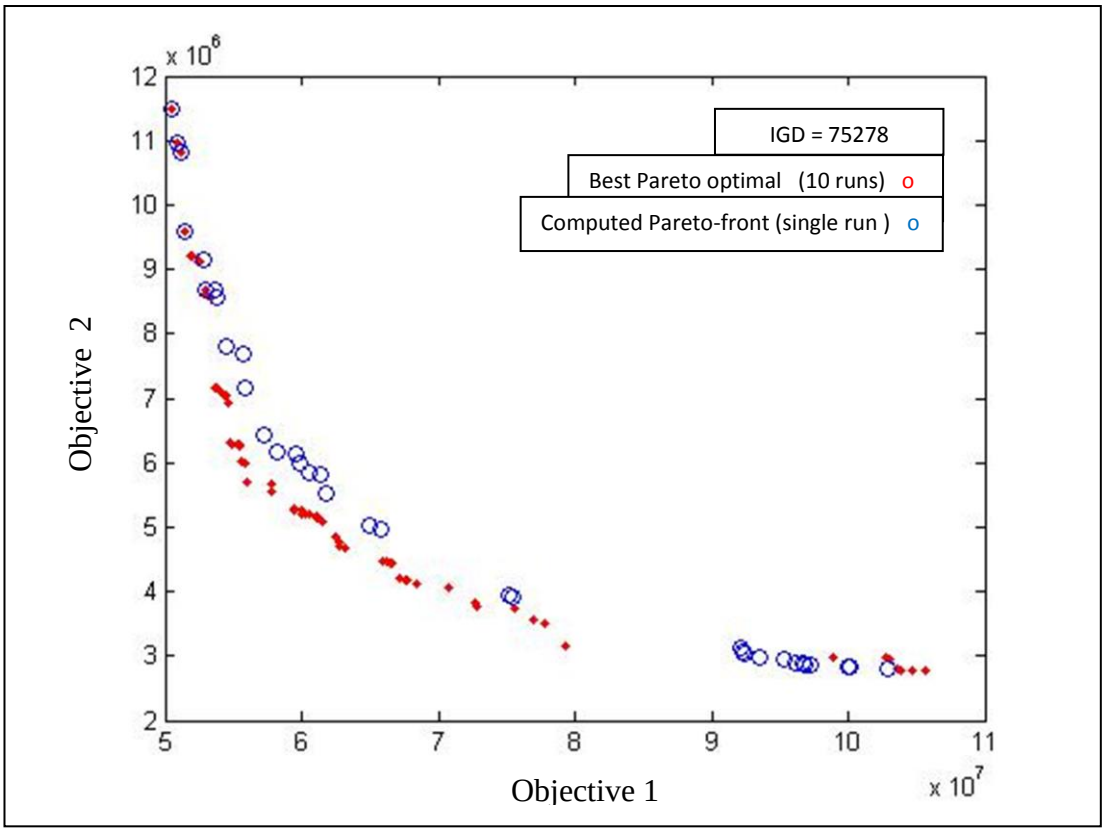

Figure 4.15 : The ABC result for 2-objective QAP (KC20-2fl-4rl) test problem with real-life distribution.

In Figure 4.15 shows how the Pareto-front set (single run) performed to (KC20-2fl-4rl) compared to the data from Pareto optimal set (10 runs). The non-dominated sets found by (single run) are close to the ones on the (10 runs). The number of Pareto points computed by Pareto-front set (single run) and Pareto optimal set (10 runs) are 34 and 68, respectively.

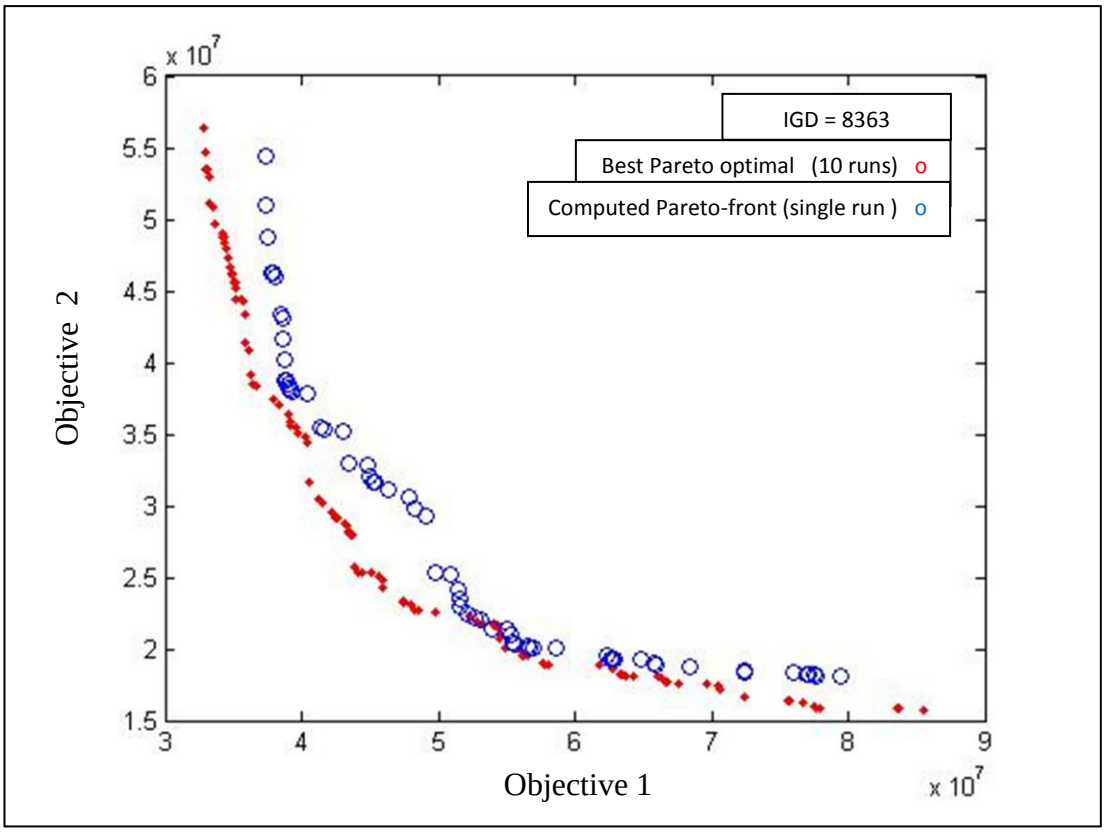

Figure 4.16 : The ABC result for 2-objective QAP (KC20-2fl-5rl) test problem with real-life distribution.

In Figure 4.16 shows how the ABC Pareto-front set (single run) performed to (KC20-2fl-5rl) compared to the data from Pareto optimal set (10 runs). The non-dominated sets found by (single run) are close to the ones on the (10 runs). In this respect, the number of solutions on the Pareto-front set (single run) is 65 solutions while the number of non-dominated solutions found by (10 runs) is 111 .

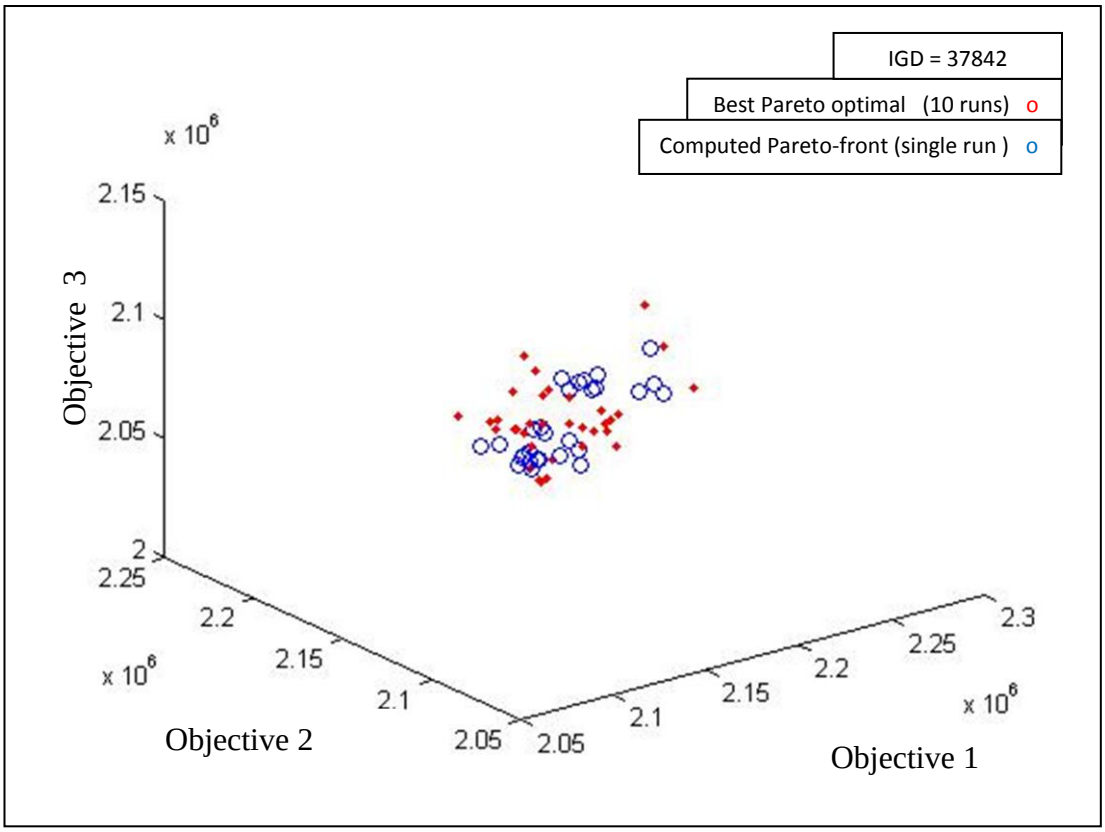

Figure 4.17 : The ABC result for 3-objective QAP (KC30-2fl-1uni) test problem with real-life distribution.

In Figure 4.17 shows how the Pareto-front set (single run) performed to (KC30-3fl-1uni) compared to the data from Pareto optimal set (10 runs). The non-dominated sets found by (single run) are very close to the ones on the (10 runs). In this respect, the number of solutions on the Pareto-front set (single run) is 28 solutions while the number of non-dominated solutions found by (10 runs) is 34.

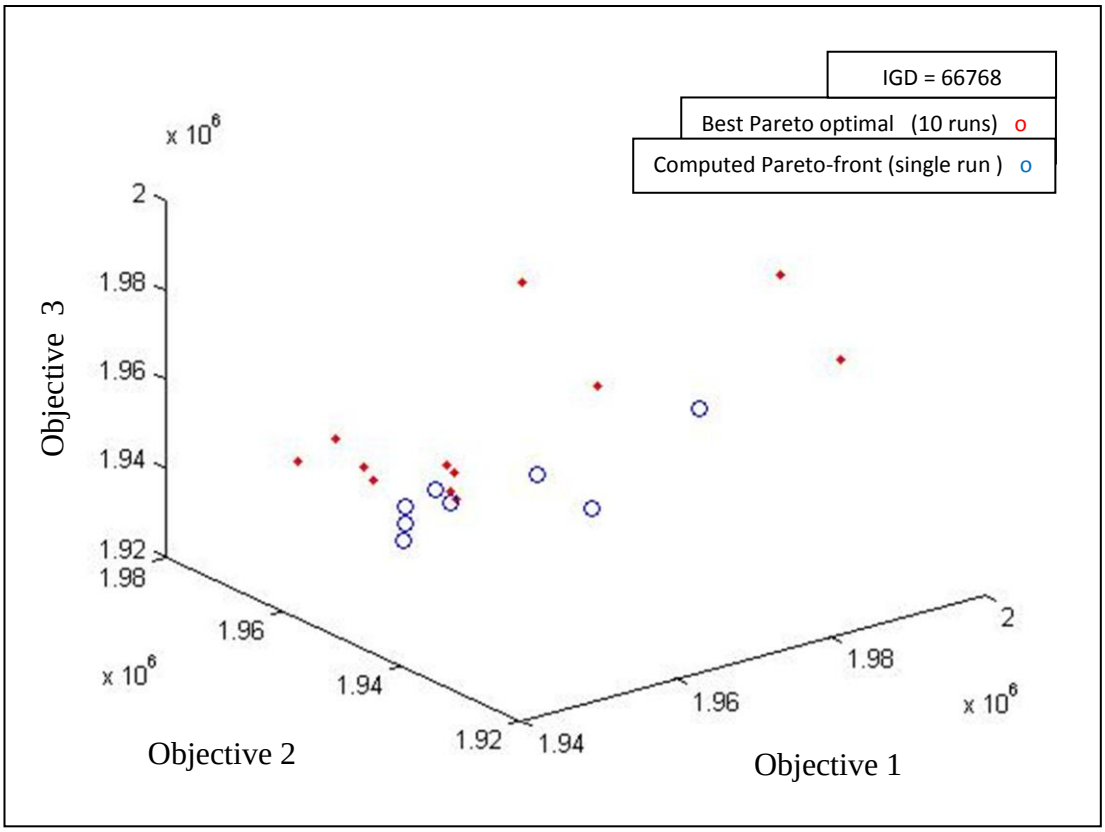

Figure 4.18: The ABC result for 3-objective QAP (KC30-3fl-2uni) test problem with uniformly random distribution.

In Figure 4.18 shows how the Pareto-front set (single run) performed to (KC30-3fl-2uni) compared to the data from Pareto optimal set (10 runs). The non-dominated sets found by (single run) are nearly to the ones on the (10 runs). The number of Pareto points computed by Pareto-front set (single run) and Pareto optimal set (10 runs) are 8 and 12, respectively.

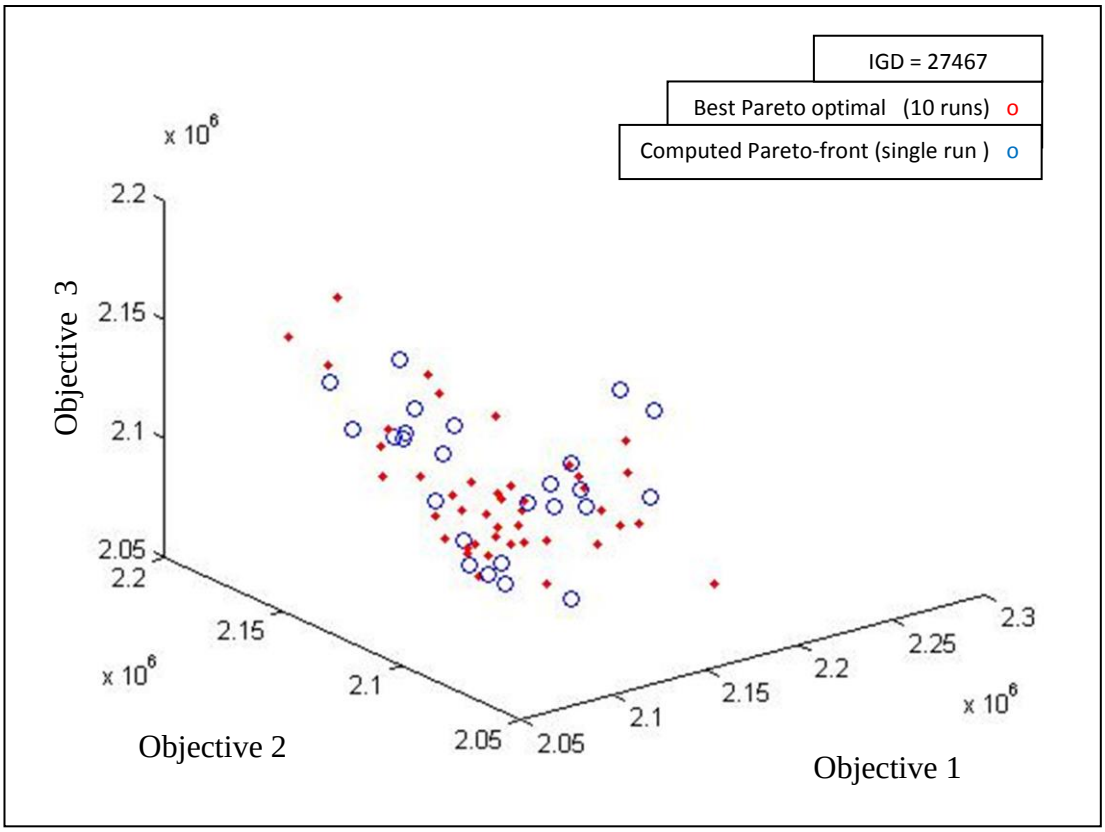

Figure 4.19: The ABC result for 3-objective QAP (KC30-3fl-3uni) test problem with uniformly random distribution.

In Figure 4.19 shows how the Pareto-front set (single run) performed to (KC30-3fl-3uni) compared to the data from Pareto optimal set (10 runs). The non-dominated sets found by (single run) are very close to the ones on the (10 runs). Considering the performances in terms of the number of Pareto points computed, the score of Pareto-front set (single run) is 25 while that of (10 runs) is 43.

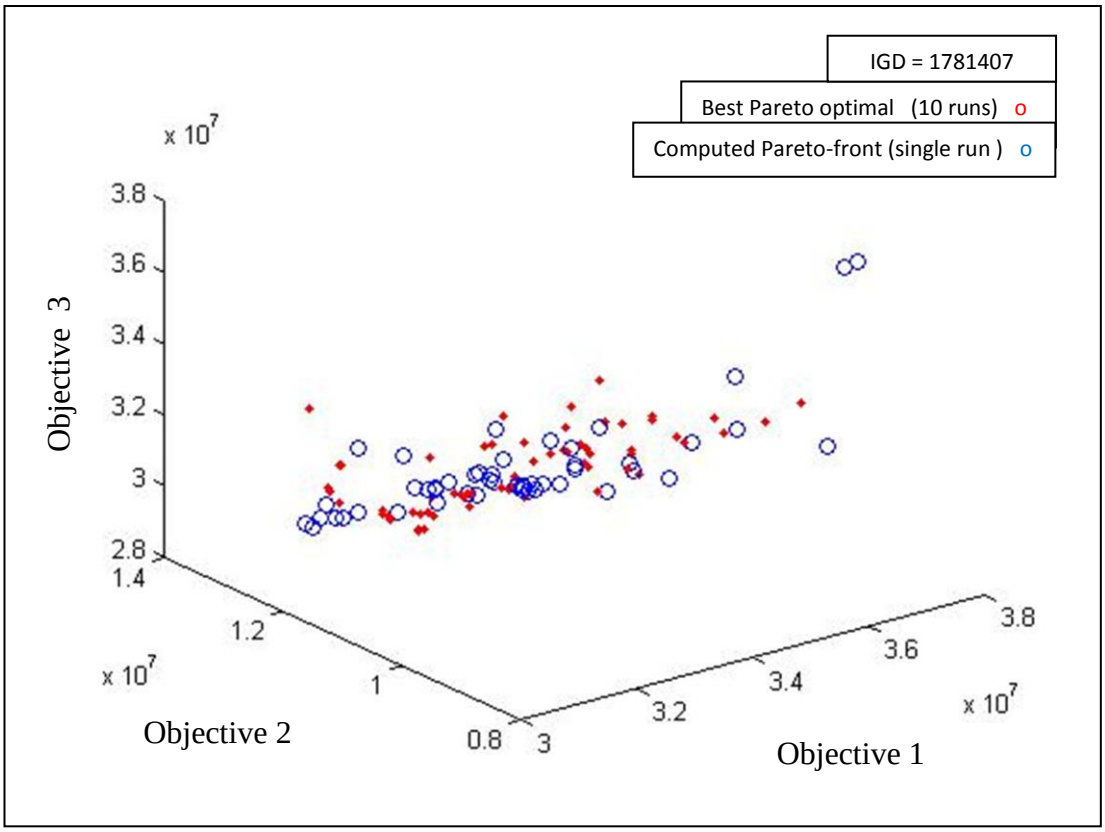

Figure 4.20: The ABC result for 3-objective QAP (KC30-3fl-1rl) test problem with real-life distribution.

In Figure 4.20 shows how the Pareto-front set (single run) performed to (KC30-3fl-1rl) compared to the data from Pareto optimal set (10 runs). In this respect, the number of solutions on the Pareto-front set (single run) is 195 solutions while the number of non-dominated solutions found by (10 runs) is 198 .

.

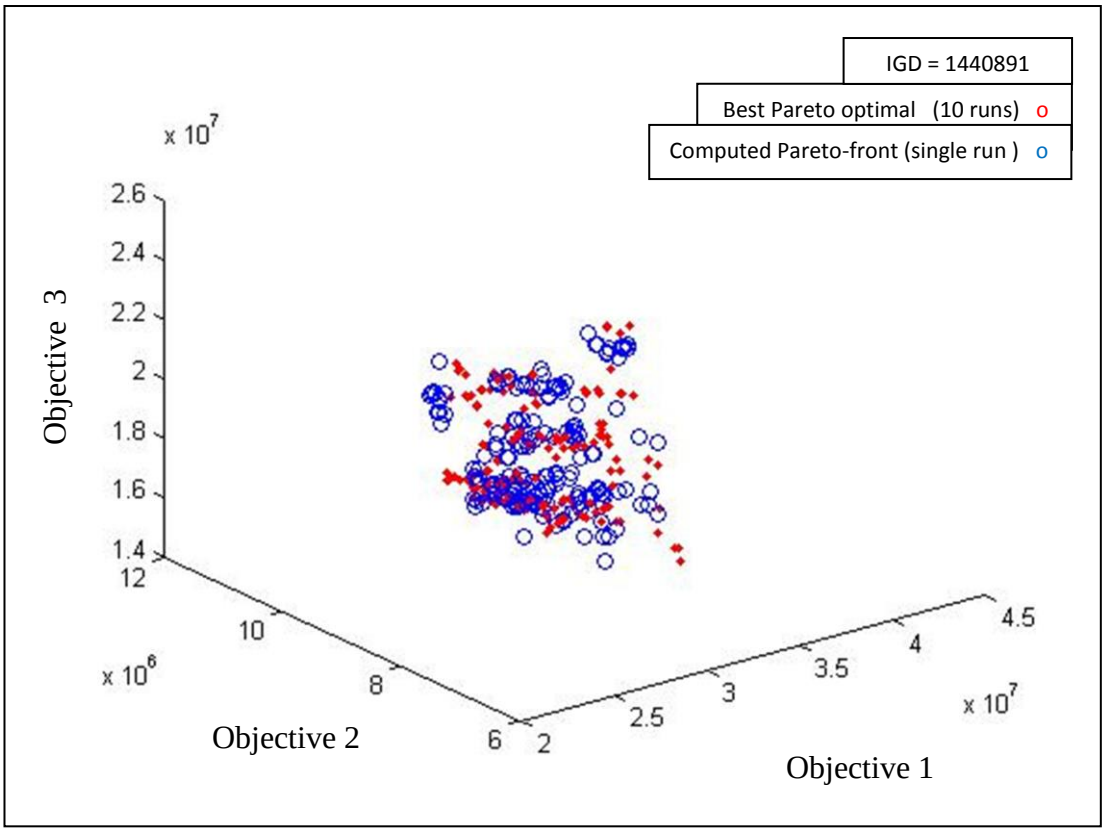

Figure 4.21 : The ABC result for 3-objective QAP (KC30-3fl-2rl) test problem with real-life distribution.

In Figure 4.21 shows how the Pareto-front set (single run) performed to (KC30-3fl-2rl) compared to the data from Pareto optimal set (10 runs). The non-dominated sets found by (single run) are very close to the ones on the (10 runs). Considering the performances in terms of the number of Pareto points computed, the score of Pareto-front set (single run) is 195 while that of (10 runs) is 198.

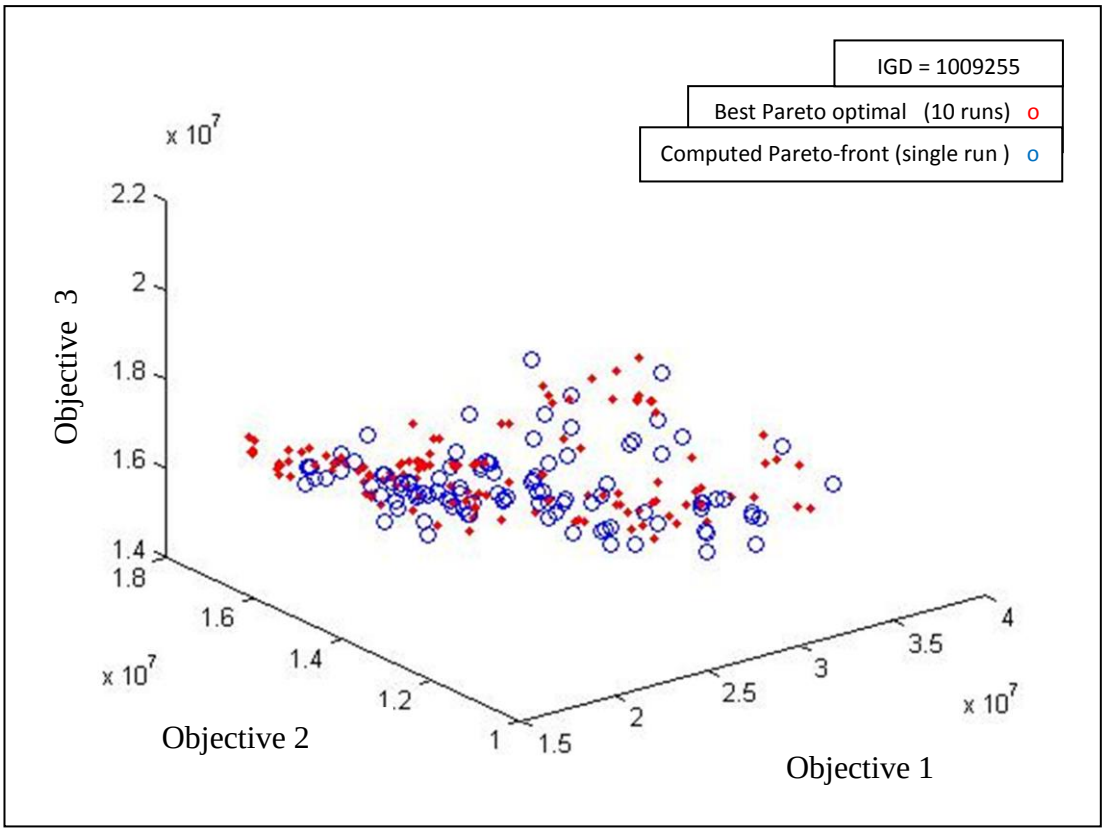

Figure 4.22: The ABC result for 3-objective QAP (KC30-3fl-3rl) test problem with real-life distribution.

In Figure 4.22 shows how the Pareto-front set (single run) performed to (KC30-3fl-3rl) compared to the data from Pareto optimal set (10 runs). The non-dominated sets found by (single run) are nearly to the ones on the (10 runs). The number of Pareto points computed by Pareto-front set (single run) and Pareto optimal set (10 runs) are 105 and 137, respectively.

Table 2: Comparison of Results

	IGD Values				
Test Name	ABC	mGRASP	MOEA	NSGA2	SPEA2
KC10-2fl-1uni	65	460	2211	6590	7795
KC10-2fl-2uni	0	0	4915	11284	13196
KC10-2fl-3uni	27	6	147	2393	4387
KC10-2fl-1rl	9709	3555	45512	318513	382993
KC10-2fl-2rl	25188	10460	128988	212026	226922
KC10-2fl-3rl	12725	2403	37239	300822	357818
KC20-2fl-1uni	195	21360	12058	53492	58575
KC20-2fl-2uni	2509	58830	16987	66425	64604
KC20-2fl-3uni	7.9	6677	4878	35764	44289
KC20-2fl-1rl	11641	1980730	433020	2914559	2623621
KC20-2fl-2rl	116820	1259521	192956	1895681	1520627
KC20-2fl-3rl	2516	653760	153859	1594337	1534329
KC30-3fl-1uni	37842	54552	20578	141325	167422
KC30-3fl-2uni	66768	110308	26415	161061	163284
KC30-3fl-3uni	27467	36927	16133	127932	154106
KC30-3fl-1rl	1781407	3350333	474028	5962007	7018525
KC30-3fl-2rl	1440891	2837723	421962	4538068	4986717
KC30-3fl-3rl	1009255	1658247	395310	4072323	4503648

The IGD-metric can assess the quality of approximation sets with respect to both convergence and diversity. The IGD values found by the five algorithms are shown in Tables 2. From these results, it is evident that ABC and mGRASP outperforms the three other algorithms in terms of both performance indicators on all mQAP instances. The superiority of ABC is due to the stronger selection pressure to move

towards the POF by having competition among individuals. Among the five algorithms, NSGA2 and SPEA2 show the worst performance with respect to minimizing the IGD values. The main reason for this might be that no local search is used to improve offspring solutions in these two approaches.

Chapter 5

CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

An artificial bee colony optimization algorithm is implemented for the solution of multiobjective quadratic assignment problem. Problem specific search operators are also within the scope of the implemented algorithm.

Experimental evaluations using well-known benchmark problem instances exhibited that the implemented algorithm is a powerful alternative for the solution of multiobjective quadratic assignment problem. Assessment with respect to inverted generational distance has also showed that the developed method is better than several well-known metaheuristics. This point is illustrated in Table 2 that clearly shows that the implemented algorithm outperforms widely known metaheuristics MOEA, NSGA2 and SPEA2.

Another observation from Table 2 is the success of the presented algorithm with respect to problem size and number of objectives. As expected, IGD scores increase with increasing problem size and number of objectives. However, relative powers of ABC compared to its competitors remain higher for all problem sizes.

Future work is planned to hybridize the ABC algorithm with other metaheuristics

and local search methods. Also using the presented method for the solution other hard optimization problems is also within the future work plans.

REFERENCES

[1] E. Mezura-Montes, Constraint-Handling in Evolutionary Optimization, Heidelberg: Springer Berlin, 2009.

[2] K. Deb, "An Efficient Constraint Handling Method for Genetic Algorithms," Computer. Methods Appl. Mech. Eng., vol. 186, no. 2-4, pp. 311-338, 2000.

[3] V. Pareto, Cours d'Economie Politique. Droz, Geneva, 1896.

[4] G. Chen., X. Huang., X. Yang., "Vector optimization: set-valued and vibrational analysis", Springer Verlag, 541 (2005).

[5] J. Knowles., D. Corne, "Instance Generators and Test Suites for the Multiobejective Quadratic Assignment Problem." Evolutionary Multi-Criterion Optimization, Second International Conference, EMO 2003, Faro, Portugal, April 2003, Proceed- ings , number 2632 in LNCS, edited by Carlos Fonseca, et al. 295–310. Springer,2003.

[6] M. P. Kleeman., R. O. Day., G.B. Lamont., Solving the Multi-objective Quadratic Assignment Problem Using a fast messy Genetic Algorithm. In: Congress on Evolutionary Computation (CEC'2003). Volume 4., Piscataway, New Jersey, IEEE Service Center (2003) 2277–2283

[7] H. Li ., D. Landa-Silva (2009), An Elitist GRASP Meta heuristic for the

Multi-objective Quadratic Assignment Problem, Springer-Verlag Berlin Heidelberg, LNCS 5467, 481–494.

[8] M. P. Kleeman., R. O. Day., G.B. Lamont., "Multi-Objective Evolutionary Search Performance with Explicit Building-Block Sizes for NPC Problems." To appear in Congress on Evolutionary Computation (CEC'2004)4 . Piscataway, New Jersey: IEEE Service Center, May 2004.

[9] M. Lopez-Ibanez, L. Paquete, and T. Stutzle, "Hybrid population based algorithms for the bi-objective quadratic assignment problem, "FG Intellektik, FB Informatik, TU Darmstadt, Tech. Rep. AIDA–04–11, DEC 2004, accepted by Journal of Mathematical Modelling and Algorithms.

[10] A. Acan, A. Unveren., "Evolutionary Multiobjective Optimization with a Segment-Based External Memory Support for the Multiobjective Quadratic Assignment Problem", 2005 IEEE Congress on Evolutionary Computation, 2-5 September 2005, Edinburg.

[11] M. Zhao, A. Abraham, C. Grosan, H. Liu, A Fuzzy Particle Swarm Approach to Multiobjective Quadratic Assignment Problems. Asia International Conference on Modelling and Simulation, 2008 .

[12] D. Garrett., D. Dasgupta, An Empirical Comparison of Memetic Algorithm Strategies on the Multiobjective Quadratic Assignment

Problem. IEEE Symposium on Computational Intelligence in Multicriteria Decision Making. Nashville, TN. March, 2009.

[13] A. Zinflou., C. Gagné., M.Gravel., A hybrid genetic/immune strategy to tackle the multiobjective quadratic assignment problem, Proceedings of the 12th European Conference on Artificial Life (ECAL 2013), Taormina, Italy, September 2-6 2013.

[14] D. Garrett., D. Dasgupta, Analyzing the Performance of Hybrid Evolutionary Algorithms for the Multiobjective Quadratic Assignment Problem. IEEE Congress on Evolutionary Computation (CEC '06). Vancouver, Canada. July, 2006.

[15] M. Drugan, Multi-objective Quadratic Assignment Problem instances generator with a known optimum solution. Artificial Intelligence lab, Vrije Universiteit Brussel, Pleinlaan 2, B-1050 Brussels, Belgium.

[16] N. Quijano., K.M. Passino., 2007a. Honey Bee Social Foraging Algorithms for Resource Allocation, Part I: Algorithm and Theory. Submitted, American Control Conference.

[17] N. Quijano., K.M. Passino., 2007b. Honey Bee Social Foraging Algorithms for Resource Allocation, Part II: Application. Submitted, American Control Conference.

[18] T. D. Seeley, (1995). The wisdom of the hive: The social physiology of

honey bee colonies. Cambridge, MA: Harvard University Press.

[19] K. von Frisch, "Decoding the language of the bee" Decoding the language of the bee, Science, vol. 185, no. 4152, pp. 663-668, 1974.

[20] S. Nakrani., C. Tovey., 2004. On Honey Bees and Dynamic Allocation in Internet Hosting Centers. Adaptive Behavior, 12(3-4):223-240.

[21] X.S. Yang., 2005. Engineering Optimization via Nature-Inspired Virtual Bee Algorithms. IWINAC 2005, LNCS 3562, 17-323.

[22] A. Afshar., O.B,Hadad., M.A,Marino, B.J,Adams,., 2007. Honey-bee mating optimization (HBMO) algorithm for optimal reservoir operation. J. Franklin Institute, (344):452-462.

[23] D. Karaboga., 2007. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of global optimization, 39:459-471.

[24] K.,F., LI, J., MA, Z., LI, H., 2011. Artificial Bee Colony Algorithm with Local Search for Numerical Optimization. Journal of Software, 3(6):490-497.

[25] D. Karaboga, & B. Basturk, (2008). Artificial Bee Colony (ABC) optimization algorithm for solving constrained optimization problems. Advances in Soft Computing, Vol (8), Issue (1), 687-697.

[26] D. Karboga, (2005). An idea based on honey bee swarm for numerical optimization, Technical Report TR06. 2005: Erciyes University.

[27] É. D. Taillard. Robust taboo search for the quadratic assignment problem. Parallel Computing, 1991, 17: 443-455.

[28] F. Glover (1986) Future paths for integer programming and links to artificial intelligence. Computers and Operations Research, 13, 533–549.

[29] J. Skorin-Kapov. Tabu search applied to the quadratic assignment problem. ORSA. Journal on Computing, 1990, 2: 33-45.

[30] H. A. Thompson and P. J. Fleming, "An Integrated Multi-Disciplinary Optimization Environment for Distributed Aero-engine Control System Architectures," in Proceedings of the Fourteenth World Congress of International Federation of Automatic Control, pp. 407-412. 1999

[31] A. J. Chipperfield and P. J. Fleming, "Multiobjective Gas Turbine Engine Controller Design Using Genetic Algorithm s,"IEEE Transactions on Industrial Electronics, vol. 43, no. 5, pp. 583-587. 1996.

[32] C. M. Fonseca and P. J. Fleming, "Multiobjective Optimal Controller Design with Genetic Algorithms," in Proceedings on IEE Control, pp. 745-749, 1994.

[33] T. H. Liu and K. J. Mills, “Robotic Trajectory Control System Design for Multiple Simultaneous Specifications: Theory and Experimentation,” in Transactions on ASME, vol. 120, pp. 520-523. 1998.

[34] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," in Neural Networks,1995. Proceedings. IEEE International Conference on, pp. 1942-1948 vol.4,1995.

[35] J.H. Holland, Adaptation in Natural and Artificial Systems, The University of Michigan Press, 1975.

[36] M. Dorigo, "Optimization, Learning and Natural Algorithms," Ph. D. Thesis,1992.

[37] X.S. Yang, Nature-Inspired Metaheuristic Algorithms, United Kingdom: Luniver Press, 2008.

[38] V. Chankong and Y. Y. Haimes. Multiobjective Decision Making – Theory and Methodology. Elsevier Science, New York, 1983.