DOKUZ EYLÜL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

METAHEURISTIC APPROACHES TO P-MEDIAN PROBLEM AND ITS EXTENSIONS

by

Ali Erel KASIRGA

January, 2022

İZMİR

METAHEURISTIC APPROACHES TO P-MEDIAN PROBLEM AND ITS EXTENSIONS

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University In Partial Fulfillment of the Requirements for the Degree of Master of Science in Industrial Engineering, Industrial Engineering Program

by

Ali Erel KASIRGA

January, 2022

İZMİR

M.Sc THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"METAHEURISTIC APPROACHES TO P-MEDIAN PROBLEM AND ITS EXTENSIONS"** completed by **ALİ EREL KASIRGA** under supervision of **ASSOC. PROF. DR. FEHMİ BURÇİN ÖZSOYDAN** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Fehmi Burçin ÖZSOYDAN

Supervisor

Asst. Prof. Dr. İlker GÖLCÜK

(Jury Member)

Assoc. Prof. Dr. Mualla Gonca AVCI

(Jury Member)

Prof. Dr. Okan FISTIKOĞLU

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGEMENTS

Firstly, I would like to state my sincere gratitude to Assoc. Prof. Dr. Fehmi Burçin ÖZSOYDAN for his supervision, support and understanding. He not only served as my supervisor but also encouraged me throughout the progress of this dissertation.

Finally, I would like to express my gratefulness to my parents, Pelgin and Hasan Erhun KASIRGA, for their endless support and mentoring they provided me through my entire life, their endless love, cheering, and trust in me. Without the priceless support and motivation of them, I could not have succeeded.

Ali Erel KASIRGA

METAHEURISTIC APPROACHES TO P-MEDIAN PROBLEM AND ITS EXTENSIONS

ABSTRACT

Facility location problems have grabbed attention of researchers, as they closely related to long-term strategic decisions that are of great importance for survival of companies. They play important roles in increasing the profitability of those companies and in reducing their costs. There exist various types of location problems in the related literature. This study aims to develop solution approaches to the P-median and the Uncapacitated Facility Location Problems, which are regarded as the two widely known facility location problems. Since these two problems are in the NP-hard category, a metaheuristic approach, namely, Flower Pollination Algorithm (FPA) is employed. As one can see from the reported literature, performance of FPA has not been studied much in facility location problems. Moreover, applications of this promising optimizer in discrete solution spaces are still lacking. This is the main motivation of the present study. In this context, FPA is modified to be able to run in binary space by making use of some evolutionary operators. Thus, while numerous metaheuristic algorithms require for some auxiliary operators, the proposed algorithm can directly work in binary spaces. Secondarily, FPA is further improved by an adaptive mutation procedure that introduces a greater level of diversity at earlier iterations and encourages intensification towards the end of the search. A comprehensive experimental study including appropriate statistical demonstrations is conducted to analyze performance of the proposed approach. As shown by the experimental results, the proposed solution approach achieves significant improvements over some of the existing results.

Keywords: P-median problem, uncapacitated facility location problem, flower pollination algorithm, metaheuristic algorithms

P-MEDYAN PROBLEMİ VE UZANTILARI İÇİN METASEZGİSEL ÇÖZÜM YAKLAŞIMLARI

ÖZ

Tesis yeri seçimi problemleri, şirketlerin hayatlarını sürdürebilmeleri için büyük önem taşıyan uzun vadeli stratejik kararlar olması nedeniyle araştırmacıların dikkatini çekmiştir. Tesis yeri seçimi problemleri, şirketlerin karlılıklarının artmasında ve maliyetlerinin düşürülmesinde önemli rol oynamaktadırlar. İlgili literatürde pek çok tesis yeri seçimi problemi türü bulunmaktadır. Bu çalışma, yaygın olarak bilinen iki tesis yerleşim problemlerinden P-medyan ve Kapasitesiz Tesis Yeri Seçimi Problemlerine çözüm yaklaşımları geliştirmeyi amaçlamaktadır. Bu iki problem NP-zor kategorisinde olduğu için, Çiçek Tozlaşma Algoritması adlı metasezgisel bir yaklaşım kullanılmıştır. Belirtilen literatürden görülebileceği gibi, tesis yerleşim problemlerinde Çiçek Tozlaşma Algoritması'nın performansı bu alanda yeteri kadar çalışılmamıştır. Ayrıca, bu umut verici algoritmanın kesikli çözüm uzaylarındaki uygulamaları günümüz itibariyle seyrektir. Bu, mevcut çalışmanın ana motivasyonunu oluşturmaktadır. Bu bağlamda Çiçek Tozlaşma Algoritması, bazı evrimsel operatörler kullanılarak ikili uzayda çalışabilecek şekilde düzenlenmiştir. Bu nedenle, çok sayıda metasezgisel algoritma için bazı yardımcı operatörler gerekirken, önerilen algoritma doğrudan ikili uzaylarda çalışabilmektedir. İkincil olarak, Çiçek Tozlaşma Algoritması, başlangıç iterasyonlarında daha fazla çeşitlilik sağlayan ve aramanın sonuna doğru çeşitliliği azaltıp yoğunlaşmayı teşvik eden uyarlanabilir bir mutasyon prosedürü ile daha da geliştirilmiştir. Önerilen yaklaşımın performansını analiz etmek için uygun istatistiksel testleri de içeren kapsamlı bir deneysel çalışma yürütülmüştür. Deneysel sonuçların da gösterdiği gibi, önerilen çözüm yaklaşımı, mevcut sonuçların bazılarına göre önemli iyileştirmeler sağlamıştır.

Anahtar kelimeler: P-medyan problemi, kapasitesiz tesis yeri seçimi problemi, çiçek tozlaşması algoritması, metasezgisel algoritmalar

CONTENTS

Page

M.SC THESIS EXAMINATION RESULT FORM II

ACKNOWLEDGEMENTS III

ABSTRACT IV

ÖZ V

LIST OF FIGURES VIII

LIST OF TABLES IX

CHAPTER ONE - INTRODUCTION 1

CHAPTER TWO - PROBLEMS USED IN THIS STUDY 4

2.1 Uncapacitated Facility Location Problem 4
2.2 The P-median Problem 5
2.2.1 Uncapacitated P-Median Problem 8
2.2.2 Capacitated P-Median Problem 9
2.2.3 Obnoxious P-Median Problem 10

CHAPTER THREE - LITERATURE REVIEW 12

CHAPTER FOUR - PROPOSED SOLUTION APPROACHES 16

4.1 Canonical Flower Pollination Algorithm 16
4.2 Proposed Modifications 19
4.2.1 Solution Encoding 20
4.2.2 Uniform-Based Xover for Inheritance 20
4.2.3 Bit-Flip Mutations to Avoid Local Optima 22
4.2.4 Greedy Repair Procedure 23

CHAPTER FIVE - EXPERIMENTAL STUDY 26

5.1 Calibration of Parameters 26
5.1.1 Calibration of Parameters for UFLP 26
5.1.2 Calibration of Parameters for UPMP 26
5.2 Computational Results for UFLP 27
5.3 Computational Results for UPMP 30
5.4 Statistical Analysis 34

CHAPTER SIX - CONCLUDING REMARKS 38

REFERENCES 40

LIST OF FIGURES

Page

Figure 2.1 1-Median Problem Example 6
Figure 4.1 Pseudo code of the original FPA 18
Figure 4.2 Flowchart of FPA 19
Figure 4.3 Binary encoding technique 20
Figure 4.4 Uniform-based xover 21
Figure 4.5 A pseudo code for the mutation operator 22
Figure 4.6 An illustration of various step size values 23
Figure 4.7 A pseudo code for the random repair procedure 24
Figure 4.8 A pseudo code for the greedy repair procedure 25
Figure 4.9 A pseudo code for the proposed bFPA 25
Figure 5.1 Convergence plots of bFPA 30
Figure 5.2 Convergence plots of gbFPA and rbFPA 31

LIST OF TABLES

Page

Table 3.1 Tabular form of literature review 12
Table 5.1 Determination of the paramters phiStep and initRate 27
Table 5.2 Determination of the paramters xOverProb and prob 27
Table 5.3 Results for UFLP 29
Table 5.4 Results for UPMP 32
Table 5.5 Computational results for P-Median instances 26-40 33
Table 5.6 Results of friedman test for UFLP 34
Table 5.7 Post-hoc results for UFLP 35
Table 5.8 Comparisons results of gbFPA and rbFPA for sign test 36

CHAPTER ONE
INTRODUCTION

Location science is of great importance for companies in today's world that is globalizing. Companies are working to reduce costs to keep up with this competitive environment without compromising quality and increasing prices. The facility location problem (Alizadeh, 2009) has a great potential here since it significantly affects the performance and cost of supply chains. The facility location problem includes determining the site of civil service points for instance schools, fire stations hospitals, police stations and private company plants such as warehouses, power plants, logistics centers, waste centers, stores. A facility is actually a place, where the business will be established and will continue to work throughout its life. In other words, a facility is a place, where an enterprise can perform its essential functions such as supply, production, storage, and distribution and their associated economic objectives. Once it is established, it is not easy to change the location of a facility whether it is a small-scaled or a large-scaled one. For this reason, selection facility location is a vital decision that has long-term impacts on the profitability of businesses.

The facility location problem can be considered as determining the facility locations among lots of candidate locations so as to service demand points (users) as profitably as possible. When the facility location is determined, it might have some positive or negative effects on users depending on the facility's distance to the user. Assuming that the impacts are positive, it is desirable to locate the facilities as close as possible to those demand points. In contrast, if the impacts are negative (undesirable or disturbing facilities), locating the facilities as far away as possible is desirable. In cases where the facilities is distant from the customers, additional costs are paid. Thus, o a compromise decision should be made considering the distance and the effects that the users will be exposed to. Hospitals, schools, banks and markets are examples of some desirable facilities. Power plants, treatment plants and landfills are some examples of undesirable or disturbing facilities.

Many facility location problems include constraints such as capacity, demands, restrictions on candidate locations or restrictions on allocations of the demand points to facilities. Consequently, this assignment process of demand points to the established facilities has a cost referred to as the allocation cost. The allocation cost is directly related to the distance of the facility to the demand points. The merit here is to keep the total cost at minimum, which in most cases is encountered as summation of allocation costs and facility opening costs. In this regard, for a company in private sector, when it is necessary to determine the location of facilities in the supply chain such as factories, warehouses and distribution systems, allocation costs correspond to the sum of delivery and transportation costs. Therefore, the decision maker tries to minimize these costs. However, one should pay attention that the efficiency of the service in both cases depends on the facility's location, which determined according to the demand points. For this reason, choosing facility location is a strategically important decision.

Plants are categorized by various features for instance capacity, cost, supply service and number. In some cases while the number of facilities to be placed is fixed, in some other cases it is also possible that the number of facilities to be placed is variable. Furthermore, while some facilities have infinite capacity, some facilities do not. To sum up, the location problems can be reported as producing solutions in serving demand points most appropriately by considering the characteristics of the problem and the defined or emerging constraints.

Facility location problems are mostly known to be NP-hard (Kariv and Hakimi, 1979) problems. Consequently, using metaheuristic algorithms is very common among researchers, especially in solving large-scaled instances. As one category of the metaheuristic algorithms, bio-inspired algorithms (Selvaraj et al., 2014) have become popular among researchers in recent years. Bio-inspired algorithms mimic the nature of living beings in solving optimization problems. Even though these algorithms may not assure optimality, researchers use them commonly since they produce high-quality solutions in acceptable time and they are open to various improvements. In this study, a bio-inspired algorithm, namely, Flower Pollination Algorithm (FPA) (Yang, 2012) is adopted to be used in solving facility location problems. The key reason to select

this algorithm is its simplicity in terms of complexity and intelligibility and being open to numerous improvements.

The main contribution of the present is study is as follows: FPA is modified to run in binary spaces, first. In this context, some evolutionary operators are borrowed from evolutionary algorithms and they are adopted in the proposed FPA. Thus, while numerous metaheuristic algorithm require for some auxiliary operators such as sigmoid transfer functions, the proposed algorithm can directly work in binary spaces. Secondarily, FPA is further improved by an adaptive mutation, which introduces a greater diversity at earlier iterations and encourages intensification in later iterations. Thus, local optima and premature convergence issues are avoided while an intensified search in later iterations is conducted. The performance of the suggested algorithm is tested both on the on the Uncapacitated Facility Location Problem (UFLP) and Uncapacitated P-median Problem (UPMP) (Zaferanieh and Abareshi, 2021), which are some of the renowned location science optimization problems. The details of these problems are presented in the upcoming sections of this study.

The rest of this thesis is organized as follows: in Section 2, information about the focused problems are given. Additionally, studies that use metaheuristic algorithms in solving the introduced problems are included. In Section 3, detailed information about FPA and the used auxiliary procedures are given. In Section 4, a preliminary work where the proposed FPA is applied to (UFLP) of which the data is obtained through OR-Library is presented. Section 5 is dedicated to the experimental study and discussion. Lastly, Section 6 presents concluding remarks.

CHAPTER TWO
PROBLEMS USED IN THIS STUDY

2.1 Uncapacitated Facility Location Problem

UFLP (Cournuejols et al., 1990) is a well-known location science problem and it forms the basis for most related models used in supply chain design. As discussed before, the aim in UFLP is to determine a set of facilities to open among candidate locations so as to minimize the total cost, which is equal to the the sum of facility opening costs and transshipment costs between the opened facilities and demand points (customers). More formally, UFLP can be formulated as follows:

$$Minimize \sum_{i=1}^{N}\sum_{j=1}^{M} c_{ij}x_{ij} + \sum_{i=1}^{N} f_i y_i \tag{2.1}$$

$$\sum_{i=1}^{N} x_{ij} = 1 \quad j = 1, \ldots, M \tag{2.2}$$

$$x_{ij} \leq y_i \qquad i = 1, \ldots, N \quad j = 1, \ldots, M \tag{2.3}$$

$$x_{ij} \in \{0,1\} \quad i = 1, \ldots, N \quad j = 1, \ldots, M \tag{2.4}$$

$$y_i \in \{0,1\} \quad i = 1, \ldots, N \quad j = 1, \ldots, M \tag{2.5}$$

Eq. 2.1 is the objective function. In Eq. 2.1, N represents the number of potential facility locations, M denotes the number of customers (demand points), f_i is the facility opening cost associated to ith point, c_{ij} is the cost occurring between facility point i and the demand point j. Eq. 2.2 provides that demand restrictions are satisfied. Eq. 2.3 provides that customers can be assigned only to opened facilities. Eq. 2.4 and Eq. 2.5 impose sign restrictions on decision variables.

Since UFLP is known to be in the category of NP-hard problems (Johnson & Garey, 1979), various metaheuristics solution methods are reported in the related literature. To the best of our knowledge, Balinski (1965) first introduce a mathematical model

for UFLP. One year later, Efroymson and Ray (1966) report a modification on this model and develop a branch and bound algorithm. In 1972, Khumawala (1972) propose a more efficient branch and bound algorithm. In the late ’70s, Erlenkotter (1978) introduce the dual-based branch and bound algorithm by using ascent and adjustment procedures. Subsequently, by rearranging the primal-dual version of the algorithm developed by Erlenkotter (1978), Koerkel (1989) propose a more efficient algorithm for the required solution time. In 1999, Al-Sultan and Al-Fawzan (1999) report a new strategy that is based on Tabu Search (TS) to solve UFLP. In 2003, Ghosh (2003) studies the performance of generic local search techniques. Three years later, Şevkli and Güner (2006) develop a Particle Swarm Optimization (PSO) to solve UFLP. Later, Wang et al. (2008) and Güner and Şevkli (2008) propose a PSO to solve UFLP. Next, Kiran (2015) implements a bio-inspired algorithm called Firefly Algorithm (FA) for UFLP. A constructive algorithm is employed by Montoya-Torres et al. (2011). According to the reported results, the proposed algorithm can be considered as efficient in solving UFLP. In 2017, Tsuya et al. (2017) use FA to solve UFLP. Later, Armas et al. (2017) solve the stochastic extension of UFLP. Subsequently, Baykasoglu et al. (2020) present a binary Weighted Superposition Attraction Algorithm (bWSA) to solve UFLP. One year later, Ozsoydan (2019) proposes a modified swarm-based intelligence algorithm to solve some binary optimization problems, including UFLP. Recently, Ozsoydan (2019) and Golcuk and Ozsoydan (2020) implement another bio-inspired algorithm, namely, Grey Wolf Optimization (GWO), to solve UFLP.

2.2 The P-median Problem

PMP, which is another famous location science problem that is frequently considered by researchers, is introduced by Hakimi (1964). The P-median problem is the problem of determining the total weighted distance between *n* demand points and *p* facilities with the minimum cost. Cost here can be the distance between facilities, as well as a value such as time or money. In the PMP, *p* facilities are placed at specific points (nodes) in the network. Therefore, the P-median facility location problem is among discrete selection problem types.

The fundamental form of PMP is the 1-median problem (Figure 1). As the name infers, only one facility meets the needs of demand points. As the median number increases, the *p* value takes different values.

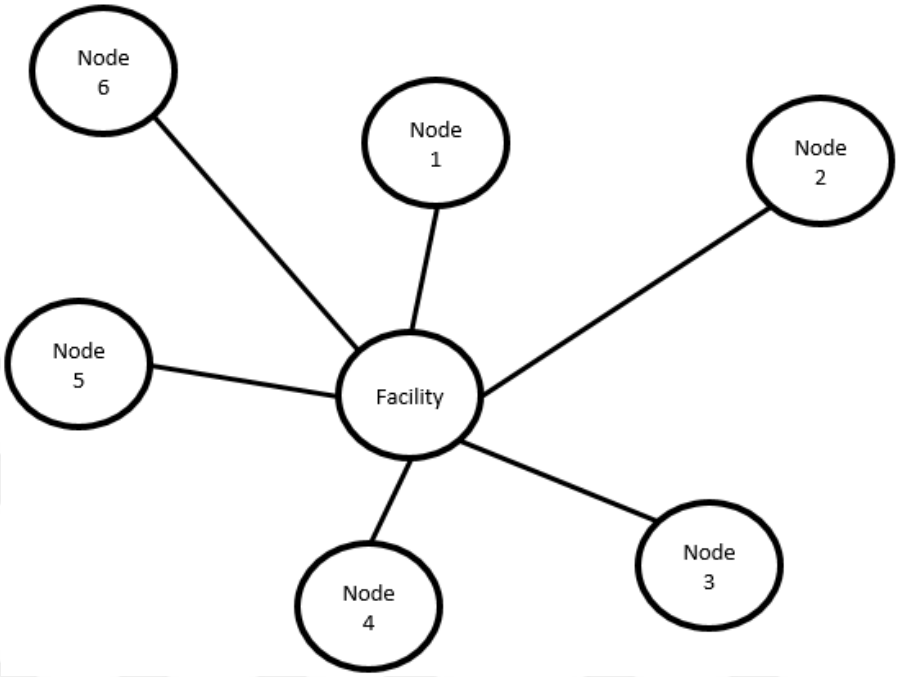


Figure 2.1 1-Median Problem Example

Figure 2.1 illustrates the 1-median problem solution, where the demands of six nodes are met by a single facility.

In the classical PMP, the number of facilities to be opened and the number of customers are priori known. It is also assumed that the points, locations, where facilities can be opened are known. Additionally, according to the general assumption, there is no capacity limit for the facilities that serve and there is no cost to open a facility (Farahani and Hekmatfar, 2009). In the PMP, it is assumed that, the serving points (serving facilities) can only be placed on the nodes on the given network. Considering this feature, one can show the possible solution number of a problem consisting of *n* nodes and *p* facilities to be opened with the following Eq 2.6.

$$\binom{n}{p} = \frac{n!}{p!\,(n-p)!} \tag{2.6}$$

For example, in a PMP, if the number of nodes (*n*) is 15 and the number of facilities to serve (*p*) is 5, the number of possible solutions is 3003. In another example, the number of possible solutions is 75.287.520, if *n* value is 100 and the number of service providers does not change. Considering that the PMP is an NP-hard problem, it is not easy to receive the optimal solution by exact methods. Therefore, this situation has led researchers to develop many alternative methods and algorithms to solve the P-median problem.

Revelle and Swain (1970) develop the mathematical notation of the this problem in 1970. Twenty-seven years later, Rolland et al. (1997) improve this formulation. In this context, the mathematical model of the P-median problem is formulated as follows:

$$y_j = \begin{cases} 1\ if\ facility\ j\ is\ opened \\ 0\ otherwise \end{cases}$$

$$x_{ij} = \begin{cases} 1\ if\ a\ node\ i\ is\ assigned\ to\ facility\ j \\ 0\ otherwise \end{cases}$$

$$a_i = demant\ at\ node\ i$$

$$d_{ij} = travel\ distance\ from\ node\ i\ to\ node\ j$$

$$p = number\ of\ facilities\ to\ be\ located$$

$$Min \sum_{i=1}^{n} \sum_{j=1}^{n} a_i\, d_{ij}\, x_{ij} \tag{2.7}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \tag{2.8}$$

$$x_{ij} \leq y_j \qquad \forall i \in I, \forall j \in J \tag{2.9}$$

$$\sum_{i=1}^{n} y_j = p \tag{2.10}$$

$$x_{ij}, y_j \in \{0,1\} \qquad \forall i \in I, \forall j \in J \tag{2.11}$$

Eq. 2.7 represents the objective function. Eq. 2.8 provides that customers are assigned exactly to one facility. Eq. 2.9 prohibits assignment to a closed facility. Eq. 2.10 denotes the requirement of exactly *p* facilities to be located. Eq. 2.11 impose sign restrictions to variables. The PMP (Reese, 2006) has numerous application and modifications. The rest of this chapter is devoted to present to most commonly referred variants.

2.2.1 Uncapacitated P-Median Problem

When there is not any limitation on the capacity of facilities, the problem (Bastı, 2012) is also known as the Uncapacitated PMP (UPMP). The aim in UPMP is to select *p* locations for facilities to serve customers at a minimum cost, where the capacity of each facility is equal or greater than the sum of the demand of all customers, which is served by the related facility.

$$y_j = \begin{cases} 1 \; if \; facility \; j \; is \; opened \\ 0 \; otherwise \end{cases}$$

$$x_{ij} = \begin{cases} 1 \; if \; a \; node \; i \; is \; assigned \; to \; facility \; j \\ 0 \; otherwise \end{cases}$$

$$\min \sum_{i \epsilon M} \sum_{j \epsilon N} d_{ij} x_{ij} \tag{2.12}$$

$$\sum_{i \epsilon M} x_{ij} = 1, \quad J \epsilon N \tag{2.13}$$

$$x_{ij} \leq y_j \qquad i \epsilon M, j \epsilon N \tag{2.14}$$

$$\sum_{j \epsilon N} y_j = p \tag{2.15}$$

$$y_j, x_{ij} \epsilon \{0,1\}, \qquad \forall i \epsilon M, \forall j \epsilon N \tag{2.16}$$

Eq. 2.12 represents the objective function. Eq. 2.13, Eq. 2.14 and Eq. 2.15 are analogous to Eq. 2.8, Eq. 2.9 and Eq. 2.10, respectively.

2.2.2 Capacitated P-Median Problem

In the Capacitated PMP (CPMP), one attempts to obtain optimal solution by additionally considering capacity of *p* service points, serving to *n* customers (Yaghini et al., 2013). CPMP is formulated in he following.

$$y_j = \begin{cases} 1\ if\ facility\ j\ is\ opened \\ 0\ otherwise \end{cases}$$

$$x_{ij} = \begin{cases} 1\ if\ a\ node\ i\ is\ assigned\ to\ facility\ j \\ 0\ otherwise \end{cases}$$

$$q_i = demand\ of\ each\ entity$$

$$Q_j = capacity\ of\ each\ possible\ median$$

$$d_{ij} = distance\ matrix$$

$$x_{ij} = allocation\ matrix$$

$$Min \sum_{i \in N} \sum_{j \in M} d_{ij} x_{ij} \tag{2.17}$$

$$\sum_{j \in N} x_{ij} = 1; \quad i \in M, \tag{2.18}$$

$$\sum_{j \in M} y_j = p; \quad j \in M, \tag{2.19}$$

$$\sum_{i \in N} q_i x_{ij} \leq Q_j x_{jj}\ ; \quad \forall i \in N, j \in M, \tag{2.20}$$

$$x_{ij}\ , y_j \in \{0,1\} \qquad \forall i \in N, \forall j \in M \tag{2.21}$$

Eq. 2.17 represents the objective function. Eq. 2.18, Eq. 2.19 and Eq. 2.20 are analogous to Eq. 2.8, Eq. 2.9 and Eq. 2.10, respectively with only difference that Eq.

2.20 brings about capacity limitation for the opened facilities. Finally, Eq. 2.21 impose sign restrictions to variables.

2.2.3 Obnoxious P-Median Problem

In the early 1970s, the emergence of stricter environmental standards and the awareness of the dangers of pollution led to new methodologies in settlement of environmentally harmful facilities. Obnoxious facilities are undesired facilities desired to locate as far as possible from the customers (Lin and Guan, 2018). The obnoxious p-Median problem (OpMP) aims to position serving points in a way that maximizes the sum of the distances between customers and the nearest facility out of a given set of possible locations.

Church and Garfinkel (1978) led the beginning of the undesirable facility location problem. The maxian problem is nearly the same as the median problem, except its objective is maximizing, not minimizing. In summary, the median problem tries to find a location close some points, while the maxian problem tries to find locations as far from these points as possible.

Let I denote the set of clients and J denote the set of facilities. There is $d_{ij}(i,j) \in I\ x\ J$ distance between each facility and customer. A client always allocated to the closest open location. To 0-1, if $x_{ij} = 1$ than that means client i is served by by facility j, otherwise $x_{ij} = 0$.

For each $j \in J$,

$$y_j = \begin{cases} 1\ if\ a\ facility\ located\ at\ j \\ 0\ otherwise \end{cases}$$

For each $i \in I,\ j \in J$

$$x_{ij} = \begin{cases} 1\ if\ client\ i\ is\ allocated\ to\ j \\ 0\ otherwise \end{cases}$$

$$S_{ij} = \{k \in j \,|(d_{ik} > \ d_{ij}) \ or \ ((d_{ik} = d_{ij}) \ and \ (k > j))\} \tag{2.22}$$

$$Max \sum_{i \in I} \sum_{j \in J} c_{ij} \, x_{ij} \tag{2.23}$$

$$\sum_{j \in J} y_j = p \tag{2.24}$$

$$x_{ij} \ \leq \ y_j \qquad \forall i \in \ I, \forall j \ \in J \tag{2.25}$$

$$y_j + \sum_{k \in S_{ij}} x_{ik} \ \leq 1 \qquad \forall i \in \ I, \forall j \in \ J \tag{2.26}$$

$$x_{ij} \, , y_j \ \in \{0,1\} \qquad \forall i \in \ I, \forall j \in \ J \tag{2.27}$$

Eq. 2.22 represents client *i* always served by its nearest facility, so S_{ij} can be defined to record the sites that can't serve client *i* when site *j* is selected. Eq. 2.23 denotes that the objective function to be maximized. Eq. 2.24 certifies that exactly p facility locations are chosen. Also set *J* covers all potential nodes. Eq. 2.25 certifies that the clients can only assigned to serving plants. Eq. 2.26 guarantees that assigning all clients to the nearby serving facility. Eq. 2.27 specifies the discreteness of the problem variables.

CHAPTER THREE
LITERATURE REVIEW

Table 3.1 Tabular form of literature review

Year	Researcher	Method
1996	Rolland et al.	TS
1997	Rosing and ReVelle	Heuristic Concentration
2003	Alp and Erkut	GA
2004	Correa et al.	GA
2006	Fathali and Jafar	GA
2007	Yapicioglu et al.	PSO
2008	Al-Khedhairi	SA
2011	Li et al.	Scatter Search
2012	Özçakar and Bastı	PSO
2014	Kromer and Platos	DE
2016	Colmenar et al.	GRASP
2018	Lin and Guan	PSO
2019	Montaya et al.	TS
2019	Mladenović et al.	VNS

P-median problems still grab attention of researchers. The complexity issue of this problem leads researchers to make optimization studies with metaheuristic algorithms rather than deterministic methods that give exact solutions. Since the PMP is the category of NP-hard problems, studies with metaheuristic methods come to the fore in problem-solving. In Table 3.1, illustrates the studies in the literature that make use of heuristic and metaheuristic methods in solving the P-median problem and its variants.

In 1996, Rolland et al. (1996) develop a new algorithm based on TS principles to solve the PMP. The purpose here is to compare their solutions to some other heuristics. The proposed algorithm uses strategic oscillation and random tabu list sizes. It is evaluated by using test problems ranging up to 500 nodes. As a result, it has been observed that the TS method is effective in developing heuristic algorithms for the P-median problem.

In the late ’90s, Rosing and ReVelle (1997) develop a heuristic concentration algorithm for the PMP. Their purpose is to provide an alternative heuristic procedure to generate better solutions than existing methods when dealing with large problems.

In the early '00s, Alp and Erkut (2003) propose a new genetic algorithm (GA) for PMP. The encoding mechanism in this study makes use of a technique that the genes in its chromosome corresponds to the indices in the distance matrix of the facilities selected as the medians. Rather than traditional crossover methods, greedy search heuristics are included.

One year later, Correa et al. (2004) adapt the CPMP problem to a real-life problem in their study. Their proposed method includes well-known GA operators and a new heuristic namely, the hypermutation heuristic. The efficiency of the algorithm is evaluated by comparing it to the TS algorithm.

In 2006, Fathali (2006) proposes a GA to solve an extension of PMP. The results of the developed GA are compared to results of the variable neighborhood search method. It is reported that the results of the proposed method can achieve better result.

After that, Yapicioglu et al. (2007) develop a bi-objective model, and they propose Particle Swarm Optimization (PSO) to solve it. The first function is reported as the

standard weighted minisum and the latter one is developed to model undesirable effects with minisum distance.

One year later, Al-khedhairi (2008) uses the Simulated Annealing (SA) Algorithm for the PMP. According to the reported results, author finds the optimum solutions for 33 instances out of 40 problems. As reported, one of the disadvantages of this algorithm is that it does not have any mechanisms to avoid previously visited solutions.

A few years later, in their study, Li et al. (2011) use a hybrid GA and try to increase the performance of the used algorithm by adopting random initial population generation strategies. They use the greedy search method for a high-quality initial populations.

One year later, Özçakar and Bastı (2012) study the performance of the PSO to solve the PMP. The performance of the algorithm is tested on all available datasets reported in the literature. The algorithm results are compared to various metaheuristics such as GA, SA and TS.

In 2014, Krömer and Platos (2014) propose a Differential Evolution (DE) for the PMP. DE is one of the well-known evolutionary optimization methods with successful applications in many fields. Solutions are expressed as real-valued vectors in this method, which is population-based algorithm that uses evolutionary operators such as crossover and mutation. The performance of the algorithm is tested on the OR-Library problem set.

Later, Colmenar et al. (2016) study the PMP and employ a Greedy Randomized Adaptive Search Procedure (GRASP). By adding the local search procedure to the proposed method, the method's efficiency is tried to be increased. The outcomes of the offered approach and the test problems in the related literature are compared.

Two years later, Lin and Guan (2018) propose a binary PSO for solving OpMP. They develop a TS-based mutation operator and a position updating rule to generate solutions. The results bring out that the developed algorithm finds high-quality solutions in short processing time.

In 2019, Montaya et al. (2019) develop a TS algorithm to solve the CPMP. The reported method is based on TS, which generates a random initial solution. It is demonstrates that it performs well in achieving optimum solutions.

In the same year, Mladenović et al. (2019) implement less is more approach (Costa et al., 2017) to solve OpMP. They use a modified variable neighborhood search to solve OpMP and they achieve promising results.

As reported earlier, in the present thesis, the UFLP and the UPMP are solved by using a bio-inspired metaheuristic algorithm, namely, FPA. As far as we know, the performance of this algorithm has not been tested on these problems yet. Moreover, as given in the upcoming sections, FPA is further enhanced by several procedures can run in binary spaces and to achieve a more efficient search. The details regarding the mentioned procedures are presented next.

CHAPTER FOUR
PROPOSED SOLUTION APPROACHES

Metaheuristic algorithms are proposed to solve mostly NP-hard problems that traditional exact solution methods cannot solve in polynomial time. Unlike heuristic methods, these algorithms are not tailor-made for problems. They are flexible and contain many local or advanced search methods. Metaheuristic algorithms are stochastic methods, and they are widely used in solving combinatorial, that is, problem types where the variables of the problem can take discrete values.

Metaheuristic algorithms can be classified according to their underlying procedures. For example, they can be classified into two main categories as single-point (neighborhood-based, trajectory-based) and population-based search methods. Decisions about which method to use depends on the optimization problem being attacked. In the literature, there are many types of metaheuristics inspired by nature (Binitha and Sathya, 2012). Such bio-inspired strategies have grabbed attention particularly in the last decade. The FPA, which we will use as a solution method in this study, is an example for these methods.

4.1 Canonical Flower Pollination Algorithm

FPA (Yang, 2012) is a metaheuristic, which is inspired by the pollination process of flowers. There are various flowering plants in nature, and they exist many different means of pollination to achieve the purpose of reproduction. In general, pollination methods can take two forms, namely self-pollination and cross-pollination. Cross-pollination (biotic) depends on living existences like birds and insects at long distances. This type of pollination is used as a global search in FPA. On the other hand, self-pollination (abiotic) takes place at a shorter distance, which inspires the local search process of FPA. To sum up, as global search and local search, FPA has two primary and simple mechanisms, and according to a switching probability, the algorithm either switches from global to local or vice versa.

Before diving into details of the developed procedures in this thesis, the canonical FPA is presented first. In the global search procedure of FPA, solution vectors

implement Lévy flight, following the best-found position. Thus, the movement pattern resembles living beings like bees and insects. As the second procedure of FPA, self-pollination takes place as random changes on the solution vectors. One can control these search patterns via a parameter referred to as switch probability denoted by $p \in (0,1)$. While greater values of p encourage global search, lower values yield to more frequent local searches. More formally, all movements of FPA are formulated as given in the following equations.

$$x_i^{t+1} = x_i^t + \mathrm{L}\,(g^* - x_i^t) \tag{4.1}$$

$$L \sim \frac{\lambda \Gamma(\lambda) \sin(\pi\lambda/2)}{\pi s^{1+\lambda}}, (s \gg s_0 > 0) \tag{4.2}$$

$$x_i^{t+1} = x_i^t + \in (x_j^t - x_k^t) \quad k \neq j \tag{4.3}$$

Where g^*,x_i^t, L, and ϵ represent the best-found solution, the *i*th solution vector at the *t*th generation, the Lévy distribution, and a random vector, respectively. Eq. 4.1 here represents the global pollination, while Eq. 4.3 is the local pollination. Eq. 4.2 is employed along with Eq. 4.1, and $\Gamma(\lambda)$ is represents the standard gamma function. To better understand the mechanisms in the canonical FPA, a pseudo-code is presented by Figure 4.1 and flowchart is presented by Figure 4.2.

There are various FPA related studies in the reported literature. Lately, researchers have conducted extensive studies on FPA. Inspired by pollination, Yang (2012) propose a meta-heuristic algorithm in 2012 called the FPA. In the last eight years, FPA has been researched extensively. In 2015, Dubey et al. (2015) propose a hybrid FPA for wind in integrated multi-objective dispatch. During that year, Abdel-Baset et al. (2015) propose a new algorithm to answer constrained optimization problems which are hybrid of FPA and GA to improve the accuracy of searching. Later, Draa (2015) propose various modifications by explicitly employing opposition-based learning and search procedures. After that, Nabil (2016) present a modified FPA called the opposition-based learning flower pollination algorithm (OFPA). This algorithm qualitatively and quantitatively analyzes the FPA with the proposition of 5 variants. Three year later, Zhou et al. (2019) present a discrete greedy FPA based on the

integration between pollen discarding behavior and order-based crossover to get the global pollination process. In same year, Fouad et al. (2019) propose a novel modified FPA based on novel local and global pollination configurations to get the best solution exploration. Subsequently, Ozsoydan and Baykasoğlu (2019) analyze the effects of switch probability of FPA.

Set $f(x)$, $x = (x_1, x_2, \ldots, x_d)$
Create initial population with n solutions
Define $\boldsymbol{g}_*$ (best-found-solution)
Determine the switching probability $prob \in [0, 1]$ betwixt local and gloabal pollination
while (t <MaximumGeneration)
  for i = 1 : n
    if rand < $prob$
    Generate Lévy vector L
    Apply the global pollination with $x_i^{t+1} = x_i^t + L\,(g^* - x_i^t)$
    else
    Generate a random vector
    Apply the local pollination with $x_i^{t+1} = x_i^t + \in (x_j^t - x_k^t)$
    end if
  Evaluate new solutions
  Update them if necessary
  end for
  Update the $\boldsymbol{g}_*$
end while

Figure 4.1 Pseudo code of the original FPA

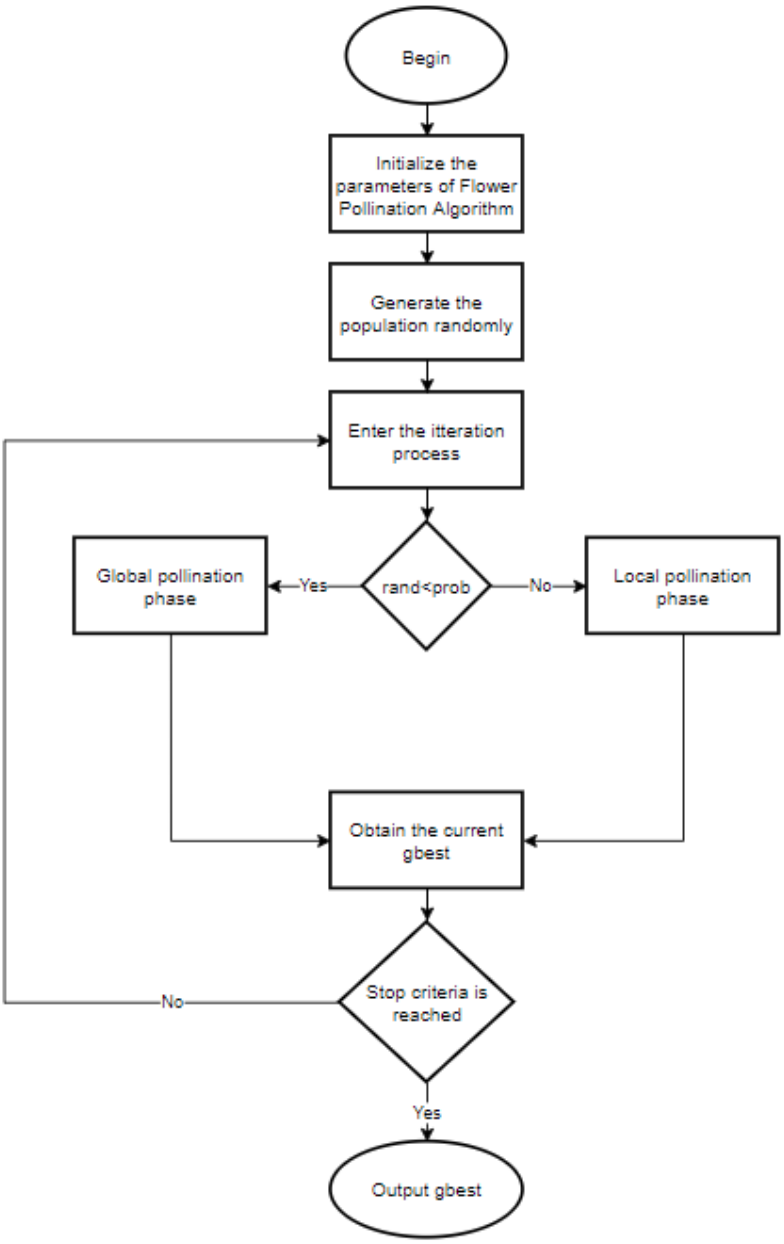


Figure 4.2 Flowchart of FPA

4.2 Proposed Modifications

It is clear that the canonical FPA is proposed for real-valued domains. Thus, adopting this algorithm to binary domains as in the present thesis might require additional procedures such as employing sigmoid or any types of transfer functions.

Most of these procedures use probability distributions. It further means that two binary vectors derived from the same real-valued vector do not necessarily have to be

exact. This undesirable circumstance brings about the spatial distance between these vectors. Therefore, although such transfer functions are commonly used in numerous studies, their performances tend to be poor in most cases. In this regard, the present paper adopts some evolutionary operators to use them in the notion of FPA.

4.2.1 Solution Encoding

In order to represent solution vectors, binary encoding technique is used in this thesis. According to this technique, the length of a vector is of size *n*, where *n* represents the number of candidate facility location points on the map. Next, a number of *p* bits in this vector takes 1 value representing that a facility is opened on the related candidate location, while the remaining 0s represent that a facility is not opened on the related location. An example for a solution vector for *p*=3 and *n*=10 is given by Figure 4.3. According to this figure, facilities (medians) are opened on the 1se, 3rd and 7th candidate locations.

	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Solution Vector	1	0	1	0	0	0	1	0	0	0

Figure 4.3 Binary encoding technique

4.2.2 Uniform-Based Xover for Inheritance

As mentioned earlier, one of the notable advantages of FPA is that it has a small number of parameters. One of the parameters used in FPA is the switching probability that is denoted by *prob* (Algorithm 1, line: 5) and this parameter defines the search characteristic. If *prob* is increased, FPA tends to exploit around the best found solution. Lower values for *prob* on the other hand, yield to making use of randomly selected solution vectors from the population more frequently. However, according to the Eqs. 4.1-4.3, these procedures cannot be directly used in binary domains. Consequently, in this thesis, a uniform-based xover is adopted to inherit information between individuals.

In this procedure, any *j*th solution vector from population undergoes a uniform-based xover with a parent vector that is either a random vector or the best-found

solution vector. For each dimension of solution vectors, a random number $rand \in (0,1)$ is generated and it is compared to a user-supplied parameter $xOverProb \in (0,1)$. If $rand \leq xOverProb$ allele of the corresponding locus is inherited from the jth vector. Otherwise, it is inherited from the parent vector, which probably yields to change of the corresponding bit. Thus, one can conclude that lower values of $xOverProb$ results in dampening the changes on the bits. In other words inertia is encouraged in lower values of $xOverProb$. Greater values on the other hand yield to possibly changing the related loci. An illustration of this procedure is presented by Fig. 4.4. As one can see from this figure, solution vectors are of size 10 bits. Thus, 10 $rand$ values are to be generated. Let's assume that these random numbers appear to be 0.50, 0.10, 0.80, 0.70, 0.05, 0.90, 0.15, 0.45, 0.75, 0.80, for each dimension respectively. Given that $xOverProb$ is set to be equal to 0.60, new solution vector, denoted by offspring is generated.

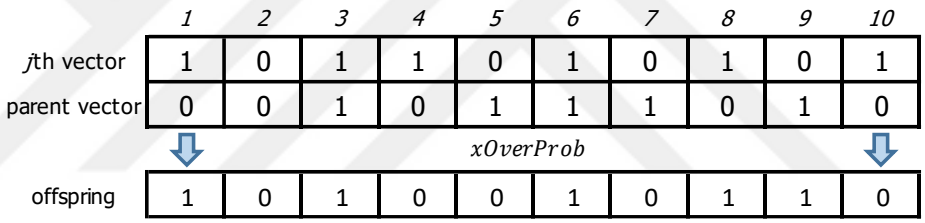


Figure 4.4 Uniform-based xover

It should be kept in mind that the variables of $xOverProb$ and *prob* should be carefully chosen. Since, greater values of p and lower values of $xOverProb$ yield to a greater exploitation of the top-found solution. There is no doubt that overuse of this solution vector might yield to a premature convergence and local optima issues. On the contrary, lower values of *prob* and greater values of $xOverProb$ yield to a more random and slower search that might also result in a poor convergence. Thus, one can conclude that now both of these parameters affect the search characteristic of the proposed FPA modification.

4.2.3 Bit-Flip Mutations to Avoid Local Optima

Subsequent to xover, a bit-flip mutation is performed. The aim here is to introduce diversity so as to avoid local optima problems. In this procedure, randomly selected bits of a solution vector are flipped. It is clear that greater number of flipped bits introduce greater level of diversity. This becomes useful whether the population loses diversity. However, high level of mutation also prevents convergence and algorithm tends to perform random search. Lower level of mutation yields to a better exploitation of the found regions, however, continuously use of lower level of mutation might result in loss of diversity and a slower search, where local optima issues might arise. Therefore, an adaptive mutation that encourages the number of mutations at earlier iterations and discouraging mutation gradually over iterations is adopted (Ozsoydan, 2019; Baykasoglu et al., 2020). In other words, a greater level of diversity at earlier iterations along with an decreased level of mutation towards the end of the search so as to encourage exploitation, is satisfied. Thus, local optima issues at earlier iterations are avoided while providing a more intensified search over iterations.

$$stepSize^{t+1}=stepSize^{t}\text{-}e^{-t/(t+1)} \times \phi \times stepSize^{t} \tag{4.4}$$

$$mutSize^{t+1} = \lceil stepSize^{t+1} \rceil \tag{4.5}$$

```
for m = 1: mutSize^t
    select a random locus ω
    while locus ω is already chosen
        select a random locus ω
    end while
    offSpring_ω = 1 − offSpring_ω
end for
```

Figure 4.5 A pseudo code for the mutation operator

This procedure is formulated by Eq. 4.4, where $stepSize^{t} \in [1, nDim]$ and $\phi \in (0,1)$ denote the step size at the tth iteration and the severity of the decrement, respectively. Therefore, both $stepSize^{1}$ and ϕ are user supplied parameters. Greater values of ϕ induce steeper decrements in step size, whereas lower values yield to a smoother decrement. Moreover, it is clear that the step size values are continuous. In

order to make them to represent the number of bit-flips on a solution vector, ceiling function is used (Eq. 4.5). Subsequently, a number of randomly selected $mutSize^t$ bits of a solution at the tth iteration is flipped. A pseudo code and an illustration for the mentioned mutation operator for $stepSize^1$=30 are presented by Fig 4.5 and Fig. 4.6, respectively.

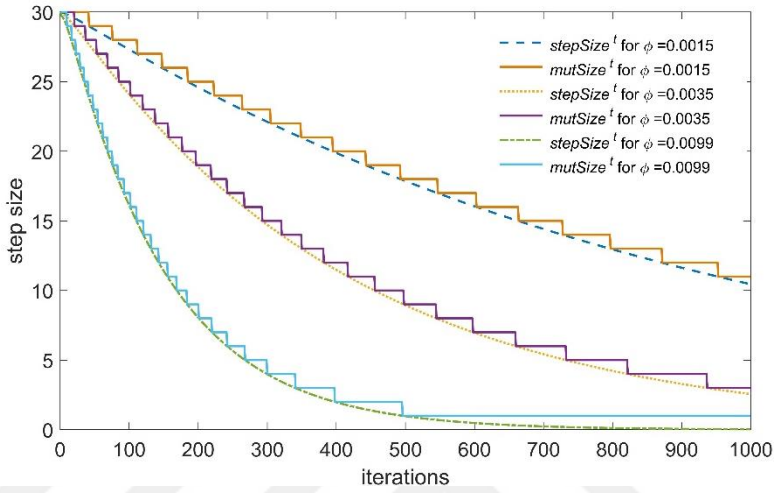


Figure 4.6 An illustration of various step size values

4.2.4 Greedy Repair Procedure

A formed solution vector might not meet the median capacity constraint subsequent to uniform-based xover, and bit-flip mutation operations. If the number of the opened facilities is not equal to *p*, it means that the solution is infeasible. In this study, two different methods are used to repair this possible infeasibility. The first method randomly decreasing or increasing the number of opened facilities without using any problem data. The second method on the other hand makes of problem data and it uses the greedy repair algorithm, which is proposed by Lin and Guan (2018). In both methods, new facilities (medians) are added to the solution vector if the number of opened plants is less than *p*. On the contrary, if *p is* exceeded, a number of opened facilities are removed from the solution vector. While the first method removes those medians at random, the greedy repair procedure makes of problem data. For a better understanding, a pseudo code for both methods are presented by Fig. 4.7 and Fig. 4.8,

respectively. Finally, bringing all modifications together, a pseudo code for the proposed binary FPA (bFPA) is presented by Fig. 4.9.

$$add_loss(y,x) = f(x) - f(x') \tag{4.6}$$

$$T_i = \min(d_{iy}, j \in T_x) \tag{4.7}$$

$$T_i' = \min(T_i, d_{iy}) \tag{4.8}$$

$$add_loss(y,x) = \sum \min\{0, (d_{iy} - T_i)\} \tag{4.9}$$

$$remove_gain(y,x) = f(x') - f(x) \tag{4.10}$$

$$remove_gain(y,x) = \sum (\min\{d_{iy}, y \in T_x\{j\}\} - T_i) \tag{4.11}$$

```
Choose a solution vector to repair
while the number of opened facilities is greater than p
    Select an opened facility at random
    Change the value of the related facility to 0
    p = p − 1
end while
while the number of opened facilities is less than p
    Select an empty location at random
    Change the value of the related location to 1
    p = p + 1
end while
```

Figure 4.7 A pseudo code for the random repair procedure

```
Define total open facility sum T_x
Define population vector (Pop_vec)
Define objective functions value f(x)
while (T_x < p)
Calculate losses through eq.4.6 and find minimum according to eq.4.7
Calculate add_loss according to eq.4.8
Let i_z = 1, and T_x = T_i ∪ {z} (Increasing the value 0 to 1 at Pop_vec)
Update T values
end while
while (T_x > p)
Calculate losses through eq.4.11
Let x_z = 0, and T_x = T_x\{z} (Decreasing the value 1 to 0 at Pop_vec)
Update T values
end while
```

Figure 4.8 A pseudo code for the greedy repair procedure

```
Generate popSize vectors
evaluate f(x_i), i = 1, ..., n
while (t < maxIt)  // maxIt: maximum number of iterations
    calculate stepSize^t and mutSize^t
    for j = 1: popSize // popSize is the population size
        if rand < p
            offSpring=xOver(pop_j, g*)
        Else
            offSpring=xOver(pop_j, pop_k)  // j ≠ k
        end if
        offSpring=mutation(offspring, mutSize^t)
        apply repair procedure
            evaluate offSpring
        update it if necessary
    end for
    update g* if necessary
end while
// pop_k is any solution vector randomly selected from the population.
```

Figure 4.9 A pseudo code for the proposed bFPA

CHAPTER FIVE
EXPERIMENTAL STUDY

5.1 Calibration of Parameters

All assessments are completed on a computer with an i5-4590 CPU and 32 GB of RAM. Termination criterion is used as the maximum number of iterations, denoted by $maxIt$. In parallel with the related literature, the parameters $maxIt$ and population size ($popSize$) are fixed to 1000 and to the number of locations (n) for the related instance, respectively. As another parameter, $xOverProb$ represents inertia.

5.1.1 Calibration of Parameters for UFLP

For UFLP, according to the preliminary tests, it is overserved that improved outcomes can be gained when $xOverProb$ is fixed to 0.20. Therefore, this parameter is used as 0.20 for all tests. There exist two additional parameters that define the mutation characteristic in the proposed approach. They are denoted by ϕ and $stepSize^1$, representing the speed of decrement in the step sizing function and the initial step size, respectively. With respect to the preliminary tests again, it is observed that ϕ =0.0025 and $stepSize^1$ =0.10× n yield to more promising solutions. Therefore, the parameters ϕ and $stepSize^1$ are fixed to 0.0025 and 0.10× n, correspondingly for all tests. The last constraint to be calibrated is the switching probability of FPA that is denoted by *prob*. This parameter is fixed to 0.75 for all tests, which is close to the value proposed by Yang (2012).

5.1.2 Calibration of Parameters for UPMP

For the UPMP a mid-level (n *x* m) sized problem picked to determine the parameters. After careful computations; $xOverProb = 0.20$, $\phi = 0.025$ and $stepSize^1 = 0.01$ parameters are determined. For comparison and better understanding for this determination, Table 5.1 and Table 5.2 is presented.

Table 5.1 Determination of the paramters phiStep and initRate

	phiStep									
	0.001		*0.0025*		*0.005*		*0.0075*		*0.01*	
initRate	*hit*	*best(mean)*	*hit*	*best(mean)*	*hit*	*Best(mean)*	*hit*	*best(mean)*	*hit*	*best(mean)*
0.01	0	2940 (2970.87)	0	2889 (2922.33)	0	2880 (2902.93)	0	2879 (2899.70)	0	2875 (2897.93)
0.025	0	3063 (3097.77)	0	2935 (2969.50)	0	2883 (2913.03)	0	2879 (2902.97)	0	2871 (2902.37)
0.05	0	3276 (3272.27)	0	2990 (3059.50)	0	2883 (2928.33)	0	2879 (2913.80)	0	2881 (2900.53)
0.075	0	3324 (3381.97)	0	3094 (3148.80)	0	2905 (2949.17)	0	2890 (2917.37)	0	2887 (2911.03)
0.1	0	3357 (3472.47)	0	3156 (3226.10)	0	2933 (2970.77)	0	2884 (2915.30)	0	2875 (2913.57)

Table 5.2 Determination of the paramters xOverProb and prob

	xOverProb									
	0.2		*0.4*		*0.5*		*0.6*		*0.7*	
prob	*hit*	*best(mean)*	*hit*	*best(mean)*	*hit*	*Best(mean)*	*hit*	*best(mean)*	*hit*	*best(mean)*
0.2	0	2890 (2917.40)	0	2894 (2919.73)	0	2893 (2924.27)	0	2899 (2926.53)	0	2895 (2934.23)
0.025	0	3063 (3097.77)	0	2935 (2969.50)	0	2883 (2913.03)	0	2879 (2902.97)	0	2871 (2902.37)
0.35	0	2873 (2913.07)	0	2888 (2918.63)	0	2892 (2917.47)	0	2898 (2923.10)	0	2910 (2936.87)
0.075	0	3324 (3381.97)	0	3094 (3148.80)	0	2905 (2949.17)	0	2890 (2917.37)	0	2887 (2911.03)
0.5	0	2881 (2903)	0	2897 (2918)	0	2887 (2921.87)	0	2880 (2921)	0	2887 (2928.17)

5.2 Computational Results for UFLP

All found outcomes for UFLP are given by Table 5.3 and the values in Table 5.3 are calculated over 30 independent replications. The rows best, mean, worst, std. and hit in this table represent the best-found result by the proposed algorithm, mean of all replications, worst-found result, standard deviances of the best-found results over all replications and the number of replications that the best-known solution is achieved, respectively. According to the results of small-scaled instances of size 16 × 50, most of the algorithms exhibit competitive performances. It is clear that particularly more recently published algorithms ABCbin, bWSA, PSO, prioGWO, learnGWO, prLeGWO and the proposed algorithm bFPA are capable of finding the most promising hit values in these instances.

Results of medium-scaled instances of size 25 × 50 point out that the difference in performances of all compared algorithms become more apparent in these cases. Clearly only the suggested algorithm bFPA reaches the best-known solutions in all replications for all instances of medium-scaled benchmarks. bFPA is followed by prLeGWO, which is capable of achieving a hit value of 26 in cap103. In this regard, the algorithm prLeGWO is followed by learnGWO, which shows a better performance in contrast to the rest of the compared algorithms.

Difference between the compared algorithms become even more apparent in large-scaled instances of size 50 × 50. Except for the instance cap133, where the proposed algorithm achieves a *hit value* of 26, bFPA is able to find the best-known solutions in all replications in the remaining instances of the same size. This can be considered as a remarkable performance since the closest algorithm in terms of *hit values*, prLeGWO is clearly outperformed by the proposed algorithm. In parallel, bFPA is superior to the compared algorithms in large-scaled instances in regard to the *hit values*. Moreover, the standard deviance values of all compared algorithms for cap133 point out that the proposed algorithm exhibits a more robust performance in this instance. Additionally, the standard deviation values obtained by the proposed algorithm bFPA are often better than those found by other algorithms. To sum up, according to the results of Table 5.3, one can conclude that the proposed algorithm bFPA obtains the best-known solutions in all replications for all instances of UFLP except for the cap133. Nevertheless, it is clear that even in this instance it is able to find the most promising results in comparison to the rest of the compared algorithms. Finally, convergence plots of the proposed algorithm for all instances are depicted by Fig. 5.1. Data points in these plots are represented by the mean results of the related iteration over all replications. According to this figure, bFPA shows a faster convergence particularly in small-scaled and medium-scaled instances. It is clear for such instances that results that are similar to those given by Table 5.3 could be obtained also by using smaller maximum number of iterations. However, one should note that in such a case the parameters of the step sizing function (Eqs. 2.26-2.27) should be recalibrated according to the used maximum number of iterations. Additionally, it is apparent that the step sizing function seems to be performing efficiently since local optima issues are not apparent in these plots.

Table 5.3 Results for UFLP

Algorithms		*Instances*											
		16 × 50				25 × 50					50 × 50		
		cap71	cap72	caf73	cap74	cap101	cap102	cap103	cap104	cap131	cap132	cap133	cap134
CPSO	*best*	932615.75	977799.40	1010641.45	1034976.98	796648.44	854704.20	893782.11	928941.75	795291.86	851495.33	893076.71	928941.75
	mean	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*
	worst	934199.14	983713.81	1012643.69	1045342.23	802457.23	857380.85	899424.91	944394.83	804549.64	865667.16	909908.7	951803.25
	std.	562.23	1324.3	702.13	2124.54	1480.72	1015.64	1695.79	3842.64	2429.54	4297.07	4210.93	6619.05
	hit	25	25	22	0.00	0.00	10	0.00	18	0.00	0.00	0.00	7
ABCbin	*best*	932615.75	977799.40	1010641.45	1034976.98	796648.44	854704.20	893782.11	928941.75	793439.56	851495.33	893076.71	928941.75
	mean	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*
	worst	932615.75	977799.40	1010641.45	1034976.98	796648.44	854704.20	894008.14	928941.75	794910.64	851636.70	895407.93	928941.75
	std.	0.00	0.00	0.00	0.00	0.00	0.00	85.67	0.00	1065.73	213.28	561.34	0.00
	hit	**30**	**30**	**30**	**30**	**30**	**30**	25	**30**	6	14	5	**30**
bWSA	*best*	932615.75	977799.40	1010641.45	1034976.97	796648.43	854704.20	893782.11	928941.75	793439.56	851495.32	893076.71	928941.75
	mean	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*	*Na*
	worst	932615.75	977799.40	1010641.45	1034976.97	797508.72	854704.20	895027.18	928941.75	798449.03	852257.97	894801.16	934586.97
	std.	0.00	0.00	0.00	0.00	380.43	0.00	470.95	0.00	1025.78	251.65	501.91	1016.14
	hit	**30**	**30**	**30**	**30**	22	**30**	10	**30**	6	23	7	26
PSO	*best*	932615.75	977799.40	1010641.45	1034976.97	796648.43	854704.20	893782.11	928941.75	803826.01	879785.46	921608.03	944335.68
	mean	932615.75	977799.40	1010641.45	1034976.97	799006.37	857115.01	896226.99	933746.35	817976.18	889889.93	943216.76	1002804.33
	worst	932615.75	977799.40	1010641.45	1034976.97	801769.38	860573.81	900406.76	941871.31	824751.43	896935.53	957975.85	1035373.88
	std.	0.00	0.00	0.00	0.00	1085.48	1268.63	1763.88	3358.73	4545.68	4697.11	9163.94	19102.20
	hit	**30**	**30**	**30**	**30**	1	2	3	2	0	0	0	0
GWO	*best*	932615.75	977799.40	1010641.45	1034976.97	796648.43	854704.20	893782.11	928941.75	793439.56	851495.32	893076.71	928941.75
	mean	932615.75	978056.06	1010702.63	1034976.97	797123.96	854830.95	894083.99	928941.75	794606.70	852034.68	894088.39	929716.82
	worst	932615.75	981649.35	1012476.97	1034976.97	799144.68	855971.75	895027.18	928941.75	797570.30	854704.20	896529.71	935122.70
	std.	0.00	976.76	335.11	0.00	772.61	386.76	480.63	0.00	1292.51	789.94	1006.16	1783.31
	hit	**30**	28	29	**30**	19	27	20	**30**	11	13	5	20
prioGWO	*best*	932615.75	977799.40	1010641.45	1034976.97	796648.43	854704.20	893782.11	928941.75	793439.56	851495.32	893076.71	928941.75
	mean	932615.75	977799.40	1010641.45	1034976.97	796963.87	854788.70	894102.76	928941.75	794187.91	851735.81	893925.90	929612.32
	worst	932615.75	977799.40	1010641.45	1034976.97	797508.72	855971.75	895027.18	928941.75	796662.92	853419.13	895049.66	934586.97
	std.	0.00	0.00	0.00	0.00	421.65	321.58	459.7	0.00	1000.79	446.02	587.93	1699.92
	hit	**30**	**30**	**30**	**30**	19	28	15	**30**	15	17	6	21
learnGWO	*best*	932615.75	977799.40	1010641.45	1034976.97	796648.43	854704.2	893782.11	928941.75	793439.56	851495.32	893076.71	928941.75
	mean	932615.75	977799.40	1010641.45	1034976.97	796849.17	854704.2	894016.05	928941.75	794749.63	852229.93	894390.11	929834.29
	worst	932615.75	977799.40	1010641.45	1034976.97	797508.72	854704.2	894801.16	928941.75	798338.45	855005.57	897548.76	934586.97
	std.	0.00	0.00	0.00	0.00	370.08	0.00	406.54	0.00	1476.05	1212.19	1214.52	1786.97
	hit	**30**	**30**	**30**	**30**	23	**30**	20	**30**	10	16	5	20
prLeGWO	*best*	932615.75	977799.40	1010641.45	1034976.97	796648.43	854704.20	893782.11	928941.75	793439.56	851495.32	893076.71	928941.75
	mean	932615.75	977799.40	1010641.45	1034976.97	796648.43	854704.20	893782.11	928941.75	793781.44	851616.90	893717.00	928941.75
	worst	932615.75	977799.40	1010641.45	1034976.97	796648.43	854704.20	894801.16	928941.75	794373.41	853525.52	894801.16	928941.75
	std.	0.00	0.00	0.00	0.00	0.00	0.00	260.79	0.00	426.88	429.35	571.45	0.00
	hit	**30**	**30**	**30**	**30**	**30**	**30**	26	**30**	18	26	10	**30**
bFPA	*best*	932615.75	977799.40	1010641.45	1034976.975	796648.438	854704.20	893782.113	928941.75	793439.563	851495.325	893076.713	928941.75
	mean	932615.75	977799.40	1010641.45	1034976.975	796648.438	854704.20	893782.113	928941.75	793439.563	851495.325	893170.766	928941.75
	worst	932615.75	977799.40	1010641.45	1034976.975	796648.438	854704.20	893782.113	928941.75	793439.563	851495.325	893782.113	928941.75
	std.	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	243.89	0.00
	hit	**30**	**30**	**30**	**30**	**30**	**30**	**30**	**30**	**30**	**30**	26	**30**

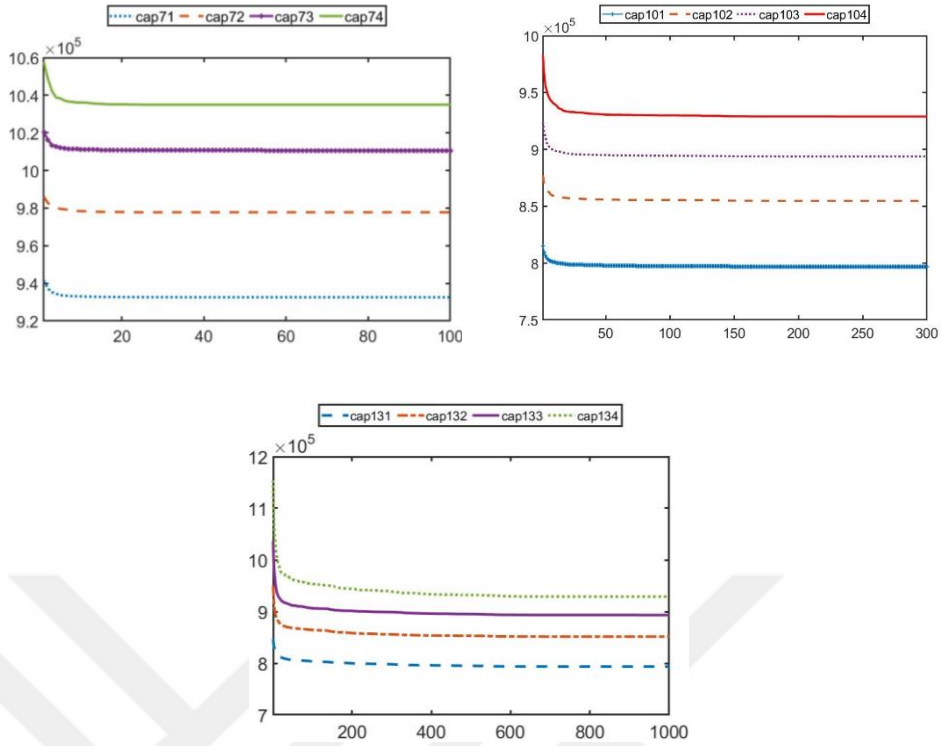


Figure 5.1 Convergence plots of bFPA

5.3 Computational Results for UPMP

All obtained results for UPMP are given in Table 5.4 and Table 5.5. The values in Table 5.4 and Table 5.5 are calculated over 30 independent replications except for greedy bFPA. Greedy bFPA calculated over 30 independent replications for the instances to pmed15 problem. After that greedy bFPA calculated over 20 independent replications due to the complexity of the greedy procedure. The rows *best*, *mean*, *worst*, *std.* and *hit* in this table represent the best-found result by the proposed algorithm, mean overall replications, worst-found result, standard deviances of the best-found results overall replications and the number of replications that the best-known solution is found, respectively. Moreover, the bFPA modification that uses the greedy repair procedure is denoted by gbFPA while the other modification with random repair is denoted by rbFPA in the rest of this thesis.

According to the results of small-scaled instances of size 100 × 100, two methods exhibit competitive performances. After pmed4, gbFPA performs better than the rbFPA. The algorithm gbFPA is able to find the best-known solutions except for

pmed19 and pmed25. As one can see from these tables gbFPA is found as superior to the rbFPA in all instances.

Convergence plots for pmed1, pmed5, pmed10, pmed15, pmed20 and pmed25 instances are depicted by Fig. 5.2. Data points in these plots are represented by the mean results of the related iteration over all replications. According to this figure, gbFPA exhibits a faster convergence than rbFPA in all instances.

All findings mention the efficiency of the suggested algorithm bFPA in UFLP and gbFPA in UMPM. However, for the purpose of prove whether significant improvements are achieved by this algorithm, a further statistical test is crucially needed. Therefore, the following subsection is dedicated to the statistical demonstrations.

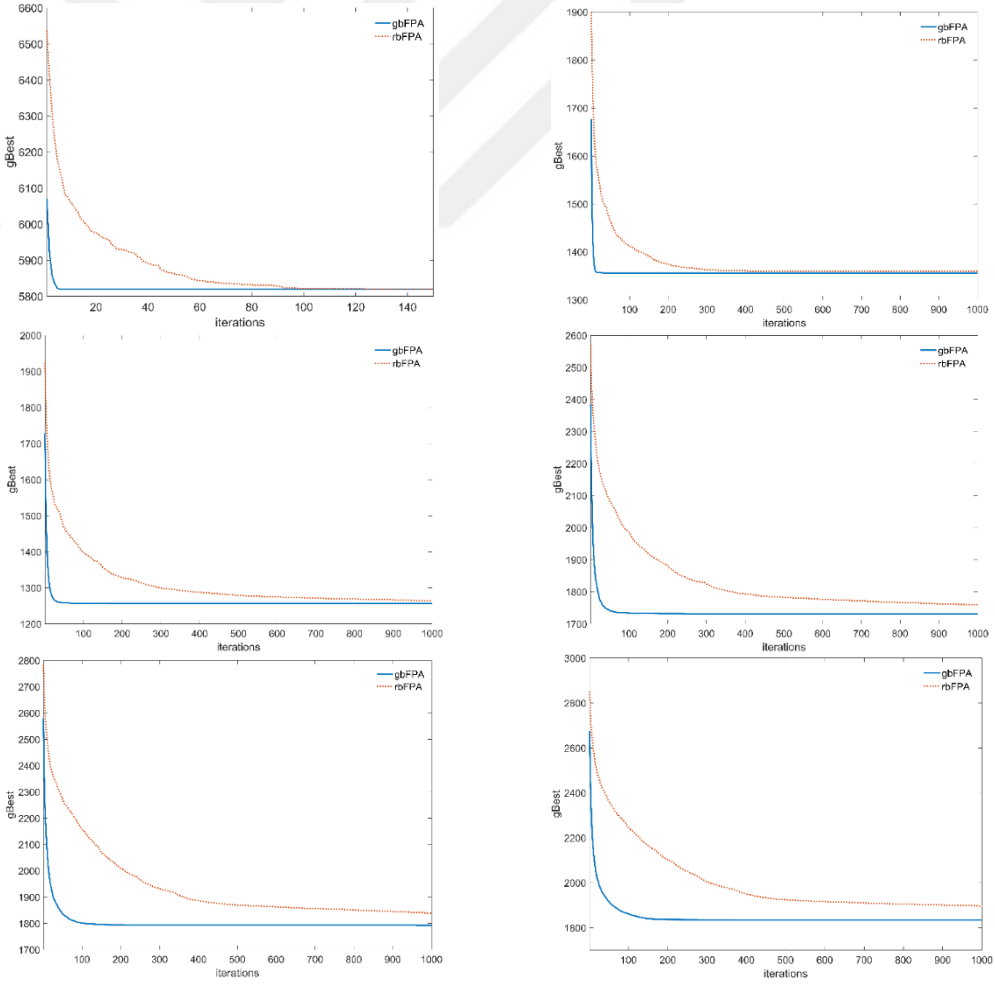


Figure 5.2 Convergence plots of gbFPA and rbFPA

Table 5.4 Results for UPMP

Optimizers		*Instances*																								
		5 x 100	10 x 100	10 x 100	20 x 100	33 x 100	5 x 200	10 x 200	20 x 200	40 x 200	67 x 200	5 x 300	10 x 300	30 x 300	60 x 300	100 x 300	5 x 400	10 x 400	40 x 400	80 x 400	133 x 400	5 x 500	10 x 500	50 x 500	100 x 500	167 x 500
	Pmed	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Optimum Results		5819	4093	4250	3034	1355	7824	5631	4445	2734	1255	7696	6634	4374	2968	1729	8162	6999	4809	2845	1789	9138	8579	4619	2961	1828
gbFPA	Best	5819	4093	4250	3034	1355	7824	5631	4445	2734	1255	7696	6634	4374	2968	1729	8162	6999	4809	2846	1789	9138	8579	4619	2961	1829
	Mean	5819	4097	4251	3036	1356	7824	5634	4447	2737	1256	7696	6634	4374	2970	1731	8162	7001	4813	2851	1792	9138	8579	4623	2969	1835
	Worst	5819	4105	4287	3046	1358	7824	5645	4454	2747	1262	7696	6634	4374	2987	1733	8162	7003	4823	2858	1798	9138	8579	4631	2977	1839
	Std.	0.00	5.66	6.64	4.09	1.41	0.00	5.11	3.98	5.01	1.77	0.00	0.00	0.00	4.02	1.39	0.00	1.99	3.92	3.08	2.23	0.00	0.00	3.49	3.99	3.01
	Hit	30	20	29	25	20	30	22	22	23	22	30	30	30	6	2	20	11	4	0	1	20	20	5	1	0
rbFPA	Best	5819	4093	4250	3034	1355	7824	5631	4445	2734	1255	7696	6634	4374	2979	1741	8162	6999	4809	2875	1800	9138	8579	4646	3005	1876
	Mean	5819	4098	4252	3042	1359	7824	5633	4452	2749	1265	7699	6634	4384	3002	1759	8166	7004	4840	2903	1839	9138	8590	4675	3041	1897
	Worst	5819	4117	4270	3056	1379	7824	5645	4496	2759	1283	7723	6634	4414	3049	1787	8183	7014	4867	2933	1857	9138	8689	4718	3076	1919
	Std.	0	6.56	5.07	6.58	5.89	0.00	5.43	10.46	6.83	7.02	8.10	0.00	10.07	18.18	11.92	7.83	5.23	13.95	14.03	11.41	0.00	28.59	19.28	16.34	11.27
	Hit	30	19	24	5	9	30	16	16	3	2	27	30	8	0	0	25	11	1	0	0	30	22	0	0	0

Table 5.5 Computational results for P-Median instances 26-40

Algorithms		Instances (facilities x demand points)														
		5 x 600	10 x 600	60 x 600	120 x 600	200 x 600	5 x 700	10 x 700	70 x 700	140 x 700	5 x 800	10 x 800	80 x 800	5 x 900	10 x 900	90 x 900
	pmed	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
Optimum Results		9917	8307	4498	3033	1989	10086	9297	4700	3013	10400	9934	5057	11060	9423	5128
rbFPA	Best	9917	8307	4527	3088	2042	10086	9297	4766	3089	10400	9934	5138	11060	9423	5210
	Mean	9918.9	8308.8	4572.6	3134.2	2069.6	10086.3	9298.9	4806.3	3122.3	10400	9952.8	5180.3	11063.4	9433.1	5264.5
	Worst	9924	8316	4640	3173	2094	10087	9313	4856	3164	10400	9989	5266	11105	9472	5309
	Std.	3.096	2.146	23.15	19.97	Ara.60	0.471	3.222	23.988	19.448	0.000	15.643	25.766	11.324	15.148	28.763
	Hit	22	15	0	0	0	20	19	0	0	30	8	0	27	17	0

5.4 Statistical Analysis

Generally, either parametric or nonparametric statistical tests can be used to compare multiple optimizers since, to be able to use parametric tests, the requirements for the independence, normality and homoscedasticity should be met, which is not the case in the present study (Derrac et al., 2011). In cases, where these assumptions cannot be achieved, nonparametric tests can be used. Accordingly, nonparametric tests are adopted here. In this context, the nonparametric test, namely, Friedman Test is applied first so as to demonstrate whether a single pair with significant differences exists in terms of *hit values*.

The results of the Friedman Test and obtained average ranks, where the best algorithm is assigned by the *1*st rank for the related instances, are given for UFLP by Table 5.6. According to the *p*-value of this test, which is adjusted for ties, the null hypothesis H_0 that is based on the equivalence of medians should be rejected for the significance level α=0.10. This shows that at least one pair of algorithms is significantly different from each other. Thus, a further analysis to find the significantly different algorithms should be conducted.

Table 5.6 Results of friedman test for UFLP

Algorithms	*overall avg. ranks*
CPSO	8.625
PSO	7.125
GWO	5.875
prioGWO	4.625
learnGWO	4.583
bWSA	4.375
ABCbin	4.125
prLeGWO	3.000
bFPA	**2.667**
p-value (*adjusted for ties*)	0.000
decision on H_0	*reject* H_0

Since the aim of this study is to demonstrate the efficiency of the suggested algorithm bFPA, a $1 \times N$ design rather than $N \times N$, is adopted. In other words, for UFLP, bFPA is compared to all remaining algorithms according to the average ranks over all instances. For UPMP, Greedy bFPA is compared to Random bFPA according

to the average ranks over all instances. One crucial issue here is the possibility of losing control on the family-wise error rate while comparing the algorithms (Derrac et al., 2011).

To avoid this, the Benferroni procedure is employed as a post-hoc test. In this context, the z-scores, unadjusted *p*-values and adjusted *p*-values according to the Benferroni procedure are given in Table for UFLP

According to the unadjusted *p*-values of Table 5.7, the proposed algorithm is found as significantly better than CPSO, PSO, GWO, prioGWO and learnGWO in terms of *hit values* for the significance level α=0.10. According to the adjusted *p*-values on the other hand, bFPA is significantly better than CPSO, PSO and GWO. In other words, although the proposed algorithm shows the most promising performance according to Table 5.6, Table 5.7 points out that there is not enough statistical evidence to demonstrate the significant difference of bFPA from prioGWO, learnGWO, bWSA, ABCbin and prLeGWO. This might be due to the small number of available benchmark data in dataset of UFLP.

Table 5.7 Post-hoc results for UFLP

bFPA *vs.*	*z-score*	*unadjusted p-values*	*adjusted p-values*
CPSO	5.329	0.000 (+)	0.000 (+)
PSO	3.988	0.000 (+)	0.000 (+)
GWO	2.870	0.004 (+)	0.033 (+)
prioGWO	1.752	0.079 (+)	0.639 (~)
learnGWO	1.714	0.086 (+)	0.692 (~)
bWSA	1.528	0.127 (~)	1 (~)
ABCbin	1.304	0.192 (~)	1 (~)
prLeGWO	0.298	0.766 (~)	1 (~)

Table 5.8 Comparisons results of gbFPA and rbFPA for sign test

#	Greedy	Random	Wins (+) / Loses (-)	
1	30	30	0	~
2	20	19	1	+
3	29	24	5	+
4	25	5	20	+
5	20	9	11	+
6	30	30	0	~
7	22	16	6	+
8	22	16	6	+
9	23	3	20	+
10	22	2	20	+
11	30	27	3	+
12	30	30	0	~
13	30	8	22	+
14	6	0	6	+
15	3	0	3	+
16	30	25	5	+
17	16,5	11	5,5	+
18	6	1	5	+
19	0	0	0	~
20	1,5	0	1,5	+
21	30	30	0	~
22	30	22	8	+
23	7,5	0	7,5	+
24	1,5	0	1,5	+
25	0	0	0	~
	N		19	
	Wins (N+)		19	
	Loses (N-)		0	
	p-value		0.064	
	decision on H0		*reject H0*	

Since the conditions of the previous statistical test for UFLP are not identical with those of the UPMP's, Sign Test is applied here to show if there exists significant difference between gbFPA and rbFPA in terms of *hit values*. In Sign Test, N value, wins (N+) and loses (N-) are found first. Wins (N+) and loses (N-) are identified according to the *hit values*. N is equal to the sum of the wins (N+) and loses (N-). The results of this procedure is given by Table 5.8.

According to wins (N+) and loses (N-), B_s value is determined by *max*{N+, N-}. Finally, N-value and B_s *p*-value is found for the significance level α=0.10. According to the found *p*-value, the null hypothesis H_0 that is based on the equivalence of medians should be rejected. This shows that gbFPA and rbFPA are significantly different from each other. Thus it can be finalized that gbFPA is considerably superior to rbFPA in UPMP under the given test conditions.

CHAPTER SIX
CONCLUDING REMARKS

The current study introduces a modified FPA to solve well-known facility location problems, UFLP and UPMP. Since these problems are in the category of binary optimization problems, FPA is modified for binary domains first. In this context, some operators such as crossover and mutation that are burrowed from evolutionary algorithms are adopted in the modified FPA. By using the proposed crossover operator, FPA is able to run in binary domains without employing additional auxiliary procedures such as transfer functions. Next, proposed binary bit-flip mutation is further enhanced by making use of an adaptive step-sizing procedure that introduces greater level of diversity at earlier iterations and encourages intensification towards the end of search. Thus, while local optima issues are avoided at earlier iterations are avoided, a more intensified search in the upcoming generations is conducted.

Diversely from UFLP, UPMP has a limitation that the number of opened facilities should be equal to a parameter, known as *p*, which is given in problem data. Therefore, although facility capacity limitations do not exist, solution vectors should be repaired in order to avoid any possible infeasibilities. Since, it is clear that, subsequent to the evolutionary operators reported in this thesis, each solution vector might include different number of 1s (denoting the number of opened facilities) of which the sum might not equal to *p*. That is why an additional repairing procedure is required. In this context, two different repairing methods are tested. In the former method, solution vectors are repaired at random while in the latter method, a greedy heuristic, which is adopted from the related literature is employed. As reported in this thesis, the greedy method performs better, however, it requires for greater CPU times whereas the first method seems to be performing more naïve.

For UFLP, all available benchmarking data is solved by the proposed algorithm and the obtained results are compared to those of the promising and competitive algorithms' that are lately reported in the related literature. According to the experimental study, the proposed algorithm is found as the most promising algorithm among all compared algorithms. Moreover, proper statistical tests are used to

demonstrate whether noteworthy developments are achieved. Outcomes of the statistical tests confirm that the suggested algorithm brings about significant improvements over some of the existing algorithms.

For UPMP, all available benchmarking data is solved again by the proposed algorithm that diversely adopts two different repairing strategies, namely random and greedy repair methods. All tested algorithms are compared to each other and appropriate statistical tests are conducted. According to the experimental and statistical study, proposed algorithm with greedy approach shows promising results while the random repair procedure exhibits a naïve performance as expected.

As one can understand from the relevant literature, most of the metaheuristic algorithms need additional procedures such as sigmoid function (logistics transfer function) to be adopted to binary problems. Moreover, such procedures yield to spatial distance, which is an undesirable circumstance. In this context, the most notable issue in the proposed approach is that it does not require such additional auxiliary procedures to run in binary spaces. Furthermore, the proposed strategy does not require for additional computational burden. On the other hand, one of the drawbacks of the proposed algorithm is that it makes use of two additional parameters to be calibrated in step sizing function. Thus, it brings about an additional effort in parameter calibration stage before proceeding with whole experimental tests.

The simplicity and efficiency of FPA deserves a further research. Since the proposed modification runs on binary populations, evaluating the population diversity by using hamming distance metric is quite practicable. In this regard, analyzing the effects of avoiding loss of diversity procedures along with the employed procedures can be scheduled as the future work.

REFERENCES

Abdel-Baset, M., & M Hezam, I. (2015). An effective hybrid flower pollination and genetic algorithm for constrained optimization problems. *Advanced Engineering Technology and Application*, *4*(3), 28-35.

Alizadeh, M. (2009). Facility location in supply chain. In R. Z. Farahani & M. Hekmatfar, (Eds.). *Facility location: concepts, models, algorithms and case studies* (473-504). Berlin: Springer-Verlag.

Al-khedhairi, A. (2008). Simulated annealing metaheuristic for solving p-median problem. *International Journal of Contemporary Mathematical Sciences*, *3*(28), 1357-1365.

Alp, O., Erkut, E., & Drezner, Z. (2003). An efficient genetic algorithm for the p-median problem. *Annals of Operations Research*, *122*(1), 21-42.

Al-Sultan, K. S., & Al-Fawzan, M. A. (1999). A tabu search approach to the uncapacitated facility location problem. *Annals of Operations Research*, *86*, 91-103.

Balinski, M. L. (1965). Integer programming: methods, uses, computations. *Management Science*, *12*(3), 253-313.

Bastı, M. (2012). P-medyan tesis yeri seçim problemi ve çözüm yaklaşımları. *AJIT-e: Bilişim Teknolojileri Online Dergisi*, *3*(7), 47-75.

Baykasoğlu, A., Ozsoydan, F. B., & Senol, M. E. (2020). Weighted superposition attraction algorithm for binary optimization problems. *Operational Research*, *20*(4), 2555-2581.

Binitha, S., & Sathya, S. S. (2012). A survey of bio inspired optimization algorithms. *International Journal of Soft Computing and Engineering*, *2*(2), 137-151.

Church, R. L., & Garfinkel, R. S. (1978). Locating an obnoxious facility on a network. *Transportation Science, 12*(2), 107-118.

Colmenar, J. M., Greistorfer, P., Martí, R., & Duarte, A. (2016). Advanced greedy randomized adaptive search procedure for the obnoxious p-median problem. *European Journal of Operational Research*, *252*(2), 432-442.

Correa, E. S., Steiner, M. T. A., Freitas, A. A., & Carnieri, C. (2004). A genetic algorithm for solving a capacitated p-median problem. *Numerical Algorithms, 35*(2), 373-388.

Costa, L. R., Aloise, D., & Mladenović, N. (2017). Less is more: basic variable neighborhood search heuristic for balanced minimum sum-of-squares clustering. *Information Sciences, 415*, 247-253.

Cournuejols, G., Nemhauser, G. L., & Wolsey, L. A. (1990). The Uncapacitated Facility Location Problem, Discrete Location Theory. *Lecture Note in Artificial Intelligence*, *1865*, 119-171.

de Armas, J., Juan, A. A., Marquès, J. M., & Pedroso, J. P. (2017). Solving the deterministic and stochastic uncapacitated facility location problem: from a heuristic to a simheuristic. *Journal of the Operational Research Society*, *68*(10), 1161-1176.

Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation, 1*(1), 3-18.

Draa, A. (2015). On the performances of the flower pollination algorithm–Qualitative and quantitative analyses. *Applied Soft Computing, 34*, 349-371.

Dubey, H. M., Pandit, M., & Panigrahi, B. K. (2015). A biologically inspired modified flower pollination algorithm for solving economic dispatch problems in modern power systems. *Cognitive Computation, 7*(5), 594-608.

Efroymson, M., & Ray, T. L. (1966). A branch-bound algorithm for plant location. *Operations Research*, *14*(3), 361-368.

Erlenkotter, D. (1978). A dual-based procedure for uncapacitated facility location. *Operations Research*, *26*(6), 992-1009.

Fathali, J. (2006). A genetic algorithm for the p-median problem with pos/neg weights. *Applied Mathematics and Computation*, *183*(2), 1071-1083.

Farahani, R. Z., & Hekmatfar, M. (Eds.). (2009). *Facility location: concepts, models, algorithms and case studies*. (1st ed.). Berlin: Springer-Verlag.

Fouad, A., & Gao, X. Z. (2019). A novel modified flower pollination algorithm for global optimization. *Neural Computing and Applications*, *31*(8), 3875-3908.

Garey, M. R., & Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness.* San Francisco: W.H. Freeman and Company.

Ghosh, D. (2003). Neighborhood search heuristics for the uncapacitated facility location problem. *European Journal of Operational Research*, *150*(1), 150-162.

Gölcük, İ., & Ozsoydan, F. B. (2020). Evolutionary and adaptive inheritance enhanced grey wolf optimization algorithm for binary domains. *Knowledge-Based Systems*, *194*, 105586.

Guner, A. R., & Sevkli, M. (2008). A discrete particle swarm optimization algorithm for uncapacitated facility location problem. *Journal of Artificial Evolution and Applications*, *2008, 1-9*.

Hakimi, S. L. (1964). Optimum locations of switching centers and the absolute centers and medians of a graph. *Operations Research*, *12*(3), 450-459.

Kariv, O., & Hakimi, S. L. (1979). An algorithmic approach to network location problems. I: The p-centers. *SIAM Journal on Applied Mathematics*, *37*(3), 513-538.

Khumawala, B. M. (1972). An efficient branch and bound algorithm for the warehouse location problem. *Management Science, 18*(12), B-718.

Kiran, M. S. (2015). The continuous artificial bee colony algorithm for binary optimization. *Applied Soft Computing*, *33*, 15-23.

Körkel, M. (1989). On the exact solution of large-scale simple plant location problems. *European Journal of Operational Research*, *39*(2), 157-173.

Krömer, P., & Platoš, J. (2014). Solving the p-median problem by a simple differential evolution. In *2014 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, (3503-3507).

Lin, G., & Guan, J. (2018). A hybrid binary particle swarm optimization for the obnoxious p-median problem. *Information Sciences*, *425*, 1-17.

Li, X., Xiao, N., Claramunt, C., & Lin, H. (2011). Initialization strategies to enhancing the performance of genetic algorithms for the p-median problem. *Computers & Industrial Engineering*, *61*(4), 1024-1034.

Mladenović, N., Alkandari, A., Pei, J., Todosijević, R., & Pardalos, P. M. (2020). Less is more approach: basic variable neighborhood search for the obnoxious p-median problem. *International Transactions in Operational Research*, *27*(1), 480-493.

Montoya-Torres, J. R., Aponte, A., & Rosas, P. (2011). Applying GRASP to solve the multi-item three-echelon uncapacitated facility location problem. *Journal of the Operational Research Society*, *62*(2), 397-406.

Montoya, M. R., Velázquez, R. G., Analco, M. E., Flores, J. L. M., & Loranca, M. B. B. (2019). Solution search for the capacitated p-median problem using tabu search. *International Journal of Combinatorial Optimization Problems and Informatics*, *10*(2), 17-25.

Nabil, E. (2016). A modified flower pollination algorithm for global optimization. *Expert Systems with Applications*, *57*, 192-203.

Ozcakar, N., & Basti, M. (2012). P-medyan kuruluş yeri seçim probleminin çözümünde parçacık sürü optimizasyonu algoritması yaklaşımı. *İstanbul Üniversitesi İşletme Fakültesi Dergisi*, *41*(2), 241-257.

Ozsoydan, F. B. (2019). Artificial search agents with cognitive intelligence for binary optimization problems. *Computers & Industrial Engineering*, *136*, 18-30.

Ozsoydan, F. B. (2019). Effects of dominant wolves in grey wolf optimization algorithm. *Applied Soft Computing*, *83*, 105658.

Ozsoydan, F. B., & Baykasoglu, A. (2019). Analysing the effects of various switching probability characteristics in flower pollination algorithm for solving unconstrained function minimization problems. *Neural Computing and Applications*, *31*(11), 7805-7819.

Reese, J. (2006). Solution methods for the p-median problem: An annotated bibliography. *Networks*, *48*(3), 125-142.

ReVelle, C. S., & Swain, R. W. (1970). Central facilities location. *Geographical Analysis*, *2*(1), 30-42.

Rolland, E., Schilling, D. A., & Current, J. R. (1997). An efficient tabu search procedure for the p-median problem. *European Journal of Operational Research*, *96*(2), 329-342.

Rosing, K. E., & ReVelle, C. S. (1997). Heuristic concentration: Two stage solution construction. *European Journal of Operational Research*, *97*(1), 75-86.

Selvaraj, C., Kumar, R. S., & Karnan, M. (2014). A survey on application of bio-inspired algorithms. *International Journal of Computer Science and Information Technologies*, *5*(1), 366-70.

Sevkli, M., & Guner, A. R. (2006). A continuous particle swarm optimization algorithm for uncapacitated facility location problem. In M. Dorigo, L. M.

Gambardella, M. Birattari, R. Poli & T. Stützle, (Eds.). *Ant Colony Optimization and Swarm Intelligence* (316-323). Berlin: Springer-Verlag.

Tsuya, K., Takaya, M., & Yamamura, A. (2017). Application of the firefly algorithm to the uncapacitated facility location problem. *Journal of Intelligent and Fuzzy Systems*, *32*(4), 3201-3208.

Wang, D., Wu, C. H., Ip, A., Wang, D., & Yan, Y. (2008). Parallel multi-population particle swarm optimization algorithm for the uncapacitated facility location problem using openMP. In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)* (1214-1218).

Yaghini, M., Karimi, M., & Rahbar, M. (2013). A hybrid metaheuristic approach for the capacitated p-median problem. *Applied Soft Computing*, *13*(9), 3922-3930.

Yang, X. S. (2012). Flower pollination algorithm for global optimization. J. Durand-Lose & N. Jonaska, (Eds.), *Unconventional computation and natural computation* (240-249). Berlin: Springer.

Yapicioglu, H., Smith, A. E., & Dozier, G. (2007). Solving the semi-desirable facility location problem using bi-objective particle swarm. *European Journal of Operational Research*, *177*(2), 733-749.

Zaferanieh, M., Abareshi, M., & Fathali, J. (2021). The minimum information approach to the uncapacitated p-median facility location problem. *Transportation Letters*, 1-10.

Zhou, Y., Wang, R., Zhao, C., Luo, Q., & Metwally, M. A. (2019). Discrete greedy flower pollination algorithm for spherical traveling salesman problem. *Neural Computing and Applications*, *31*(7), 2155-2170.