DETECTING BEST GROCERY STORE LOCATION BASED ON A MULTI-OBJECTIVE OPTIMIZATION APPROACH

by

İpek Çebi

Submitted to Graduate School of Natural and Applied Sciences
in Partial Fulfillment of the Requirements
for the Degree of Master of Science in
Computer Engineering

Yeditepe University
2016

DETECTING BEST GROCERY STORE LOCATION BASED ON A MULTI-OBJECTIVE OPTIMIZATION APPROACH

APPROVED BY:

Assist. Prof. Dr. Dionysis Goularas .....
(Thesis Supervisor)

Assoc. Prof. Dr. Nafiz Arıca .....

Assist. Prof. Dr. Tacha Serif .....

DATE OF APPROVAL: ..../..../2016

ACKNOWLEDGEMENTS

I am especially thankful to my supervisor, Assist. Prof. Dr. Dionysis Goularas, whose support and guidance from the initial to the final level enabled me to accomplish this research.

I would like to express my gratitude to Dr. Zeynep Zerrin Turgay for her guidance, suggestions and encouragement towards this project.

To my family, I offer sincere gratitude for their unconditional support. I also thank Gökçe Seymen, Büşra Kahraman Ağır, Yusuf Can Semerci for their useful discussions and great friendship.

ABSTRACT

DETECTING BEST GROCERY STORE LOCATION BASED ON A MULTI-OBJECTIVE OPTIMIZATION APPROACH

Multi-objective optimization is an approach that can be utilized to optimize more than one objective function simultaneously. NSGA-II is a non-dominated sorting genetic algorithm that has been widely used for solving different multi-objective optimization problems. Location decision is a critical problem for the facilities such as markets and stores. The goal of this study is to decide where a new facility should be located. In the literature, there are studies that utilize multi-objective optimization techniques for the location decision problem. However, such applications try to determine the location for big facilities like hospitals, big grocery stores and so on. In this study, a novel approach based on multi-objective genetic algorithms (MOGAs) have been proposed for the location decision problem of moderate-sized grocery stores. The genetic search is also supported with image processing techniques in a hybrid framework and the NSGA-II is utilized as the multi-objective genetic algorithm. The proposed method optimizes two objectives aiming to minimize the distance to restaurants and metro stations in the area and maximize the distance to other markets in the area. The mean value of the surface areas, an important factor for the location decision is found with Google Maps using image processing techniques that have the goal to detect the surface of individual buildings. The mean value of the surface areas are utilized to determine the final result among the set of solutions produced by NSGA-II. Various tests are carried out and it has been observed that the system can propose appropriate locations based on the objective functions utilized.

ÖZET

ÇOK AMAÇLI OPTİMİZASYON YAKLAŞIMINA DAYALI EN İYİ MARKET KONUMUNUN TESPİTİ

Çok amaçlı optimizasyon aynı anda birden fazla amaç fonksiyonunu optimize etmek için kullanılabilen bir yaklaşımdır. NSGA-II farklı niteliklere sahip çok amaçlı optimizasyon problemlerini çözmek için kullanılan genetik algoritmadır. Algoritma birbirini amaçlar açısından domine edemeyen kromozomların popülasyonda sıralanarak yer alması temelinde arama işlemini gerçekleştirmektedir. Literatürde, konum karar problemi için çok amaçlı optimizasyon teknikleri kullanan çalışmalar vardır. Ancak, bu tür uygulamalar hastaneler, büyük marketler vb. büyük tesisler için konum belirlemek için gerçekleştirilmiştir. Bu çalışmada, orta büyüklükte marketlerin yer karar probleminin çözülmesi için çok amaçlı genetik algoritmalara (MOGAs) dayalı yeni bir yaklaşım öne sürülmüştür. Genetik arama süreci görüntü işleme teknikleri ile desteklenmektedir. NSGA-II çok amaçlı genetik algoritma olarak kullanılmaktadır. Önerilen yöntem restoranlar ve metro istasyonlarına mesafeyi en aza indiren ve bölgedeki diğer mağazalara mesafeyi maksimize eden iki hedefi optimize eder. Diğer taraftan, binaların yüzey alanlarının büyüklüğü lokasyon seçimi için önemli bir faktördür. Yapılan çalışmada haritada yer alan binaların yüzey alanları ve bu alanların ortalama değerleri görüntü işleme teknikleri ile belirlenmektedir. Hesaplanan bu değer NSGA-II tarafından üretilen çözüm kümesi arasından son çözümü belirlemek için kullanılmaktadır. Yapılan testlerde sistemin kullanılan amaç fonksiyonlarına göre uygun yerleri önerdiği gözlenmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS / ABBREVIATIONS

CRF	The Conditional Random Field
DSM	Digital Surface Model
FL-MICROGA	Fuzzy Logic guided Micro Genetic Algorithm
FL-SPEA2	Fuzzy Logic guided Strength Pareto Evolutionary Algorithm
GIS	Geographic Information System
HOPM	Hierarchical Object Process Model
LRCD	Left-Right Crowding Distance
MCS	Monte-Carlo-Simulation
MICROGA	Micro Genetic Algorithm
MOEAs	Multi-Objective Evolutionary Algorithms
MOGA	Multi-Objective Genetic Algorithm
MOPSO	Multi Objective Particle Swarm Optimisation
NSGA-II	Non Sorting Genetic Algorithm
PESA	Pareto Envelope-based Selection Algorithm
PICEA	Preference-Inspired Coevolutionary Algorithm
RGB	Red Green Blue
SIFT	Scale Invariant Feature Transform
SPEA2	Strength Pareto Evolutionary Algorithm
SVM	Support Vector Machines

1. INTRODUCTION

Location decision is a critical problem for facilities such as markets, hospitals, airports, etc. The problem is important for both public and private sectors. The main question is to define the criteria that will determine where a particular location should be placed. The answer to this question depends on the context in which the localization problem is solved and on the criteria utilized for the problem. The facilities should be located according to different demands and conditions. As an example, for ambulance sitting problem, the facilities should be located near to the demand sites as much as possible. In another example, radioactive waste repositories should be located far from highly populated areas and they should be localized in stable regions.

In order to solve real world problems in finance, transportation, engineering and health areas simultaneous optimization techniques are required. In many cases multi-objective evolutionary algorithms are proposed to solve such problems. Very often there are more than one conflicting constraints in multi-objective optimization problems. By consequence, the solution of these problems becomes a challenging task. Facility location problems can be solved by multi-objective optimization methods. Similarly, grocery store location decision problem has a number of conflicting constraints and hence the problem is suitable to be handled by the multi-objective optimization approach. Different important factors exist that determine the success of the grocery store locations. The success of the store is based on the endorsement of sale and the stores which have the highest sales, are usually located near to public areas with a dense population.

In the literature, various methods, problem definitions and algorithms exist for facility location problem [33,34,35]. These works propose real-life extensions to the basic models utilized for the localization process. For each specific location model, the related concepts and the applications are discussed broadly. Also customer choice model is explained with a deterministic utility function.

There are some studies in the literature that solve the localization problems with multi-objective optimization techniques [18,25,27,31,32]. These studies include localization of organic farms, allocation of parking lots, bus stop placement along a single bus route, localization of maternity hospitals in France and site layout planning problems. The

common approach in these studies is the localization by using multi-objectives. There are different multi-objective evolutionary algorithms such as Pareto Envelope-based Selection Algorithm (PESA), Strength Pareto Evolutionary Algorithm (SPEA2), Non Sorting Genetic Algorithm (NSGA-II) and Multi Objective Particle Swarm Optimisation (MOPSO) that are used for solving optimization problems [19,20,21,22,23]. These evolutionary algorithms are used in different areas such as finance, business processes and supply chain management. In finance, the evolutionary algorithms are used to maximize the profit by satisfying feasibility requirements and matching a proxy for the return. Duration and process costs are the objective functions in the business processes. In supply chain, the evolutionary algorithms are used to minimize the total inventory, variance of order quantity and the total cost of a two-echelon serial supply chain.

Concerning the building detection from satellite images, there exist a variety of different methods and approaches. In [4,5,6,7], shadow information is used for the detection of buildings. Also building detection methods are used in various remote sensing applications: the Conditional Random Field (CRF) which extracts the existing objects in an area by fusing various features, a Hierarchical Object Process Model which finds the location of objects and the corner detection model which describes the objects' vector shapes [9,10].

This study proposes a multi-objective optimization solution for the grocery store location problem. NSGA-II is utilized in order to optimize location decisions of the stores in this study based on location coordinates which are displayed on Google Maps. Two objectives are utilized for the solution of the problem. The first aims to position the new store close to points of interests such as metro stations and restaurants. The second objective aims to maximize the distance from other stores in the area. Then, among the proposed locations found by the genetic algorithm we select the one which has the minimum distance from the location of the weighted mean value of the surface areas of the individual buildings.

The difference of this study from other studies in the literature is that a new grocery store location is determined by using multi-objective optimization technique. In the literature, there are many studies that utilize multi-objective optimization technique for the location decisions, but to the best of our knowledge, multi-objective genetic algorithms have not been still utilized for the grocery store location decision. In general, many location

problems are solved by heuristic optimization algorithms. Solving the grocery store location problem with a genetic algorithm is the main contribution of this study

The parameters of the genetic algorithm will be tested in order to generate the best possible results. We will select different locations in Kadıköy, a region of Istanbul and the coordinates of restaurants, metro stations and stores will be set manually. A part of testing will be to remove one of the existing stores in a selected area from the map in order to create a place for a new store in this area. Then the proximity to the removed store will be calculated with NSGA-II and our objective functions. It has been observed the results have been near the restaurants and metro stations and away from the grocery store locations.

This thesis is organized as follows: Preliminary information about Image Processing techniques and Multi-Objective optimization and the related work are presented in the next chapter. The proposed method that consists of building detection and the application of NSGA-II algorithm can be found in Chapter 3. The user interface and the system description are presented in Chapter 4. The experimental results are given in Chapter 5. Finally, the last chapter contains the discussion and the conclusion of this study.

2. BACKGROUND

2.1. Problem Definition

In retail industry a big importance is given to the localization of stores. Different factors can be affective on the success of a store according to the choice of its location. As an example, a store which is located near public areas would have more endorsement compared to others located in a bigger distance from these areas. In general, the localization of stores affects directly the sales of their products. However, it is difficult to decide where the new store will be located. The automatic detection of a store location based on a multi-objective optimization according to criteria related with the success of the store has emerged from this need.

One of the important factors is the population density of the area where the new store will be located. Higher sales could be obtained in stores near high populated areas. In general, the population density can be roughly estimated from the surface area of the buildings. When the surface areas of the buildings in a place are large enough, this is a sign for high population in that area. Another critical factor is the distance of the grocery store to the point of interests such as metro stations, schools and restaurants. The stores near to such public areas, are more successful compared to the other stores. The other criterion is that the new store location should as far as possible from other grocery stores in the area in order to avoid competition and be located in a place that no competitor stores are around.

2.2. Image Processing Background

Satellite images provide useful data that can be utilized by researchers in order to extract information. However, detecting man-made objects from a satellite image is not a straight forward process since the resolution of satellite images in general is not high enough. Furthermore, the urban area changes during time and by consequence, the detection of objects should be done periodically. Extracting information from satellite images by using automatic techniques is an important task which provides valuable information in many areas such as military, government agencies and municipalities. Automatic techniques come with some difficulties as for example the detection of buildings in the area. Buildings

have different characteristics such as color, shape and their appearance changes according to the view angle, scaling and illumination. As can be seen in literature [1,2,3,4,5,6,7], solving such problems related to building detection, requires different pattern recognition and image processing techniques.

Buildings have different lighting conditions, viewpoints, sizes and present various structural differences. The fixed properties of scale invariant feature transform (SIFT) which leads to localing image descriptors, are invariant to rotation, scaling and translation. Therefore SIFT is convenient for building detection in satellite images. In the literature, there are many studies that apply SIFT keypoints for building detection. The use of SIFT and graph theory are proposed for the urban area and building detection problem in [1]. SIFT key points, are effective on detecting objects in an image but they are not sufficient for detecting buildings and urban areas alone. Consequently, in [1] graph theory and SIFT key points are both used for detecting the buildings. Each SIFT key point is utilized as the vertex of a graph and each relationship represents the edges in the graph between the corresponding vertices. Based on this graph constructed by the key points and the relations between them, the urban area can be detected by using multiple sub-graph matching methods and the buildings are extracted with a graph cut method in the urban area. The results show that urban area detection is performed 89.62% correctly, and with a 8.03% false-alarm rate. Then, after detecting the urban area, the building detection method is performed 88.4% correctly with a 14.4% false-alarm rate. These false-alarm rates occur because of similar items of buildings in the image.

3D urban maps provide data that can be used in different areas such as disasters, airport and urban planning, geographic localization and military situations. In [2], for building detection, a novel approach which utilizes segmentation, detection of corners and Hough Transform, is proposed. The corners are detected with SIFT features in the original image. The image is separated into a number of classes by applying the mean shift segmentation which gives the approximate borders of the buildings. After applying segmentation, the adaptive windowed Hough Transform is used for finding the straight lines of the buildings' borders.

Automatic building rooftop methods are used with 3D map reconstruction in [3]. The method utilizes a set of a four sided rooftop which is formed from the combination of the

border line and the line segments. The results show that the rooftops which have triangular shapes, are detected with different reflection properties.

There are many studies that use shadow information for the detection of buildings. The relationship between shadows and buildings are used in [4]. The corners which are labeled as bright or shadow, are extracted from the original image. Rectangles are detected by using the bright corners and the detection of buildings is based on these rectangles which are verified by using shadow corners. In [5], buildings are detected from a PAN image of high resolution satellite images with shadow evidences. The images are separated by histogram peak selection with an unsupervised clustering method for splitting the image into classes and then shadow information of the buildings which is found from the lowest gray level of the image, is used together with the classes to detect the buildings. In order to verify the detected buildings, Canny operator is applied for edge detection. In [6], it is proved that shadows of the buildings and their borders have important information such as the shapes of the roofs and the heights of the buildings. Therefore, in this method the shadows are detected by thresholding. In [7], the invariant color features are extracted from the shadows and the red rooftops. Then the views of buildings are found by using the shadow information. The building shapes are determined by the box fitting algorithm. In [8], multiple elements such as information of shadow, color, edge, texture and elevation data are used together and the buildings are finally extracted using blobs.

Building detection in satellite images is an important topic in various remote sensing applications such as disaster recovery and Geographic Information System (GIS) data management. Urban buildings have complex structures and different shapes in high-resolution images. Also it is hard to separate the borders between similar objects. Therefore, automatic building detection is a critical problem in high resolution remote sensing images. There is a certain number of studies on this problem in the literature. The Conditional Random Field (CRF) which extracts the existing objects in an area by fusing various features, a Hierarchical Object Process Model (HOPM) which finds the location of objects and the corner detection model which describes the objects' vector shapes are proposed in [9]. A Marked Point Process approach is proposed for the extraction of buildings in remotely sensed images in [10]. Various buildings are characterized with various features that represent the relationships between the neighbor buildings.

To overcome the difficulties of building detection on high resolution satellite images, probability theory and the structural features are used in [11]. A steerable filter set is used to find the structural features that show geometrical properties of the objects in the original image. The probability density function (pdf) showing the locations of the detected buildings is calculated from these extracted structural features. The feautures are classified as curved or straight features. In [12], the linear features are extracted from the given image and they are used as the vertices of a graph and the detection of the buildings is carried out by using this graph. Sub-graph matching is also applied to extract buildings and intensity. The shadow information is used to verify the appearance of the buildings.

In the literature, the methods proposed for detecting buildings can be divided into two main groups. Multispectral and panchromatic information are used to detect buildings in the first group. Digital surface model (DSM) data is used by some studies as the second approach to detect buildings. In [13], DSM and panchromatic data are combined to detect the buildings. In the first method, the corner points found from Harris corner detection method are used with DSM data. Then, shadow information and DSM data are combined to detect the buildings. A method is proposed to separate the trees and the buildings using DSM in [14]. According to this method, the width and the height information which are found from DSM are utilized. The trees are removed based on the color and entropy information.

A method using Gabor filters to extract local features is proposed in [15,16]. These local features which represent urban area properties, are utilized to create a spatial matrix for voting candidate urban areas in [15]. An optimum decision making approach is applied to detect urban areas on the vote distribution. In [16], the local features representing building properties, form descriptor vectors to create a spatial voting matrix. The buildings are detected with local maximum votes in the spatial voting matrix. These methods are tested on Ikonos satellite image set and other panchromatic aerial image sets.

Machine learning techniques are also utilized in the detection of buildings. The combination of support vector machines (SVM) and k-means clustering is applied for detecting rooftops in [17]. The image is segmented into a set of rooftop candidates by k-means clustering. The segments are homogenous regions that are associated with rooftop areas in the image. Then an SVM classifier differentiates the nonrooftop and rooftop regions. However, SVM results a high rate of misclassification. Histogram method is

applied to detect rooftops that are not found in the SVM method. The results show that rooftops are correctly detected with a high percentage rate.

2.3. Multi-Objective Optimization Background

Synchronous optimization techniques are required to solve real world problems with multiple goals. Multi-objective evolutionary algorithms (MOEAs) are one of the techniques that have emerged based on such needs. There are different constraints which conflict each other in a multi-objective optimization problem. Hence, obtaining the solution of a multi-objective optimization problem is a challenging task. Therefore, various algorithms are proposed to solve these problems in the literature. In the real world, multi-objective optimization is done for different facilities. MOEAs have also been applied to different real world problems in engineering, finance, transportation etc.

As an example, an organic farm design model in Netherlands is presented in [18]. This model evaluates both economic and productive features of an organic farm by using a multi-objective evolutionary algorithm. The algorithm generates a large set of Pareto-optimal farm models. The features optimized in this model are provided from the literature and a lot of inquiries are also done with the farmers. The criteria utilized for this multi-objective optimization process are to minimize the soil nitrogen losses and labor requirements, to maximize organic matter balance and the operating profit. The results show that when compared to the existing farms, better performance has been obtained using these four alternative farm objectives.

In the finance sector, the portfolio optimization is important. This optimization process has three different objectives. These are maximizing the profit and the satisfying feasibility requirements. Matching a proxy for return is also needed. Three Evolutionary Algorithms PESA, SPEA2 and NSGA-II, are used in [19] for the portfolio optimization problem.

A multi-objective optimization application defining the duration and the process cost as objective functions, is used in the business processes. Five test problems which are business process designs of varying complexities, are solved with the multi-objective business model described in [20,21]. The following different optimization algorithms, MOPSO, SPEA2 and NSGA-II are utilized on the test problems. The results show that optimal solutions are obtained on process design of different complexities with the NSGA-

II and SPEA2 algorithms. For the business analyst, these optimal solutions provide different alternative process designs.

In [22], an optimization problem based on the tasks of business processes is handled. These processes have their own libraries of alternatives. The criteria are to address processes of different sizes for alternative libraries and subscribing tasks. MOPSO, SPEA2 and NSGA-II algorithms again are applied to produce the optimized solutions to tri-objective and bi-objective problem formulations. They both produce near optimal solutions for all instances.

The multi-objective optimization approach has been utilized in supply chain management. The modification of an existing multi-objective evolutionary algorithm NSGA-II is proposed in [23]. There are three objective functions for this problem. These objective functions are minimizing the total inventory, variance of order quantity and the total cost of a two-echelon serial supply chain. The proposed modified algorithm gives better solutions compared to NSGA-II and SPEA2.

The preference-inspired co-evolutionary algorithm (PICEA) aims to design a multi-objective hybrid renewable energy system [24]. The study tries to combine different resources in a convenient way in order to find a solution. Minimizing fuel emissions, the loss of power supply probability and the annualized cost of system are the objective functions for this problem. The proposed method produces optimal solutions on wind turbines, batteries, PV panels and diesel generators.

In [25], for the problem of allocation of parking lots, a multi-objective optimization model which determines the size of parking lots needed and the optimal capacity is proposed. The model is tested with nine buses. The voltage profile distribution of the system can be improved by the optimal allocation of parking lots in the simulation results. According to the authors, sizing of parking lots and optimal sitting have economical benefits on electric vehicles.

Multi-objective routing problem with multiple customers, depots and products is studied in [26]. Fuzzy logic non-dominated sorting genetic algorithm which assesses the rate of mutation and crossover after ten sequential generations is utilized to minimize both the total travelling time and the total distance. Furthermore, other genetic algortihms as SPEA2, Micro Genetic Algorithm (MICROGA), NSGA-II, Fuzzy Logic guided Strength

Pareto Evolutionary Algorithm (FL-MICROGA), Fuzzy Logic guided Micro Genetic Algorithm (FL-SPEA2) are applied to assess the success rate of the proposed method (FL-NSGA-II). The results show that FL-NSGA-II gives better performance on this problem compared to other genetic algorithms.

The multi-objective evolutionary algorithm can be applied for problems with conflicting objectives such as designing a bus stop placement along a single bus route [27]. Adding new stops on the bus route would reduce the access time for the passengers. However, deceleration delays can be incurred by buses serving the additional stops. The supplier cost and the corresponding user in-vehicle time can increase due to the extra stops. The model for this problem is developed by taking into consideration the walking distance, the acceleration rates at stops and the attractiveness of an access path to a transit service. A heuristic evolutionary algorithm is utilized for the optimal solution in [27]. The method is implemented as a Geographic Information System software (GIS). Applying the access paths in the model provides a high degree of accuracy for the route accessibility. The method applies topographic information which is an important criteria in the design of a bus network. The framework of this model utilizes GIS data that covers an important geographic area.

In some applications, the multi-objective evolutionary approach might have a limited success. However, it is possible to improve the success rate of the algorithm by utilizing extra operators and techniques. For instance, the most popular multi-objective algorithm NSGA-II has a special density estimation method named as crowding-distance. The density estimation method calculates the distance among the individuals in the non-dominated set. Then, these distances determine the distribution of individuals in the non-dominated set and the individuals in the less crowded areas are favored during the genetic selection process. Hence, a better exploration of the search space is achieved with the method described in [28]. This method is tested on five test problems and it gives better optimal solutions in the non-dominated front compared to the original algorithm. As shown in this study, diversity distribution has an importance role in the performance of multi-objective evolutionary algorithms. Therefore, to improve NSGA-II further, Left-Right Crowding Distance (LRCD) approach and also two other methods based on LCRD named as E-LRCD and W-LRCD are proposed in [29]. The improvements obtained in this study are based on improving the uniformity of the non-dominated set. Six test problems

evaluate the performance of the new methods and the results demonstrate that the performance of E-LRCD is the best, LRCD is the second, W-LRCD is third and the original NSGA-II has the lowest performance on these six test problems.

The multi-objective optimization method is also applied in hybrid frameworks in order to improve efficiency. An efficient method for planning a probabilistic distribution network expansion is proposed in [30]. The method combines local search with the multi-objective genetic algorithm. The objective functions utilized in this algorithm are minimizing the installation cost of distribution sub-stations, feeders, power quality and network loss. Monte-Carlo-Simulation (MCS) which provides the correlations between the nodal specified values, is applied to find the uncertainty of distribution network expansion. Hence this local search facility provided by MCS fine-tunes the solutions created by the evolutionary approach.

Multi-objective optimization is also an important method for localization problems. In the literature, there are many studies aiming to solve these problems with multi-objective optimization techniques. A method for optimizing locations of maternity hospitals in France is proposed in [31]. This is a domain where health costs rise rapidly. The method proposed in [31] utilized the maximum covering/p-median hierarchical model. There are 703 public and private maternity hospitals that are categorised in three levels in France. The level 1 maternity hospitals consist of pregnant mothers with no health problems. The level 2 maternity hospitals can deposit babies who are born premature in 33 weeks with no treatment. The level 3 maternity hospitals can monitor pathological pregnancies, and accomodate premature babies with health problems such as diabetes and hypertension. The changes in France's population affect public services given in maternity hospitals. Therefore, a location-allocation model is required for reconstructing public services in this domain. The hierarcical location-allocation method proposed has three steps. In the first step, k maternity hospitals are located without considering their level. The second step provides searching for p best facilities among these k locations in order to correspond the second and the third level of maternity hospitals in a p-center model. An m-center model which corresponds at the third level of maternity hospitals defines the m best maternity hospitals between the p facilities in the third step. The myopic algorithm is used to optimize locations of the facilities in the network. When the objective function for each location node is minimized, the best location for the first activity is simply chosen.

Locating a second facility is provided by assigning the closest nodes to the considered node. The objective function values is minimized at the second facility node. This method can be repeated until every facility is located. The weighted m-median and p-median models are solved by a Lagrangian relaxation heuristic that uses a substitution improvement algorithm. The suggested locations by the algorithm are compared with actual locations to show the success of the proposed method. Additionally, the same problem exists in service chains or retail. In the supply chain case the same example problem is described by the unavailability of a store to the customers. In the chain organization, we can reduce delivery costs by reducing or regulating the distance between stores.

Site layout planning problems are also solved by multi-objective genetic algorithms. A multi-objective optimization method is proposed for automated support to airport operators and construction planners [32]. The method optimizes airport site layouts. The system is implemented with the following four modules: a relational database, a multi-objective optimization engine, a visualization module and an input/output module. The relational database stores all generated optimal solutions and integrates data. The input/output module provides possible optimization parameters to identify planning and provide optimal solutions. The visualization module enables the system to communicate with external CAD software in order to support the visualization of the optimal site layout plans. The multi-objective optimization engine optimizes the construction security and all relevant site layout costs. The objective functions utilized in the system aim to maximize construction safety, construction-related security level, construction-related aviation safety such as debris and wildlife control and on the other side to minimize all relevant site layout costs. This optimization system is designed and identified according to some decision variables. These are the optimal usage of wreck holding measures, the security control systems used on site, the optimal use of wildlife management measures, the location of security hedge with respect to secure facilities and the x, y coordinates of the each transient service area's center of gravity.

2.4. Store Location Background

The location of a facility is of great importance for the private and public sectors. For instance, state governments need to determine locations for the bases of emergency

highway patrol vehicles. As another example, local governments have to define adequate locations, fire stations and ambulances. For such examples, poor locations could increase the possibility of property damage or loss of life. In the private sector, the companies have to define locations for distribution centers, production plants and retail outlets. In this domain, wrong decisions about the location of a facility can cause a decrease in competitiveness and increase costs. Therefore, it is important to decide the place where a new facility will be located in a competitive environment that already consists some other facilities.

The customer choice model is described with a deterministic utility function in [33]. In this study, bicriterion maxcovering-minquantile problem's solution methods are used. The solution for maximal profit can be obtained with a finite polynomial algorithm. For the problem sets where optimal solutions may not exist, alternative tie-resolution rules are also proposed.

Several different algorithms for finding optimal and attractive facility locations are presented in [34]. Quantifiable objectives which depend on facility locations, are defined in the study. Additionally, a model based on factors such as median, center, fixed charge and covering, is also proposed to solve classical location problems in [34]. Moreover, this study inlcudes real-life extensions to the basic models utilized for locating undesirable, distribution, production or interacting facilities.

In [35], facility site selection problem and different concepts, models, algorithms and applications for location problems are discussed. In this study, a set of facilities established to minimize the cost of providing a set of demands is taken into consideration. Customers, type of facility, the space where customers and the facility are located together with the distance metric utilized are the four important components for describing location problems. The customers are assumed to be on routes or already inside the facility. The distance metric is assumed to show the geographical distance between facilities and customers and for each specific location model, these components are described. Location models are used in a wide range of domains such as locating dangerous material by maximizing its distance to the public or locating warehouses by minimizing the average access time to the market.

The problem of making concurrent decisions on service capacity, price and location for facilities on a network is presented in [36]. A Poisson Process is followed by demand of each node of the network. At each facility, service times are assumed to be exponentially distributed and the demand is assumed to depend on the closeness to the facility and the price. Multinomial logit function is used to model all facilities based on the assumption of having the same price and the same distances. To find the locations of the facilities, the tabu search method is utilized. The results determine what service capacity has to be operated for each facility and also where the facilities will be located and what price will be invoiced to customers. Additionally, the balking affects, the decisions of the system and the expected profit are determined in the solution.

A basic stochastic model based on customer preference is proposed in [37]. The Huff model utilized in this study is based on the proportion of customers who are shopping at a certain shopping area, the distance between the customers and the shopping area. Moreover, assumptions related to the degree of competition exist in the model. Such assumptions have been validated with a wide field of use. The literature is also discussed in this publication in detail considering different studies related to this domain.

A probabilistic customers' choice model in competitive locations is proposed in [38]. This model depends on the logit regression analysis and the random utility and by consequence a link between operations and marketing managers is presented in [38]. The main properties of objective functions in this model are concavity, submodularity and a sum of rational ratios. The objectives are optimized by the bound and branch method. The proposed methods are tested on a set of random problems. Concavity provides the best upper bound of the objective functions, which makes it is the most useful property. Some results show that the optimal location for facilities can be modified by customers' behaviors. Therefore more random or more deterministic logit models may be needed based on the customers on the flow.

3. METHODOLOGY

3.1. Genetic Algorithms

In this study, genetic algorithms are used to solve the new grocery store location problem. Multi-objective genetic algorithms (MOGAs) have been preferred since the actual problem has more than one objective to be optimized. In general, genetic algorithms provide a search and optimization method that works in a similar way with the evolutionary process observed in nature. Instead of working on a single solution, genetic algorithms carry out the search by producing a set of solutions. Thus, in the search space alternative solutions are evaluated simultaneously and the possibility of ultimately reaching a holistic solution is increased.

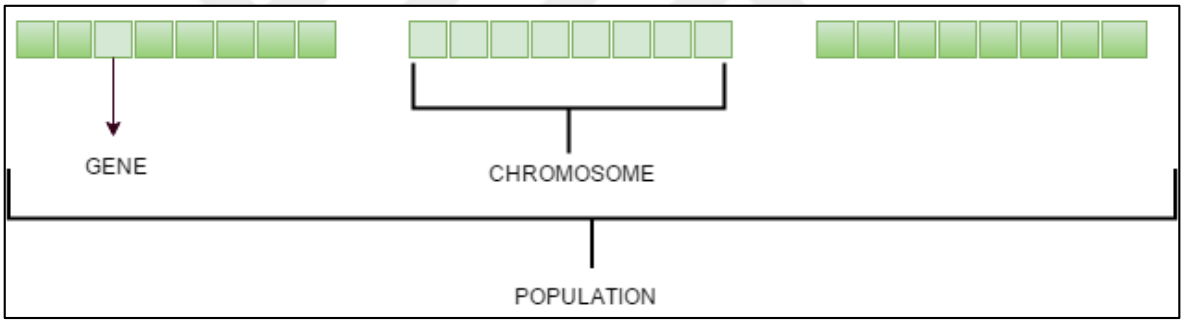

Figure 3.1. The notations of genetic algorithms: gene, chromosome and population

To solve problems, genetic algorithms mimic the evolutionary process in a computing environment. Instead of using a single element for developing the solution as in other optimization methods, genetic algorithms utilize a group of elements for the search process. This group is named as a “population” in genetic algorithm terminology. Each element of the population, consisting of an array of numbers is called a “chromosome” or an “individual”. Each element of this array is named as a gene. Individuals in the population are created by the genetic algorithm processors in the evolutionary process. Each individual represents a possible solution in a search space. Individuals are generally represented in terms of binary alphabet {0,1}. Figure 3.1 gives a graphical representation of the terms gen, chromosome, and population.

The representation of a problem as an individual is a crucial issue in genetic algorithms. The most important factor that determines the success in a problem solution is the representation utilized for the individual. A fitness function is also utilized and this function evaluates the individuals in terms of their success to solve the problem at hand. The individuals with high fitness value have more chance to be selected for the genetic operations. These individuals produce new individuals named as “offspring” by crossover and mutation operations. The offspring has the properties of the parents. The individuals with low fitness values will be excluded from the population since they will have less chance for being selected for the following genetic operations. The new population created by the offspring would contain the majority of the properties characterized by resulting a high fitness value to the members of the previous population. Thus, good features spread in the population throughout the generations. Hence, the solution is expected to appear in the population if sufficient number of generations is utilized.

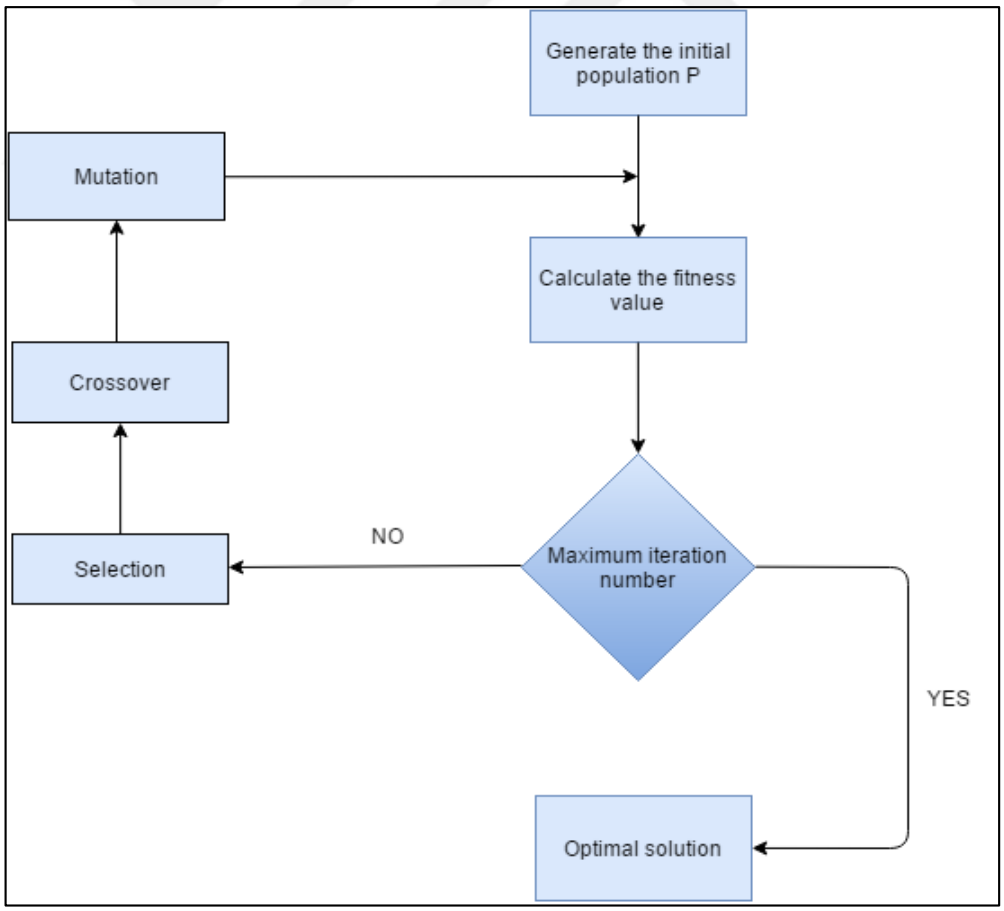

Figure 3.2. Flow chart of genetic algorithms

As presented in Figure 3.2, genetic algorithms generate offspring by using operators such as mutation, crossover and inheritance. These techniques are inspired by natural evolution. The algorithm applies crossover and mutation on the selected individuals after the initial population is randomly generated. A selection process allows "better" individuals to pass their genes to the next generation. However, the low fit individuals also have a certain chance to be selected for the genetic operations. The selection of an individual depends on its fitness value which is determined by an objective function. In crossover, the gene values are chosen from two selected individuals in order to create the offspring. For instance, in one point crossover, if S_1=111000 and S_2=000111 are the parents selected and if the crossover point is 2, two offsprings will be created which are S_1'=001000 and S_2'=110111. Then these two new offspring are placed into the population in the next generation. Figure 3.3 presents an example of a crossover operation.

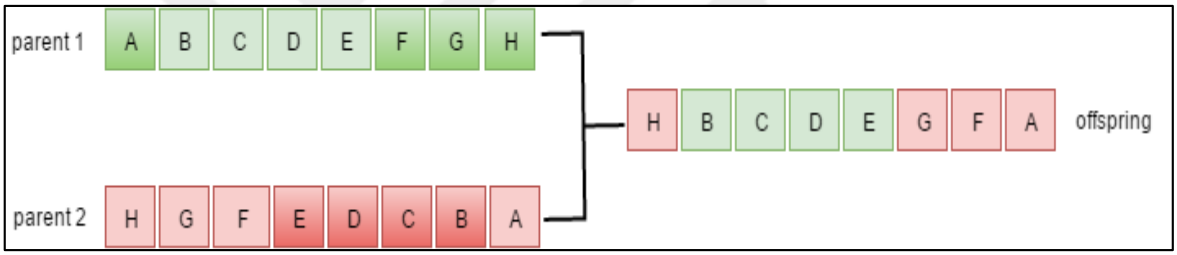

Figure 3.3. Crossover operation in genetic algorithms

After the new offspring is generated by crossover, usually, the mutation operator is also applied on the offspring. The operator provides random changes on gene values and hence deduces a random walk through the search space. This procedure keeps a diversity in the population. Figure 3.4 shows an example of Mutation.

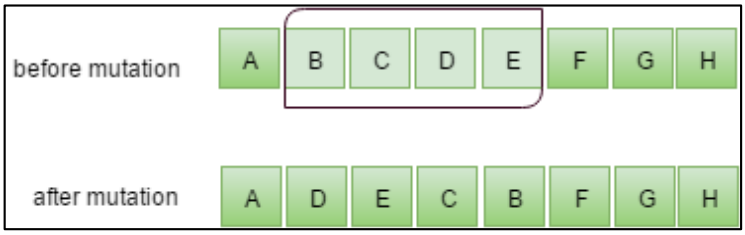

Figure 3.4. Mutation operation in genetic algorithms

Genetic algorithms solve problems by coding the problem as a bitstring. Therefore, genetic algorithms do not contain information about the problem at hand; they just provide a problem independent search on the individuals in the population. The algorithm starts by using a set of points in the search space. Hence, the search process does not usually stay trapped in a local optimal solution. However, in some cases, genetic algorithms converge to the local optimum rather than the global optimum solution. To prevent this situation, diversity can be increased or fitness function can be changed at each reproductive stage.

3.2. Multi-Objective Optimization

Genetic algorithms generate particular solutions with respect to a single objective. Nevertheless, many problems have a multi-objective nature. Additionally, the objectives can be in conflict with each other in many real-life problems. Solving these problems with respect to a single objective can cause unacceptable results. Even if normalization of different objectives into a single objective is possible, a more reasonable solution for the multi-objective problems is to generate a set of non-dominated solutions [39].

There are two approaches solving the multi-objective problems. As mentioned above, the first approach is to associate all objective functions into a single composite function. A single objective can be created by using the weighted sum of the objectives or by using a utility function. Then the problem is reduced to the appropriate selection of the utility function or the weighted sum. As a matter of fact, it can be very difficult to select these weigths accurately. The other solution is to create a constraint set containing a constraint value for each objective. The optimization method would return a set of solutions in this case, too. Therefore, researchers prefer to use the multi-objective approach instead of the first method.

In resume, the second approach is a multiobjective approach which defines a “Pareto optimal solution set” defined as a complete set of solutions for a Multi-Objective Optimization problem. The set of solutions that can be generated by a search algorithm is called “Pareto Front” and indicates the nature of the trade-off between the different objective functions. A Pareto optimal solution set consists of non-dominated optimal solutions with respect to each other. In multi-objective optimization problems, the output can be maximized or minimized in terms of each objective. If you consider the individuals in a population, a set of nondominated individuals would exist when the multi-objective

approach is used. The genetic search is carried out by improving this Pareto front towards a Pareto set that will be considered as optimal. Figure 3.5 (left) describes a Pareto front in terms of objectives.

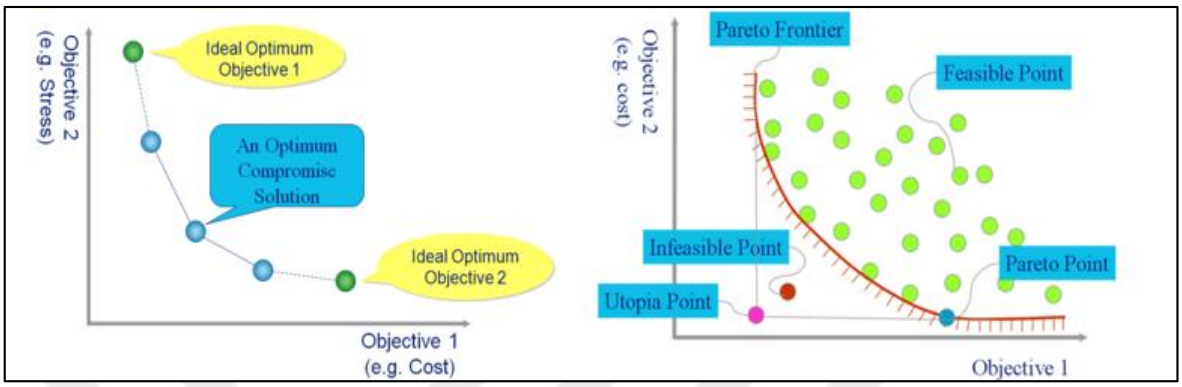

Figure 3.5. Pareto Front [40]

When the non-dominated solutions are connected to each other, they define a line which is called “Pareto line”. Pareto line separates the feasible and infeasible regions of the search space. Figure 3.5 (right), presents the Pareto line. In the case of more than two objectives this border can be represented as a surface. In resume, the Pareto line defines a border which contains feasible optimal combinations of the objectives. If the value of one objective is increased and the values of other objectives are kept constant, the corresponding individual would be no longer optimal. If the value of one objective is decreased and the values of other objectives are kept constant, the individual would move into the infeasible domain. The aim of the Pareto optimization methods is to find a number of points on the Pareto front giving distinct weigthing factors to distinct objectives. There is always an amount of loss in one objective in order to obtain a certain amount of gain in another objective while moving on the Pareto line. Pareto optimal solution sets are more preferable because they provide practical and more realistic solutions to real-life problems. The size of Pareto set usually gets larger when the number of objectives is increased.

The aim of a multi-objective optimization algorithm is to obtain the solutions in the Pareto optimal set. Therefore, a multi-objective approach has to achieve three important challenges. The first aim is that the obtained Pareto front should be as near as possible to the true Pareto front. True Pareto front indicates the set of globally optimal non-dominated solutions in the entire search space. Secondly, the solutions in the best-known Pareto set

have to be uniformly distributed in the Pareto front. The third one is that the whole spectrum of the Pareto optimal set should be contained in Pareto front obtained by the algorithm. It is worthy to mention that the approach investigates solutions at the excessive ends of the objective function space.

The formulation of the approach described above is provided in [39]. A minimization multi-objective problem with M objectives can be defined formally as follows:

$$f(y^*) = \{f_1(y^*), \dots, f_m(y^*)\} \tag{3.1}$$

where y^* is a k-dimensional decision variable vector $y = \{y_1, \dots, y_k\}$ in the solution space Y, which minimizes a given set of M objective functions. The solution space Y is generally limited by a series of constraints like $h_i(y^*) = a_i$for $i = 1, \dots, n$ which can be considered as bounds on the decision variables.

For minimization, a feasible solution z dominates another feasible solution w $(z > w)$, if and only if $f_j(z) \leq f_j(w)$ for $j = 1, \dots, M$ and $f_i(z) < f_i(w)$ for least one objective function i. The solution is Pareto optimal if it is not dominated by any other solution in the solution space. Pareto optimal set is the set of all feasible non-dominated solutions in Y.

3.3. NSGA-II

NSGA-II is a non-dominated sorting genetic algorithm implementation for multi-objective optimization. NSGA-II associates elitism. Elitism is the strategy that guarantees that the best individuals in the current generation contribute to the construction of the new population. The approach utilized determines the best individuals among the parents and the offspring generated. The individuals are classified according to the level of non-domination. Then, non-dominated individuals are selected for genetic operations for sure and they are directly transferred to the next population. What is more, the approach tries to obtain a uniform distribution for the Pareto front. This method is based on the crowding distance in order to guarantee diversity. The crowding distance is a measure of how close an individual is to its neighbors. Finally, constraints are applied by using dominance instead of the use of penalty functions.

The algorithm uses an evolutionary process including crossover, mutation operators and a selection method. The population is sorted into a hierarchy of sub-populations based on Pareto dominance. Then, similarity between members of each sub-grup is evaluated by using the crowding distance. Figure 3.6 presents the flow chart of the NSGA-II algorithm.

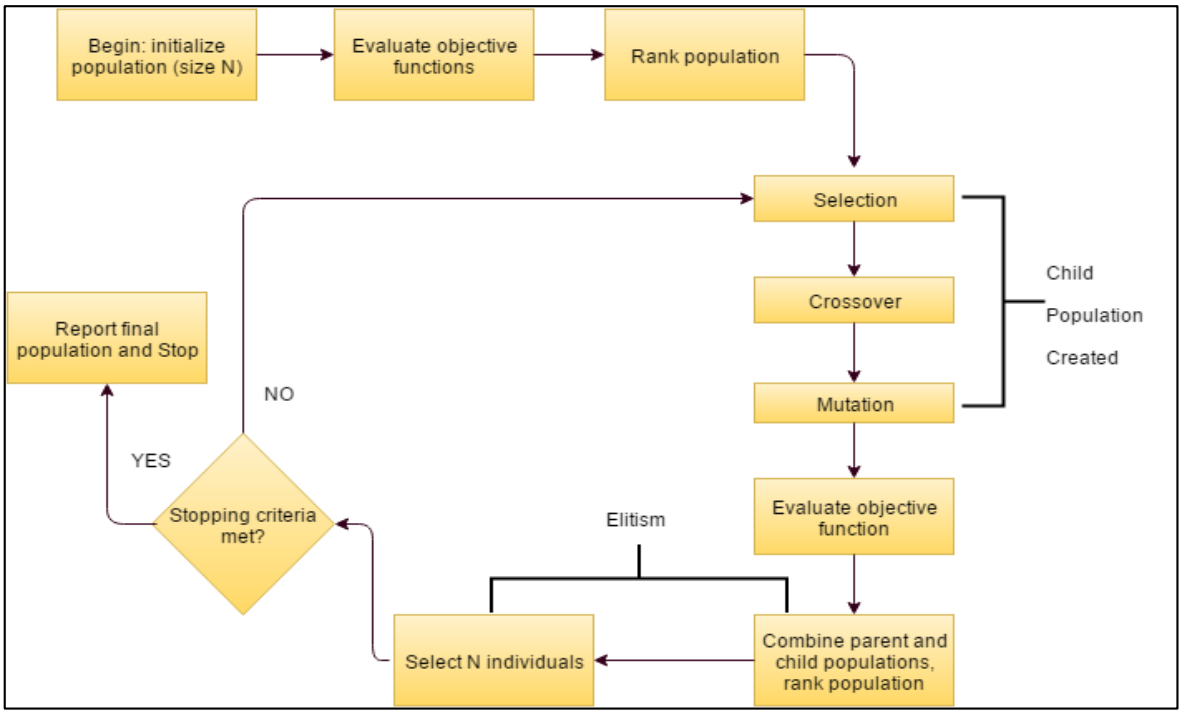

Figure 3.6. Flow chart of NSGA-II

The first step of the algorithm is to initialize the population based on the constraints and the range of variables used. After the initialization, the population is sorted based on non-domination. The first front consists of a completely non-dominant set in the current population. Namely, these are the individuals that cannot be dominated by any other individual in the population. The second front is dominated by the individuals in the first front only. And it goes on like this. Fitness values are assigned to each individual according to the front they belong to. The fitness value of 1 is given to individuals in first front and the fitness value of 2 is assigned to individuals in the second one and so on. After assigning fitness values, crowding distance is calculated for each individual. As noted above, the crowding distance is a measure of how close an individual is to its neighbors. In the m dimensional space, the crowding distance provides to find the euclidian distance

between individuals that exist in the same front in a front by using their m objective values.

Algorithm 3.1. Algorithm for NSGA-II

```
Input : Population_{size,ProblemSize}, P_crossover, P_mutation
Output : Children
Population = InitializePopulation(Population_{size,ProblemSize})
EvaluateAgaintsObjectiveFunctions(Population)
FastNondominatedSort(Population)
Selected = SelectParentsByRank(Population, Population_size)
Children = CrossoverAndMutation(Selected, P_crossover, P_mutation)
while (−StopCondition ())
    EvaluateAgainstObjectiveFunctions(Children)
    Union = Merge(Population, Children)
    Fronts = FastNondominatedDort(Union)
    Parents = ∅
   Front_L = ∅
    for (Front_{i ∈Fronts}) do
       CrowdingDistanceAssignment(Front_i)
       if (Size(Parents)+Size( Front_i )> Population_size) then
          Front_L = i
          break()
       else
          Parents = Merge( Parents, Front_i )
       end
   end
   if (Size(Parents) < Population_size) then
      Front_L = SortByRankAndDistance( Front_L )
       for (P_1 To P_{Population_size−SizeFront_L}) do
            Parents = P_i
       end
   end
   Selected = SelectParentsByRankAndDistance(Parents, Population_size)
   Population = Children
   Children=CrossoverAndMutation(Selected,Population_crossover, Population_mutation)
end
return(Children)
```

Parents are selected from the population according to the fitness values and crowding distance. A tournament is carried out between two individuals to determine a parent. An individual is selected if its fitness is better than the other or if its crowding distance is greater than the other. Since all the previous and current best individuals are added in the population, elitism is ensured. Offsprings are created by the selected parents using

crossover and mutation. The population with the current offspring and the current individuals are sorted again based on non-domination and the best individuals form the next population. Algorithm 3.1 presents the NSGA-II algorithm.

In the above pseudocode, the *SortByRankAndDistance* function directs the population into a hierarchy of non-dominated Pareto fronts. The *CrowdingDistanceAssignment* function calculates the distance between members of each front. The *CrossoverAndMutation* function operates the genetic operators crossover and mutation. The members of the population are differentiated by rank (fitness values) with both *SelectParentsByRankAndDistance* and *SortByRankAndDistance* functions.

4. ANALYSIS & DESIGN

4.1. Determining the Mean Value of the Surface Areas

In this section, the techniques of determining the mean value of the surface areas by the detection of buildings are explained in detail. For this study, Google Maps images are utilized. The method comprises the following steps: Building detection on Google Maps images, surface area calculation of the buildings and weighted mean value calculation of the surface areas.

4.1.1. Building detection on Google maps images

In literature, there are a lot of work and methods for the detection of buildings. In recent years, Google provides the information of the buildings areas on Google Maps and this study will be based on this information.

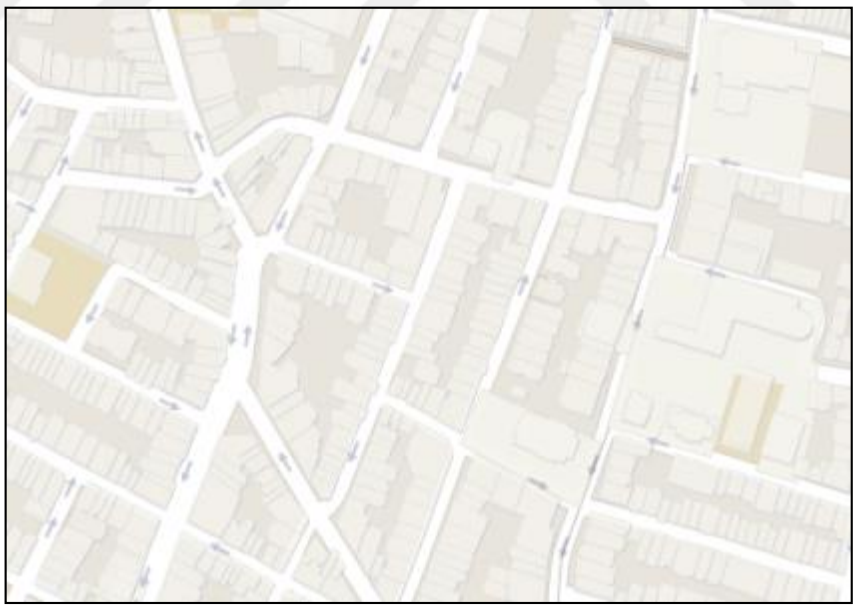

Figure 4.1. Building representation on Google Maps

First, an area is determined as the region where buildings are localized. At this stage, the labels are removed from Google Maps in order to avoid confusion created by letters lying in the buildings area. After this step the resulted view is saved as an image. The detection

of buildings is based on the detection of the particular color that Google Maps assigns to buildings. It is worth to mention that in general, in Google Maps, the representation of buildings is provided through a very limited number of colors. We are based on the RGB color model that defines the intensity of red, green and blue values. Gray color represents the roof of the buildings on Google Maps images. The values of RGB color model representing the color of the buildings are found from Google Maps images. R value is between 236 and 243, G value is between 235 and 242 and B value is between 230 and 236. By using the RGB color model of the scene, the corresponding gray color is easily isolated from the saved image. Then the buildings are found in a scene. We finally create a binary image where we assign a white color for buildings and the other colors of the image are assigned black. Finally, this binary image will give the position and number of buildings. Algorithm 4.1 represents the procedure of binarization for building detection.

Algorithm 4.1. Building detection based on RGB color model

```
Mat R = channels[0]
Mat G = channels[1]
Mat B = channels[2]
for i = 0, i < myImage.rows, i = i + 1 do
  for j = 0, j < myImage.cols, j = j + 1 do
    if ((R(i,j) >= 236 AND R(i,j) <= 243) AND
        (G(i,j) >= 235 AND G(i,j) <= 242) AND
        (B(i,j) >= 230 AND B(i,j) <= 236)) then
        myImage(i,j)[0] = 255
        myImage(i,j)[1] = 255
        myImage(i,j)[2] = 255
    else
        myImage(i,j)[0] = 0
        myImage(i,j)[1] = 0
        myImage(i,j)[2] = 0
    end
  end
end
```

4.1.2. Surface area of the buildings

In the previous sections we show that RGB color model is used to provide us the place of the buildings and a binary image is created. Then, a morphological operation based on the

erode procedure is applied in order to clearly separate the building between them. The erosion operation is applied to remove pixels on object boundaries. In erosion, the value of the output pixel is the minimum value of all the pixels in the input pixel's neighborhood. If any of the pixels is set to 0, the output pixel is set to 0 in a binary image. The erosion formulation can be described as follows:

$$dst(x,y) = min_{(x',y'):element(x',y')\neq 0} src(x+x', y+y') \tag{4.1}$$

After applying the erode operation, the bounding boxes of the blobs are taken. A blob is a group of pixels in an image that shares some common properties such as brigthness or color and is different to its surrounding region. In other words, a blob represents a region of the image in which some properties are constant. All the points in the blob can be considered in some sense to be similar to each other. In figure 4.2 we can observe a binary image extracted from Google Maps which represents the buildings extracted from the RGB color model in the form of blobs (white regions) together with their corresponding bounding boxes (green rectangles). In this image an erosion is being applied. After erosion, the buildings are the white blobs and the green rectangles are their corresponding bounding boxes.

Figure 4.2. Building detection based on the RGB color model

4.1.3. Mean value of the surface areas

After finding the blobs which correspond to the detected buildings, we calculate the weigthed mean of the surface area of the buildings based on the center and the area of the bounding boxes of the blobs. The weighted mean formula can be described as follows:

$$X_C = \frac{\sum_{i=1}^{N} S_i x_i}{\sum_{i=1}^{N} S_i} \tag{4.2}$$

$$Y_C = \frac{\sum_{i=1}^{N} S_i y_i}{\sum_{i=1}^{N} S_i} \tag{4.3}$$

where N is the number of bounding boxes and $C_i(x_i, y_i)$ is the current center of the bounding box. Finally, $C_{mean}(X_C, Y_C)$ is the weighted average mean location. Figure 4.3 represents the result of finding the weighted mean in an area. There, the weighted average mean location is shown on Google Maps with orange marker.

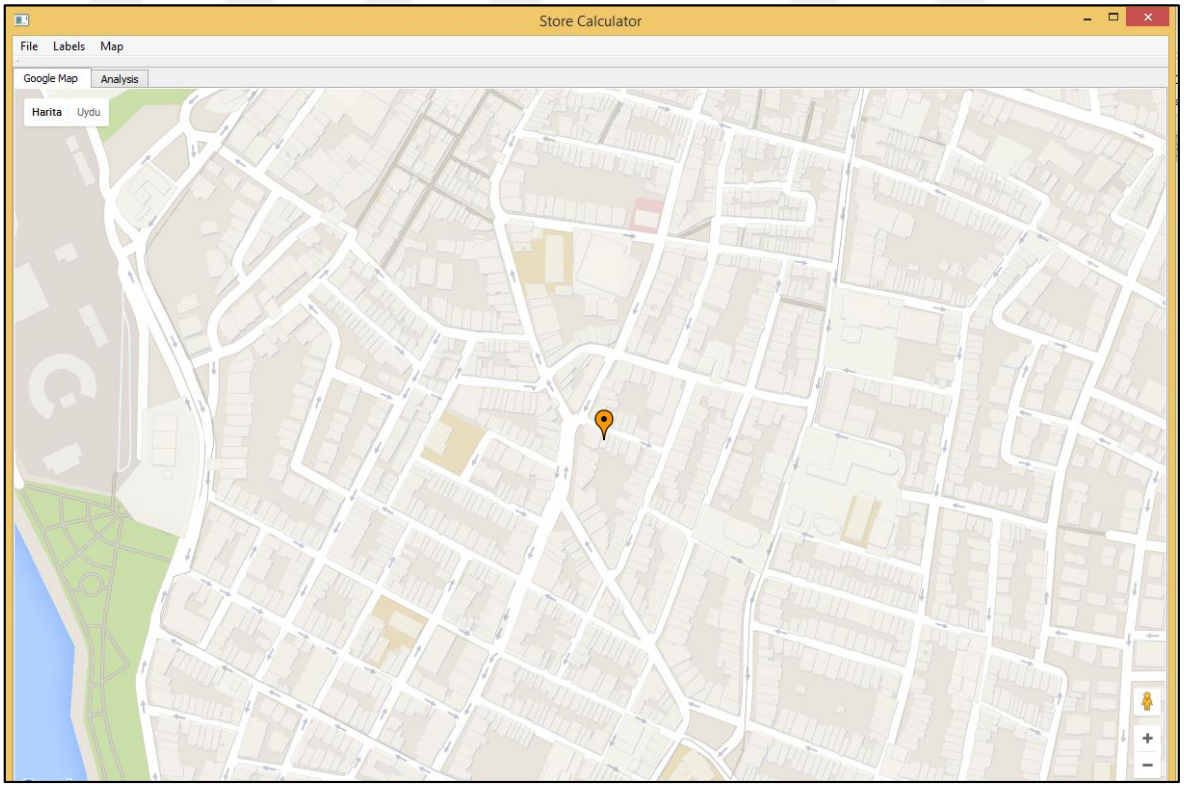

Figure 4.3. The weighted average mean location with orange marker

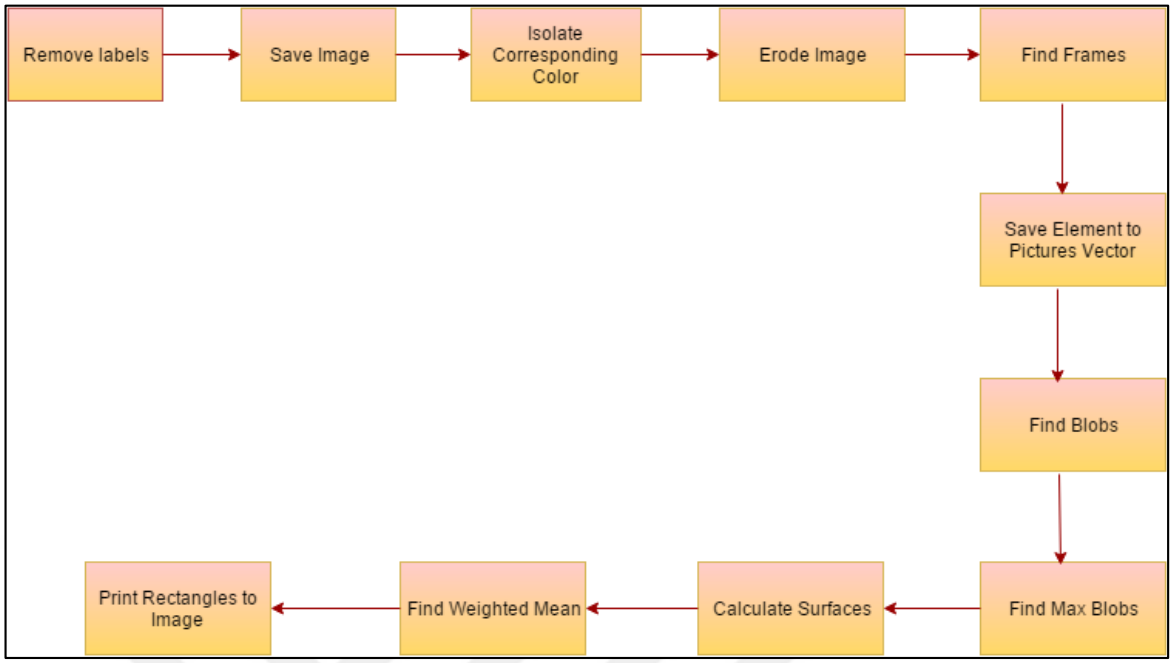

Figure 4.4. Flow chart of the building detection

Figure 4.4 represents the flow chart for detecting buildings from Google Maps. The method for the detection of the buildings starts with a color detection based on the RGB color model. After the corresponding color is isolated, a binary image is created based on the previous detection method. Then, the morphological erode operation is applied in order to clearly separate buildings. After, the boundry boxes of the blobs are taken from the image. Finally, the weighted mean of the surface areas is calculated based on them.

4.2. Finding Locations

The restaurants, the metro stations and the existing grocery stores are the places that we will take in consideration in this study for the calculation of the location of a candidate grocery store. The latitude and longitude values of the restaurants, the metro stations and the existing grocery stores in a particular place are taken from the Maptriks application. Maptriks is a web program based on Google Maps. It gives address information, population information, latitude and longitude values of public locations, grocery stores and important places. The latitude and longitude values of the metro stations and the restaurants of a particular location are taken from Maptriks and written on a specific text

file. The latitude and longitude values of the existing grocery store locations are taken from Maptriks and written on another text file.

In this study, a user can select an area in the range of Kadıköy. Google Maps gives the latitude and longitude information of the north-east and the south-west of the selected region. The system is able to then find the boundary latitude and longitude values of the selected place.

The latitude and longitude values of the restaurants, the metro stations and the existing grocery locations are read from the text files. The latitude and longitude values of these places which are in this selected place are found by the system and written again in different text files. These text files will be used in the optimization procedure. The system can show the location information of the restaurants, the metro stations and the existing grocery stores with the pins on Google Maps.

4.3. Determining Store Locations by Using Multi-Objective Optimization

In this study, two different objectives have been used for determining the location for a grocery store. The first objective takes into consideration the nearby restaurants and metro stations. The second objective is based on the locations of other grocery store locations. Certainly, the first objective has to be minimized and the second one maximized. In other words, the store locations close to restaurants and metro stations is preferred, but on the other side the location has to be away from other grocery stores as much as possible. It would be hard to solve this problem with a genetic algorithm that generates solutions with respect to a single objective. Therefore, this multi-objective problem is solved with NSGA-II.

The coordinates of public places like restaurants and metro stations and the other grocery store coordinates are read from input text files. After the read operations, the coordinates are normalized into values between 1 and 1000. These coordinate values are kept simply in an array structure. Then the initialization of the population is carried out by NSGA-II and the chromosomes for the first generation are created. Note that each chromosome is a potential solution to the problem and each chromosome provides a possible location for the new grocery store in our case. Therefore each chromosome in the population consists of two variables that will denote the (X, Y) coordinates of the new store. Initially, the

chromosomes are initialized with random numbers between 1 and 1000. Then, the objective function values and constraints for the individuals in the population are evaluated. As noted in the previous section, the chromosomes are also sorted based on non-dominance and ranking and crowding distance values are also assigned to the individuals in the population.

After these first steps, the process of generating the next population is started. A selection operation is done in order to choose the parents for the crossover operation. In NSGA-II a tournament selection is used. The tournament selection randomly selects two individuals from the population. It compares the selected individuals in terms of objective functions in order to determine which one is going to be selected as the parent. In fact, the two individuals are compared with each other in terms of ranking values and crowding distance. Then a random value between zero and one is generated. This value is compared to a pre-determined selection probability. If the generated random value is greater than the selection probability, the weak candidate is chosen, otherwise the better candidate is chosen as the parent. Algorithm 4.2 presents the selection operation procedure. In Algorithm 4.2. $F_i(d_p)$ is the crowding distance, $<_n$ is the crowded comparison operator, p and q are individuals.

Algorithm 4.2. Selection Operation

Individuals in front F_i have their rank $p_{rank} = i$
if $p <_n q$ **then**
  $p_{rank} < q_{rank}$
  if p and q belong to the same front F_i then crowding distance **then**
    $F_i(d_p) > F_i(d_q)$
  end
end

A Simulated Binary Crossover (SBX) procedure is used in the NSGA-II algorithm. SBX simulates the binary crossover observed in nature. It can be described as follows:

$$c_{1,k} = \frac{1}{2}\left[(1 - \beta_k)p_{1,k} + (1 + \beta_k)p_{2,k}\right] \tag{4.4}$$

$$c_{2,k} = \frac{1}{2}\left[(1 + \beta_k)p_{1,k} + (1 - \beta_k)p_{2,k}\right] \tag{4.5}$$

where $c_{i,k}$ is the i^{th} child with k^{th} component, $p_{i,k}$ is the selected parent. The value of the k^{th} component in the offspring is determined based on the values of the corresponding genes in the parents. The contribution of each parent is based on the value $\beta_k (\geq 0)$ which is a sample from a random number generated having the density:

$$p(\beta) = \frac{1}{2}(n_c + 1)\beta^{n_c}, \text{ if } 0 \leq \beta \leq 1 \tag{4.6}$$

$$p(\beta) = \frac{1}{2}(n_c + 1)\frac{1}{\beta^{n_c+2}}, \beta > 1 \tag{4.7}$$

n_c is the distribution index for crossover in equation 4.6 and 4.7. After the selection operation, the NSGA-II algorithm does the polynomial mutation. Polynomial mutation can be described as follows:

$$c_k = p_k + (p_k^u - p_k^l)\delta_k \tag{4.8}$$

where c_k is again the k^{th} component of the child. This value is determined based on the corresponding gene p_k in the parent with p_k^u being the upper bound on the parent component, p_k^l is the lower bound and δ_k is a small variation which is calculated from a polynomial distribution by using.

$$\delta_k = (2r_k)^{\frac{1}{n_m+1}} - 1, \text{ if } r_k < 0.5 \tag{4.9}$$

$$\delta_k = 1 - [(2(1 - r_k)]^{\frac{1}{n_m+1}}, \text{ if } r_k \geq 0.5 \tag{4.10}$$

where r_k is an uniformly sampled random number between $(0,1)$ and n_m is a mutation distribution index that takes any non-negative value.

Then the algorithm performs the evaluation of individuals in the population. In this part, the algorithm performs distance calculations in order to determine the objective values. The objective value calculations are explained in sections 4.3.2. and 4.3.3.

After that, the non-domination sorting explained in the previous section is performed. The sorting algorithm can be described in Algorithm 4.3.

Algorithm 4.3. Non-domination sorting Algorithm

```
for each individual p in main population P do
  S_p = ∅
  n_p = 0
  for each individual q in P do
    if p dominated q then
      Add q to the set S_p i.e. S_p = S_p ∪ {q}
    else if q dominates p then
      n_p = n_p + 1
    end
  end
  if n_p = 0 then
    p belongs to the first front
    set rank of individual p to one p_rank = 1
    update the first front set by adding p to front one F_1 = F_1 ∪ {p}
  end
end
this is carried out for all the individuals in main population P.
initialize the front counter to one i = 1
following is carried out when F_i = ∅
θ = ∅
for each individual p in front F_i do
  for each individual q in S_p do
    n_q = n_q − 1
    if n_q = 0 then
      q_rank = i + 1
      update the set θ with individual q i.e. θ = θ ∪ q
    end
  end
end
i = i + 1
the set θ is the next front and hence F_i = θ
```

In algorithm 4.3, S_p is the set that contains all the individuals dominated by p. n_p is the number of individuals that dominate p and n_q is the domination count for individual q. After non-domination sorting is completed, the crowding distance is assigned to the individuals. Crowding distance is assigned front wise since comparing the crowding distance between two individuals in different fronts does not make sense. The crowding distance algorithm is presented in algorithm 4.4.

Algorithm 4.4. Algorithm of Crowding-Distance

for each front F_i, n **do**
  $F_i(d_j) = 0$
  for each objective function m **do**
    $I = \text{sort}(F_i, m)$
    $I(d_1) = \infty$ and $I(d_n) = \infty$
    for $k = 2$ to $(n-1)$ **do**

$$I(d_k) = I(d_k) + \frac{I(K+1) * m - I(K-1) * m}{(f_m^{max} - f_m^{min})}$$

    end
  end
end

In the algorithm, n is the number of individuals. First, the distance is initialized to be zero for all the individulas. j corresponds to the j^{th} individual in front F_i and $I(k) * m$ is the value of the m^{th} objective function of the k^{th} individual again in front F_i .

In the proposed method, when NSGA-II is initiated, the population is initialized. Then it evaluates the objective functions for the first generation. In this study, the first objective function comprises the minimization to the locations of restaurants and metro stations and the second objective is doing the maximization to the locations of stores. At first, a minimum distance calculation between the random numbers generated from NSGA-II and the coordinates of public locations is done. Then, a maximum distance calculation between the random numbers generated from NSGA-II and the coordinates of store locations are performed. After evaluating the objective functions, the algorithm ranks the population. Next, the algorithm proceeds to the procedures of selection, crossover and mutation. For the next generations the algorithm evaluates again the objective functions mentioned above. After evaluating the objective functions, the algorithm combines parent and child population followed by the ranking of the population. Finally, N individuals are selected. If the stopping criterion is met, the final population is reported and the algorithm ends. If the criterion for ending the algorithm is not met, the algorithm goes to the selection operation and continues from that point. The flow chart of the algorithm is shown in Figure 4.5.

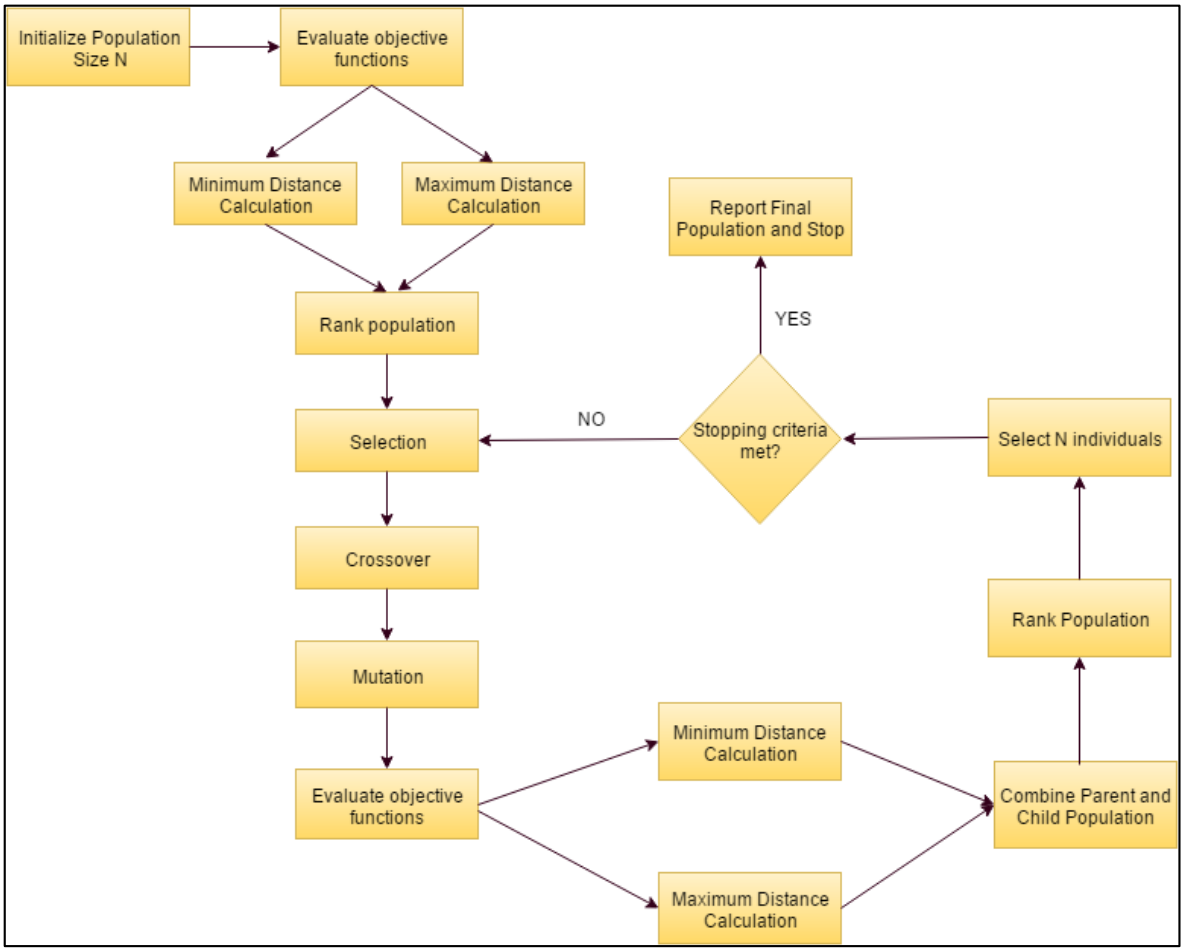

Figure 4.5. Flow chart of the algorithm

4.3.1. Conversion of latitude and longitude values

As defined in the introduction section, the two objective functions utilized in this study are based on the locations of public places like metro stations and restaurants and locations of other grocery stores. The stores and the public location coordinates are read directly from files which are provided by the software mentioned before. The coordinates are in terms of latitude and longitude values and as the geographical area we select it relatively small (some hundreds of meters). They have a very small range such as between 49.5678 to 49.5692. Therefore, the values are converted to values between 1 and 1000 before they are used in the genetic algorithm. The aim is to get rid of possible truncation errors. The conversion formula can be described as follows:

$$X_{convert} = \frac{(\|X - X_2\|)}{\|X_1 - X_2\|} * 1000, \quad \text{if } X > X_2 \tag{4.11}$$

$$X_{convert} = \frac{(\|X - X_1\|)}{\|X_1 - X_2\|} * 1000, \quad \text{if } X > X_1 \tag{4.12}$$

$$Y_{convert} = \frac{(\|Y - Y_2\|)}{\|Y_1 - Y_2\|} * 1000, \quad \text{if } Y > Y_2 \tag{4.13}$$

$$Y_{convert} = \frac{(\|Y - Y_1\|)}{\|Y_1 - Y_2\|} * 1000, \quad \text{if } Y > Y_1 \tag{4.14}$$

where X_1 and X_2 are the latitude values of the corresponding location on the map. Certainly, $X_{convert}$is a value between 1 and 1000. Similarly, Y_1 and Y_2 are the longitude values of the selected location. And again, $Y_{convert}$ is between 1 and 1000.

At the end, the genetic search is carried out on coordinates having values between 1 and 1000. After the genetic search is finished, the values are converted back into longitude and latitude values in order to determine the positions of the solutions on the map.

$$X_{final} = \frac{(X_1 - X_2)}{1000} * X + X_2, \quad \text{if } X_1 > X_2 \tag{4.15}$$

$$X_{final} = \frac{(X_2 - X_1)}{1000} * X + X_1, \quad \text{if } X_2 \geq X_1 \tag{4.16}$$

$$Y_{final} = \frac{(Y_1 - Y_2)}{1000} * Y + Y_2, \quad \text{if } Y_1 \geq Y_2 \tag{4.17}$$

$$Y_{final} = \frac{(Y_2 - Y_1)}{1000} * Y + Y_1, \quad \text{if } Y_2 \geq Y_1 \tag{4.18}$$

The above equations denote this conversion procedure X and Y are the coordinates generated by NSGA-II. X_{final} and Y_{final} are the longitude, latitude values of the solution produced by NSGA-II.

4.3.2. Objective 1: Distance calculation for restaurants and metro stations

The chromosomes generated by NSGA-II consist of two values between 0 and 1000. These two numbers represent the coordinates of a possible location for the new grocery store. The first criterion utilized in this study is to have the new store location close to the metro stations and the restaurants. Therefore, the distances between the public locations and the coordinates generated by NSGA-II are calculated.

In this study, the public locations are selected from distinct categories such as restaurants and metro stations. The first objective is to find the average distance from the five closest restaurants or metro stations. The distance calculation formula between two points i and j can be defined as follows:

$$d_{ij} = \sqrt{((X_i - X_j)^2 + (Y_i - Y_j)^2)} \tag{4.19}$$

where X_i and Y_i are the coordinates given by the chromosome i, X_j respresents the converted latitude value of the public place j and Y_j represents the converted longitude value of the same place.

According to the method utilized, initially the distances between the coordinates of the new store location and five random public places are calculated and placed into an array. Then these distances are sorted. Since only five individuals exist in the array, a simple sorting algorithm is preferred for the procedure.

After having these five distances sorted in the array, for each remaining public place, the distance calculation is carried out and then if this distance is smaller than the last distance in the array, this new distance is inserted into the correct position in the array. For this operation, the distances in the array that are larger than this new instance is shifted one position to the right and the largest distance is removed from the array. Certainly by using this method only an array of five elements is sufficient to determine the five minimal distances. When the procedure is applied to all public places, the array would contain the five minimum distances for the set. The average of these five minimum distances gives the final value for the first objective of the corresponding chromosome.

Algorithm 4.5. Calculation of an array containing five places with minimum distance

```
i = 0
array_size = 5
for i = 0, i < array_size, i = i + 1 do
   min_dist = sqrt((X1 − store [i][0]) ∗ (X1 − store [i][0]) ∗ +(Y1 − store [i][1])
                ∗ (Y1 − store[i][1]))
   min_fivedist[i] = min_dist
end
for k 0, k < array_size − 1, k = k + 1 do
   for m = 0, m < array_size − 1 − k, m = m + 1 do
      if min_fivedist[m] > min_fivedist[m + 1] then
        Swap(min _fivedist[m], min _fivedist[m + 1])
      end
   end
end
for i = array_size − 1 , i < 1000, i = i + 1 do
   min _dist = sqrt((X1 − store [i][0]) ∗ (X1 − store [i][0]) ∗ +(Y1 − store [i][1])
                ∗ (Y1 − store[i][1]))
   for j = 0, j < array_size, j = j + 1 do
      if min_dist > min_fivedist[j] then
         if min_dist <= min_fivedist[j + 1] AND j ! = array_size then
           n = 1
            while array_size − n ! = j + 1 do
               min_fivedist[array_size − n] = min_fivedist[array_size − n − 1]
              n = n + 1
            end
           min_fivedist[j + 1] = min_dist
         end
       else
         n = 1
         while array_size − n ! = j do
            min_fivedist[array_size − n] = min_fivedist[array_size − n − 1]
            n = n + 1
         end
         min_fivedist[j] = min_dist
         break
     end
  end
end
for i = 0, i < array_size, i = i + 1 then
   sum_min_dist = sum_min_dist + min_fivedist[i]
end
 mean_min_dist = sum_min_dist / array_size
```

Algorithm 4.5 presents the procedure for the creation of the array containing the five minimum distances. In Algorithm 4.5 store array includes the coordinates of the stores which are converted between 1 and 1000. Min_dist is the distance between the coordinates produced by NSGA-II and store coordinates. Min_fivedist array includes the minimum five distances between the stores and coordinates proposed.

4.3.3. Objective 2: Maximum distance to other grocery stores

The second objective in this study is to guarantee that the new store location should be away from the other grocery store locations. Therefore, the distances between the other store locations and the coordinates given by a chromosome are calculated in order to set the second objective value.

The second objective is defined by the distance of the closest grocery store to the coordinates of the chromosome. As a consequence, distance calculations between the given coordinates and all other stores are carried out and the minimum distance is simply set as the second objective of the chromosome. The minimum distance is used here, since this is a maximizing objective and the aim is to maximize the distance to the closest store by the genetic search process. The same distance calculation presented in the previous section in equation 4.19 is utilized for the second objective too.

$$d_{final} = -1 * d_{min} \tag{4.20}$$

NSGA-II has been designed for minimization objectives. However, our second objective has to be maximized in order to find a store location far away from other stores. For this purpose, the objective value is utilized after multiplying it by -1 according to the equation 4.20. Therefore when NSGA-II tries to minimize the objective it is possible to maximize the distance to other store locations. Algorithm 4.6 presents the calculation of the second objective function. Store array holds the coordinates of the store locations which are converted to a scale between 1 and 1000. f_1 holds the first distance between the first store location and coordinates produced by NSGA-II. f_2 holds the distance between the store locations and coordinates proposed.

Algorithm 4.6. Calculation of the second objective function maximizing the distances from existing stores

```
i = 0
f1 = sqrt((X1 − store [0][0]) ∗ (X1 − store [0][0]) ∗ +(Y1 − store [0][1]) ∗ (Y1
                − store[0][1]))
f2 = sqrt((X1 − store [i][0]) ∗ (X1 − store [i][0]) ∗ +(Y1 − store [i][1]) ∗ (Y1
                − store[i][1]))
if (f2 < f1) then
    f1 = f2
end
i = i + 1
do...while(store [i][0]! = −1 AND store [i][1]! = 0)
if (i == 0) then
   result = 0
else
   if (f1 < 0) then
      f1 = 0.1
   end
   result = 1/f1
end
```

4.3.4. Reporting feasible results

Throughout the genetic search, at each iteration the offspring population is combined with the current population and a selection is performed to set the individuals of the next generation. The selection process guarantees that all the previous and current best individuals would be added in the next population. Population is sorted based on non-domination as explained in section 4.3. The new generation is filled by each front subsequently until the population size exceeds the current population size. The process is repeated in order to generate the subsequent generations. Hence elitism is ensured and it is possible to have the best individuals in the final population of the genetic search.

After the genetic search is finished, a Pareto front consisting of non-dominated solutions is obtained. Usually the number of solutions that exist in this Pareto front is quite high. Returning all of them as possible store locations would not be reasonable. That is why only five store locations are selected from the Pareto front and they are returned as the possibilities for the new store locations. In order to determine these five solutions, the

individuals in the Pareto front are filtered according to a strategy. The method used is to filter out one individual that is the worst in the set in terms of one of the objectives. For instance, if we start with the first objective, the individual that is worst in terms of the first objective would be removed from the set initially, then we would do the same thing in terms of second objective and then the first objective. By altering between the objectives, the individuals at the two ends of the Pareto front would be eliminated. Hence, the five individuals that will be selected at the end would be from the central part of the Pareto front where the individual would have a balanced value for both objectives.

Algorithm 4.7. Selection of five solutions from the central region of the pareto front

```
k = −1
for j = 0, j < pop_size − 5, j = j + 1 do
    temp_maxobj1 = 0
    temp_maxobj2 = 0
    k = k + 1
    if k == 2 then
      k = 0
      if k == 0 then
        for i = 1, i < pop_size, i = i + 1 do
            if objectives1[i] > temp_obj1 then
              temp_obj1 = objectives1[i]
              count_maxobj1 = i
            end
        end
      else if k == 1 then
        m = 0
        while objectives2[m] == −1 do
          m = m + 1
        end
        temp_obj2 = objectives2[m]
        count_minobj2 = m
        for i = m + 1, i < pop_size, i = i + 1 do
            if objectives2[i] < temp_obj2 AND objectives2[i] > 0 then
              temp_obj2 = objectives2[i]
              count_minobj2 = i
            end
        end
       objectives1[count_minobj2] = −1
       objectives2[count_minobj2] = −1
    end
end
```

Algorithm 4.7 shows the procedure of keeping five solutions from the central region of the Pareto front. In algorithm 4.7 *objective1* array holds the values of *objective1* and *objective2* array holds the values of the objective2. pop_{size} is the size of the population. k, j, m are the constant variables. *Temp_obj1* holds the maximum value of the *objective1* and *temp_obj2* holds the minimum value of the objective2. Count_maxobj1 holds the index of the maximum value of the objective1 and *count_minobj2* holds the index of the minimum value of the objective2.

After five locations are determined, these results are converted into geographical coordinates as latitude and longitude. The final five results are shown on Google Maps. The location which has the minimum distance to the weighted average surface areas point is chosen as the solution of the system.

5. IMPLEMENTATION

In this study, the proposed method is implemented with C++ and OpenCV libraries. The language for NSGA-II library is C and the recommended image processing library is OpenCV 2.4.5. The NSGA-II library contains routines for plotting the objective data on realtime using gnuplot. The C code in NSGA-II library has been written for posix compliant operating systems and uses standard piping method provided by GNU C library. The Makefile in the NSGA-II library has been provided for compiling the program on linux systems. The Makefile is edited according to this study's need. The C library of NSGA-II is compiled on Windows with a QT 5.3.1 compiler. A Graphical User Interface (GUI) is designed to simulate the tests in a computer environment and this GUI is implemented using QT 5.3.1.

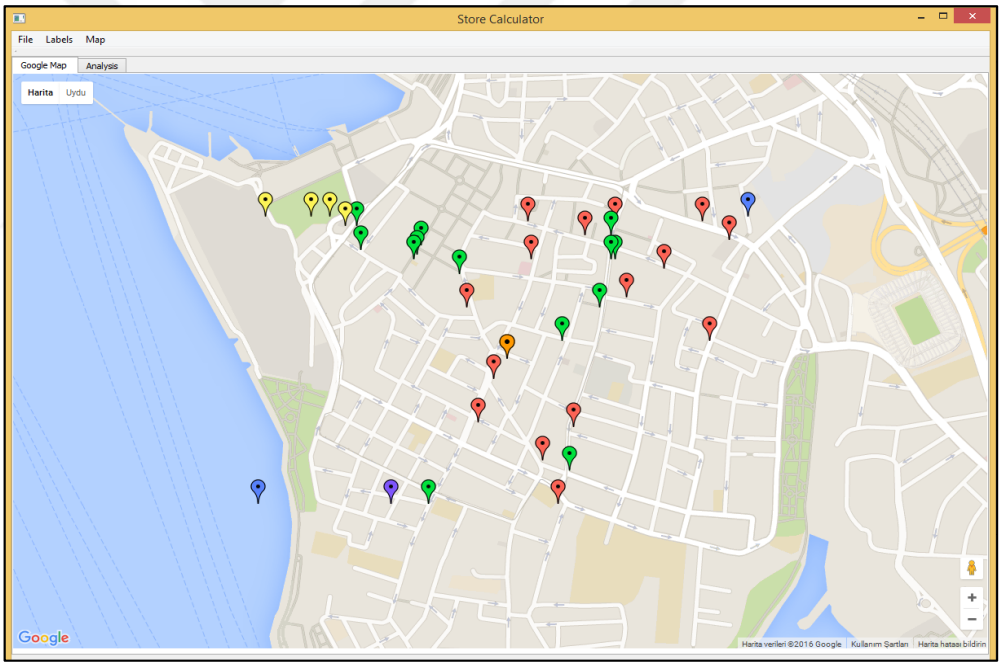

Figure 5.1. The graphical user interface

In the GUI the simulated field is displayed to the user on the screen. The results of the tests are shown on this image (Figure 5.1). The restaurants and metro stations are shown on Google Maps with green markers and the store locations are shown on Google Maps with red markers. The suggested store locations which are calculated in NSGA-II algorithm are shown with yellow markers, the range of the selected area is shown with blue markers at

the bottom left and top right of the region, the solution of the multi-objective algorithm is shown by yellow markers and the location of the weighted mean point of the surface areas is shown on Google Maps with an orange marker. When the user selects a marker with the mouse, the balloon showing the name of the current location is poped out. Figure 5.1 presents the markers in a Google Map, when the application has been executed for a particular region.

The proposed method is implemented with an object oriented approach. The system is composed of a number of classes and a class hierarchy. The classes and their relations are shown in the Figure 5.2. The user interaction with the GUI is handled by a class named 'MainWindow'. This class is responsible for the interactions with the user and displaying the results. When the system starts, the current area information from Google Maps is downloaded to our application.

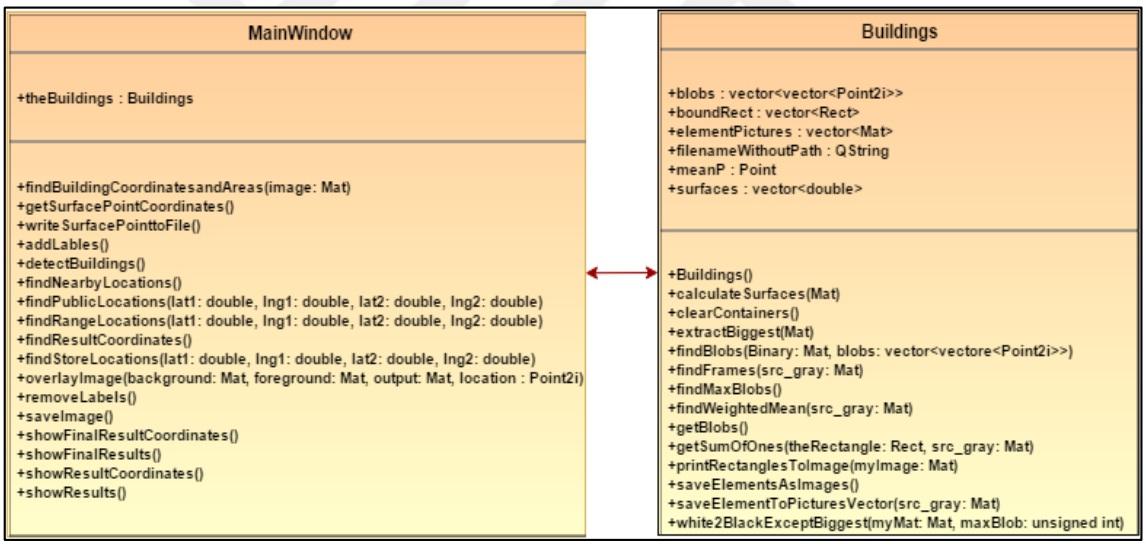

Figure 5.2. The class diagram

The 'MainWindow' class is responsible for the following tasks: saving image from Google Maps, finding store and public locations, removing and adding labels on Google Maps, initiating NSGA-II algorithm and displaying results on Google Maps. It also calls 'Buildings' class responsible for the buildings' detection. The 'Buildings' class comprises tasks as calculating surfaces, finding bounding boxes, clearing containers and calculating the average mean 2D point value extracted from the bounding boxes of the surface areas. The 'Buildings' class is called by the 'MainWindow' class when the user wants to detect

buildings. The detection of the buildings, the calculation of the bounding boxes of the surface areas and the calculation of the average mean point are found in ‘Buildings’ class and the results are returned on ‘MainWindow’ classes.

After starting the system, when the user chooses the ‘Labels’ dialog menu, she/he has the option add or remove labels from ‘Add Labels’ and ‘Remove Labels’ dialog menus. If the user chooses ‘File’ dialog menu, the user can exit from the system with the ‘Exit’ dialog menu. When the user chooses ‘Map’ dialog menu, he/she can select the following ‘Detect Buildings’, ‘Save Image’, ‘Find Nearby Locations’, ‘Show Results’, ‘Show Final Results’ operations. When the user chooses the ‘Save Image’ menu, the selected area from Google Maps is saved as an image. Moreover, when the user chooses ‘Detect Buildings’ operation, the following set of actions is performed: first the labels are removed from Google Maps, then the selected area on Google Maps is saved as an image. After the image is saved, the buildings are detected by calling the ‘Buildings’ class from the ‘MainWindow’ in order to execute the appropriate functions. The result of detection of buildings and weighted mean average point of the surface areas is shown on the ‘Analysis’ tab. When the user chooses the ‘Find Nearby Locations’ menu, the coordinates of public locations such as restaurants and metro stations and the coordinates of store locations are found in the selected area. These coordinates are read from the corresponding text files and the coordinates in the selected area are found from these coordinates and those found in the selected area are written on a file. Then the results are shown on Google Maps. By selecting the ‘Show Results’ menu, first the labels are removed from Google Maps, then the selected area on Google Maps is saved as an image. After saving the image, the buildings are detected and the average weighted mean point of the surface areas is calculated and displayed on Google Maps. Moreover, the nearby public and store locations are found and shown on Google Maps. Then the NSGA-II algorithm is initiated. The coordinates of restaurants, metro stations and stores are read from the file and transferred to the NSGA-II algorithm. The objective functions comprise the minimization of the distance between public and proposed locations and the maximization of the distance between store and proposed locations. Finally, the algorithm reports the best five locations. By selecting the ‘Show Final Results’ menu, the results reporting from NSGA-II algorithm are shown on Google Maps. The results are shown on Google Maps with a yellow pin. The user interactions described above are shown on Figure 5.3.

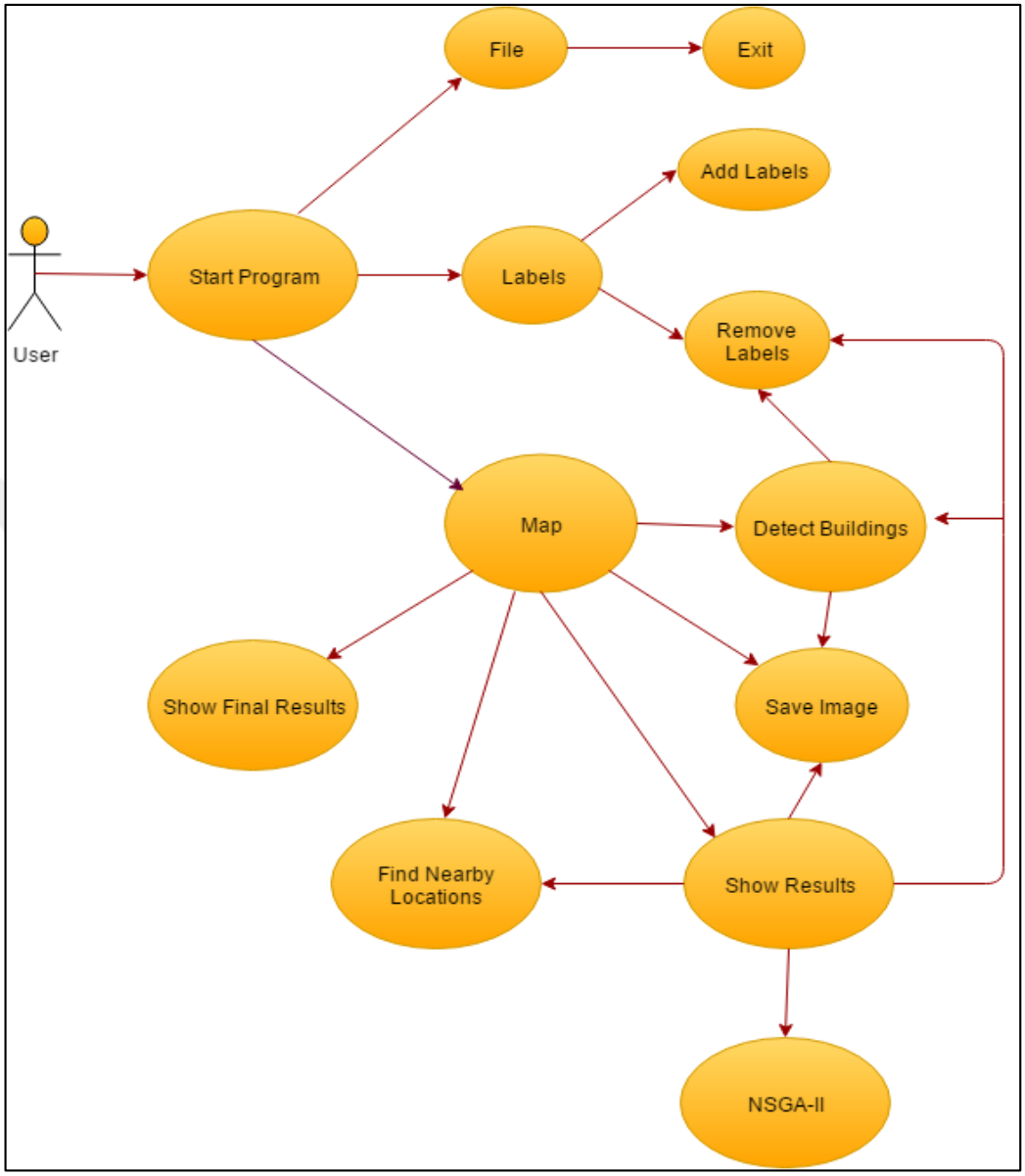

Figure 5.3. The application's interaction with the user

6. TEST AND EVALUATION

In this section, initially some experiments are carried out to tune the parameters of NSGA-II algorithm. Then different test scenarios have been created on Google Maps and the application is tested according to these scenarios. Because the Kadıköy region is characterized by a dense population we have obtained the coordinates of public places and other grocery stores for this region and the tests are done on maps at that area. The solutions generated by NSGA-II are shown on QT 5.3.1 which is the IDE of our application. OPENCV 2.4.5. library and C++ code are used for the detection of the buildings and the locations determined by NSGA-II are displayed on Google Maps. NSGA-II library is executed by an external call from the application in order to tune the parameters of genetic algorithm. The data which include the coordinates of the stores, restaurants and the metro stations in a certain area are taken from the application *Maptriks*. *Maptriks* is an online platform which is based on Google Maps and it stores the coordinates and addresses of different kinds of locations in different areas.

As noted above, the parameters of the genetic algorithm are initially tested in order to decide about the parameter set that can generate the best possible results. In the second phase of the evaluation process, different test scenarios are created for the system. Different locations in Kadıköy are selected and coordinates of the restaurants, metro stations and the stores are set manually in order to have cases where a reasonable solution for the coordinates of the new score could be calculated. Trivial setups are created in order to demonstrate that the system can produce a convergence to reasonable areas according to our criteria and some other scenarios are also created to test different aspects of the system. Note that the mean value of the surface areas is also calculated by the application. In the third phase of the experiments, having the minimum distance from the coordinates of the average mean surface point is added to NSGA-II as the third objective. In this test, the influence of using the mean value as the third objective is observed. In the last phase, one of the existing stores in a selected area is removed from the map. Then the proximity of results generated by the NSGA-II algorithm to the removed store is calculated. In these tests, it is checked if NSGA-II algorithm could produce solutions that are close to an existing store.

6.1. Parameters of Genetic Algorithm

The parameters of the genetic algorithm are tuned in this part of the evaluation process. The best parameters for the number of generations, the population size, the probability of mutation and the probability of crossover are determined by using various tests. After the parameters are set, the same parameters are used for the tests carried out in the following phases. As noted above, the tests are done in Kadıköy region with latitude and longitude values 40.9896315 and 29.0264541 respectively.

First, different generation numbers are tested by using 20 different runs in each experiment. For each run a different seed value is utilized. The results are shown in Figure 6.1 and 6.2.

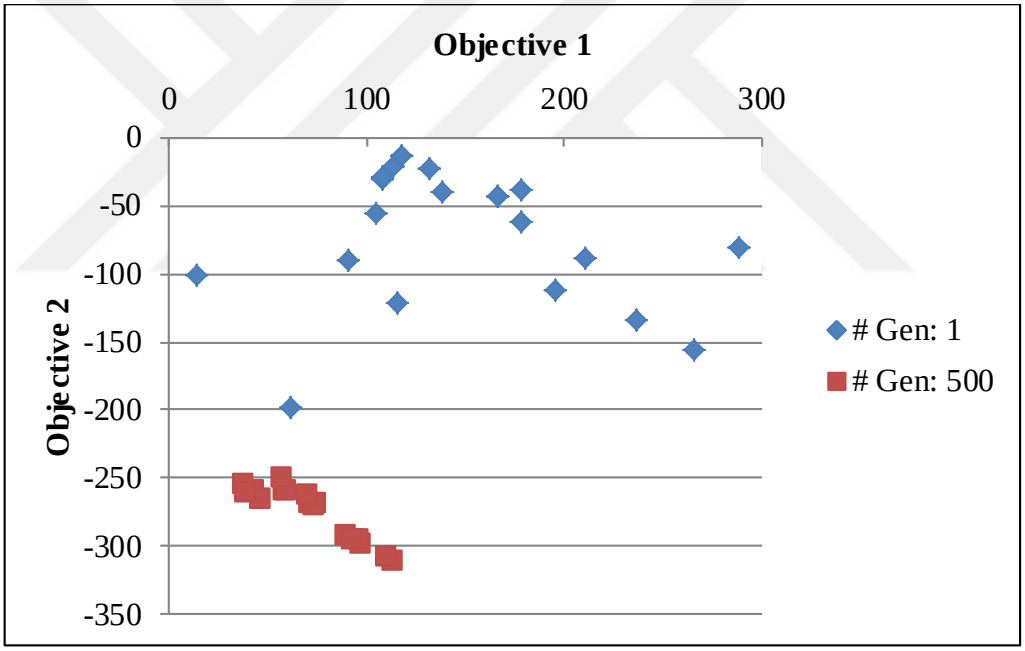

Figure 6.1. Distribution of population from generation 1 to 500

Figure 6.1 presents the distribution of the initial population and the best individuals obtained at the end of 500 generations. As it can be observed, the randomly created individuals are quite scattered in terms of objective 1 and objective 2 values. However, after 500 generations a convergence could be obtained and the best individuals line up forming the pareto front.

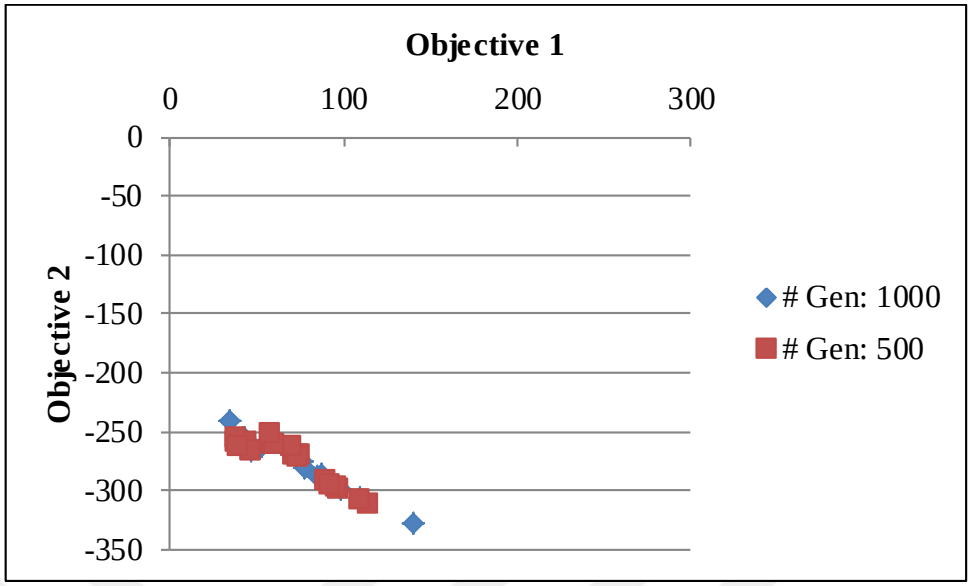

Figure 6.2. Pareto fronts for 500 and 1000 generations

Additionally, the NSGA-II algorithm is executed with 1000 generations. The comparison of the Pareto fronts obtained by 500 and 1000 generations are shown in Figure 6.2. We can observe that the results do not differ considerably between 500 and 1000 generations. As the Pareto fronts for 500 and 1000 generations are close to each other, we decided to keep 500 generations to be used for the system.

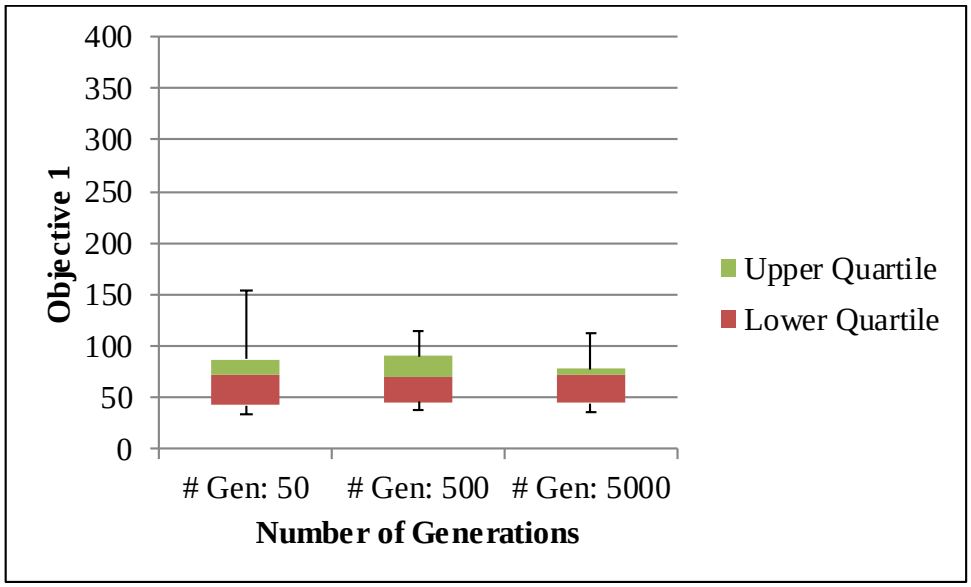

Figure 6.3. Median, lower and upper quartile for objective 1

Using different number of generations is also analyzed statically. The median, lower and upper quartiles for objective 1 are shown on Figure 6.3. The results are presented for 50, 500 and 1000 generations. The brown color indicates the lower quartile and the green color indicates the upper quartile in Figure 6.3. The lower quartile is the first quartile in the graph. The first quartile (Q_1) is described as the middle number between the smallest number and the median of the data set. The upper quartile is the third quartile in the graph. The third quartile (Q_3) is the middle value between the median and the highest value of the data set. The upper quartile is the middle value between the median and the highest value of the objective 1 for 20 runs of 50, 500 and 1000 generations. The lower quartile is the middle number between the smallest number and the median of the objective 1 values for 20 runs of 50, 500 and 1000 generations. The y axes shows the objective 1 value which is the average distance to restaurants and metro stations. The results are displayed using a scale between 1 and 1000 on the map according to the equations 4.15, 4.16, 4.17 and 4.18. Note that, the results are obtained by using 20 runs. As seen in the figure, the deviation with 50 generations is higher than the others. The deviation is the difference between the smallest and the largest distance found in these 20 runs. A stochastic algorithm is considered to be more robust and reliable if the deviation throughout different runs is not large. On the other side, this deviation is similar for 500 and 1000 generations. We can see this deviation in figure 6.4. Consequently, we can conclude that there is no further recruitment after 500 generations for objective 1 in this analysis.

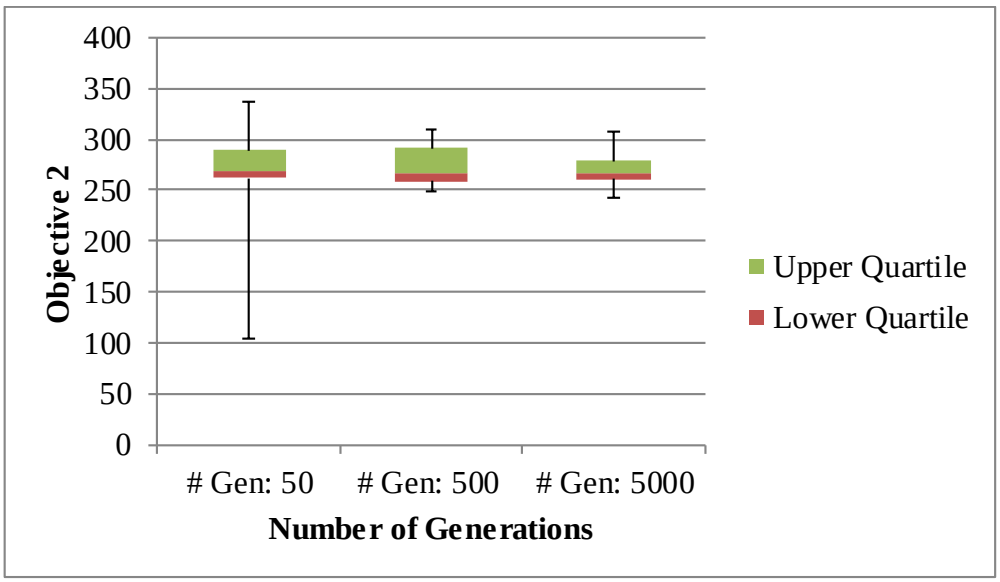

Figure 6.4. Median, lower and upper quartile for objective 2

In Figure 6.4, the same analysis is carried out for the second objective. Again a similar result can be observed in the figure. The brown color indicates the lower quartile and the green color indicates the upper quartile in Figure 5.3. The lower quartile is the first quartile (Q_1) in the graph. The upper quartile is the third quartile (Q_3) in the graph. The upper quartile is the middle value between the median and the highest value of the objective 2 for 20 runs of 50, 500 and 1000 generations. The lower quartile is the middle number between the smallest number and the median of the objective 2 values for 20 runs of 50, 500 and 1000 generations. The high deviation that exists in 50 generations disappears when 500 generations are utilized. The difference between the smallest and largest distance is more than 200 when 50 generations are utilized. The y axes indicates the values of objective 2 which can again vary between 1 and 1000. Again, the values are calculated from the coordinates which are converted to the scale between 1 and 1000 according to the equations 4.15, 4.16, 4.17 and 4.18. However, a further improvement could not be obtained when generation number is increased more.

After the tests on generation number, experiments have been carried out for the population size. The population size is tested by using 500 generations number and again 20 different runs have been carried out with different seed values in each experiment. Figure 6.5 shows the results obtained with population size 8 and 20. The results are the average of 20 runs for both population sizes. As seen in the figure, results for objective 1 and objective 2 do not form a clear pareto pront when the population size is 8. However, when the population is increased to 20, an acceptable pareto front is obtained at the end of the genetic search.

The comparison of Pareto fronts obtained with population size 20 and 40 are shown on Figure 6.6. The results do not change considerably when population size is increased to 40. However, it can be claimed that population size 40 generates a pareto front that is a little better than the result generated with population size 20. Also the system is tested with population size 80. It is observed that the results do not change with population size 40 and 80. Therefore, the population size parameter is set as 40 for the system.

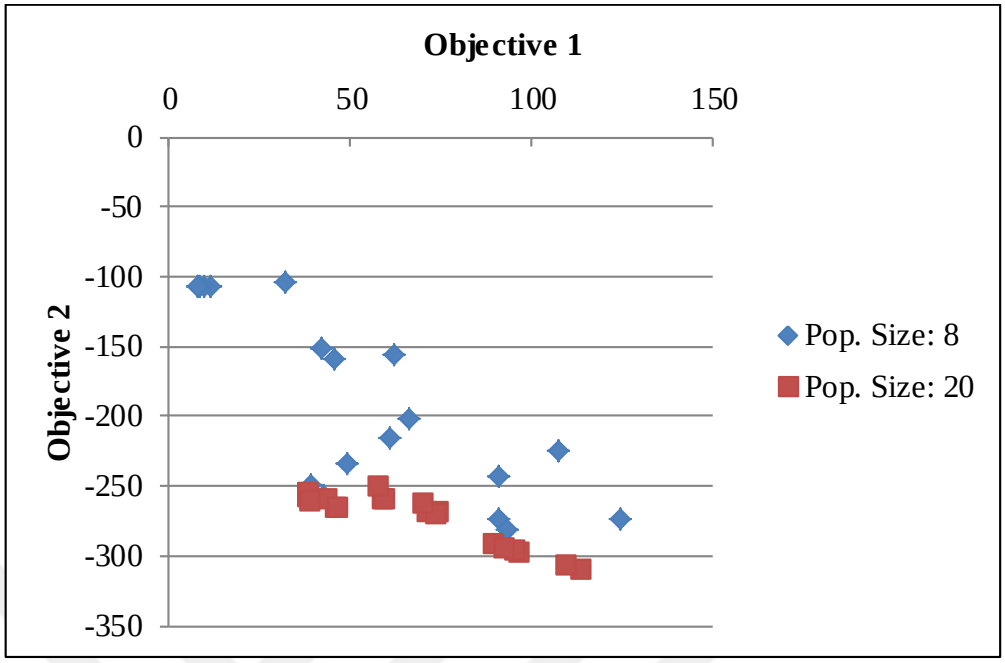

Figure 6.5. Distribution of population for a size 8 and 20

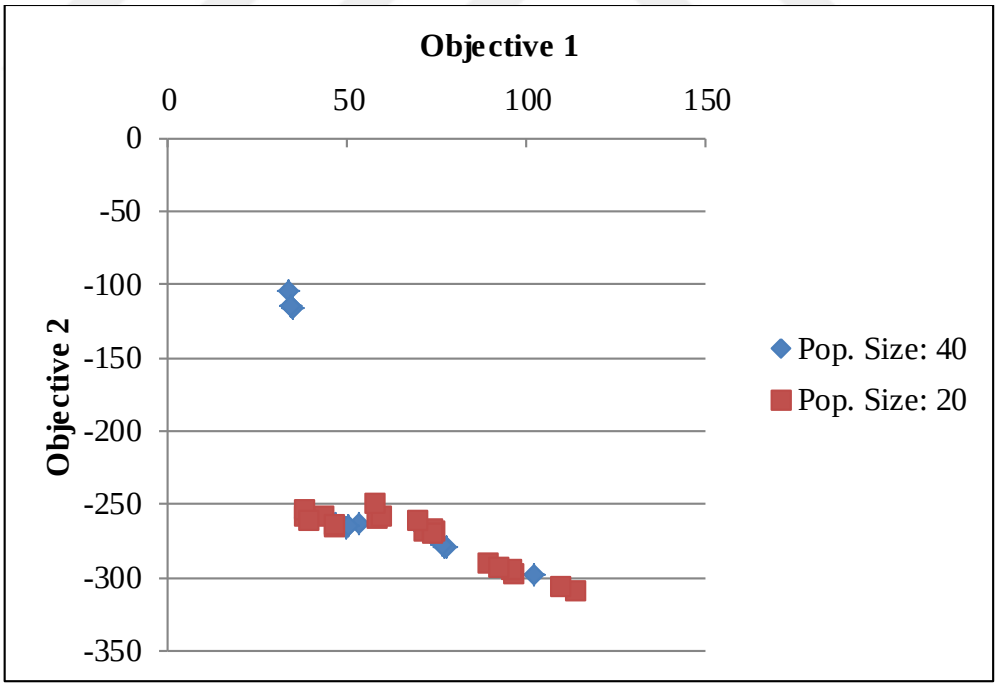

Figure 6.6. Pareto fronts for 20 and 40 Population Size

After the generation and population parameters are tuned, new experiments are carried out for the probability of crossover operation. Again the probability of crossover is determined

by using 20 different runs in the experiments. The generation number and population size determined above are used in the experiments. The average results for objective 1 and objective 2 are shown on Table 6.1 for different crossover probabilities. It was not possible to observe vital changes in the results when different crossover probabilities are tried. However, with probability 0.6 the most reasonable pareto front has been obtained. Therefore, the probability of crossover is set as 0.6.

Table 6.1. Results Obtained with Different Crossover Probabilities

Probability of Crossover	**Average of Objective 1**	**Average of Objective 2**
0.6	46.83433	-238.83
0.7	51.98837	-242.027
0.8	57.15888	-231.835
0.9	49.14705	-234.391
1.0	53.71848	-239.664

On the other side, experiments are carried out to determine the probability of mutation operation. The previously determined crossover rate, generation number and population size are utilized in the experiments and in each experiment again 20 runs have been utilized with different seed values. The designers of NSGA-II algorithm suggest that the best probability for mutation operation is one over the number of variables used in the genetic search. This probability is 0.5 since we have been using two variables which represent the coordinates of the store. Although different rates have been tried, it has been observed that 0.5 is the best value for the probability of mutation for our problem too.

Table 6.2. Best Parameters on NSGA-II

Generation Number	500
Population Size	40
Probability of Crossover	0.6
Probability of Mutation	0.5

Finally, the parameters determined for the generation number, the population size, the probability of crossover and mutation are shown on Table 6.2. These parameters are used on other tests.

6.2. Different Test Senarios

Different test scenarios are created on Google Maps in order to analyze the behaviour of the system in depth. For this tests we have selected different locations on Kadıköy. The coordinates of restaurants, metro stations and stores are distributed over the area manually in order to create different configurations for the algorithm.

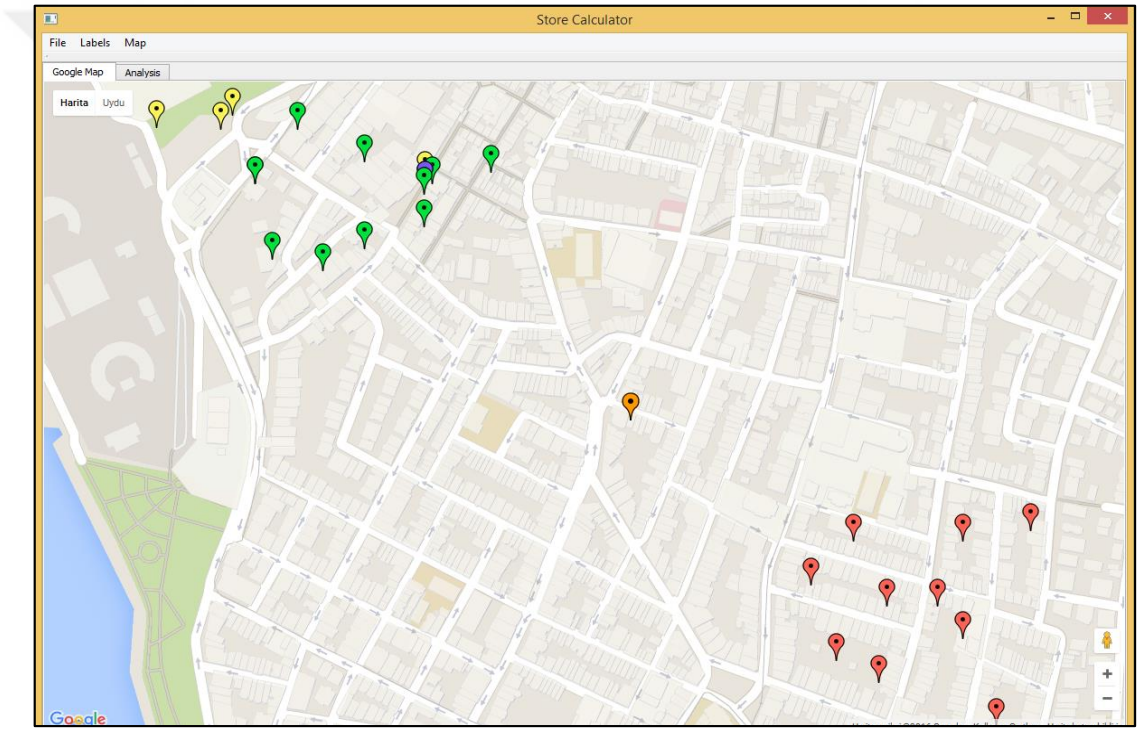

Figure 6.7. Solutions with 10 public and 10 store locations

In the first scenario, the restaurants and metro stations are grouped together in different regions of the selected area. As seen in Figure 6.7, the restaurants and metro stations are in the top left corner of the selected area (green markers). On the other side, the stores are gathered in bottom right corner of the selected area (red markers). For this example and the rest of the examples the weighted mean surface area point is denoted with an orange marker. The number of stores is the same with the number of restaurants and metro stations. Hence a configuration is obtained where the solution is trivial. The final results

are shown on Figure 6.7. The yellow marker are the solutions returned by the system and the purple marker is the solution that is closest to the the mean surface area location. In this example since the restaurants and metro stations are clearly away from the other grocery stores, certainly the trivial solution is to place the new store near to the metro stations and restaurants. It is observed the system could produce this solution and that the new store locations are proposed near the restaurants and far away from the other grocery stores. With this trivial scenario, it has been verified that the system could have a convergence based on the objective functions utilized.

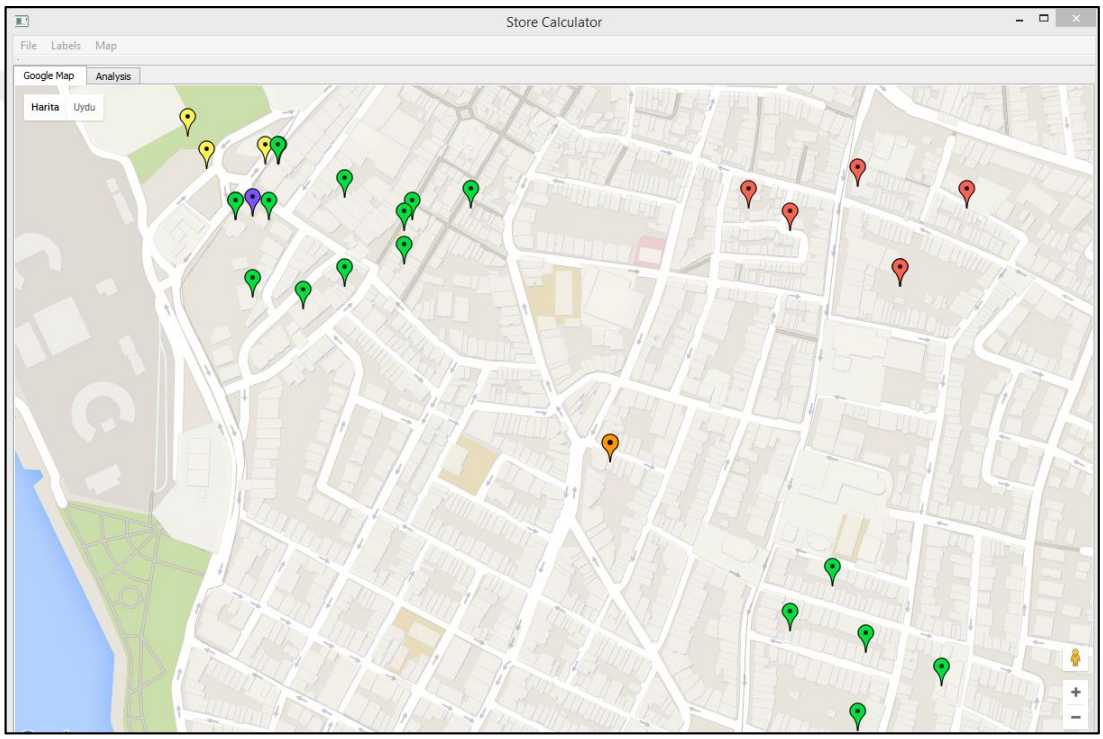

Figure 6.8. Solutions with 16 public and 5 store locations

In the second test scenario, 11 restaurants and metro stations are placed in the left top corner of the area (green markers), 5 stores are placed in the right top corner (red markers) and another 5 restaurants and metro stations are placed in the right bottom corner of the area (green markers). The results are shown on Figure 5.8 where the yellow markers are the solutions and the purple marker is the solution closest to the weighted mean surface areas point. In this scenario, two groups of restaurants are utilized. Both of the groups are away from other grocery stores. However, there are more restaurants in the first group. Hence it is more reasonable to open a store close to this first group. In this scenario, we

wanted to see if the system would be misleaded by the second group of restaurants, however it can be seen that all locations proposed are near the 11 restaurants and metro stations group and no proposal has been proposed for the second group of restaurants and metro stations.

In the third scenario, 11 restaurants and metro stations (green markers) and 3 stores (red markers) are placed in the left top corner of the selected area and another 5 restaurants (green markers) are placed in the right bottom corner of the area. The results are shown on Figure 6.9 again with yellow (the solutions) and purple (the solution closer to the weighted mean surface point) markers. In this scenario, two groups of restaurants are utilized. The first group of restaurants are near the stores, where the algorithm is configured to maximize the distance from them and the second group of restaurants are away from the stores. Hence, it is more reasonable to open a store close to the second group. With this scenario, it has been observed that the solutions have been proposed near the group of restaurants and metro stations which are far away from other stores on the map. In the first group, only one location is proposed but this location is also relatively away from the stores that exist in the region.

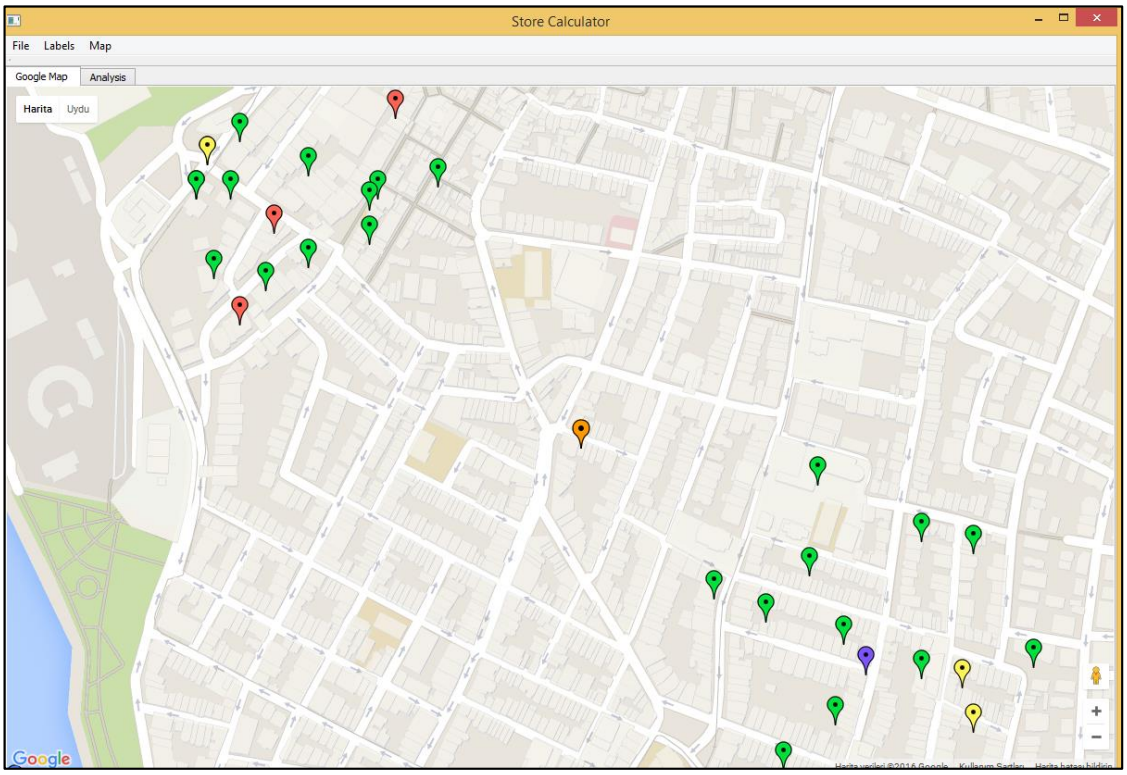

Figure 6.9. Solutions with 22 public and 3 store locations

In the fourth test scenario, the restaurants (green markers), metro stations (green markers) and stores (red markers) are distributed in the area totally randomly. The results are shown on Figure 6.10. Again, it has been observed that the results are away from the store locations. However, since a complicated environment is created in this scenario, different alternatives exist for the new store. In this case, the solutions proposed by the system are scattered to different regions of the selected area as seen in the figure. In this figure the blue markers on the bottom left and top right indicate the region of search of the algorithm.

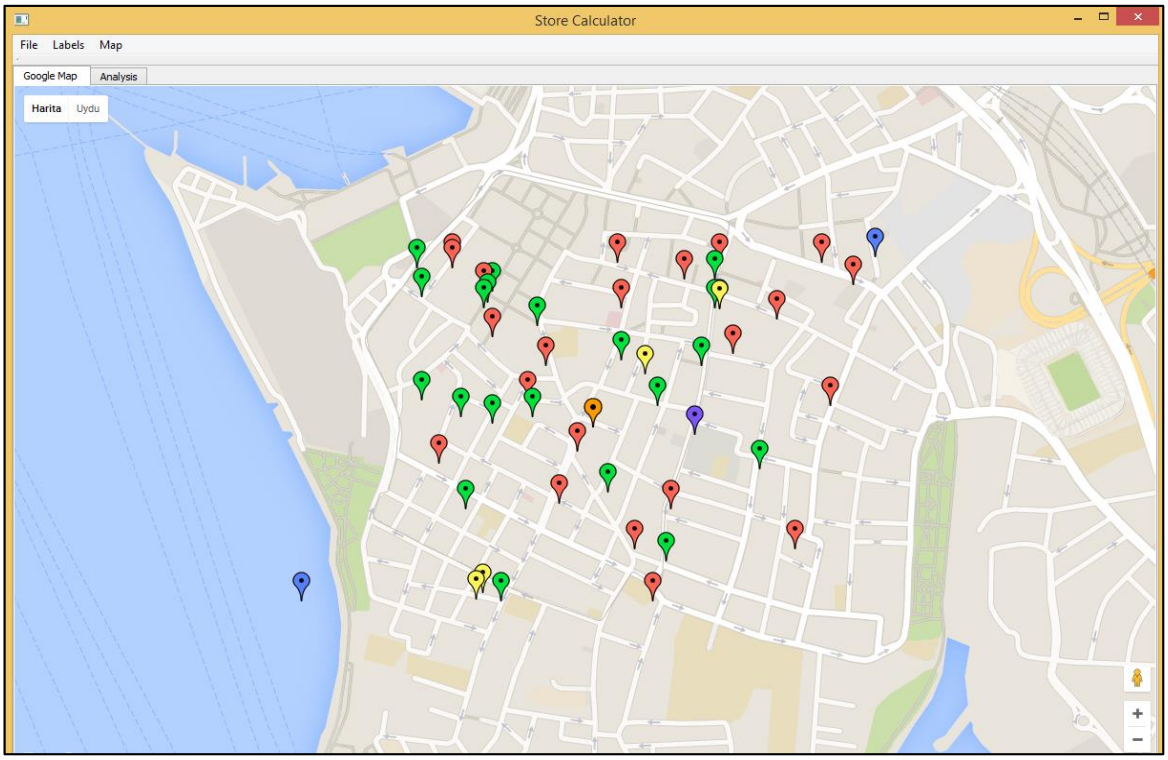

Figure 6.10. Solutions with a hybrid framework

In the fifth test scenario, the real data of restaurants, metro stations and store locations are used. The results are shown on Figure 6.11. Again, the results are observed to be reasonable. If we consider the locations proposed by the system, they are not close to other stores. However, the system based on the first objective seems to chose locations close to the restaurants and metro stations as much as possible. As in the previous example, the blue markers in the bottom left and top right region indicate the region of search of the algorithm.

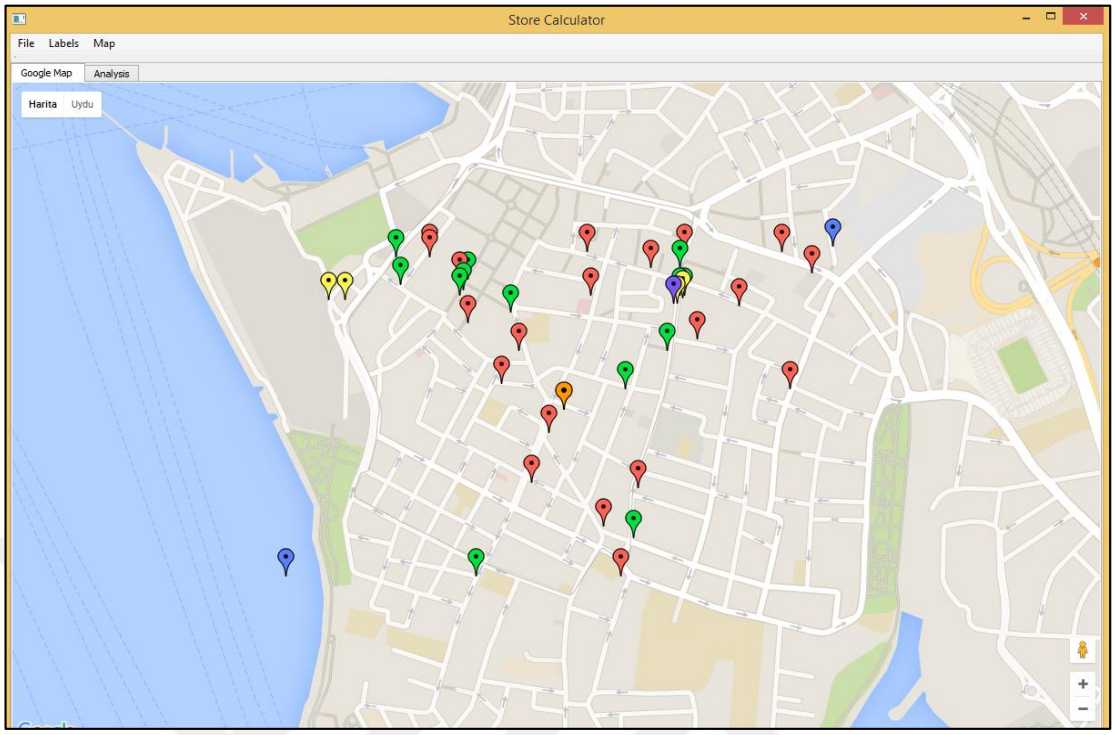

Figure 6.11. Solutions with real data

6.3. Utilizing Mean Surface Area as the Third Objective

As explained in the previous section, the weighted mean surface area point has been used in order to determine the most adequate location between the solutions given in the Pareto front generated by the genetic search by selecting the closest to the weighted mean surface area point. However, the distance of the locations to the mean surface area could be used as the third objective in the system. Hence, experiments have been carried out where the third objective is set as the minimization of the distance between the proposed store locations and the location of the weighted mean point of the surface areas.

In Figure 6.12 the result generated by two objectives in an area is shown. In Figure 6.13. the result generated by three objectives in the same area is shown. With two objectives, the results are away from the other store locations in the area. However, when the third objective is added to the system, the locations proposed by the system also has a convergence towards the location of mean value and hence, it becomes impossible to preserve the distance from other the store locations.

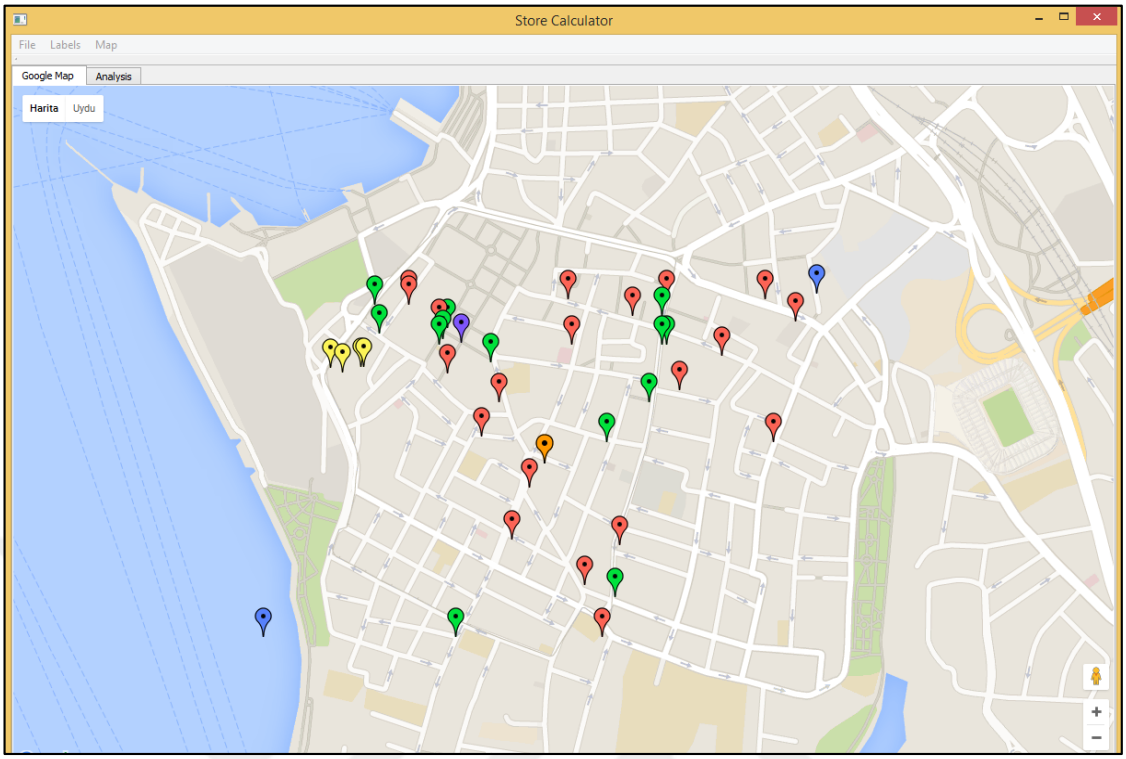

Figure 6.12. Final results with 2 objectives

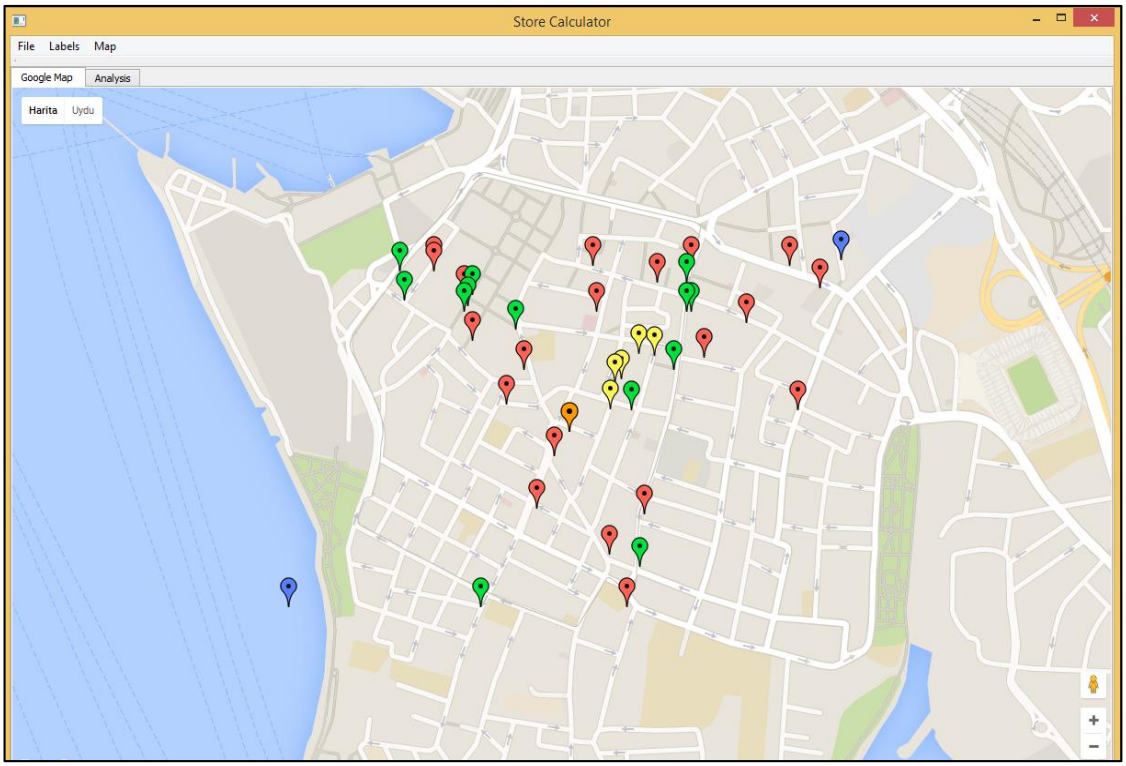

Figure 6.13. Final results with 3 objectives

In figure 6.14, the results of experiment with 2 objectives are presented. The median, lower and upper quartiles for the objectives are shown in Figure 6.14. The brown color indicates the lower quartile and the green color indicates the upper quartile in this figure. The lower quartile is the first quartile (Q_1) in the graph. The upper quartile is the third quartile (Q_3) in the graph. The upper quartile is the middle value between the median and the highest value of the objectives. The lower quartile is the middle number between the smallest number and the median of the corresponding objective. Hence, the comparison of using two or three objectives can be seen in these figures. Again each result is obtained by using 20 runs with different seeds. As seen in the figure 6.15, the deviation is higher for objective 1 and objective 2 when 3 objectives are utilized. As seen in the figure 6.14, the deviation is lower for objective 1 and objective 2 when 2 objectives are utilized. The deviation is the difference between the smallest and the largest distance found in these 20 runs. However, this is an expected result, since when the third objective is added, it becomes more difficult to find a position close the mean value location and the public places and away from other stores in the area.

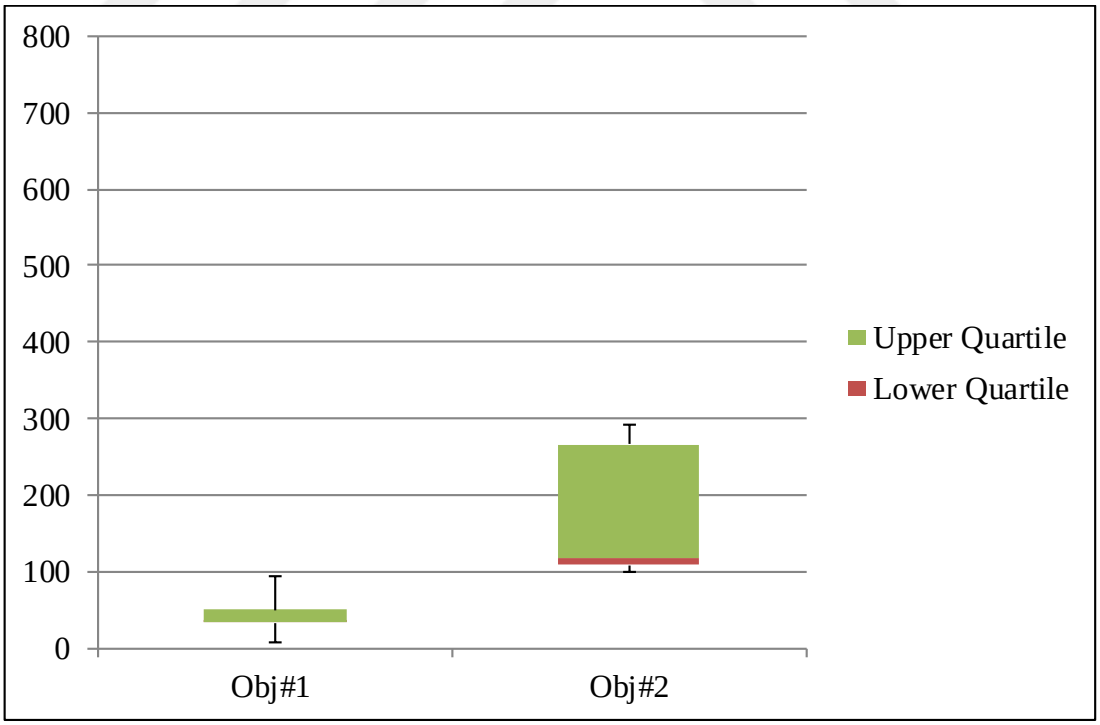

Figure 6.14. Median, lower and upper quartile for 2 objectives

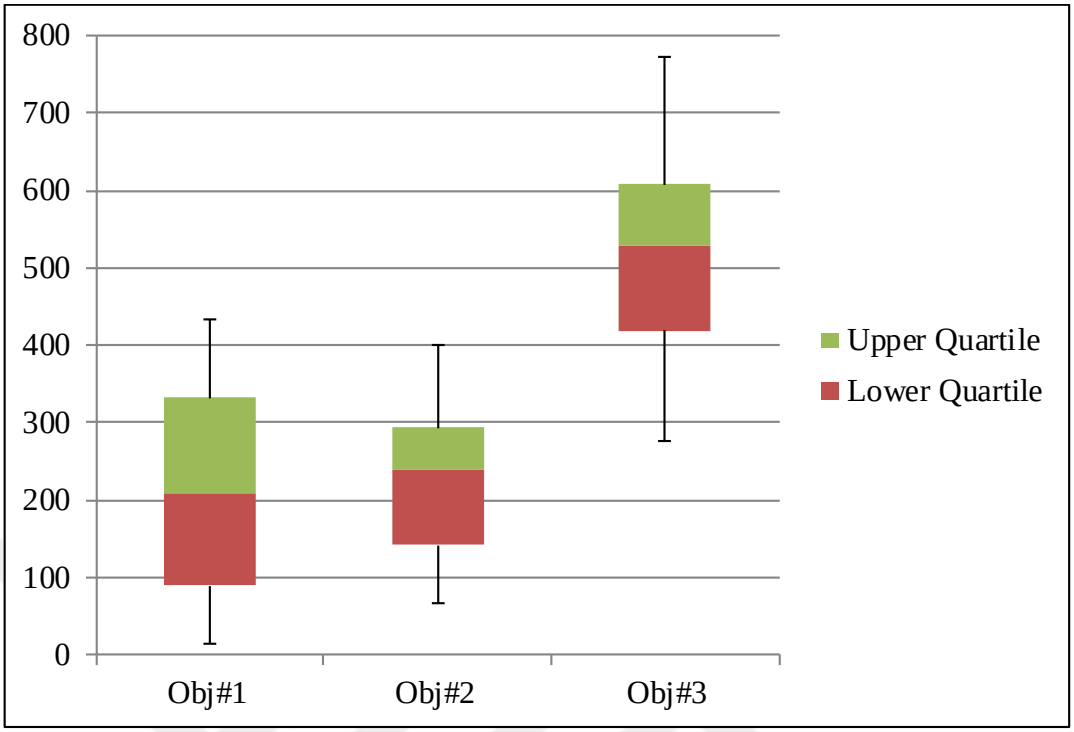

Figure 6.15. Median, lower and upper quartile for 3 objectives

6.4. Removing Existing Store Locations From Google Maps

In this test scenario, the real locations for restaurants, metro stations and other stores are utilized. However, one of the existing stores in a selected area is removed from the map. Then the proximity of results generated by the NSGA-II algorithm to the removed store is calculated. In these tests, it is checked if NSGA-II algorithm could produce solutions that are close to an existing store. Four different places in Kadıköy are selected for this test.

One of the selected regions which is in a real case of stores, restaurans and metro stations is shown on the top figure of Figure 6.16. The store that exists on the top left corner is removed from the map. The removal store location is shown inside a circle. Again, the proposed store locations are shown with yellow markers and the proposed location which has the minimum distance to the location of the mean value of the surface areas is shown with a purple marker (the range of the selected area is shown with blue markers at the bottom left and top right of the region). It is observed that the proposed location with purple marker is close to the store which has been removed from the map as shown in the

Figure 6.17. This result indicates that the system can make realistic proposals which can overlap with the previous decisions to select the location of stores.

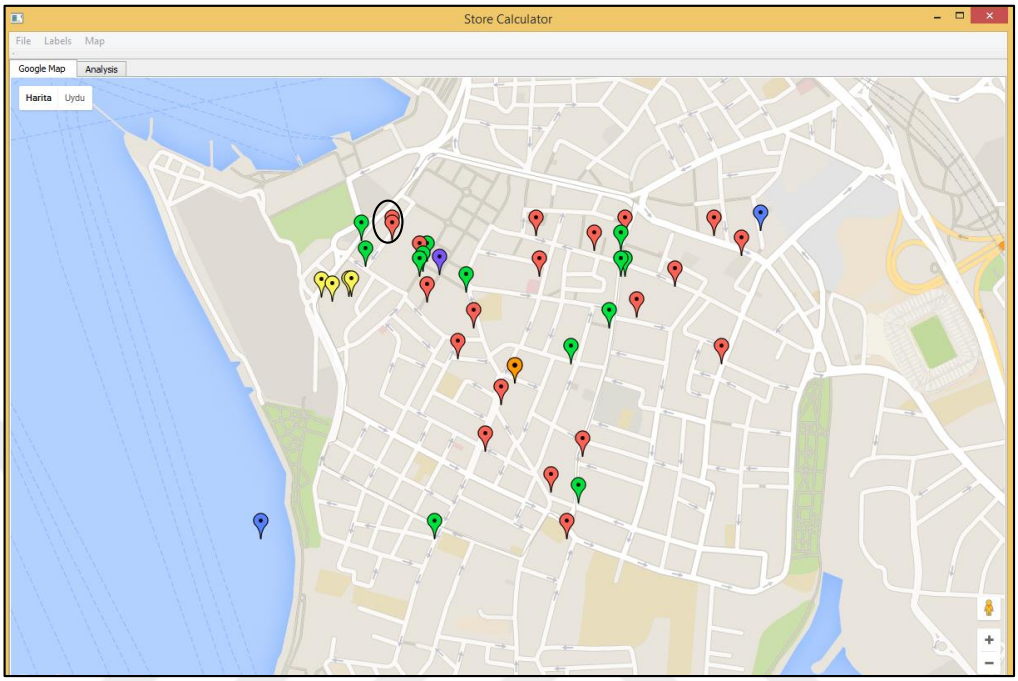

Figure 6.16. First scenario: the results before removing an existing store

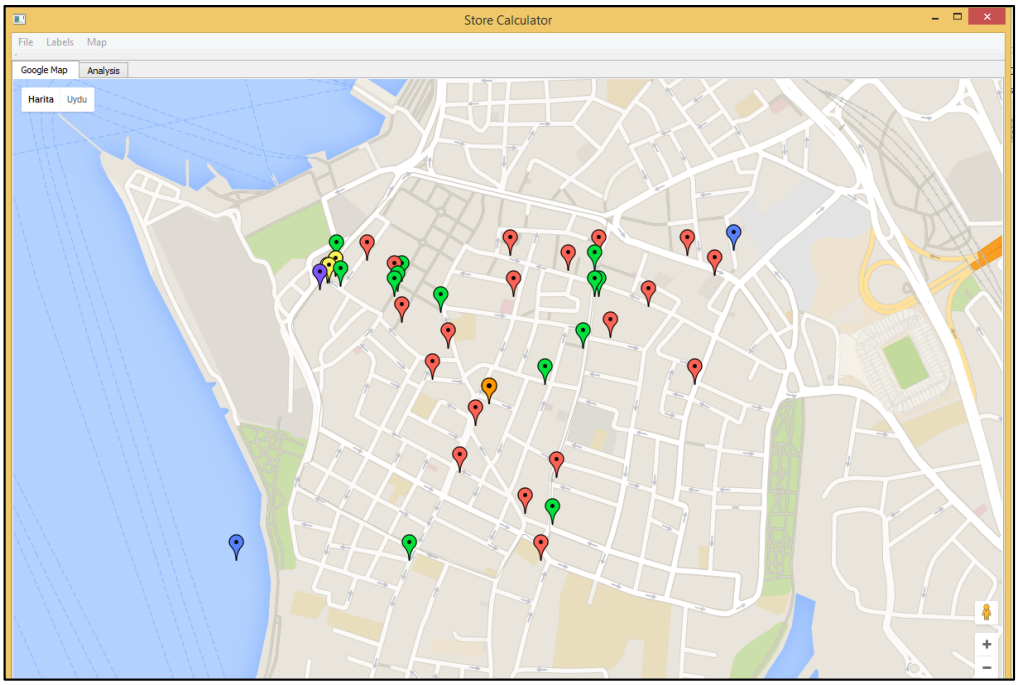

Figure 6.17. First scenario: Final results after removing an existing store

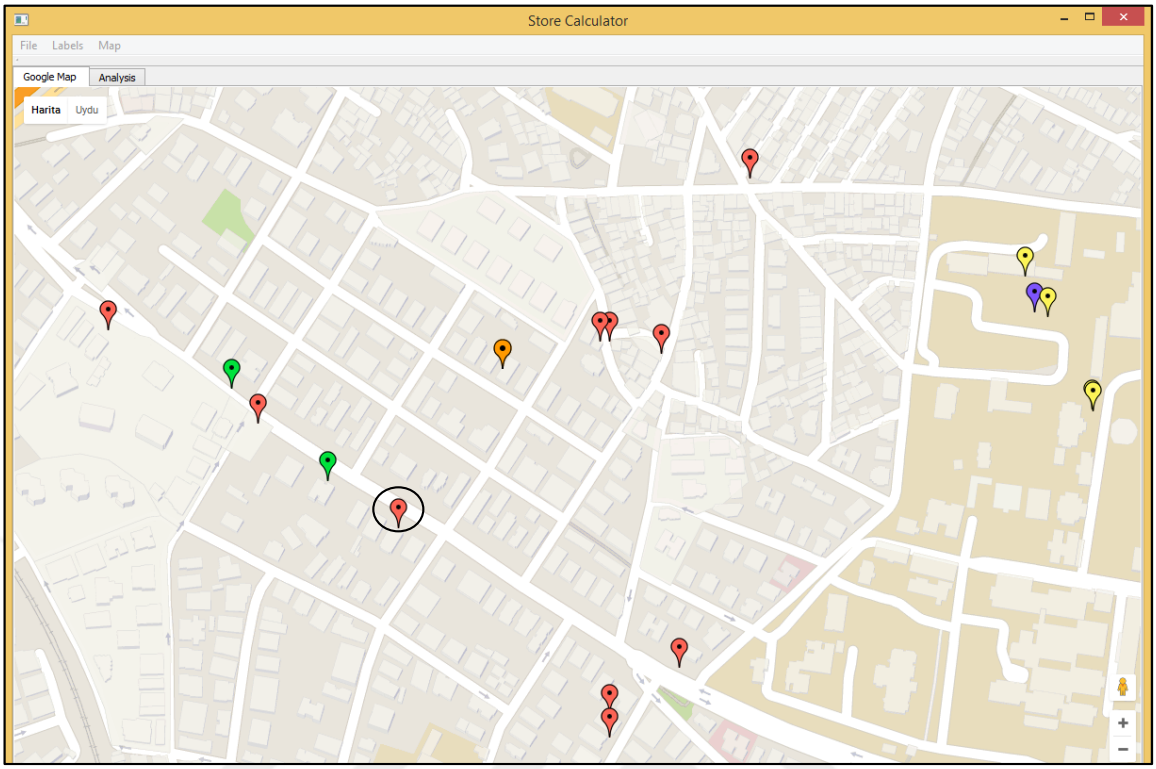

Figure 6.18. Second scenario: The results before removing an existing store

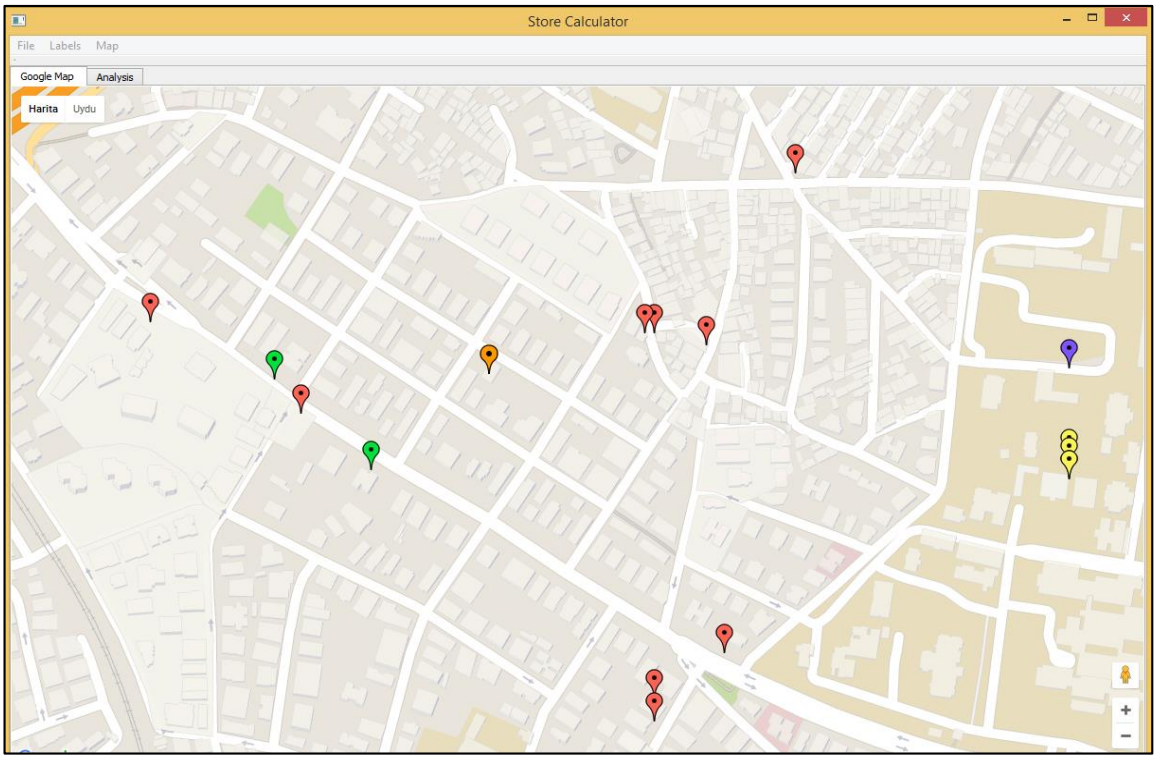

Figure 6.19. Second scenario: Final results after removing a store location

The second scenario is shown on Figure 6.18. The removed store location is shown inside a circle. In this scenario, the number of stores is greater than the number of restaurants and metro stations. Again, the proposed store locations are shown with yellow markers and the proposed location which has the minimum distance to the location of the mean value of the surface areas is shown with a purple marker. In this area, there is a group of other store locations close to the store that has been removed from the map (the red marker inside circle in the top figure of Figure 6.18), so the system proposes locations that are away from these stores as shown in the bottom figure of Figure 6.19, and this is consistent with the objectives utilized by the framework. The result is now very close, because there are other stores near the removed store location. So the results become closer to the restaurants and metro stations.

The third scenario is shown on Figure 6.20. In this test scenario, the number of restaurants and metro stations is greater than the number of stores. Again, the proposed store locations are shown with yellow markers and the proposed location which has the minimum distance to the location of the mean value of the surface areas is shown with a purple marker. The removed store location is shown inside a circle in the top figure of Figure 6.20. The removed store location is away placed in a distance from the other stores and restaurants locations. After removed the store, the result is close to the restaurants locations as shown in the bottom figure of Figure 6.21. The system proposes thus locations that are away from the stores and but comes closer to restaurants when is possible, and this is consistent with the objectives utilized by the framework. In this test area, it is shown that the result become closer to the removed store location.

The fourth scenario is shown on Figure 6.22. In this test scenario, the number of metro stations and restaurants is equal to the number of stores. The removed store location is away from the other store and restaurant locations. In Figure 6.23, the result is shown after the store location removed. The removing store location is away from the other store and restaurant locations. After removing the store location, the result is near the previous position. In that case the removal of a store that was away from the main concentration of location didn’t influence dramatically the results.

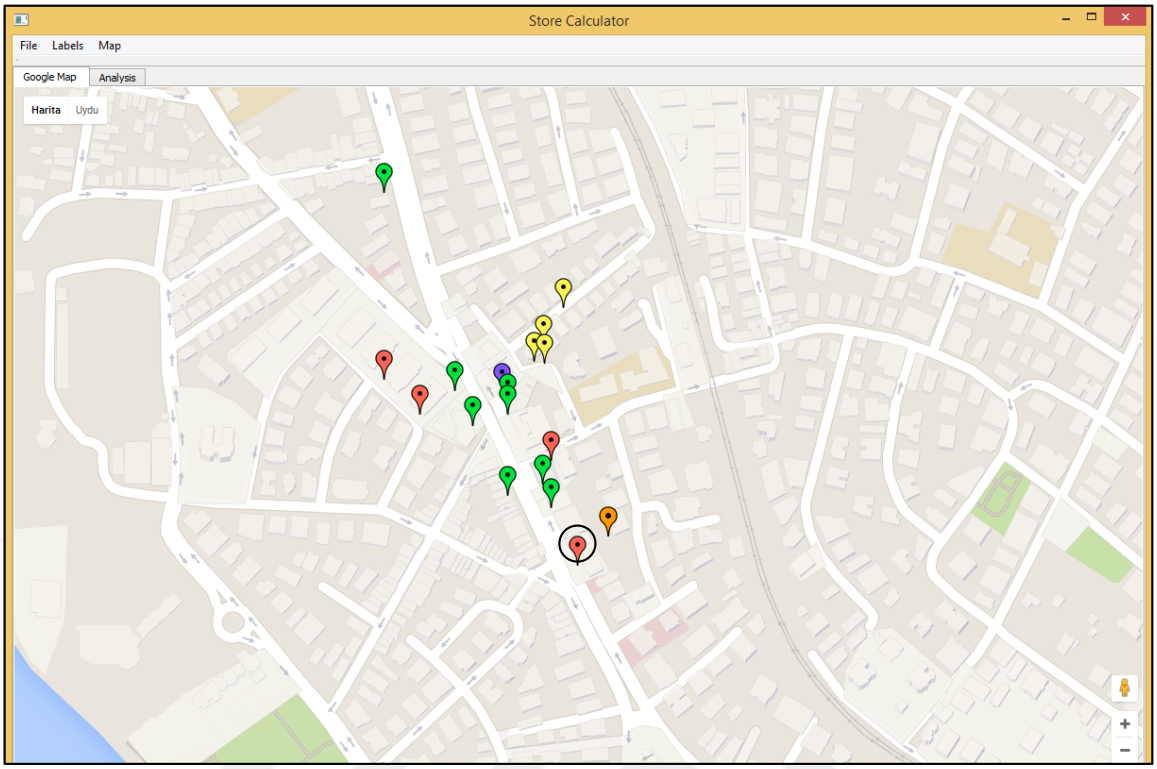

Figure 6.20. Third scenario: The results before removing an existing store

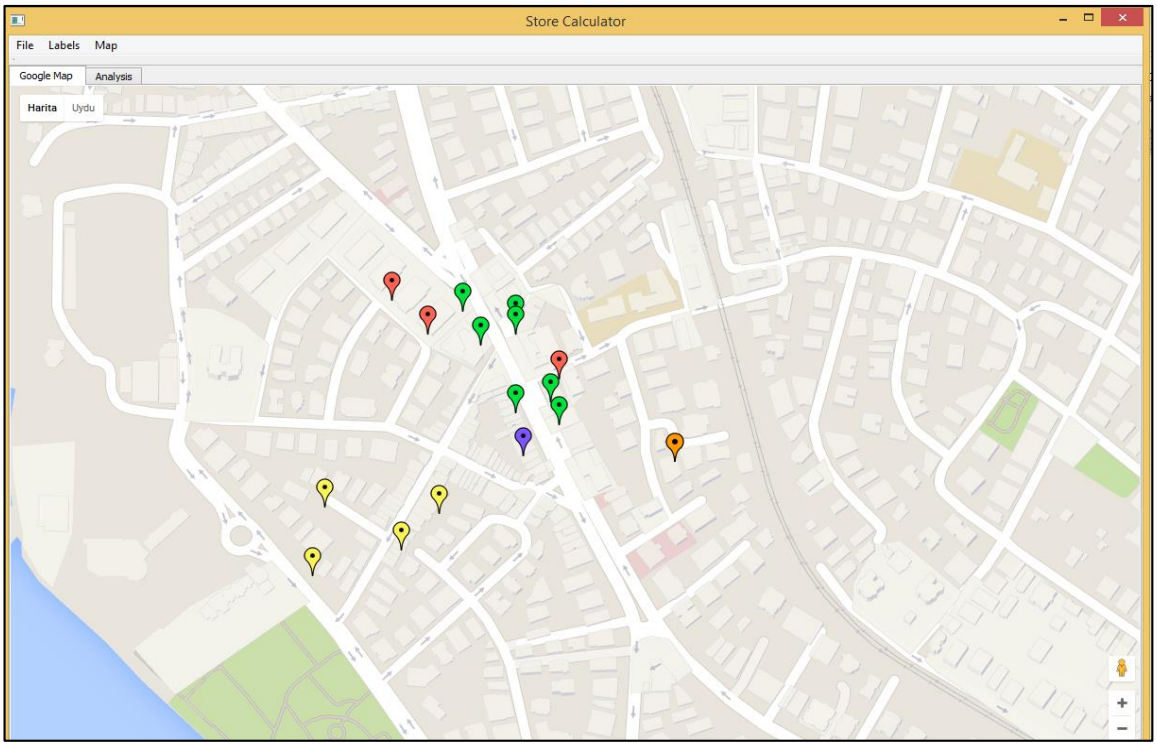

Figure 6.21. Third scenario: Final results after removing a store location

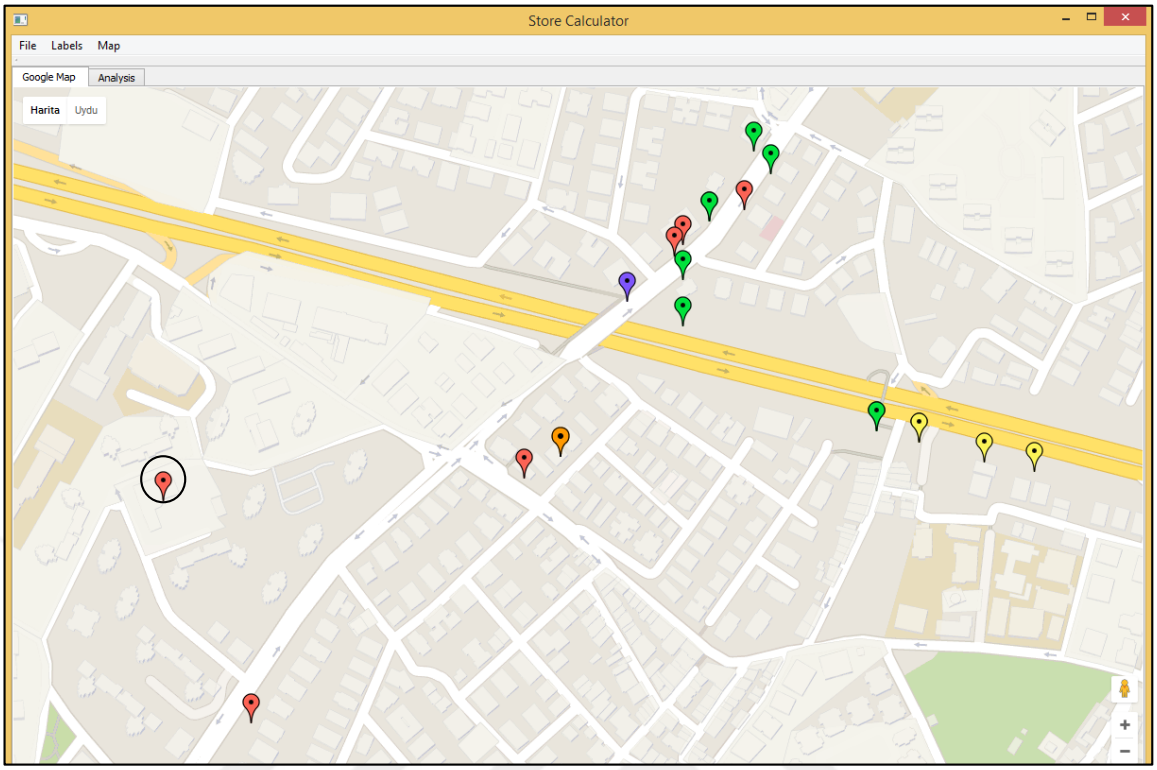

Figure 6.22. Fourth scenario: The results before removing an existing store

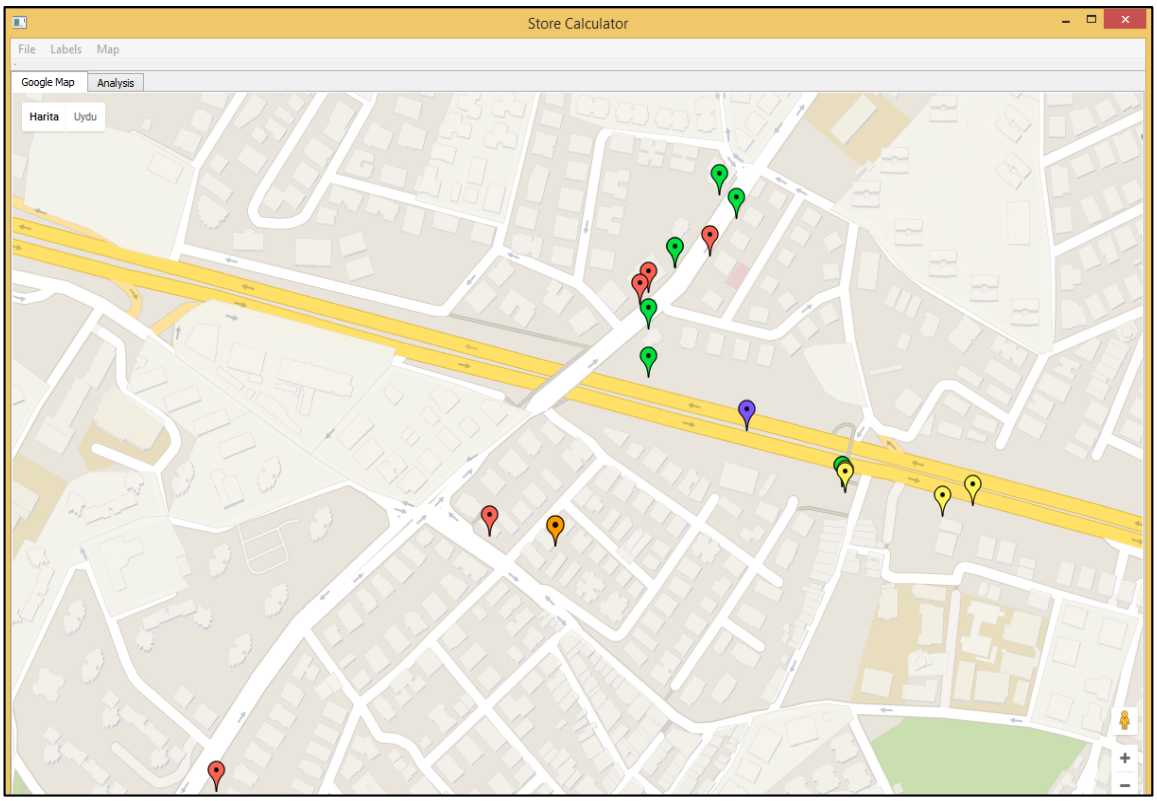

Figure 6.23. Fourth scenario: Final results after removing a store location

7. CONCLUSION

In this study, a multi-objective optimization technique that optimizes more than one objective function is proposed for the grocery market location decision problem. NSGA-II is utilized to optimize the location decision process in this study. The first objective function used in this study is minimizing the distance to public places like restaurants and metro stations in the area. Then, the second objective is maximizing the distance to other stores in the area. Additionally, the mean value of the surface areas is calculated based on building detection from Google Maps. This process is using the RGB color model and the bounding boxes of blobs corresponding to the buildings. Finally, the NSGA-II algorithm produces a set of solutions according to the selected objective functions. These solutions form the Pareto front. As no solution can dominate the other, the weighted mean value of the surface areas is used to determine the final solution, which is this solution from the pareto front that is closer to this weighted mean surface area point.

The parameters of the genetic algorithm are tested in order to generate the best possible results. Different locations in Kadıköy are selected and coordinates of the restaurants, metro stations and stores are set manually in order to have cases the algorithm could prove its efficiency and its capacity to be coherent to the criteria of the objective functions. The testing procedure comprises the removal of the map of one of the existing stores in a selected area. Then the proximity of results generated by the NSGA-II algorithm to the removed store is calculated and the solution is most of cases in proximity with the removed store. In general, it is observed that the results are near the restaurants and metro stations and away from the grocery store locations.

In this study, only two objective functions have ben used. However, in a future work, other criteria can be added as new objective functions to this framework. For the moment, only the coordinates of restaurants and metro stations are used. But we may consider to use the locations of other facilities like hospitals or schools. The system is only tested in Kadıköy region because we had the coordinates of public places and other grocery stores only for this region. But the system can be utilized for any area on Google Maps if the necessary data are provided. In resume, this study can contribute to the location problem by focusing more particularly in the location of small entities by proposing simple but efficient criteria via a multi-objective optimization approach.

REFERENCES

1. B. Sirmacek, and C. Unsalan. Urban-Area and Building Detection Using SIFT Keypoints and Graph Theory. *Geoscience and Remote Sensing, IEEE Transactions on,* 47:1156-1167, 2009.

2. M. Wang, S. Yuan, and J. Pan. Building Detection in High Resolution Satellite Urban Image Using Segmentation, Corner Detection Combined with Adaptive Windowed Hough Transform. *Geoscience and Remote Sensing Symposium (IGARSS), 2013 IEEE International,* Melbourne, 508-511, 2013.

3. P. Saeedi, and H. Zwick. Automatic Building Detection in Aerial and Satellite Images. *Control, Automation, Robotics and Vision, 2008. ICARCV 2008. 10th International Conference on,* 623-629, 2008.

4. A. Huertas, and R. Nevatia. Detecting Buildings in Aerial Images. *Computer Vision, Grpahics, and Image Processing,* 41:131-152, 1988.

5. W. Yanfeng, Z. Zhongming, and S. Jianghong. Urban Building Extraction from High-Resolution Satellite Panchromatic Image Using Clustering and Edge Detection. *Geoscience and Remote Sensing Symposium, 2004. IGARSS '04. Proceedings. 2004 IEEE International,* Anchorage, 3:2008-2010, 2004.

6. R.B. Irvin and D. M. McKeown. Methods for Exploiting The Relationship Between Buildings and Their Shadows in Aerial Imagery. *Systems, Man and Cybernetics, IEEE Transcations on,* 19:1564-1575, 1989.

7. B. Sirmacek, and C. Unsalan. Building Detection From Aerial Images Using Invariant Color Features and Shadow Information. *Computer and Information Sciences, 2008. ISCIS '08. 23rd International Symposium on*, Istanbul, 1-5, 2008.

8. P. Zimmermann. A New Framework for Automatic Building Detection Analyzing Multiple Cue Data. *IAPRS*, 33:1063-1070, 2000.

9. Z. Zhang, M. Zhou, L. Tang, and C. Li. Automatic Detection and Mapping of Urban Buildings in High Resolution Remote Sensing Images. *Geoscience and Remote Sensing Symposium (IGARSS), 2012 IEEE International*, Munich, 5721-5724, 2012.

10. C. Benedek, X. Descombes, and J. Zerubia. Building Detection in A Single Remotely Sensed Image with a Point Process of Rectangles. *Pattern Recognition (ICPR), 2010 20th International Conference on*, Istanbul, 1417-1420, 2010.

11. B. Sirmacek, and C. Unsalan. Using Structural Features to Detect Buildings in Panchromatic Satellite and Aerial Images. *Recent Advances in Space Technologies (RAST), 2011 5th International Conference on*, Istanbul, 107-111, 2011.

12. T.J. Kim, and J.P. Muller. Image and Vision Computing. In: *Development of A Graph-Based Approach for Building Detection*, 17: 3-14, 1999.

13. A. H. Ozcan, C. Unsalan, and P. Reinartz. Building Detection Using Local Features and DSM Data. *Recent Advances in Space Technologies (RAST), 2013 6th International Conference on*, Istanbul, 139-143, 2013.

14. M. Awrangjeb, C. Zhang, and C.S. Fraser. An Improved Building Detection Technique for Complex Scenes. *Multimedia and Expo Workshops (ICMEW), 2012 IEEE International Conference on,* Melbourne, 516-521, 2012.

15. B. Sirmacek, and C. Unsalan. Urban Area Detection Using Local Feature Points and Spatial Voting. *Geoscience and Remote Sensing Letters, IEEE,* 7:146-150, 2010.

16. B. Sirmacek, and C. Unsalan. Building Detection Using Local Gabor Features in Very High Resolution Satellite Images. *Recent Advances in Space Technologies, 2009. RAST '09 4th International Conference on*, 283-286, 2009.

17. H. Baluyan, B. Joshi, A. Al Hinai, and W. L. Woon. Novel Approach for Rooftop Detection Using Support Vector Machine. *ISRN Machine Vision*, 2013.

18. J. C.J. Groot, G. J. M. Oomen, and W. A.H. Rossing. Multi-Objective Optimization and Design of Farming Systems. *Agricultural Systems*, 110:63-77, 2012.

19. L.Diosan. A Multi-Objective Evolutionary Approach to The Portfolio Optimization Problem. *Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on*,Vienna, 2:183 – 187, 2005.

20. A.Tiwari, K. Vergidis, and B. Majeed. Evolutionary Multi-objective Optimization of Business Processes. *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, Vancouver, 3091-3097, 2006.

21. K. Vergidis, A. Tiwari, B. Majeed, and R. Roy. Optimisation of Business Process Designs: An Algorithmic Approach with Multiple Objectives. *International Journal of Production Economics*, 109:105-121, 2007.

22. K. Vergidis, A. Tiwari, and B. Majeed. Composite Business Processes: An Evolutionary Multi-Objective Optimization Approach. *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, Singapore, 2672-2678, 2007.

23. S. Bandyopadhyay and R. Bhattacharya. Solving Tri-Objective Supply Chain Problem with Modified NSGA-II Algorithm. *Journal of Manufacturing Ssytems*, 33:41-50, 2014.

24. Z. Shi, R. Wang, and T. Zhang. Multi-Objective Optimal Design of Hybrid Renewable Energy Systems Using Preference-Inspired Coevolutionary Approach. *Solar Energy*, 118:96-106, 2015.

25. M. Moradijoz, M.P. Moghaddam, M.R. Haghifam, and E. Alishai. A multi-Objective Optimization Problem for Allocating Parking Lots in a Distribution Network. *International Journal of Electric Power and Energy Systems*, 46:115-122, 2013.

26. H.C.W. Lau, T.M. Chan, W.T. Tsui, F.T.S. Chan, G.T.S. Ho, and K.L. Choy. A Fuzzy Guided Multi-Objective Evolutionary Algorithm Model for Solving Transportation Problem. *Expert Systems with Applications*, 36:8255-8268, 2009.

27. A Ceder, M. Butcher, and L. Wang. Optimization of Bus Stop Placement for Routes on Uneven Topography. In: *Transportation Research Part B: Methodological*, pages 40-61, Elsevier Science Publishers Limited Company, 2015.

28. L. Huiyuan, and S. Yixin. An improved Density Estimation Method in NSGA2. *Automatic Control and Artificial Intelligence (ACAI 2012), International Conference on*, Xiamen, 429-432, 2012.

29. W. Jiasen, L. Huiyuan, and Z. Zejiu. A Novel Strategy to Preserve Diversity in Solving MOEAs Based on Artificial Emphasis on Uniformity. *Intelligent Computing and Intelligent Systems (ICIS), 2010 IEEE International Conference on*, Xiamen, 3:812-816, 2010.

30. H. Mori, and T. Yoshida. Probabilistic Distribution Network Expansion Planning with Multi-objective Memetic Algorithm. *Electric Power Conference, 2008. EPEC 2008. IEEE Canada*, Vancouver, 1-6, 2008.

31. J. Baray, and G. Cliquet. Optimizing Locations Through a Maximum Covering/P-Median Hierarchical Model: Maternity Hospitals in France. *Journal of Business Research,* 66:127-13, 2013.

32. A Khalafallah, and K. El-Rayes. Automated Multi-Objective Optimization System for Airport Site Layouts. *In: Automation in Construction,* 20:313-320, Elsevier Science Direct Limited Company, 2011.

33. F. Plastria and E. Carrizosa. Optimal Location and Design of a Competitive facility. *Mathematical Programming,* 2:247, 2004.

34. M. S. Daskin. *Network and Discrete Location: Models, Algorithms, and Applications,* John Wiley and Sons New Jersey, Inc., New Jersey, 2013.

35. R. Z. Farahani, and M. Hekmatfar. *Facility Location Concepts, Models, Algorithms and Case Studies. Contributions to Management Science,* Springer Dordrecht Heidelberg London, Inc., New York, 2009.

36. H. Abouee-Mehrizi, S. Babri, O. Berman, and H. Shavandi. Optimizing Capacity, Pricing and Location Decisions on A Congested Network with Balking. *Mathematical Methods of Operations Research,* 74:233-255, 2011.

37. D. L. Huff. A Programmed Solution for Approximating an Optimum Retail Location. *Land Economics,* 42:293-303, 1966.

38. S. Benati, and P. Hansen. The Maximum Capture Problem with Random Utilities: Problem Formulation and Algorithms. *European Journal of Operational Research,* 143:518-530, 2002.

39. A Konak, D. W. Coit, and A. E. Smith. Multi-Objective Optimization Using Genetic Algorithms: A tutorial. *Reliability Engineering and System Safety,* 91:992-1007, 2006.

40. “Multi-Obejctive Optimization”, http://www.noesissolutions.com/Noesis/design-optimization/optimize/multi-objective-optimization [retrieved 10 March 2016].