SINGLE-FACILITY LOCATION PROBLEM AMONG
TWO-DIMENSIONAL EXISTING FACILITY LOCATIONS

A Thesis Presented to

The Faculty of the
Fritz J. and Dolores H. Russ
College of Engineering and Technology
Ohio University

In Partial Fulfillment
of the Requirements for the Degree

Master of Science

by
Balajee Thangavelu
March 2003

OHIO UNIVERSITY
LIBRARY
# TABLE OF CONTENTS

LIST OF TABLES ........................................................................................................ VI
LIST OF FIGURES ...................................................................................................... VII

1. INTRODUCTION ....................................................................................................... 1
   1.1 MOTIVATION FOR RESEARCH ............................................................................ 2
   1.2 SCOPE AND SIGNIFICANCE OF RESEARCH ....................................................... 4
       1.2.1 Objective Function ................................................................................... 4
       1.2.2 Distance Metric ....................................................................................... 6
       1.2.3 Feasible Space ......................................................................................... 6
       1.2.4 Number of Facilities ............................................................................... 7
       1.2.5 Factors Related to Facility Location Problem ............................................. 8
   1.3 TWO-DIMENSIONAL FACILITY LOCATION PROBLEM .................................... 9
   1.4 STRUCTURE OF THESIS ................................................................................ 11

2. LITERATURE REVIEW ............................................................................................ 13
   2.1 WEISZFELD’S ALGORITHM ............................................................................... 13
   2.2 OVERVIEW OF WEBER PROBLEM AND WEISZFELD’S ALGORITHM .......... 15
   2.3 APPROACH ON AREA-DEMAND LOCATION PROBLEMS ................................ 19

3. METHODOLOGY ...................................................................................................... 22
   3.1 NUMERICAL EXAMPLE FOR WEISZFELD’S ALGORITHM ................................. 22
   3.2 EXTENSION OF WEISZFELD’S ALGORITHM .................................................... 24
   3.3 COMPARISON OF RESULTS OF WEISZFELD’S ALGORITHM FOR VARIED ....... 25
       DEMARCATION POINTS OF SAME 2-DIMENSIONAL SHAPE .............................. 25
       3.3.1 Four-Squares Model ............................................................................... 26
           3.3.1.1 Four Square – Four Points Case ......................................................... 27
           3.3.1.2 Four Squares – One point Case ......................................................... 28
       3.3.2 Three Squares Model ................................................................................ 30
           3.3.2.1 Three Square – Four Point Case ....................................................... 30
           3.3.2.2 Three Square – One Point Case ....................................................... 32
       3.3.3 Irregular Shape Model .............................................................................. 34
   3.4 CALCULATION OF EXPECTED DISTANCES USING CLOSED INTEGRAL FORM FOR VARIED DEMARCATION POINTS ......................................................... 39

4. CONCLUSIONS ........................................................................................................ 44
5. FUTURE WORK .................................................................................................48
BIBLIOGRAPHY ....................................................................................................51
APPENDIX: CODE FOR DEVELOPED ALGORITHM ...........................................56
ABSTRACT ..............................................................................................................79
LIST OF TABLES

Table 1. Coordinates of Four Squares – Four point model ........................................ 28
Table 2. Coordinates of Four square one-point model ........................................ 29
Table 3. Coordinates of Three square four-point model ........................................ 31
Table 4. Coordinates of Three square one-point model ........................................ 33
Table 5. Comparison of new location for Three-square models .............................. 34
Table 6. Coordinates of Irregular shape model ...................................................... 36
Table 7. Analysis of Irregular shape model ............................................................ 37
Table 8. Comparison of expected distances for Three-square models .................... 42
Table 9. Comparison of new location for Four square models ............................... 44
Table 10. Comparison of new location for Three-square models .......................... 46
LIST OF FIGURES

FIGURE 1. REPRESENTATIVE FEASIBLE SPACES ................................................................. 7
FIGURE 2. EXAMPLE OF A 2-DIMENSIONAL FACILITY LOCATION PROBLEM ............. 10
FIGURE 3. FOUR SQUARES – FOUR POINT FACILITY LOCATION PROBLEM ............... 27
FIGURE 4. FOUR SQUARES – ONE POINT FACILITY LOCATION PROBLEM .................. 29
FIGURE 5. THREE SQUARES – FOUR POINT FACILITY LOCATION PROBLEM ............... 31
FIGURE 6. THREE SQUARES – ONE POINT FACILITY LOCATION PROBLEM ............... 33
FIGURE 7. IRREGULAR SHAPE FACILITY LOCATION PROBLEM .................................. 35
FIGURE 8. X-COORDINATE: WEIGHTS VS DEMARCATION UNITS ............................... 38
FIGURE 9. Y-COORDINATE: WEIGHTS VS DEMARCATION UNITS ............................... 38
FIGURE 10. EXPECTED DISTANCE BETWEEN ORIGIN AND AN AREA A ....................... 39
FIGURE 11. POINT TO AREAS ...................................................................................... 49
FIGURE 12. AREA TO POINTS ...................................................................................... 49
FIGURE 13. AREA TO AREAS ...................................................................................... 50
Chapter 1

1. Introduction

Facility location problems play a prominent role in the industrial world. Locations of factories, warehouses and distribution centers have a direct impact on a company’s bottom line. One especially appreciates the importance of these types of problems in more complex facility design and location problems. The diversity of these problems ranges from one to many new facility locations and location problems on a plane versus on a network. Although the study of facility location can be dated back to Galileo, the modern-day facility location problem was first introduced in 1909 by Alfred Weber [1]. He considered the problem of locating a warehouse to minimize the travel distance between the warehouse and its customers.

Facility location is an area of analytical study that has been active for the past thirty years. The areas spanned by facility location are as wide as they are deep. The field of location theory had many primary applications like locating firefighting vehicles, classification yards in a rail network, solid waste disposal sites, exchange locations in a telephone network, factory sites and track checking stations on a rail line. Geographers, urban planners, retailers, civil engineers, industrial engineers, park rangers, distribution analysts, purchasers, hospital administrators, and even politicians on the campaign all deal with facility location problems [2].
1.1 Motivation for research

The problem discussed here has a long and convoluted history. Players from many fields of study stepped on this stage, and some of them stumbled. The problem seems disarmingly simple, but is so rich in possibilities and traps that it has generated an enormous literature dating back to the seventeenth century [3].

Modern-day location theory was formally introduced in 1909 by Alfred Weber, who considered the problem of locating a single warehouse to minimize the total travel distance between the warehouse and a set of partially distributed customers. This work was reconsidered by Israd [4] with his study of industrial location, land use, and related problems. Another early location problem was formulated by Hotelling [5], an economist who considered the problem of locating two competing vendors in a straight line. This work was later extended by Smithies [6] and Stevens [7].


During the mid-1960s, work in the field of location theory consisted primarily of a number of separate applications which were not tied by a unified theory. Those include the location of: firefighting vehicles (Valinsky) [11]; classification yards in a rail network (Mansfield and Wein) [12]; solid waste disposal sites (Wersen et al.) [13]; exchange
locations in a telephone network (Rapp) [14]; factory sites (Burstall et al.) [15]; and track checking stations on a rail line (Young) [16].

More theoretical interest in location problems was sparked by the seminal paper by Hakimi [17] who considered the general problems of locating one or more facilities on a network to minimize either the sum of distances or the maximum distance between facilities and points on a network. Since then, considerable research has been carried out in the field of location theory. A number of different classes of problems have been identified and solved, and location methodologies have been extended to a variety of practical applications.

Although location theory has been an active area of research for the last 30 years, only limited attempts have been made to systematically review and describe location problems. Francis et al. [18], in an excellent survey of selected location research, reviewed four classes of problems (continuous planar; discrete planar; mixed planar; and discrete network problems), concentrating on optimization models for which "reliable algorithms" have been developed. Krarup and Pruzan [19] reviewed planar and network versions of the p-median and p-center problems, and in another work surveyed capacitated and incapacitated plant location problems [20]. Aikens [21] presented a survey of warehouse location models. Scott [22] reviewed a selected set of multi-server location problems, while ReVelle et al. [23] surveyed public and private sector location models. Francis and White [24] examined problems relating to facility layout and location, and Handler and Mirchandani [25] considered network location problems.
Domschke and Drexl [26] provided an extensive bibliography of location-related problems.

As evidenced by a growing body of literature, location theory is an active field of research, with many new types of problems emerging in recent years.

1.2 Scope and significance of research

Four mandatory components of every facility location problem are Objective Function, Distance Metric, Feasible Subspace and the Number of Facilities to be located [27]. The significance of these components will be discussed individually in the following sections.

1.2.1 Objective Function

Typically, the objective of a location problem is to either minimize or maximize the sum of the weighted distances between the facilities and the customers. The objective function of a facility location problem could be

- Minisum
- Minimax
- Maxisum
- Maximin
- Set covering
- Maximal covering
- Other
Minisum is minimizing the sum of the distances between the new facility to be located and the existing facility locations (customers). It is also known as the median problem. Minimax is nothing but minimizing the maximum distance between the existing facility location and new facility to be located. It is also known as the center problem.

Maxisum is maximizing the sum of the distances between the new facility to be located and the existing facility locations. Maximin is maximizing the minimum distance between the existing facilities and the new facility to be located. A set covering objective function refers to the problem in which a minimal number of new facilities to be located is determined having a constraint that the new facility cannot be further than some pre-specified distance away from the existing facility. The maximal covering problem knows the number of new facilities to be located, but the distance between the existing facility and the new facilities is to be maximized. These are the frequently used objective functions of a facility location problem.

Other than the objective functions mentioned above there are few more, including synchronous hybrid, adversarial hybrid, and equisum. These objective functions are yet to be explored in detail. Synchronous hybrid objective function is one in which, a linear combination of like objective functions is defined. Synchronous hybrid objective function can be either partially minimax and partially minisum or partially maximin and partially maximum. Adversarial hybrid objective function has both attraction (positive demand) and repulsion (negative demand) from existing facilities. Equisum objective function seeks for equilibrium for all customers of a particular problem domain.
1.2.2 Distance Metric

The most common distance metric is the $l_p$ distance metric, where $l_p$ is given by,

$$ l_p = \left( \sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p} \quad \text{(1.2.2.1)} $$

where,

- $X_i = X$-coordinate of the facility location
- $Y_i = Y$-coordinate of the facility location
- $p = 1$, rectilinear or “Manhattan” distance metric
- $p = 2$, Euclidean or “Straight-line” distance metric
- $n = \text{Number of facility locations}$

The less common forms of distance metrics include the weight $1-\infty$ norm, block norm, round norm, mixed norms, Jaccard, expected, central, and mixed gauges among others.

1.2.3 Feasible Space

There are three main types of feasible space: continuous, network, and discrete.

Examples of these spaces are shown in Figure 1.
Figure 1. Representative feasible spaces

A third mandatory feature of facility location problems is the space in which they are modeled. Feasible space is the spatial topology in which both the existing facilities and the new facility reside. Feasibility of the space is problem specific.

A facility location problem could have discrete, continuous, mixed, acyclic network, generalized network, vertex-constrained network or spherical among others. In continuous space, there is no restriction on the location of the new facility. The new facility could be located anywhere inside the convex hull. In the case of discrete space problems, the new facility has to be located among one of the discrete points. Network distances are considered in the case of network location problems. Network location problems include aisle and road networks, river networks, air transport networks made up of flight corridors, and ocean transport networks made up of shipping lanes.

1.2.4 Number of Facilities

Next to the feasible subspace, the number of facilities to be located is considered to differentiate a facility location problem. It could be single, multiple, multiple allocation, or as a decision variable. Single-facility location is the simplest of those four
different types in the number of facilities category. In the case of multiple facility locations, more than one new facility is to be located but with assumption that all new facilities can satisfy same amount of demand to the customers. In this multiple allocation case, the amount of demand satisfied by the new facilities is also determined. In the decision variable case, the number of new facilities to be located is unknown.

1.2.5 Factors Related to Facility Location Problem

Facility location problems can also vary based upon demand portrayal, market competition, facility setup costs, facility capacity, region symmetry, facility utilization, facility type and planning horizon. Demand portrayal deals with how the demand of a particular problem is portrayed. Demand portrayal could be either discrete (points) or continuous demand. In each case, the portrayal could be either weighted (non-uniform demand) or unweighted (uniform demand). Demand portrayal could also be a mix of point demands and continuous demands. Market competition discusses about the aberrations of location problems that incorporate economic competition models.

Facility setup costs involve cost incurred by new facilities. New facilities to be located could also be either capacitated or uncapsicitated (no capacity restriction). Region symmetry categorizes the problem to have symmetric distances or asymmetric distances (one-way street). Facility utilization depicts the difference between a congested or overburdened facility and a normal facility. Facility type could be either fixed (bricks and mortar) or mobile (idle automated guided vehicle systems). Planning horizon of a facility investigates about the problems that consider multiple planning horizons. The first four
indexes of the facility location taxonomy (objective function, distance metric, feasible space and number of facilities) attract the maximum attention from researchers.

A Literature survey shows the depth of research that has been carried on in the area of location science. Of all the problems portrayed earlier, the Weber problem is the most famous. The Weber problem is also known as the “spatial median” problem. This problem is a minisum Euclidean distance planar single-facility location problem.

1.3 Two-Dimensional Facility Location Problem

The objective of this thesis is to develop an analytical model for locating a single new facility on a plane modeled as a point among several existing facilities, which are modeled as 2-dimensional shapes as shown in Figure 2. Analytical models exist for facility location problems, which are modeled as points. In this thesis Weiszfeld’s algorithm has been extended to locate a single-facility among the existing facilities, which are modeled as 2-dimensional shapes.

Two-dimensional facility location problems have real life applications. Location of a tool crib in a cell with a job shop environment would be an example for its application. A job shop environment with shared resources has non-dedicated tools. Location of a tool crib has to more precise in that small area. The solution space for these problems is always convex. Solution space cannot be convex only if there is repulsion (negative weights) from the existing facility locations. An application with an extension of Weiszfeld’s algorithm was developed for solving problems of this nature as a part of
this thesis. It locates a single new facility among the existing facilities, which are represented as areas.

![Diagram of facility location problem](image)

**Figure 2. Example of a 2-dimensional facility location problem**

Each existing location is represented by an area. Each of these areas is demarcated by a rectangular grid into several points. The accuracy of the new facility location depends on the procedure to divide the existing facility location into small squares. This behavior would be prominent when working with irregular shapes as facility locations.

An existing algorithm (Weiszfeld’s Algorithm) has been incorporated for locating the best new location among the given existing locations. The inputs for this algorithm are the various existing facility locations in the form of Cartesian coordinates and their corresponding weights. The accuracy in the results of this algorithm depends on the closeness of the existing facility locations and the number of existing facility locations.
This algorithm had to be run several times to get the optimal solution. Visual Basic and Microsoft Excel applications are the most likely tools that could be used for automating this solution. Manual calculation of these iterations would consume hours together as the number of iterations to be performed before we reach the minimal difference between the new facility locations, would be more than five hundred.

The proposed algorithm deals with the new facility location as a point. In practice this may or may not be true. The new facility to be located might be an area. Another issue to be discussed is about the number of new facilities to be located. In our problem we have located single new facility, however one may have to locate multiple new facilities. The existing facilities may overlap among themselves. The above-mentioned factors could provide a good scope for the future study on this topic. The proposed model is the just the first step into the facility location problems but it is devised in such a way that any further improvement would be compatible without much change to the existing algorithm.

1.4 Structure of Thesis

This thesis consists of four chapters. Chapter one introduces the concept of facility location and it also discusses the significance of this research. This chapter also states the problem to be solved in a descriptive manner. Chapter two discusses the extensive literature review of this thesis. It provides the details on the articles that paved the way for this research. Chapter three covers the methodology of this thesis. It states the situation of the problem, methods to develop the algorithm, test cases used to prove the
accuracy of the developed algorithm and also the comparative study. Chapter four gives
the inferences of the thesis talking about the merits and demerits of the approach to the
problem. Chapter five gives recommendations for future study on this topic.
Chapter 2

2. Literature Review

The problem of locating a new facility with respect to the locations of a number of existing facilities (demand points) has generated a large body of literature. Models of the location problems often assume that the transportation costs from the new facility to a demand point are proportional to the Euclidean distance between these points, with the proportionality (weight) depending on the demand point. Two kinds of objectives to be minimized arise frequently: the sum of the transportation costs (the minisum problem) and the greatest transportation cost (the minimax problem). Here, the Euclidean minisum location problem is the focus. This problem is also known as generalized Fermat problem, the Steiner problem, the Weber problem, and the Euclidean one-facility location problem.

2.1 Weiszfeld’s Algorithm

Euclidean Distance Metric:

There exist a variety of distance metrics used in facility location problems, namely $l_p$, weight $1-\infty$ norm, block norm, round norm, mixed norms, Jaccard, expected, central and mixed gauges.

The most common distance metric of all is the $l_p$ distance metric. When $p$ takes a value of 1, it is known as rectilinear distance metric and when $p$ takes a value of 2, it becomes the Euclidean distance metric. Weiszfeld’s algorithm is based on Euclidean distance metric.
Weiszfeld's algorithm locates the new facility among the existing facility locations considering the distance to be Euclidean. The objective of this algorithm is to minimize the function in equation 2.1.2.

\[ f(x, y) = \sum_{i=1}^{m} w_i \left( (x - a_i)^2 + (y - b_i)^2 \right)^{1/2} \]  \quad \ldots \ldots (2.1.2)

where,
- \( m \) = number of existing facilities
- \( w_i \) = weight for \( i^{th} \) existing location
- \( a_i \) = X-coordinate of the \( i^{th} \) existing location
- \( b_i \) = y-coordinate of the \( i^{th} \) existing location

The equation 2.1.2 calculates the sum of the distances between the existing locations and the new location. Weiszfeld's algorithm follows an iterative procedure in determining the new facility location. The termination criterion for the iterative procedure is the difference between the successive objective function values.

\[ \gamma_i (x, y) = \frac{w_i}{\left( (x - a_i)^2 + (y - b_i)^2 + \varepsilon \right)^{1/2}} \]  \quad \ldots \ldots (2.1.3)
where,

\[ X_n = \text{X-coordinate of starting point for the iterative algorithm} \]

\[ Y_n = \text{Y-coordinate of starting point for the iterative algorithm} \]

\[ \varepsilon = \text{A very small number to avoid local convergence of the algorithm} \]

\[ \Gamma (x, y) = \sum_{i=1}^{m} \gamma_i (x, y) \] ..........................(2.1.4)

\[ \lambda_i (x, y) = \frac{\gamma_i (x, y)}{\Gamma (x, y)} \] ..........................(2.1.5)

\[ WF (x, y) = \sum_{i=1}^{m} \lambda_i (x, y)(a_i, b_i) \] ..........................(2.1.6)

The variables in equations 2.1.4, 2.1.5 and 2.1.6 are utilized in the iterative procedure of the calculation of the objective function. A new facility location is chosen randomly and used for the calculation of the second trial value. This second value is substituted and the third value is found. This process is stopped when the distance between two successive trial values is less than or equal to \( \varepsilon \) (almost zero).

2.2 Overview of Weber Problem and Weiszfeld’s Algorithm

Major contributions to the literature on the facility location models have dealt with the variations and extensions of the Weber problem.
Cooper [28] presented an extension to the generalized Weber problem. According to him the cost should be proportional to the distance raised to some power. This extension of the Weber problem gives greater flexibility in realistically and accurately fitting the cost data.

Francis and Cabot [29] considered a problem of locating new facilities in a plane with respect to new facilities. The distance between the existing and new facilities is assumed to be Euclidean. They located the new facilities by minimizing the total cost function. They also developed a dual approach to this problem. They showed that the resulting new facility would lie in a convex hull of the existing facilities.

Love and Morris [30] analyzed the methods to calculate the distances between points on a road network. They introduced the concept of empirical distance functions for improved accuracy in estimating travel distances. They also concluded that the urban distances are not necessarily rectangular. The Euclidean distance is more accurate to use than the rectangular distances for urban distance calculations as the distances were not necessarily rectangular.

Morris and Verdini [31] generalized Weiszfeld’s algorithm for $l_p$ distances. They developed a differentiable approximating function, which uniformly converges to the original objective function, as the smoothing constant tends to zero. They also extended this approximating function to the case of multifacility location problems.

Morris [32] also gave a generalization to the Weber problem. He considered the case in which the distance function is some power of the $l_p$ distance. The Weiszfeld iterative procedure is given based on this new case study.
Rado [33] gave an extension to the Weiszfeld algorithm. This extension and the generalization of the Kuhn-Ostresh convergence theorem helped him to prove the convergence and optimality of the proposed algorithm.

Rosen and Xue [34] analyzed the relationship between Miehle’s multifacility location algorithm and Weiszfeld’s single-facility location algorithm. If the location problems are well structured (where convergence to local optima is possible), then Miehle’s algorithm for Euclidean multifacility location problem can easily fail to converge to the optimal solution.

Rosen and Xue [35] dealt with the Euclidean multifacility location problem. They gave a proof for the global convergence of the HAP (Hyperboloid approximation procedure). They proved that it is a descent algorithm and it converges to the minimizer of the objective function from any initial point.

Brimberg [36] clarified the open question about the convergence of Weiszfeld’s algorithm. He proved that Weiszfeld’s algorithm converges to the unique optimal solution for all but a denumerable set of starting points, provided the convex hull of the given points is of dimension N.

Xue [37] analyzed the problem of facility location on the surface of a sphere. He proved the hull property of this problem. He compared the planar Euclidean facility location problem with the corresponding approximate solution to the spherical facility location problem. His proposed algorithm converges to a global minimizer of the spherical facility location problem.
Parlar [38] considered the case of mixed distance metrics in the single-facility location problem. The distance metric could be both Euclidean and rectilinear in the same problem. He modified the Weiszfeld algorithm to suit this specific case. He also verified the results by comparing them with global optimization method.

Drezner and Wesolowsky [39] analyzed the minisum and the minimax single-facility location problem in a network containing one-way streets. This complicates the distance consideration of the problem but has the ability to improve the traffic flow. They showed that it is optimal to consider the location of new facilities only at the nodes of the network of interest.

Brimber and Love [40] showed that the convex hull property holds when any so-called “round form” is used to measure the travel distance of a location problem. They also showed that the transportation costs are increasing differentiable functions of the travel distances. The hull property also holds for multi-facility location problems. They concluded that all the local minima must also lie within the convex hull of the existing facilities.

Drezner [41] proposed a simple procedure for accelerating the convergence of Weiszfeld’s algorithm, based on Aiken’s process. This acceleration procedure involves two successive iterations and he assumed that the difference between the successive points forms a geometric series. So the next iteration is assumed to be limited by the geometric series. This accelerating procedure reduces the run time by a factor of two.

Brimberg, Chen and Chen [42] presented a method to accelerate the convergence in a generalized Fermat-Weber problem with $l_p$ distances. They suggested multiplying the
step size of the Weiszfeld algorithm by a factor, which is a function of the parameter $p$. This factor reduces the number of iterations that are required to arrive at the best solution.

Üster and Love [43] analyzed the convergence properties of the Weiszfeld algorithm applied to a single-facility location problem with $l_p$ distance metric, with $p>2$. They introduced a step-size factor into the Weiszfeld iterative procedure. This factor aids in the convergence of the algorithm when $p>2$.

Suzuki and Drezner [44] analyzed the problem of locating $p$ facilities. The objective is to minimize the maximal distance for all demand points, which originates from an area. They solved the problem by covering every point in the area by $p$ circles with the smallest possible radius. They proposed a heuristic procedure to handle the above problem.

2.3 Approach on Area-Demand Location Problems

The articles above give a history on the evolution of the optimal facility location problem involving two dimensional demands. The following articles will provide a much narrower description of how area demand has been studied in the literature.

Love and Wesolowsky [45] addressed the facility location problem in which the destinations could be single points, lines, or rectangular areas. The distance metric of interest was rectilinear. They proposed methods to solve both the single and multiple facility location problems by a gradient reduction procedure. This method deals with the optimal location of one or more facilities among any number of destinations (areas or points). The areas are restricted to be rectangles with each side parallel to the axes. The
overlapping of the areas is also taken into consideration for the location of the optimal facility. To start with, they solved single-facility location problem and then two-facility location problem. With the help of the two solved procedures they have generalized the n-facility location problem.

Drezner and Wesolowsky [46] dealt with a single-facility location problem with $l_p$ distances with $p>1$ and the destinations could be circular, rectangular or any general shape. The distance metric on the problem is Euclidean. They proposed an iterative procedure, which is based on the Weiszfeld’s algorithm to solve the above problems. The weight density was assumed to be uniformly distributed if the destinations are areas. The proposed method is more generalized than the other methods as Euclidean distances are permitted and circles and other general shapes are also accepted in addition to rectangles.

Drezner and Wesolowsky [47] introduced the problem of locating a facility among the group of demand points. The distance between the facility and the group-demand points could be either closest point in the group or the farthest point in the group or it could be the average distance of all the points in the group. And the objectives could be either minisum, maximin or minimax for a total of nine different problems.

The authors investigated the minisum maximal group distance problem and minimax average group-distance problem. They noted that there are few standard mathematical programming methods to help solve these problems. They devised special algorithms for larger problems in which computational complexity is high. The goal of the research was not to find solutions effectively; instead they took a general approach
where it is not assumed that group has any special cohesion or other justification to replace the group demand with a single point.

Chen [48] developed a new approach to the optimal location of a single-facility minisum location problem, where the facility serves a number of circular demand areas and also a few discrete demand points. Each circular demand area has been assumed to be uniformly distributed. The proposed algorithm is a Weiszfeld-like iterative procedure. He discussed three cases: service point being outside, inside or on the circumference of the circular demand area. He also considered a few cases such as the service point to be located is far off from the demand area. He described the amended Weiszfeld iterative procedure with some numerical experience.
Chapter 3

3. Methodology

3.1 Numerical Example for Weiszfeld’s Algorithm

The objective function of the Weiszfeld’s algorithm is to minimize the sum of the distances between the existing facility locations and the new facility location as given in equation 2.1.2. The matrices below gives the X and Y coordinates of the existing facility locations and the corresponding weights for the location.

Existing facility locations \((a_i, b_i)\):

Weights of the existing locations:

\[
\begin{bmatrix}
4,2 \\
10,9 \\
1,8 \\
9,2 \\
\end{bmatrix}
\begin{bmatrix}
2 \\
4 \\
6 \\
5 \\
\end{bmatrix}
\]

\((X_n, Y_n)\) gives the initial point with which the algorithm is start it’s iterations. It could be a default value of \((0,0)\). In this example the initial point is chosen to be \((6,6)\). This selection depends on the discretion of the user and based on where could the new facility be located.

\( (x_n, y_n) = (6,6) \)

\[
y_1(x, y) = \frac{2}{\left[ (6 - 4)^2 + (6 - 2)^2 + 0.00001 \right]^{\frac{1}{2}}} = 0.4472
\]
\[ y_2(x, y) = 0.9701 \]
\[ y_3(x, y) = 1.1766 \]
\[ y_4(x, y) = 1.3867 \]

With the help of the initial point and the existing facility locations, corresponding \( y \) values were calculated from equation 2.1.3. \( \Gamma \) is the addition of the \( y \) values for the existing facility locations. It is calculated based on equation 2.1.4.

\[ \Gamma(x, y) = (0.4442 + 0.9701 + 1.1766 + 1.3867) = 3.9806 \]

Using \( \Gamma \) and the corresponding \( y \) value, \( \lambda \) values were calculated for each of the existing facility locations based on equation 2.1.4.

\[ \lambda_1(x, y) = \frac{0.4472}{3.9806} = 0.1129 \]
\[ \lambda_2(x, y) = 0.2449 \]
\[ \lambda_3(x, y) = 0.2976 \]
\[ \lambda_4(x, y) = 0.3450 \]

With the above details, weighted function for \( X \) coordinate and \( Y \) coordinate is calculated using the equation 2.1.5. The difference between the \( X/Y \) coordinate of this new point with the \( X_n/Y_n \) would be the termination criteria for the algorithm. If the difference is not small enough (determined by \( \varepsilon \)), the new point calculated becomes the \( X_n, Y_n \) and the procedure is continued.
The above procedure should be repeated until the following condition is satisfied.

\[
WF (x_{n+1}) = \sum_{i=1}^{m} \lambda_i * a_i
\]

\[
= (0.1129 * 4) + (0.2449 * 10) + (0.2976 * 1) + (0.3450 * 9)
\]

\[
= 6.3032
\]

\[
WF (y_{n+1}) = \sum_{i=1}^{m} \lambda_i * b_i
\]

\[
= (0.1129 * 2) + (0.2449 * 9) + (0.2976 * 8) + (0.3450 * 2)
\]

\[
= 5.5009
\]

The above procedure should be repeated until the following condition is satisfied.

\[
(|x_{n+1} - x_n| + |y_{n+1} - y_n|) \leq \varepsilon
\]

Weiszfeld’s algorithm was coded in Microsoft Visual Basic 6.0 and the code is provided in Appendix A. The number of existing facility locations is set to be dynamic. The initial trial value to start the iterative procedure depends on the discretion of the user.

3.2 Extension of Weiszfeld’s Algorithm

The objective of the problem is to locate a single new facility on a plane modeled as a point among several existing facilities, which are modeled as 2-dimensional shapes. Weiszfeld’s algorithm locates the new facility among the existing facility locations, which are modeled as points. This concept has been extended for 2-dimensional objects.

Each 2-dimensional shape is divided into numerous points and the points are used as an input for Weiszfeld’s algorithm. This extension will also be valid with multi-dimensional objects. The Visual Basic application takes care of the conversion of a 2-dimensional shape into numerous points based on the boundary conditions of the 2-
dimensional shape. This concept is also true for irregular shapes, but the developed application has to be modified to accept the inputs of an irregular shape. Any shape that can be dissected into a square or a rectangle can be an input to the developed application. There lies an extensive scope for future study in the modifications that could be done to the application to accept irregular shapes of all nature.

The code for the application is provided in the appendix. The application also has provision for accepting different weights for the existing locations. After taking the input from the user it runs through the Weiszfeld’s procedure. When the distance between two successive trial values is less than or equal to 0.000000000001 (default value) the iteration stops. This termination criterion could be changed based on the user’s discretion. The best solution is recorded in the form of text and displayed for the user.

3.3 Comparison of Results of Weiszfeld’s Algorithm for Varied Demarcation Points of Same 2-Dimensional Shape

The performance of the Weiszfeld’s algorithm depends on the input data. The existing facility locations, which are modeled as 2-dimensional shapes, are demarcated into points. The results of Weiszfeld’s algorithm depict improvement as the number of demarcation units increases. There exists a tradeoff between the maximum number of demarcation units and the computational time of the algorithm. Experimentation results show that a 2-dimensional shape with more than 5000 points does not show much improvement on the results.
The consistency and accuracy of the algorithm was tested for various models. In this thesis, two models have been considered. The first model consists of four squares and the second model has three squares, where the squares represent the existing locations. In each model there are two test cases, which handle varied demarcation units for every existing location. The first test case has four demarcation units per square and the second test case has one demarcation unit per square.

### 3.3.1 Four-Squares Model

A four-square existing facilities location problem is used as a testing model. In this test model exist four squares, which are the existing locations. All the four existing locations are modeled as 2-dimensional shapes. The new facility to be located, which is modeled as a point should have the minimum distance from all the existing locations.

Two-dimensional shapes (square) are demarcated into points and fed as an input into the developed algorithm. The number of demarcation units per square opens up an area of further research. Results get better as the number of demarcation units is increased. The computational time of algorithm serves as a limitation in increasing the number of demarcation points. Computational time exceed five minutes for hundred or more demarcation points. In the first case of four-square model, four points represent the 2-dimensional shape and in the second case one point at the center of the square represents the square.

All the test cases use regular shapes for the experimentation (square, triangle), but they can also be extended to irregular shapes. The procedure for demarcation of the irregular shapes will differ from the former procedures. Procedure to divide the irregular
shapes into small squares needs some special algorithm but this was not the case with regular shapes.

3.3.1.1 Four Square – Four Points Case

This is the first case of the four-square model and the number of demarcation units in this case (points per square) is four as in Figure 3. This gives a total of sixteen points for four squares. The four points represent the corners of the square. All the four squares have a dimension of 10 by 10. The location of the points in the square is given in Table 1.

Figure 3. Four Squares – Four point facility location problem
The developed application computes the new facility location for the existing locations using Weiszfeld's algorithm. X-Y coordinates of the sixteen points become the input data for the application and the starting point is assumed to be (0,0). The new facility location found by the developed application is given in the table below.

Based on the modified Weiszfeld's algorithm, the results were found to be

X value: 14.9999
Y value: 14.9999

### 3.3.1.2 Four Squares – One point Case

In a Four square – One point case, the demarcation unit per square is one as shown in Figure 4. The one point represents the center of the square. That gives four points for the four squares. The squares are 10 by 10. X-Y coordinates of the four points representing the squares are given Table 2.
Figure 4. Four Squares – One point facility location problem

Table 2. Coordinates of Four square one-point model

<table>
<thead>
<tr>
<th>Square 1</th>
<th>X</th>
<th>Y</th>
<th>Square 2</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5</td>
<td>25</td>
<td></td>
<td>25</td>
<td>25</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Square 3</th>
<th>X</th>
<th>Y</th>
<th>Square 4</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5</td>
<td>5</td>
<td></td>
<td>25</td>
<td>5</td>
</tr>
</tbody>
</table>

The X-Y coordinates in table 2 serve as an input to the developed application that finds the new facility location among the existing facility locations. The starting point for the algorithm is assumed to be the default value.

Based on the modified Weiszfeld’s algorithm, the results were found to be,

X value: 14.99999

Y value: 14.99999
The second case results are same as the first case results of the four-square model. As the number of demarcation points of the existing locations increased, it was expected that the results would be more accurate. The four-square model has symmetry on both the x- and y-axes, and hence we know that the exact new facility location should be at (15,15). This may not be the case with all other problems that may include more complex shapes and can also lack symmetry, so a three-square model, which lacks symmetry along the x-axis, has been taken for experimentation.

### 3.3.2 Three Squares Model

This is the second test model, in which three squares are utilized to analyze the results using the modified Weiszfeld's algorithm. The three-square model differs from the four-square model in the aspect of symmetry. The four-square model has a symmetry on both the x- and y-axes and the three-square lacks symmetry with respect to the x-axis. This difference provides a more comprehensive test in analyzing the accuracy and consistency of the developed algorithm.

#### 3.3.2.1 Three Square – Four Point Case

The four corners of each square represent the three existing locations. This case has three squares with four points each making twelve existing locations as in Figure 5. Each square has a dimension of 10 by 10. The x and y coordinates of the twelve points representing the existing locations are given in Table 3.
Figure 5. Three Squares – Four point facility location problem

Table 3. Coordinates of Three square four-point model

<table>
<thead>
<tr>
<th>Square 1</th>
<th>X</th>
<th>Y</th>
<th>Square 2</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>20</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
<td></td>
<td>10</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>30</td>
<td></td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>30</td>
<td></td>
<td>0</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Square 3</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>
This model does not have symmetry in its x-axis, so the exact solution for the new facility location is not known. The x-and y-coordinates in Table 3 were plugged into the developed algorithm and the obtained results are shown below.

Based on the modified Weiszfeld's algorithm, the results were found to be,

X value: 14.99999
Y value: 11.23484

3.3.2.2 Three Square – One Point Case

This is the second case of the three-square model. The Three Square – One point case has three squares, each containing one existing location in the center of the square as in Figure 6. The values for the new facility location in this case would be helpful for comparing the results with the first case of the three-square model. The squares used have the same dimension as the above cases and their coordinates are given in Table 4.
Figure 6. Three Squares – One point facility location problem

Table 4. Coordinates of Three square one-point model

<table>
<thead>
<tr>
<th>Square 1</th>
<th>X</th>
<th>Y</th>
<th>Square 2</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>15</td>
<td>25</td>
<td></td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Square 3</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>25</td>
<td>5</td>
</tr>
</tbody>
</table>

The above values serves as an input for the developed application and the following results are produced.

Based on the modified Weiszfeld’s algorithm, the results were found to be,

X value: 14.99999
Y value: 10.77351

The results from both the test cases of three-square model vary as shown in Table 5. The X-coordinate of the new facility location does not show variation but the y-coordinate has a significant variation in units place.

Table 5. Comparison of new location for Three-square models

<table>
<thead>
<tr>
<th>Per Square</th>
<th>X value</th>
<th>Y value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14.99999</td>
<td>10.77351</td>
</tr>
<tr>
<td>4</td>
<td>14.99999</td>
<td>11.23484</td>
</tr>
<tr>
<td>25</td>
<td>14.99999</td>
<td>10.93097</td>
</tr>
<tr>
<td>100</td>
<td>14.99999</td>
<td>10.90221</td>
</tr>
<tr>
<td>225</td>
<td>14.99999</td>
<td>10.89329</td>
</tr>
<tr>
<td>2500</td>
<td>14.99999</td>
<td>10.88129</td>
</tr>
<tr>
<td>5625</td>
<td>14.99999</td>
<td>10.87962</td>
</tr>
</tbody>
</table>

The above table gives the results of various cases for the three-square model. The above results does not truly comply with the expected behavior, which states that better results are obtained with more number of demarcation points. The expected behavior will be a vital factor when irregular shapes are considered for the existing locations. The above test cases being perfected geometric figures do not show significant variation in the increase of the number of demarcation units.

3.3.3 Irregular Shape Model

This is the third test model, in which irregular shapes are utilized to analyze the results using modified Weiszfeld’s algorithm. In this model there is no symmetry. This model proves that the extension of Weiszfeld’s algorithm can be utilized for any shape.
This model has two existing facility locations as shown in Figure 7. One of the existing facility locations is a square and the other one is an irregular two-dimensional existing facility location. Corner points of both the locations are used as inputs for the Weiszfeld’s algorithm. Shape 2 has a dimension of 10 by 10. The X and Y coordinates of the corner points representing the existing locations are given in Table 6.

Shape 1

Shape 2

Figure 7. Irregular shape facility location problem
This model was also used to study the effect of increasing the weights for an existing location in relation with the location of the new facility. Demarcation units were varied to analyze the location of the new facility for weighted existing facility locations. Shape 2 was considered for varying the weights. It was given a weight range from 1 to 10. Demarcation units were given a range from 4 to 225 per square block. Shape 1 has thrice the area of the shape 2. So if shape 2 has 25 points, the number of demarcation units with shape 1 would be 75. The obtained results for the above-mentioned analysis are given in Table 7.

<table>
<thead>
<tr>
<th>Shape 1</th>
<th>X</th>
<th>Y</th>
<th>Shape 2</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>30</td>
<td></td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>30</td>
<td></td>
<td>30</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>20</td>
<td></td>
<td>30</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>20</td>
<td></td>
<td>20</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Figures 8 and 9 show that X and Y coordinates of the new facility location were plotted against the demarcation units for varied weights of shape 2. As the number of demarcation units increases the variation with X and Y coordinate value decreases. This behavior is more prominent when the weights are higher. It could also be concluded that increasing to a demarcation unit of 100 to 225 will yield best results considering the computational time for the algorithm. As the weights of square 2 increase, the new facility location is pulled towards square 2. This is true as the X-coordinate value increases and the y-coordinate value decreases with the increase of weights for square 2.
Figure 8. X-coordinate: Weights Vs Demarcation Units

Figure 9. Y-Coordinate: Weights Vs Demarcation Units
3.4 Calculation of Expected Distances Using Closed Integral Form for Varied Demarcation Points

The developed algorithm served as a means to locate the new facility among the existing facilities that are modeled as 2-dimensional shapes. The new facility to be located is modeled as a point in contrast to the 2-dimensional existing locations. The expected distance from the existing locations to the new facility location has to be determined. This estimation is done using the closed integral form. The three-square model with four points in each square, has been taken as an example to demonstrate the closed integral form for the estimation of the expected distance from existing locations to the new facility location.

Expected distance functions can be defined for any pair of $n$-dimensional endpoints through the use of applied probability theory (expectation of a real valued function). Expected distance functions can be defined for any real valued distance metric of interest. Scilicet, the $E(D)_{\text{Euclidean}}$ from the origin (a point) is to the area $A$ (an area) shown in Figure 10 is defined as follows:

![Figure 10. Expected distance between origin and an area A](image-url)
\[ E(D) = \int \int_A \sqrt{x^2 + y^2} \, f_{xy}(x, y) \, dx \, dy \]

where,

\( A = \text{Area of the locations involved in expected distance calculation} \)

The generalized expected distance equation is given by,

\[ \text{Min} \ldots Z \ldots = \sum_{i=1}^m \int \int_A \sqrt{(x^* - X)^2 + (y^* - Y)^2} \, f(X, Y) \, dX \, dY \]

Where,

\( x^* = \text{X-coordinate of the new facility location} \)

\( y^* = \text{Y-coordinate of the new facility location} \)

\( A = \text{Area of the locations involved in expected distance calculation} \)

\( f(X,Y) = (1/A) \)

Due to the presence of the embedded integrals, the generalized equation becomes cumbersome to solve.

As a solution to minimize the expected distances from the existing and new facility location, an aggregation procedure is followed. Weiszfeld’s algorithm locates the new facility among the existing facility locations, which are modeled as points. This concept has been extended to solve the location problems, which involves two-dimensional objects. All the existing facility locations (two-dimensional objects) are
demarcated into points and fed as input to the Weiszfeld’s algorithm. Weiszfeld’s algorithm comes out with the best possible new facility location that has the minimum expected distance from the existing locations.

Three Square – Four Point Case:

The applied probability methodology to calculate the expected distances has been used to prove the consistency of the results obtained by the extension of Weiszfeld’s algorithm.

First Square: \[ [(0,0) (10,0) (10,10) (0,10)] \]

Second Square: \[ [(20,0) (30,0) (30,10) (20,10)] \]

Third Square: \[ [(10,20) (20,20) (20,30) (10,30)] \]

The expected distance between the existing locations and the new location is based on the equation 3.3.2.

\[
\int_0^{10} \int_0^{10} \left( (X_s - x)^2 + (Y_s - y)^2 \right)^{1/2} dx \, dy + \int_{20}^{30} \int_0^{10} \left( (X_s - x)^2 + (Y_s - y)^2 \right)^{1/2} dx \, dy + \int_0^{10} \int_{20}^{30} \left( (X_s - x)^2 + (Y_s - y)^2 \right)^{1/2} dx \, dy
\]

\[ = \int_0^{10} \int_0^{10} \left( (X_s - x_1)^2 + (Y_s - y_1)^2 \right)^{1/2} dx_1 \, dy_1 \]

\[ = 100 \]

If \( X_s = 14.9999992 \) and \( Y_s = 10.773572 \),

Expected Distance =

\[
\int_0^{10} \int_0^{10} \left[ (14.9999992 - x_1)^2 + (10.773572 - y_1)^2 \right]^{1/2} dx_1 \, dy_1
\]

\[ + \]
The results for the calculation of the expected distances are given in Table 8.

Table 8. Comparison of expected distances for Three-square models

<table>
<thead>
<tr>
<th>Per Square</th>
<th>X value</th>
<th>Y value</th>
<th>Expected Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14.99999</td>
<td>10.77351</td>
<td>38.34231</td>
</tr>
<tr>
<td>4</td>
<td>14.99999</td>
<td>11.23483</td>
<td>38.35000</td>
</tr>
<tr>
<td>25</td>
<td>14.99999</td>
<td>10.93097</td>
<td>38.34181</td>
</tr>
<tr>
<td>100</td>
<td>14.99999</td>
<td>10.90221</td>
<td>38.34165</td>
</tr>
<tr>
<td>225</td>
<td>14.99999</td>
<td>10.89329</td>
<td>38.34163</td>
</tr>
<tr>
<td>2500</td>
<td>14.99999</td>
<td>10.88129</td>
<td>38.34161</td>
</tr>
<tr>
<td>5625</td>
<td>14.99999</td>
<td>10.87962</td>
<td>38.34161</td>
</tr>
</tbody>
</table>

The expected distance between the three existing locations and one new facility location is found to be 38.35 units. The same procedure is followed for all the above cases and the expected distances are calculated.

Table 8 shows the expected distances for the three-square model with various cases of demarcation points. The new facility locations for all the different cases are
obtained by plugging the respective coordinates in the developed algorithm. The expected distances are calculated using the closed integral form and the results show consistency.
Chapter 4

4. Conclusions

This thesis developed an analytical model for locating a single new facility on a plane modeled as a point among the existing facilities, which in turn are modeled as 2-dimensional shapes. This algorithm has been experimented with the existing facilities being modeled as squares. The number of existing facilities is also varied to demonstrate the robustness of the developed algorithm. The results obtained are found to be consistent as shown in Table 9.

Table 9. Comparison of new location for Four square models

<table>
<thead>
<tr>
<th>Per Square</th>
<th>X value</th>
<th>Y value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14.99999</td>
<td>14.99999</td>
</tr>
<tr>
<td>4</td>
<td>14.99999</td>
<td>14.99999</td>
</tr>
<tr>
<td>25</td>
<td>14.99999</td>
<td>14.99999</td>
</tr>
<tr>
<td>100</td>
<td>14.99999</td>
<td>14.99999</td>
</tr>
<tr>
<td>225</td>
<td>14.99999</td>
<td>14.99999</td>
</tr>
<tr>
<td>2500</td>
<td>14.99999</td>
<td>14.99999</td>
</tr>
<tr>
<td>5625</td>
<td>14.99999</td>
<td>14.99999</td>
</tr>
</tbody>
</table>

The developed algorithm has been tested varying the demarcation units for the same 2-dimensionally-modeled existing locations. A four-square problem was selected for the analysis, as the new facility location (optimal solution) to be determined is known in this case. Each 2-dimensionally-modeled existing location (square) was demarcated into 1, 4, 25, 100, 225, 2500 and 5625 points. The developed algorithm determined the new facility location in all the above cases. The results are shown in the Table 9. The new
facility location should be at (15.0,15.0). The results were compared with the known optimal solution. The results were found to be the same.

Based on the discussion of the irregular shape model the optimal solution is reached with more demarcation points. This activity of getting to the optimal solution is more pronounced when the weights of the existing locations are higher. This analysis proved that the case in which the existing locations have more demarcation points yields better results. Hence the performance of the developed algorithm depends on the demarcation units of the existing locations. There exists a limitation in increasing the demarcation units of an existing location. The upper bound for the demarcation has to be traded off with the computational time of the software developed. Hundred demarcation units per facility location would be an optimum number to solve this problem compared to the computational time of the algorithm.

The three-square model was considered for determining the behavior of the demarcation points with the accuracy of the new facility location. Table 10 shows the results of the new facility location for the various demarcation points. The optimal location of new facility for the three-square model is not known. This fact stands in the way of comparing the results with the actual solution, which is not known.
In general, we can conclude that if the number of demarcation points for existing locations is high then the location of the new facility will be more accurate. With existing locations being irregular shapes, which is true in most of the practical cases, we saw significant changes in the new facility location with higher weights for the existing facility locations.

Comparing the results of the problem for varied demarcation units tested the performance of the developed algorithm. To test the consistency of the developed algorithm, the expected distances calculation using closed integral form was used. A three square problem was selected to demonstrate this attribute of the algorithm. The new facility location was determined using the developed algorithm. These results were utilized in the expected distance calculation from the existing locations to the new facility location with the closed integral form.

The expected distances were calculated for all the different cases, in which the number of demarcation points was 1, 4, 25, 100, 225, 2500 and 5625. The calculated

<table>
<thead>
<tr>
<th>Per Square</th>
<th>X value</th>
<th>Y value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14.999999</td>
<td>10.773502</td>
</tr>
<tr>
<td>4</td>
<td>14.999999</td>
<td>11.23483</td>
</tr>
<tr>
<td>25</td>
<td>14.999999</td>
<td>10.93097</td>
</tr>
<tr>
<td>100</td>
<td>14.999999</td>
<td>10.90221</td>
</tr>
<tr>
<td>225</td>
<td>14.999999</td>
<td>10.89329</td>
</tr>
<tr>
<td>2500</td>
<td>14.999999</td>
<td>10.88129</td>
</tr>
<tr>
<td>5625</td>
<td>14.999999</td>
<td>10.87962</td>
</tr>
</tbody>
</table>
expected distances should have demonstrated a consistent value, but there existed a variation in the results in the order of third decimal. This variation is due to the difference present in the new facility location used for the calculation of expected distances and it also depends on the machine precision.
Chapter 5

5. Future Work

Even though the developed algorithm locates a single new facility among the existing facility locations that are modeled as 2-dimensional shapes, it is not comprehensive. The developed algorithm deals with the new facility location as a point. In practice this is probably not true. The new facility to be located might also be a 2-dimensionally modeled shape.

The number of new facilities to be located would be another issue to receive significant consideration. In our problem we have located a single new facility. However one may have to locate multiple new facilities. The existing facilities may overlap among themselves. The above-mentioned factors could provide a good scope for the future study on this topic.

Three multi-dimensional extensions of traditional facility location models have been shown (P-to-A, A-to-P, and A-to-A) in Figures 11, 12 and 13.
Figure 11. Point to Areas

Figure 12. Area to Points
Algorithm to solve Point to Area problems was developed as a part of this thesis. Overlapping of the areas in Point to Area problems were not considered while developing this model. It opens up an area for future research. Future research would be aimed at developing algorithms to help solve Area to Point and Area to Area location problems.
Bibliography


Appendix: Code for developed algorithm

**ModuleWeisz:**

Public ctri As Integer
Public xvalue() As Double
Public yvalue() As Double
Public data() As Double
Public gamma() As Double
Public lamda() As Double
Public weizxval() As Double
Public weizyval() As Double

Public Sub Grid_validation(grid As MSFlexGrid, KeyAscii As Integer)

Select Case KeyAscii

    Case Asc("0") To Asc("9")
        grid.Text = grid.Text & Chr(KeyAscii)

    Case Asc(".")
        If InStr(grid.Text, ".") = 0 Then
            grid.Text = grid.Text & "."
        End If

    Case Asc("-")
        If InStr(grid.Text, "-") = 0 Then
            grid.Text = "-" & grid.Text
        Else
            grid.Text = Right(grid.Text, Len(grid.Text) - 1)
        End If

Case vbKeyReturn
    With grid
        .Row = ctri

        Do While ctri < 3
            .Col = ctri
            If grid.Text = "" Then
                MsgBox ("Please fill all the columns")
                Exit Sub
            End If
        Loop
    End With

End Sub
End If

    ctrj = ctrj + 1

Loop

grid.Rows = grid.Rows + 1

ctri = tri + 1

End With

Case vbKeyBack

    If grid.Text <> "" Then
        grid.Text = Left(grid.Text, Len(grid.Text) - 1)
    End If

MsgBox ("Dont leave the cell blank, enter a value")

Case Else

' do nothing

End Select

End Sub

Public Sub celldivisionsq1()

' For Square 1

Dim i, j, k, l, m, n, o, p As Integer
Dim xparting, yparting, xdiv, ydiv As Double

With frmCellsdiv

    .txtXvalue(0) = 0
    .txtXvalue(1) = 10
    .txtXvalue(2) = 10
    .txtXvalue(3) = 0

    .txtYvalue(0) = 20
    .txtYvalue(1) = 20
    .txtYvalue(2) = 30
    .txtYvalue(3) = 30

    xparting = Val(.txtXvalue(1)) - Val(.txtXvalue(0))
    yparting = Val(.txtYvalue(2)) - Val(.txtYvalue(1))
xdiv = xparting / Val(.txtUnits)
ydiv = yparting / Val(.txtUnits)

ReDim xvalue(Val(.txtUnits))
ReDim yvalue(Val(.txtUnits))

xvalue(0) = Val(.txtXvalue(0))
yvalue(0) = Val(.txtYvalue(1))

For i = 1 To Val(.txtUnits)
    xvalue(i) = xvalue(i - 1) + xdiv
    yvalue(i) = yvalue(i - 1) + ydiv
Next i

frmweisz/Gridweiz.Rows = ((Val(.txtUnits) + 1) ^ 2) * 4
frmweisz/Gridweiz.Cols = 3

For j = 0 To ((Val(.txtUnits) + 1) ^ 2) * 4 - 1
    frmweisz/Gridweiz.Row = j
    frmweisz/Gridweiz.Col = 2
    frmweisz/Gridweiz.Text = "1"
Next j

m = 0

For k = 0 To Val(.txtUnits)
    For l = 0 To Val(.txtUnits)
        frmweisz/Gridweiz.Row = m
        frmweisz/Gridweiz.Col = 0
        frmweisz/Gridweiz.Text = xvalue(k)
        m = m + 1
    Next l
    m = m + 1
Next k

p = 0
For n = 0 To Val(txtUnits)
    For o = 0 To Val(txtUnits)
        frmweisz.Gridweiz.Row = p
        frmweisz.Gridweiz.Col = 1
        frmweisz.Gridweiz.Text = yvalue(o)
        p = p + 1
    Next o
    Next n
    frmweisz.Show
End With
End Sub

Public Sub celldivisionsq2()
    'For Square 2
    Dim i, j, k, l, m, n, o, p As Integer
    Dim xparting, yparting, xdiv, ydiv As Double
    With frmCellsdiv
        .txtXvalue(4) = 20
        .txtXvalue(5) = 30
        .txtXvalue(6) = 30
        .txtXvalue(7) = 20
        .txtYvalue(4) = 20
        .txtYvalue(5) = 20
        .txtYvalue(6) = 30
        .txtYvalue(7) = 30
        xparting = Val(.txtXvalue(5)) - Val(.txtXvalue(4))
        yparting = Val(.txtYvalue(6)) - Val(.txtYvalue(5))
        xdiv = xparting / Val(txtUnits)
ydiv = yparting / Val(txtUnits)

ReDim xvalue(Val(txtUnits))
ReDim yvalue(Val(txtUnits))

xvalue(0) = Val(txtXvalue(4))
yvalue(0) = Val(txtYvalue(5))

For i = 1 To Val(txtUnits)
    xvalue(i) = xvalue(i - 1) + xdiv
    yvalue(i) = yvalue(i - 1) + ydiv
Next i

For j = (Val(txtUnits) + 1) ^ 2 To 2 * (Val(txtUnits) + 1) ^ 2 - 1
    frmweisz.Gridweiz.Row = j
    frmweisz.Gridweiz.Col = 2
    frmweisz.Gridweiz.Text = "1"
Next j

m = (Val(txtUnits) + 1) ^ 2

For k = 0 To Val(txtUnits)
    For l = 0 To Val(txtUnits)
        frmweisz.Gridweiz.Row = m
        frmweisz.Gridweiz.Col = 0
        frmweisz.Gridweiz.Text = xvalue(k)
        m = m + 1
    Next l
Next k

Next k

p = (Val(txtUnits) + 1) ^ 2

For n = 0 To Val(txtUnits)
    For o = 0 To Val(txtUnits)
Public Sub celldivisionsq3()

'For Square 3

Dim i, j, k, l, m, n, o, p As Integer
Dim xparting, yparting, xdiv, ydiv As Double

With frmCellsdiv

   .txtXvalue(8) = 0
   .txtXvalue(9) = 10
   .txtXvalue(10) = 10
   .txtXvalue(11) = 0

   .txtYvalue(8) = 0
   .txtYvalue(9) = 0
   .txtYvalue(10) = 10
   .txtYvalue(11) = 10

   xparting = Val(.txtXvalue(9)) - Val(.txtXvalue(8))
   yparting = Val(.txtYvalue(10)) - Val(.txtYvalue(9))

   xdiv = xparting / Val(.txtUnits)
   ydiv = yparting / Val(.txtUnits)

ReDim xvalue(Val(.txtUnits))
ReDim yvalue(Val(.txtUnits))
xvalue(0) = Val(.txtXvalue(8))
yvalue(0) = Val(.txtYvalue(9))

For i = 1 To Val(.txtUnits)
   xvalue(i) = xvalue(i - 1) + xdiv
   yvalue(i) = yvalue(i - 1) + ydiv
Next i

For j = 2 * (Val(.txtUnits) + 1) ^ 2 To 3 * (Val(.txtUnits) + 1) ^ 2 - 1
   frmweisz.Gridweiz.Row = j
   frmweisz.Gridweiz.Col = 2
   frmweisz.Gridweiz.Text = "1"
Next j

m = 2 * (Val(.txtUnits) + 1) ^ 2

For k = 0 To Val(.txtUnits)
   For l = 0 To Val(.txtUnits)
      frmweisz.Gridweiz.Row = m
      frmweisz.Gridweiz.Col = 0
      frmweisz.Gridweiz.Text = xvalue(k)
      m = m + 1
   Next l
Next k

p = 2 * (Val(.txtUnits) + 1) ^ 2

For n = 0 To Val(.txtUnits)
   For o = 0 To Val(.txtUnits)
      frmweisz.Gridweiz.Row = p
      frmweisz.Gridweiz.Col = 1
      frmweisz.Gridweiz.Text = yvalue(o)
   Next o
Next n
p = p + 1
Next o
Next n
frmweisz.Show
End With
End Sub

Public Sub celldivisionsq4()
'For Square 4
Dim i, j, k, l, m, n, o, p As Integer
Dim xparting, yparting, xdiv, ydiv As Double

With frmCellsdiv
   .txtXvalue(12) = 20
   .txtXvalue(13) = 30
   .txtXvalue(14) = 30
   .txtXvalue(15) = 20

   .txtYvalue(12) = 0
   .txtYvalue(13) = 0
   .txtYvalue(14) = 10
   .txtYvalue(15) = 10

   xparting = Val(.txtXvalue(13)) - Val(.txtXvalue(12))
   yparting = Val(.txtYvalue(14)) - Val(.txtYvalue(13))

   xdiv = xparting / Val(.txtUnits)
   ydiv = yparting / Val(.txtUnits)

   ReDim xvalue(Val(.txtUnits))
   ReDim yvalue(Val(.txtUnits))

   xvalue(0) = Val(.txtXvalue(12))
   yvalue(0) = Val(.txtYvalue(13))
For i = 1 To Val(.txtUnits)
    xvalue(i) = xvalue(i - 1) + xdiv
    yvalue(i) = yvalue(i - 1) + ydiv
Next i

For j = 3 * (Val(.txtUnits) + 1) ^ 2 To 4 * (Val(.txtUnits) + 1) - 1
    frmweisz.Gridweiz.Row = j
    frmweisz.Gridweiz.Col = 2
    frmweisz.Gridweiz.Text = "1"
Next j

m = (Val(.txtUnits) + 1) ^ 2 * 3

For k = 0 To Val(.txtUnits)
    For l = 0 To Val(.txtUnits)
        frmweisz.Gridweiz.Row = m
        frmweisz.Gridweiz.Col = 0
        frmweisz.Gridweiz.Text = xvalue(k)
        m = m + 1
    Next l
Next k

p = (Val(.txtUnits) + 1) ^ 2 * 3

For n = 0 To Val(.txtUnits)
    For o = 0 To Val(.txtUnits)
        frmweisz.Gridweiz.Row = p
        frmweisz.Gridweiz.Col = 1
        frmweisz.Gridweiz.Text = yvalue(o)
        p = p + 1
    Next o
Next n
Public Sub celldivisionsq5()

' For Square 5

Dim i, j, k, l, m, n, o, p As Integer
Dim xparting, yparting, xdiv, ydiv As Double

With frmCellsdiv
    .txtXvalue(0) = 10
    .txtXvalue(1) = 20
    .txtXvalue(2) = 20
    .txtXvalue(3) = 10
    .txtYvalue(0) = 20
    .txtYvalue(1) = 20
    .txtYvalue(2) = 30
    .txtYvalue(3) = 30

    xparting = Val(.txtXvalue(1)) - Val(.txtXvalue(0))
    yparting = Val(.txtYvalue(2)) - Val(.txtYvalue(1))

    xdiv = xparting / Val(.txtUnits)
    ydiv = yparting / Val(.txtUnits)

    ReDim xvalue(Val(.txtUnits))
    ReDim yvalue(Val(.txtUnits))

    xvalue(0) = Val(.txtXvalue(0))
    yvalue(0) = Val(.txtYvalue(1))

    For i = 1 To Val(.txtUnits)
        xvalue(i) = xvalue(i - 1) + xdiv
    Next n

    frmweisz.Show

End With
End Sub
yvalue(i) = yvalue(i - 1) + ydiv

Next i

frmweisz.Gridweiz.Rows = ((Val(.txtUnits) + 1) ^ 2) * 3
frmweisz.Gridweiz.Cols = 3

For j = 0 To ((Val(.txtUnits) + 1) ^ 2) * 3 - 1

    frmweisz.Gridweiz.Row = j
    frmweisz.Gridweiz.Col = 2
    frmweisz.Gridweiz.Text = "1"

Next j

m = 0

For k = 0 To Val(.txtUnits)

    For l = 0 To Val(.txtUnits)

        frmweisz.Gridweiz.Row = m
        frmweisz.Gridweiz.Col = 0
        frmweisz.Gridweiz.Text = xvalue(k)
        m = m + 1

    Next l

Next k

p = 0

For n = 0 To Val(.txtUnits)

    For o = 0 To Val(.txtUnits)

        frmweisz.Gridweiz.Row = p
        frmweisz.Gridweiz.Col = 1
        frmweisz.Gridweiz.Text = yvalue(o)
        p = p + 1

    Next o

Next n
Public Sub Calculate()

Dim ctrl As Integer, ctrk As Integer, count As Integer
Dim x As Double, y As Double, sumgamma As Double, a As Double, b As Double
Dim sumweizxval As Double, sumweizyval As Double

frmweisz.Gridweiz.Enabled = True
Select Case Index
Case 0
    intConverge = 0
    x = 0
    y = 0
    With frmweisz.Gridweiz
        Do
            sumgamma = 0
            sumweizxval = 0
            sumweizyval = 0
            On Error GoTo errorhandler:
            For ctrk = 0 To .Rows - 1
                .Row = ctrk
                For ctrl = 0 To 2
                    .Col = ctrl
                    ReDim Preserve data(ctrl)
                    data(ctrl) = Val(.Text)
                    Next ctrl
                    ReDim Preserve gamma(ctrl)
                    gamma(ctrl) = data(2) / ((x - data(0))^2 + (y - data(1))^2 + 0.0000000001)^0.5
                    sumgamma = sumgamma + gamma(ctrl)
                    Next ctrk
                For ctrk = 0 To .Rows - 1
                    ReDim Preserve lamda(ctrl)
                    lamda(ctrl) = gamma(ctrl) / sumgamma
                    Next ctrk
            ...
For ctrk = 0 To .Rows - 1
    .Col = 0
    .Row = ctrk
    ReDim Preserve weizxval(ctrk)
    weizxval(ctrk) = lamda(ctrk) * Val(.Text)
    sumweizxval = sumweizxval + weizxval(ctrk)
    .Col = 1
    ReDim Preserve weizyval(ctrk)
    weizyval(ctrk) = lamda(ctrk) * Val(.Text)
    sumweizyval = sumweizyval + weizyval(ctrk)
Next ctrk

a = x
b = y

x = sumweizxval
y = sumweizyval

Loop While (Abs(x - a) > 0.000000000001) Or (Abs(y - b) > 0.0000000000001)

    frmweisz.Gridweiz.Enabled = True

End With

    frmweisz.Text1.Text = sumweizxval
    frmweisz.Text2.Text = sumweizyval

Case 1
End
End Select

errorhandler:

    If Err.Number = 6 Then
    MsgBox ("Dont leave the rows empty, else quit and start again")
    End If

End Sub

Public Sub Foursquares()
Call celldivisionsq1
Call celldivisionsq2
Call celldivisionsq3
Call celldivisionsq4
'Call Calculate

End Sub

Public Sub Threesquares()

    Call celldivisionsq5
    Call celldivisionsq3
    Call celldivisionsq4
    'Call Calculate

End Sub
**FrmCellsdiv:**

Option Explicit

Private Sub cmdCelldiv_Click()
    If frmCellsdiv.txtUnits <> "" Then
        Call Foursquares
        'Call Threesquares
    Else
        MsgBox "Enter the demarcation units"
    End If
End Sub

Private Sub cmdPrint_Click()
    frmCellsdiv.PrintForm
End Sub
Private Sub cmdCalculate_Click(Index As Integer)

    Call Calculate

End Sub

Private Sub cmdPrint_Click()

    frmweisz.PrintForm

End Sub

Private Sub cmdQuit_Click(Index As Integer)

    End

End Sub
Private Sub Form_Load()

    'Call Weiz_16
    'Call Weiz_4
    'Call Weiz_12
    'Call Weiz_3

End Sub

Private Sub Gridweiz_KeyPress(KeyAscii As Integer)

    Call Grid_validation(Gridweiz, KeyAscii)

End Sub

Private Sub Weiz_16()

    With Gridweiz
        .Row = 0
        .Col = 0
        .Text = "0"
        .Col = 1
        .Text = "0"
        .Col = 2
    End With

End Sub
End With

End Sub

Public Sub Weiz_4()

With Gridweiz

.Row = 0
.Col = 0
.Text = "5"
.Col = 1
.Text = "5"
.Col = 2
.Text = "1"
.Row = 14
.Col = 0
.Text = "20"
.Col = 1
.Text = "30"
.Col = 2
.Text = "1"
.Row = 15
.Col = 0
.Text = "30"
.Col = 1
.Text = "30"
.Col = 2
.Text = "1"

End With

End Sub

Public Sub Weiz_4()
.Row = 2
.Col = 0
.Text = "25"
.Col = 1
.Text = "5"
.Col = 2
.Text = "1"
.Row = 3
.Col = 0
.Text = "25"
.Col = 1
.Text = "25"
.Col = 2
.Text = "1"
End With
End Sub

Public Sub Weiz_12()
With Gridweiz
  .Row = 0
  .Col = 0
  .Text = "0"
  .Col = 1
  .Text = "0"
  .Col = 2
  .Text = "1"
  .Row = 1
  .Col = 0
  .Text = "10"
  .Col = 1
  .Text = "0"
  .Col = 2
  .Text = "1"
  .Row = 2
  .Col = 0
  .Text = "20"
  .Col = 1
  .Text = "0"
  .Col = 2
  .Text = "1"
  .Row = 3
  .Col = 0
  .Text = "30"
Private Sub Weiz_30

With Gridweiz
  .Row = 0
  .Col = 0
  .Text = "5"
  .Col = 1
  .Text = "5"
  .Col = 2
  .Text = "1"
  .Row = 1
  .Col = 0
  .Text = "25"
  .Col = 1
  .Text = "5"
  .Col = 2
  .Text = "1"
  .Row = 2
  .Col = 0
  .Text = "15"

End With

End Sub
.Col = 1
.Text = "25"
.Col = 2
.Text = "1"

End With

End Sub