SOLUTION OF CERTAIN LOCATIONAL PROBLEMS ARISING IN L_1 NORM

A THESIS
SUBMITTED FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY (MATHEMATICS)
TO THE
UNIVERSITY OF NORTH BENGAL
1994

BY
Nisith Ranjan Chakrabarti

ANANDA CHANDRA COLLEGE
JALPAIGURI, WEST BENGAL
INDIA

DEDICATED TO MY PARENTS

ACKNOWLEDGEMENTS

This work could not have been possible without the active cooperation and guidance of Dr. Prasanta Kumar Chaudhuri, Reader in Mathematics,University of North Bengal,under whose supervision this research has been completed. The teachers and staff of the Mathematics Department of this university and my colleagues in Ananda Chandra College lent me unstinted support throughout the course of my work. I am also indebted to my relatives and friends who have never failed in extending encouragement and showing warm interest in the work. The authorities of A. C. College have made available the computer facilities without which the work would have taken much more time to be completed. My heartfelt thanks are also due to the staff of the University Science Instrumentation Centre and in partcular to its Head Dr. S.K. Sengupta for providing access to the centre. I wish to thank the inmates of the Research Scholars' Hostel for their sincere cooperation. Possibly the project would never actualise without the financial support and leave made available to me under the F. I. P. scheme of the University Grants Commission. I would like to take this opportunity to express my indebtedness to Prof. A. Banerjee for giving various illuminating suggestions which I have tried to incorporate in my work. Last, but not the least, I owe my gratitude to my wife who had to bear the full brunt of the various inconveniences that the work entailed.

Nisithranjan Chakrabarti

TABLE OF CONTENTS

CHAPTER 1

Introduction and Preliminaries

This dissertation is divided into four chapters.

Chapter 1 introduces some basic concepts and notations we employ in this thesis and serves as a prelude to the current work. Geometric solution procedures find important applications in the problems involving location of facilities considered by us.

Chapter 2 considers the unconstrained problem of locating a single service centre in the plane in the presence of existing location points using the criterion of *minimising the maximum* weighted rectilinear distance, symmetric as well as asymmetric, and obtains the solution analytically by exploiting the geometrical structure of the problem. Asymmetric weight is typically exemplified by rush hour traffic and similar other situations.

Chapter 3 deals with the problem of locating a single service centre catering to the demands of customers distributed over a finite set of demand points in a two-dimensional space employing both the symmetric and the non-symmetric Manhattan metric *minimax* criterion. An exact solution technique, based on geometry, has been presented, under the assumption that the required centre should be situated within a convex polyhedral region.

Finally, chapter 4 focusses on the computational aspects of some of the above mentioned problems and presents algorithms having a polynomial time complexity.

1.1 Origin and development of different aspects of locational problems

Facility layout and location problems have been the subject of study for centuries. The ancient Greeks are known to have been fascinated by this subject. A version of the Euclidean distance location problem was posed by Fermat [16, 51] as a purely geometrical problem in the early seventeenth century and solved by Torricelli [36] around 1640, which may be stated thus: given three points in the plane, find a fourth one such that the sum of the distances to the three said points is a minimum. Cavalieri [36] in 1647 reviewed the problem and Jacob Steiner [16, 52], a Swiss mathematician, early in the nineteenth century, made an attempt to solve the problem posed as a classic geometry problem in the special case of equal weights while Alfred Weber [76], a German economist,in his pioneering work towards the beginning of this century,once again studied the weighted version, also known as the Steiner-Weber problem or the general Fermat problem, which consists in locating a warehouse in such a way that the total weighted distance travelled between the warehouse and a set of demand points is a minimum. The dual of this problem was solved by Fasbender [36] towards the middle of nineteenth century. But it was Kuhn [50] in 1963 who was the first to have attempted a purely mathematical approach in order to find a solution to the problem.

Although facility layout and location problems continued to receive considerable attention over the years it was only after practitioners in OR began exhibiting interest that the subject became the focal point of attraction for several disciplines.

Locational analysis deals with the study and development of methodologies seeking to determine the locations of new facilities in such a way that the users of the facilities are benefited most. By constructing suitable models which involve locating one or more new facilities, and solving them, the investigation is carried out. With the passage of time, however, the formulation of the location problem has undergone radical change.

In solving facility layout and location problems models simple rather than highly sophisticated although closely approximating the real world have been developed. In the analysis of facility layout and location problems the process of verifying if the model accurately represents the physical system under study is most important. In obtaining a solution to the problem its formulation and analysis have to be carried out at the outset.

1.1.1 Selection of criterion

The criterion of minimising some function of distance, either unweighted or weighted by some importance factor, the weight being interpreted as cost per unit of distance from a demand point to a facility, is perhaps the most natural choice. If the new facility is a factory supplying warehouses or a new machine to be located in a plant layout or a point in a network to be connected to known points in the network (the existing facilities), its location may be determined in such a way that the total cost which is directly proportional to the distances involved, will be a minimum. Sometimes, instead of minimising the total distance travelled, it may be required to minimise the maximum distance, which is actually the minimax counterpart of the more familiar Fermat problem. Such a criterion is most natural in locating some emergency facility for which the maximum delay is more important than the average or total delay incurred as a measure of effectiveness and has been rightly called the 'grease the squeaky wheel' criterion by some authors inasmuch as the objective is to minimise the effects of the worst situation, viz., the maximum cost.

1.1.2 Choice of norms

There are many a distance measure we may define on the plane. A general distance family is L_p distance defined as

$L_p(A_1, A_2) = \left(|x_1 - x_2|^p + |y_1 - y_2|^p\right)^{1/p}$, where p is the distance parameter $(1 \le p \le \infty)$, and (x_1, y_1), (x_2, y_2) are the coordinates of two given points A_1 and A_2. Two distance measures, however, received much attention, viz., L_1, variously called the rectilinear, rectangular or Manhattan distance and L_2, the euclidean or straight line distance. In urban location analyses travel usually takes place along an orthogonal set of streets. In problems involving machine location travel occurs along a set of rectangular aisles arranged parallel to the walls of the building where the machine is housed. In such situations rectilinear distance is the appropriate metric. Whereas for some network location or pipeline design problems or problems involving air travel euclidean distance is conveniently applied. For relatively small areas on the earth's surface, which may be treated with considerable precision as a sphere, the planar model offers a very good approximation. When the existing facilities are widely separated, the area covering these may be projected onto a plane and the location of the required facility may be determined using an appropriate location model on the plane, but only at the expense of introducing errors into the analysis. Although there exists a number of mapping techniques producing accurate projections of a sphere onto a plane, in problems involving large regions on the globe none preserves distance. On a sphere, therefore, we should apply the shortest arc distance, also called the geodesic norm, useful for global optimisation [1, 3, 5, 19, 20, 23, 32, 79]. Recently, location problems on the sphere have been the centre of much critical attention in the literature. Besides these there exist other norms notable among which are ring radial, hyper-rectilinear, block norms etc.

1.1.3 Minimax and minisum location problems in networks

Location theory evoked considerable interest after the publication of a seminal paper by Hakimi [42] who considered the general problem of locating one or many facilities on a network employing the minimax or the minisum criterion.

Problems dealing with the determination of optimal location of service centres in networks or graphs abound in practical situations. In particular, if a graph represents a road network with its nodes representing communities, one may have to optimally locate a hospital, police station, fire station or any other emergency service facility. In such situations, the optimality criterion may be the minimisation of the distance or travel time from the facility to the farthest node of the graph or the optimisation of the worst case. In a more general situation, a number of such facilities rather than a single facility, may be required to be located such that the remotest node of the graph can be reached from at least one of the facilities within a minimum distance. The problem of locating emergency facilities with a view to minimising the largest travel distance to any node from its nearest facility is naturally called the minimax location problem and the facilities the centres of the graph (Christofides [14]).

There is a different class of location problems where the objective is to minimise the sum total of the distances from the nodes to the central facility, assuming that a single such facility is to be located. The problem of locating a depot in a road network where the nodes represent customers or switching centre in a telephone network where the nodes represent subscribers, calls for such objectives. Problems of this type are consequently called minisum location problems, although the objective may be the sum of various functions of distances rather than the sum of distances. The resultant facility locations are then known as the medians of a graph.

1.1.4 The euclidean MSC problem

The minimum spanning circle problem, also known as the euclidean 1-centre problem, may be stated as the problem of covering a finite set of points in a plane with the smallest possible circle (Preparata and Shamos [67]). This is a classical problem originally posed by Sylvester[71, 72] in 1857, who while continuing the search for an efficient algorithm ultimately hit upon a graphical solution procedure only in 1860 and attributed the same to Peirce. This algorithm was rediscovered by Chrystal[15] twenty five years later. A modern account of their treatment may be found in Rademacher and Toeplitz [68]. The smallest enclosing circle thus obtained is unique and is either the circle circumscribing some three points of the set forming an acute triangle or described by two of them as diameter. Thus a finite algorithm that examines all pairs and triplets of points and determines the minimum circle enclosing the set was obtained. The complexity of this algorithm was $O(n^4)$ and Elzinga and Hearn [31, 32] suggested an improvement that would run in $O(n^2)$ time. Shamos [70] proposed an algorithm which depends on the determination of the farthest point of the Voronoi diagram requiring computational complexity of $O(n \log n)$. This amount of computational effort is at least needed for any solution algorithm. However, Megiddo [59] suggested a linear time algorithm by transforming the minimum spanning circle problem into a two - dimensional LP formulation. In the MSC problem, also called a minimax facilities location problem in Operations Research parlance, we seek a point, the required centre of the circle, the greatest distance from which to any point of the set is a minimum. The minimax criterion is most suitable in locating emergency facilities to reduce worst case response time to a minimum (Toregas et al. [74]). It has also been successfully implemented to locate a radio transmitter serving a number of discrete receivers or a radar station catering to the demands of several defence installations so that the RF power determined by the radius of the covering circle is a minimum (Nair and Chandrasekaran [65]). By treating the smallest covering circle problem as a continuous optimisation problem, a number of iterative algorithms has appeared, notable among them being Lawson [53], and Zhukhovitsky and Avdeyeva [80] algorithms. Jacobsen [48] has developed an algorithm relying on a specialised implementation of the method of feasible directions.

1.1.5 Generalisations of the single facility problem

The m-centre problem forms the most important class of problems in location theory and may be formally stated as follows: Given a set $D = \{d_1, d_2, \ldots, d_n\}$ of n demand points on a plane, find a set $S = \{s_1, s_2, \ldots, s_m\}$ of unknown locations of m supply points on the plane such that the furthest distance between the demand points and their closest supply points is as close as possible. Mathematically speaking,

$$\text{Minimise } z,$$

$$\text{where } z = \max_{1 \le i \le n} \{ \min_{1 \le j \le m} \{L_p(d_i, s_j)\}\}$$

Problems such as these find important applications in models concerning location of service facilities, as for example, hospitals, shopping centres, fire departments, police stations, radio or TV centres, or in many equity models in economics where the communities represent demand points. The m-centre problem may be treated as a generalisation of the 1-centre problem. This problem for general m has been considered by Aneja et al. [2], Drezner [21], Hwang et al. [47], Ko et al. [49] and Vijay [75].

1.1.6 Another generalisation of the single facility problem

The multifacility location problem concerns locating any given number of variable points representing facilities with respect to any given number of fixed points corresponding to potential users applying the minisum or the minimax criterion. Let x_i (i = 1, 2, , m) denote the new facilities or the so-called variable points, a_j (j = 1, 2, , n) the fixed points or the existing facilities, w_{1ij} and w_{2ik} the weighting constants between x_i and a_j and between x_i and x_k respectively. The weights allow for the model to discriminate in importance among distances. The problem where a minimum sum of weighted distances criterion is satisfied is the following:

$$\text{minimise } z = \sum_{i=1}^{m} \sum_{j=1}^{n} w_{1ij} \; L_p(x_i, a_j) + \sum_{i=1}^{m-1} \sum_{k=i+1}^{m} w_{2ik} \; L_p(x_i, x_k)$$

whereas the problem satisfying the minimax objective is given by minimise z where

$$z = \max \left\{ w_{1ij}\, L_p(x_i,\, a_j),\ 1\le i\le m,\ 1\le j\le n; \right.$$

$$\left. w_{2ik}\, L_p(x_i,\, x_k),\ 1\le i\le m-1,\ i+1\le k\le m \right\}.$$

Hakimi [42], Frank [39,40] and Goldman [41] have studied the minimax location problem in a network while Francis [35] has dealt with the same problem on a plane. Love et al. [56] and Elzinga and Hearn [33] provide solution procedures to the multifacility minimax problem using euclidean distances. Love and Morris [54] have suggested a non-linear programming approach to the problem using generalised L_p distances. However, in an urban setting the travel paths resemble more a rectangular than a straight line distance and consequently, rectilinear distances become relevant in such situations. Wesolowsky [78] has given a parametric linear programming solution to the multifacility problem using rectilinear distances. The single facility location problem can also be considered to be a special case of the multifacility location problem stated above.

1.1.7 Algorithms

There exist several algorithms for solving the minimax problem under the Euclidean norm (Blumenthal and Wahlin [5], Castells and Melville [6], Chakraborty and Chaudhuri [11], Chatelon et al. [12], Chrystal and Peirce [15], Elzinga and Hearn [31, 32], Francis [35], Hearn and Vijay [45], Jacobsen [48], Megiddo [59], Nair and Chandrasekaran [65], Oommen[66] Rademacher and Toeplitz [68] and Shamos [70]). Two papers - one by Oommen [66] and the other by Hearn and Vijay [45] - survey the literature and compare, qualitatively and computationally, the various solution procedures. Oommen has proposed a computational scheme that synthesizes three of the best known primal feasible algorithms, viz., the Chrystal-Peirce algorithm with the Chakraborty-Chaudhuri initialisation, the Jacobsen algorithm and the Castells- Melville and Francis algorithms and conjectures that the geometric algorithm has a linear time complexity. Hearn and Vijay have demonstrated that the Chrystal-Peirce algorithm with the Chakraborty-Chaudhuri starting solution is the fastest in the equiweighted case whereas for the weighted case the Elzinga-Hearn algorithm turns out to be the fastest. Chrystal-Peirce algorithm is based on primal feasibility while Elzinga-Hearn algorithm depends on dual feasibility. The primal feasibility concept to solve a minimax location problem with an efficient starting solution, was first introduced by Chakraborty and Chaudhuri [11]. The basic idea behind this method is to cover S, the set of all demand points, by a circle. The next step consists in reducing the radius of this circle so that the demand points continue to remain within the circle. The algorithm is designed in such a way that at each iteration at least one demand point could be eliminated and no future iteration would ever need any information about this point. The dual feasibility concept consists in covering any two points of S by a minimum circle. The radius of the circle is then increased at each step to accommodate more and more demand points within the circle until one gets the minimum covering circle. The basic ideas contained in our algorithm and Chakraborty-Chaudhuri's are identical. But, in order to solve the problem using rectilinear metric we have to recast the latter and effect certain changes to meet the present situation. With this end in view the spanning circle of the euclidean 1-centre problem has been replaced by the covering diamond (Francis and White [36]), defined as follows: given any point P(a, b) and any non-negative number r we define a diamond with centre P and radius r, to be denoted by D(P, r), by the set of all points X(x, y) for which $L_1(X, P) \le r$. In symbols,

$$D(P,\ r) = \left\{ X(x,\ y):\ L_1(X,\ P) \le r \right\}.$$

1.1.8 Locating an undesirable facility using the minimax criterion

There is another important class of location problems concerned with locating an obnoxious or undesirable facility that produces pollutants of the nature of radiation, noise or harmful gases, in such a way that the smallest distance or the smallest weighted distance from a given set of demand points is maximised while remaining within a prespecified region. Such problems are naturally known as maximin problems as opposed to the minimax version. Application areas include location of a noisy facility, say a school, a prescribed distance away from residential quarters, or an infectious disease hospital, an ordnance factory, a nuclear waste disposal site, a factory spewing out effluents and the like. In such situations it is imperative to locate the facility as far away as possible from the points it actually serves. Among the papers dealing with the maximin objective we may mention the ones by Dasarathy and White [17], Drezner and Wesolowsky [26 - 28], Melachrinoudis [61, 62] and Mehrez et al. [60]. For a realistic formulation of the above problem, Melachrinoudis and Cullinane [63] developed a model based on the physical laws of transfer of the unpleasant effects associated with the installation of an undesirable facility using the minimax criterion which, in this case, minimises the maximum or worst effect of the polluting facility. This model assumes that the effect of a new undesirable facility upon an existing one follows the law of inverse square of the distance between the facilities.

1.2 Solution procedures for some rectilinear distance planar single facility problem

Among the various solution procedures available in the literature for the rectilinear one-centre problem, we shall have the occasion now to dwell upon a few amongst these, which are geometric in nature and moreover, have a direct bearing on the present work. From a purely theoretical standpoint the complexity of geometric algorithm is of interest since it sheds new light on the intrinsic difficulty in computation. The solution procedures described here include Elzinga and Hearn algorithm [31] dealing with the equal weighted case using an innovative concept of a covering diamond (sec 1.1.7), Francis algorithm [34, 36] concerning both the unweighted and the symmetric weighted cases and Drezner and Wesolowsky algorithm [25] having substantial contribution to the asymmetric weighted problem.

Let (a_i, b_i), $i \in I = \{1, 2, \ldots, n\}$ be the location of the existing facilities or demand points and (x, y) be the proposed location of the new facility or the convenience centre.

1.2.1 Elzinga and Hearn Algorithm

Elzinga and Hearn [31] have studied four variants of the minimax location problem using geometric arguments, stated as follows:

$$\min_p \max_i [L_p(P, P_i) + k_i]$$

where p = 1 or 2 and k_i = nonnegative constant. We discuss the Elzinga-Hearn algorithm for the case for which all the k_i = 0 and p = 1. All the location points are at first covered by a rectangle by moving a line having slope -1 so as to touch at least one point situated farthest left and at least another point farthest right and doing the same thing with another line with slope +1. They next consider at most four points, one on each side of the rectangle. Four equal diamonds centred at each of these are then constructed in such a way that the radius of a diamond is less than the required minimax distance. The diamonds are now allowed to expand uniformly about their centres. Any point that belongs to all four diamonds is an optimal solution. If the rectangle has unequal adjacent sides Elzinga and Hearn have shown geometrically that the perpendicular bisector of the longer sides truncated by vertical and horizontal lines constructed through the extremities of one of the shorter sides of the rectangle constitutes the solution set.

1.2.2 Francis Algorithm

Francis [34] has considered the generalised unweighted one-centre problem in the following form:

$$\underset{(x,y) \in E^2}{\text{minimise}} \quad f(x, y)$$

$$\text{where } f(x, y) = \max_{i \in I} (|x - a_i| + |y - b_i| + g_i)$$

The inclusion of the term g_i is justified by the fact that if (x, y) be the location of an ambulance, then g_i may be interpreted as the travel distance from (a_i, b_i) to the nearest hospital.

This problem may be rewritten as:

$$\text{minimise } z$$

$$\text{subject to } |x - a_i| + |y - b_i| + g_i \le z, \ i \in I$$

By manipulating these absolute value inequalities they have, after some reductions, obtained the following linear programming problem:

$$\text{minimise } z$$

$$\begin{aligned} \text{subject to} \quad & x + y - z \le a_i + b_i - g_i \\ & x + y + z \ge a_i + b_i + g_i \\ & -x + y - z \le -a_i + b_i - g_i \\ & -x + y + z \ge -a_i + b_i + g_i \\ & i \in I \end{aligned}$$

Assuming $c_1 = \min_{i \in I} (a_i + b_i - g_i)$, $c_2 = \max_{i \in I} (a_i + b_i + g_i)$,

$$c_3 = \min_{i \in I} (-a_i + b_i - g_i), \ c_4 = \max_{i \in I} (-a_i + b_i + g_i)$$

the above LPP further reduces to

$$\text{minimise } z$$

subject to

$$\begin{aligned} x + y - z &\le c_1 \\ x + y + z &\ge c_2 \\ -x + y - z &\le c_3 \\ -x + y + z &\ge c_4 \end{aligned} \qquad (1)$$

These constraints define a rectangle, two of whose parallel sides are inclined at an angle of 45° with the x-axis while the other two make an angle of 135° with it. Let

$$c_5 = \max_{i \in I} (c_2 - c_1, c_4 - c_3).$$

Then any point belonging to the line segment joining the points $\lambda(c_1 - c_3, c_1 + c_3 + c_5)$ and $\lambda(c_2 - c_4, c_2 + c_4 - c_5)$ is a minimax location with λc_5 as the optimal objective value, where $\lambda = 1/2$.

The weighted version of the above problem considered by Francis [36] may be stated as:

$$\underset{(x,y) \in E^2}{\text{minimise}} \quad f(x,y)$$

$$\text{where } f(x, y) = \max_{i \in I} [w_i(|x - a_i| + |y - b_i|) + g_i]$$

where w_i (≥ 0) is the weight associated with (a_i, b_i) and g_i may be interpreted as the time required by user i to prepare to go to the centre.

With $M = \{i : i \in I, w_i > 0\}$, $\overline{M} = \{i : i \in I, w_i = 0\}$ the above problem may be rewritten as

$$\underset{(x,y) \in E^2}{\text{minimise}} \quad f(x,y)$$

$$\text{where } f(x,y) = \max \left\{ \max_{i \in M} [w_i(|x-a_i|+|y-b_i|)+g_i], \ \max_{i \in \overline{M}}(g_i) \right\}$$

Let $\overline{g} = \max_{i \in \overline{M}}(g_i)$. If $\overline{g} \le k$ then the inequality

$$\max_{i \in M} [w_i(|x-a_i|+|y-b_i|)+g_i] \le k$$

is clearly equivalent to

$$x + y \le a_i + b_i + (k - g_i)/w_i$$

$$x + y \ge a_i + b_i - (k - g_i)/w_i$$

$$-x + y \le -a_i + b_i + (k - g_i)/w_i$$

$$-x + y \ge -a_i + b_i - (k - g_i)/w_i$$

which are the same as (1) having g_i removed, z replaced by $(k - g_i)/w_i$ and the condition $i \in I$ substituted by $i \in M$. We now define $c_1(k)$ through $c_4(k)$ as follows:

$$c_1(k) = \min_{i \in M} \left(a_i+b_i+(k-g_i)/w_i\right), \quad c_2(k) = \max_{i \in M} \left(a_i+b_i-(k-g_i)/w_i\right)$$

$$c_3(k) = \min_{i \in M} \left(-a_i+b_i+(k-g_i)/w_i\right), \quad c_4(k)=\max_{i \in M} \left(-a_i+b_i-(k-g_i)/w_i\right)$$

The set of all (x, y) such that $f(x, y) \le k$ is given by

$$S(k) = \Big\{(x, y): c_2(k) \le x + y \le c_1(k),$$
$$c_4(k) \le -x + y \le c_3(k)\Big\}$$

which is a rectangle with a pair of parallel sides at 45° and another pair of parallel sides at 135° with the x-axis. For notational convenience, the linear transformation T and its inverse T^{-1}, have been defined as follows:

$$T(x, y) = (x + y, -x + y) = (x', y') \text{ (say) and}$$

$$T^{-1}(x, y) = \lambda(x-y, x+y), \text{ where } \lambda = 1/2.$$

The numbers α_{ij} and β_{ij} for all $1 \le i < j \le n$ are defined thus:

$$\alpha_{ij} = \max\Big([w_i\ w_j\ |a'_i - a'_j| + w_i\ g_j + w_j\ g_i]/(w_i + w_j),$$
$$g_i,\ g_j \Big)$$

$$\beta_{ij} = \max \left([w_i \ w_j \ |b'_i - b'_j| + w_i \ g_j + w_j \ g_i]/(w_i + w_j), \ g_i, \ g_j \right)$$

Geometrically, α_{ij} represents the value of k when both the coordinates of either user are respectively greater than those of the other and β_{ij} represents the value of k in all other cases. Let p_1, p_2 be the indices for which

$$z_1 = \max_{1 \le i < j \le n} (\alpha_{ij}) = \alpha_{p_1 p_2}$$

and q_1, q_2 the indices for which

$$z_2 = \max_{1 \le i < j \le n} (\beta_{ij}) = \beta_{q_1 q_2}$$

Also let $r^* = \left(w_{p_1} \ a'_{p_1} + w_{p_2} \ a'_{p_2} + \varepsilon \ (g_{p_1} - g_{p_2})\right)/(w_{p_1} + w_{p_2})$ where $\varepsilon = -1$ when $a'_{p_1} \le a'_{p_2}$, and $\varepsilon = 1$ when $a'_{p_1} > a'_{p_2}$; and let $s^* = \left(w_{q_1} \ b'_{q_1} + w_{q_2} \ b'_{q_2} + \varepsilon \ (g_{q_1} - g_{q_2})\right)/(w_{q_1} + w_{q_2})$ where $\varepsilon = -1$ or $+1$ according as $b'_{p_1} \le b'_{p_2}$ or $b'_{p_1} > b'_{p_2}$. Then $z_0 = \max (z_1, z_2)$ gives the minimum objective value and $T^{-1}(r^*, s^*)$ a minimax location. To decide whether a unique location or alternative locations exist the following three cases have been considered.

Case 1. $z_0 = z_1 = z_2$ implies $T^{-1}(r^*, s^*)$ is the unique location.

Case 2. $z_0 = z_1 > z_2$ implies that any point belonging to the line segment joining $T^{-1}(r^*, s_1)$ and $T^{-1}(r^*, s_2)$ can claim to be a minimax location where s_1 and s_2 are given by

$$s_1 = \max_{i \in I} \left\{ \left(b'_i - (z_0 - g_i)\right)/w_i \right\}, \ s_2 = \min_{i \in I} \left\{ \left(b'_i + (z_0 - g_i)\right)/w_i \right\}$$

Case 3. $z_0 = z_2 > z_1$ implies that any point within the line segment defined by $T^{-1}(r_1, s^*)$ and $T^{-1}(r_2, s^*)$ is an optimal location, r_1 and r_2 being given by

$$r_1 = \max_{i \in I} \left\{ \left(a'_i - (z_o - g_i)\right)/w_i \right\}, \quad r_2 = \min_{i \in I} \left\{ \left(a'_i + (z_o - g_i)\right)/w_i \right\}$$

1.2.3 Drezner and Wesolowsky Algorithm

We now discuss Drezner and Wesolowsky's method [25] of solution of the asymmetric rectilinear minimax problem. Let the distance between the proposed location (x, y) and the demand point (a_i, b_i) to be denoted by $d_i(x, y)$, be defined as follows:

$$d_i(x, y) = d_i(x) + d_i(y) \text{ where}$$

$$d_i(x) = \begin{cases} E_i\,|x-a_i| & \text{if } x \geq a_i \\ W_i\,|x-a_i| & \text{if } x < a_i \end{cases} \quad \text{and} \quad d_i(y) = \begin{cases} N_i\,|y-b_i| & \text{if } y \geq b_i \\ S_i\,|y-b_i| & \text{if } y < b_i \end{cases}$$

E_i, N_i, W_i and S_i being the four weights to east, north, west and south respectively.

The problem considered by Drezner and Wesolowsky is as follows:

$$\underset{(x,y) \in E^2}{\text{minimise}} \left\{ f(x,y) = \max_{i \in I} \{d_i(x, y)\} \right\}$$

This problem can be restated as a linear programming problem involving 3 variables and 4n constraints as shown below:

$$\text{minimise } z$$

$$\begin{aligned} \text{subject to} \quad & E_i(x - a_i) + N_i(y - b_i) \leq z \\ & W_i(a_i - x) + N_i(y - b_i) \leq z \\ & E_i(x - a_i) + S_i(b_i - y) \leq z \\ & W_i(a_i - x) + S_i(b_i - y) \leq z \end{aligned}$$

The method of solution proposed by Drezner and Wesolowsky consists in finding a set of three demand points in such a way that the solution to this 3-point problem coincides with that of the original problem. By a lemma it has been shown that the optimal point must lie in the smallest rectangle obtained by drawing sides parallel to the coordinate axes containing all the demand points. Consequently, the solution point for the 2-point problem will lie in the rectangle constructed with these two points as opposite vertices. As a result, the solution point in this case is the point of intersection of one of the sides of the rectangle and the line of equal distances from the two given points i and j, whose equation is given by

$$X_i |x - a_i| + Y_i |y - b_i| = X_j |x - a_j| + Y_j |y - b_j|$$

where $X_r = \begin{cases} E_r, \text{if } x \geq a_r \\ W_r, \text{otherwise} \end{cases}$ and $Y_r = \begin{cases} N_r, \text{if } y \geq b_r \\ S_r, \text{otherwise} \end{cases}$, $r = i, j$.

Not all the four points are feasible. From the set of feasible points the one with the minimum objective is to be chosen.

The 3-point problem has been next decomposed into three 2-point problems. A solution point (x^*, y^*) to a 2-point problem involving i, j (say), is also a solution to a 3-point problem characterised by i, j, k iff $d_i (x^*, y^*) \geq d_k (x^*, y^*)$. Otherwise, the solution (x_o, y_o) to the 3-point problem must satisfy

$$d_i(x_o, y_o) = d_j(x_o, y_o) = d_k(x_o, y_o).$$

In such a situation (x_o,y_o) will be the point of intersection between any two lines of equal distances with respect to i, j and i, k (say), which will lie in one of the sub-rectangles enclosing the three points. The algorithm given by Drezner and Wesolowsky solves the above problem using an iterative technique and may be described as under:

Step 1. Choose any three points from the set of given demand points and solve the 3-point problem. Obtain $f(x^1, y^1)$ the objective value corresponding to the solution point (x^1, y^1) and go to step 2.

Step 2. Calculate $d^r = d_j(x^r, y^r)$ the maximum distance at the rth iteration from (x^r, y^r) to (a_j, b_j) and the corresponding $f(x^r,y^r)$ and go to step 3.

Step 3. If $d^r = f(x^r, y^r)$ then (x^r, y^r) is the optimal solution; stop. Else go to step 4.

Step 4. Introduce (a_j,b_j) by dropping one of the three points in such a way that the three retained points correspond to the maximum objective value $f(x^{r+1},y^{r+1})$, increment r by one and go to step 2.

1.3 The present methodology

The complexity of the rectilinear one-centre problem increases as the number of demand points increases. But the algorithm we are about to describe easily yields exact solution even for large size problems and as far as we know no existing algorithm uses geometric concepts for the constrained case. Before we proceed to describe our method of solution let us say a few words about the usefulness of developing a specialised algorithm. Although simplex method is quite efficient in solving minimax location problems and there are readily available LP solvers capable of solving medium to large-scale problems, we give below the reasons for working out yet another algorithm. The current solution procedure requires storing of three vectors with n components each in the symmetric weighted case whereas standard LP packages need three 4n-vectors occupying that many memory locations. As a result, the former can handle a problem having 3500 data points even on a PC. In contrast, solving a problem having 1000 points with one of the above mentioned packages makes memory management more cumbersome. The results obtained by running the Pascal program of the algorithm on a 486 PC AT after randomly generating various sets of data points in the range of 1500 to 3500 are summarised below. The number of iterations never exceeded four.

No. of data points	Average running time in secs.	Maximum no. of iterations
1500	0.36	2
2000	0.48	2
2500	0.65	2
3000	0.83	3
3500	0.92	4

With the above preliminaries let us now give a brief account of the solution procedure for the rectilinear minimax problem. The locus of a point (x, y), whose weighted rectilinear distances from two given points P_i and P_j are equal, is given by

$$l_i(x - a_i) + m_i(y - b_i) = l_j(x - a_j) + m_j(y - b_j) \qquad (2)$$

where

$$l_r = \begin{cases} u_r^+ \text{ if } x \geq a_r \\ -u_r^- \text{ otherwise} \end{cases} \text{ and } m_r = \begin{cases} v_r^+ \text{ if } y \geq b_r \\ -v_r^- \text{ otherwise} \end{cases}, \; r = i \text{ or } j$$

in the asymmetric weighted case. By asymmetric weight we mean that with every location point is associated four different weights corresponding to the four principal directions viz., left, right, up and down, with respect to a pair of mutually perpendicular lines. The weights u_r^+, u_r^-, v_r^+, v_r^- are considered positive. In the symmetric case $u_r^+ = u_r^- = v_r^+ = v_r^- = w_r$ and if, additionally, the weights are equal, $w_r = 1$. This locus has been shown to be a closed polygon in the weighted case with at most six sides enclosing the demand point having greater weight. Here by greater weight in the asymmetric case we mean $u_i^{\pm} > u_j^{\pm}$ and $v_i^{\pm} > v_j^{\pm}$. The locus reduces to an open polygon in the absence of weights. This locus will be subsequently called an equipolygon EP(i-j). The optimal solution with respect to a pair of demand points P_i, P_j has been found to lie within or on the rectangle drawn with these two points as opposite vertices, to be called R(P_i, P_j) henceforth. Our method clearly obtains the direction of descent i.e., the directed edge of the equipolygon along which the objective does not increase, leading to the boundary of the rectangle.

1.3.1 The solution procedure for the unconstrained case

Although any point may be chosen as the starting point, we take, for convenience, one of the vertices of the smallest rectangle SR whose sides are papallel to the axes of coordinates enclosing all the demand points as the initial solution point. In particular, we have taken the vertex at the rightmost bottom corner P_o(xmax, ymin) as the starting solution point. We next determine the weighted farthest demand point, say P_i, from P_o. By traversing an L-shaped path joining P_o to (a_i, ymin) to P_i, to be denoted by L(P_o, P_i), we reach a point T(p, q) equidistant from P_i and at least another demand point, say P_j. T represents L, M or N in case $a_i < a_j$ as shown in figure 1a). In case $a_i > a_j$, T denotes L or N (see figure 1b). In the figures from 1a to 3d and 6a, 6b any two opposite corners of the rectangle ABCD denote P_i, P_j. From T we continue moving in the direction of descent of the equipolygon EP(i-j) until a point E is obtained so that any one of the following possibilities is true:

(i) E is on the boundary of R(P_i, P_j) and no P_k, $k \in I \setminus \{i,j\}$, is as far away from E as P_i or P_j.

(ii) E is equidistant from three points P_i, P_j and P_k.

In case (i) E is optimal. In case (ii) if E falls within or

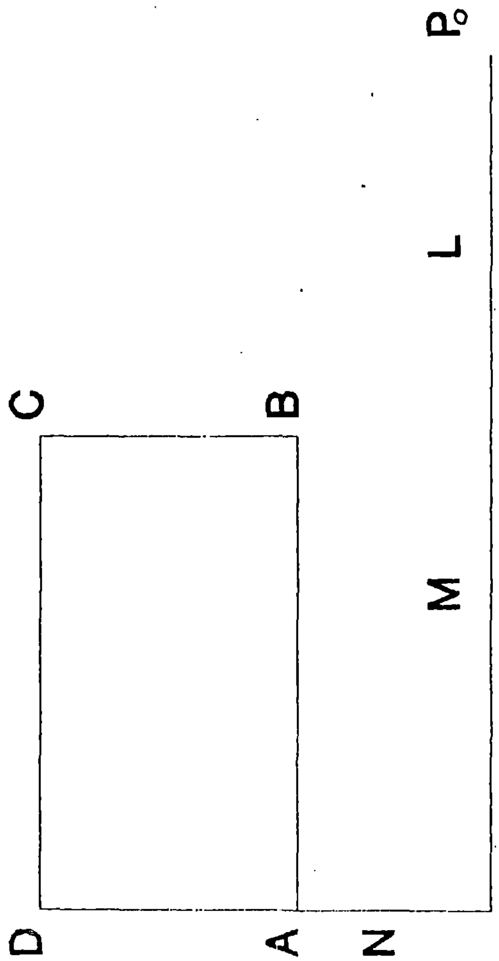


Figure 1a

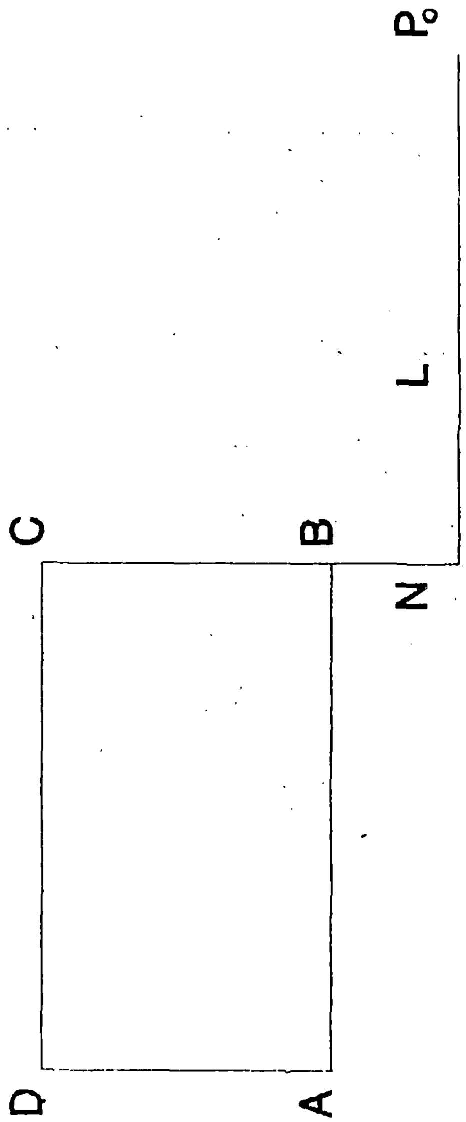


Figure 1b

on the boundary of $R(P_i, P_k)$ or $R(P_j, P_k)$ then E is optimal. Otherwise, the two points needed for the next iteration can be obtained by means of a well defined selection rule stated below. If more than three points are equidistant from E then by a repeated application of the procedure for case (ii) the pair of points required for the next iteration may be easily obtained.

Determination of the point E is carried out in two steps: first, *finding the direction of descent at T* and second, *obtaining the point of intersection of the edge of EP(i−j) containing T and EP(i−k)*, $k \in I \setminus \{i, j\}$.

Finding the direction of descent at T :

Let us denote the difference of the weighted rectilinear distances of P_i and P_j from a point X by diff(X, P_i, P_j). Refer to figures 2a and 2b for the case for which abscissa of T is greater than both a_i and a_j. If diff(A, P_i, P_j) and diff(B, P_i, P_j) be of the same sign then the descent direction TV will be given as in figure 2a; otherwise, TV will be given as in figure 2b, where V is the vertex of EP(i−j) lying on the edge containing T.

Let the abscissa of T lie between a_i and a_j. If the product of the above differences for A and B be negative then the descent direction TV is given as shown in figures 3a, 3b; otherwise, the abscissa of the demand point with the smaller weight is assigned to that of the point V and the ordinate calculated from the equation (2) of the equipolygon

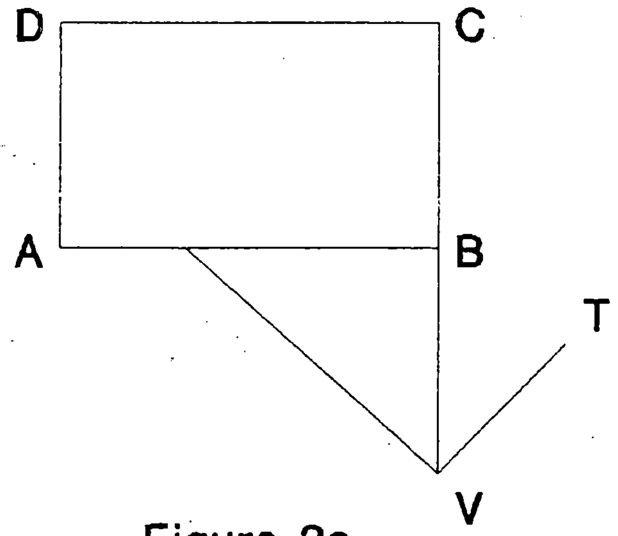


Figure 2a

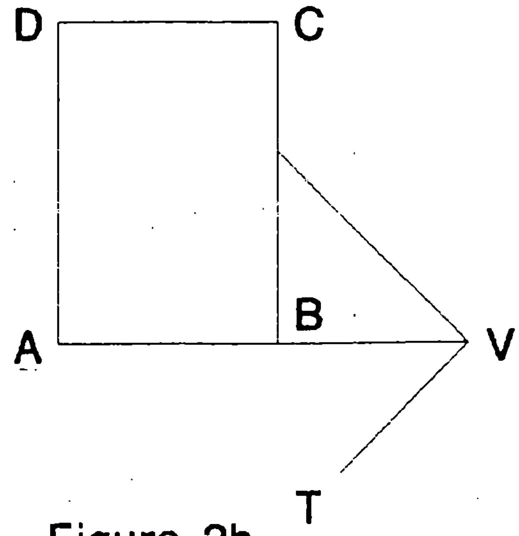


Figure 2b

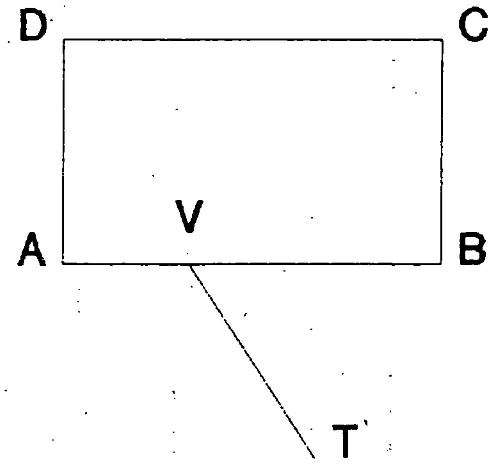


Figure 3a

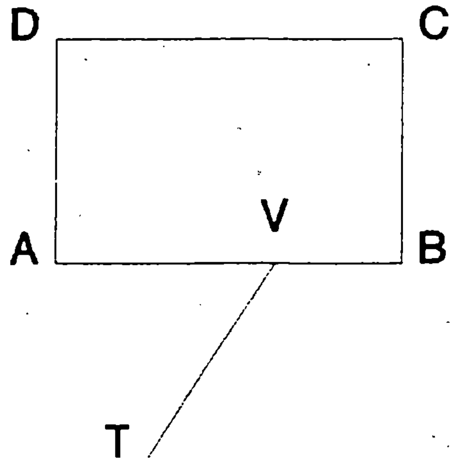


Figure 3b

(see figures 3c, 3d). If the ordinate of T lies between b_i and b_j then the position of V may be obtained in a similar manner. It should be clearly borne in mind that to reach the boundary of $R(P_i, P_j)$ we have to traverse at most three edges of EP(i-j); see figures 3c, 3d.

Obtaining the point of intersection of the edge of EP(i-j) containing T and EP(i-k) :

> If $diff(V, P_i, P_k) \le 0$, for some $k \in I \setminus \{i, j\}$, then
> $V_k := EP(i-j) \cap EP(i-k)$ exists; and $TE = \min \{TV_k\}$
> else
> if V is on the boundary of the rectangle then E ⟵ V
> else T ⟵ V and repeat the above procedure to get E.

The line segment TV of EP(i-j) may intersect either (I) $x=a_k$ and $y=b_k$ or (II) $x=a_k$ or (III) $y=b_k$ or (IV) none of the above. For case (I) see figures 4a, 4b; for cases (II),(III) and (IV) refer to figures 4c, 4d and 4e respectively. Let TG = s_1 and TH = s_2. Figure 4a corresponds to the case for which $s_1 < s_2$; figure 4b represents the case where $s_1 > s_2$. Let us consider $s_1 < s_2$. We try to obtain the point V_k first by ascertaining if it belongs to the line segment TG. If it does then no more search is necessary. Otherwise, we replace TG by GH and repeat the same procedure. If V_k is still not found we have to do the same thing with GH replaced by HV. In any case at most three searches are required to get V_k. For $s_1 > s_2$ all the above steps are needed to be performed

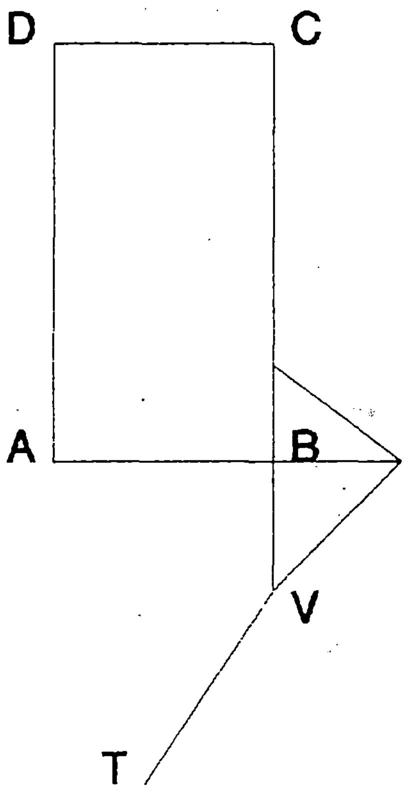


Figure 3c

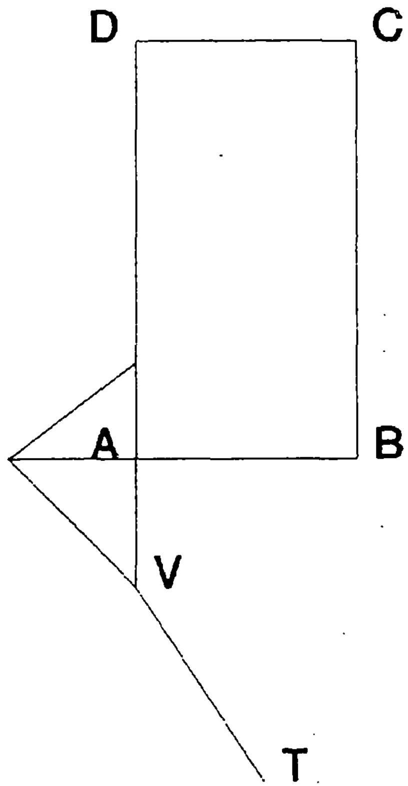


Figure 3d

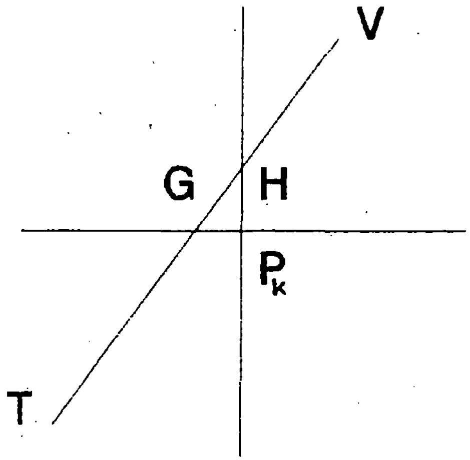


Figure 4a

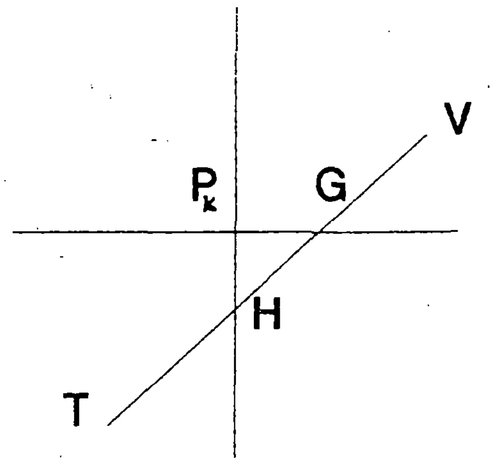


Figure 4b

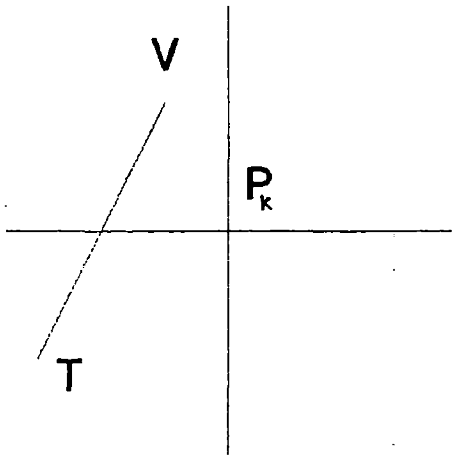


Figure 4c

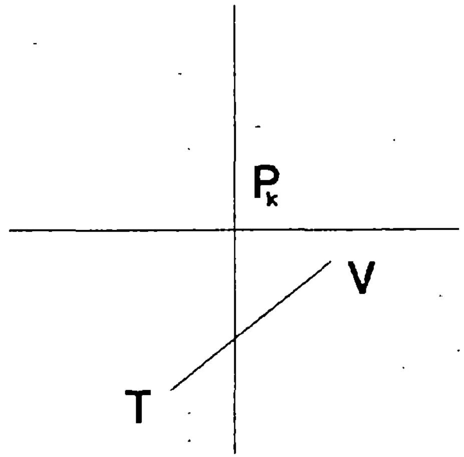


Figure 4d

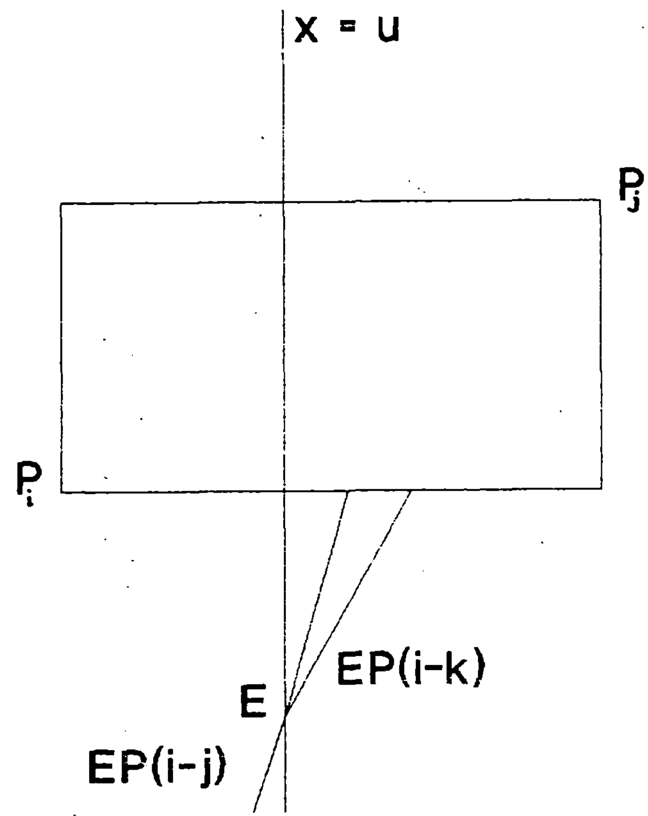


Figure 5

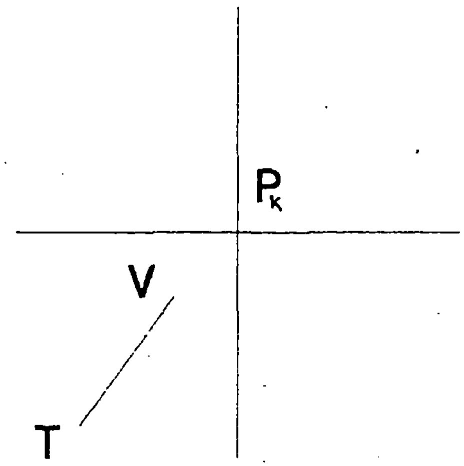


Figure 4e

taking the segments TH, HG, GV in that order. All the other cases are disposed of in a similar manner. It should be observed that cases (II) and (III) require at most two searches and case (IV) one. Although in all the figures 4a through 4e the point T has been shown to lie in the third quadrant with respect to P_k, the above arguments apply equally well had we chosen the point T in any one of the remaining three quadrants. We now describe the search technique. For an illustration let us again refer to case (I). If diff(G, P_i, P_k) < 0 then the point of intersection V_k of EP(i-j) and EP(i-k) exists within the line segment TG and V_k is given by the convex combination of the points T and G satisfying equation (2). Otherwise, if there is still some segment left to be searched we apply the above criterion to the next segment.

We now go on to describe *the criterion for selecting the pair of points needed for the next iteration when three or more equipolygons meet at E(u, v) and E is non-optimal.*

Selection Rule :

Let S_1 = {P_k| the weighted rectilinear distance from E of P_k = the weighted rectilinear distance of P_i or P_j from E; k ≠ i,j}.

All the points belonging to S_1 must be on one side of x = u (y = v). Take the points P_i, P_j and another point $P_k \in S_1$. If these three points lie on one side of x = u (y = v) then

if two of them be on the same side of y = v (x = u) then the point corresponding to the smaller of the two weights associated with these latter points is to be retained for the next iteration after relabelling them as P_i, P_j

else perform the next iteration after excluding the point with the maximum weight.

Justification of the selection Rule:

To give a concrete example let us take P_j, P_k on one side of x = u. We want to prove that *P_k has a smaller weight compared to P_j*. If possible, let us suppose that weight of P_k is greater. Since T is then outside EP(j-k) the weighted rectilinear distance of P_j is less than that of P_k contradicting primal feasibility.

We next propose to show that the direction of descent at E of EP(i-k) is pointed outside EP(i-j) when the weight of P_i is greater than that of P_j and inside, otherwise. From what has just now been proved, weight of P_j > weight of P_k implies weight of P_i > weight of P_k. Hence, by primal feasibility and also by virtue of the fact that an equipolygon encloses the greater weight, it is evident that T is outside EP(i-k). As E is the point of intersection of EP(i-j) and EP(i-k) it immediately follows that the direction of descent at E of EP(i-k) is pointed outside EP(i-j). See figure 5.

In case of the latter let us take a point $T_1 \in$ EP(i-k) in the neighbourhood of E and opposite to the direction of descent of EP(i-j) at E. Therefore, T_1 must be outside EP(i-j) in order to maintain primal feasibility thereby proving our assertion.

From the above we conclude that if we follow the direction of descent of EP(i-k) at E the weighted distances of P_i, P_k while remaining equal in relation to each other, continually diminish but this distance remains greater than that of P_j.

When P_i, P_j and P_k belong to the same quadrant with respect to E the three equipolygons coincide. Although we may select any equipolygon, without loss of generality we can exclude the point having the maximum weight from the purview of the following iteration.

Finding the stretch containing the set of optimal solutions:

Two cases are to be considered here depending on the position of the optimal point E. When E is within $R(P_i, P_j)$ and a third point P_k, as far away from E as P_i or P_j in the weighted rectilinear distance sense, is available. First the symmetric case. Following the edge of EP(i-j) through E the objective will remain unaltered. But for P_k it will decrease in one direction only which, therefore, has to be chosen as the direction of descent. $T \leftarrow E$ and proceeding exactly as the method described above for getting E we shall get TE as the required stretch. If E is on the boundary of $R(P_i, P_j)$ then we obtain a unique direction of descent which will give the stretch by applying the same arguments as has been put forward in this section. In the asymmetric case, the same arguments once again hold. But we must bear in mind that the values of the objective function at various points on the edge of EP(i-j) lying within R(P_i, P_j) are not, in general, the same.

1.3.2 The solution procedure for the constrained case

If the smallest rectangle SR containing all the demand points belongs to the convex polyhedral region CP then the unconstrained case we have already discussed is obtained. If the right hand bottom corner of SR $\in$ CP then as usual we take P_o as the starting point; otherwise, P_o is chosen arbitrarily within CP. In any case we obtain the weighted farthest point P_i from P_o and then move along L(P_o, P_i) until a point T is obtained from which P_i and another point P_j are at the same weighted rectangular distance. We denote by Y the point where the direction of movement meets the boundary.

If T $\notin$ CP then Y $\longleftarrow$ ∂CP $\cap$ L(P_o, P_i) and the boundary criterion given below is applied.

Else we follow the direction of descent of EP(i-j) so as to encounter a point E equidistant from P_i, P_j and at least one more point P_k. Now either E $\in$ CP or E $\notin$ CP.

If E $\in$ CP then

if E satisfies the criterion discussed in sec 1.3.1

then E is optimal

else apply the selection rule discussed in sec 1.3.1, drop a point from P_i, P_j and repeat this step.

Else Y ⟵ ∂CP ∩ EP(i-j) and we apply the boundary criterion. If no such E is available then

if we can move upto the point V on $\partial R(P_i, P_j)$ then

V is optimal

else Y ⟵ ∂CP ∩ EP(i-j) and the boundary criterion is applied.

Boundary criterion: We now present the procedure that determines the direction governing the movement on reaching a point Y on ∂CP. For this purpose let us introduce the concept of the *Cone of descent direction* which will be found to be useful in our subsequent discussion. Let $H_i(Y)$ denote the halfspace at Y defined by the isoline of P_i through Y containing P_i. Then $H_i(Y) \cap CP$ is the cone of descent direction provided no EP(i-j), $j \in I\setminus\{i\}$, passes through Y. Otherwise, $H_i(Y) \cap H_j(Y) \cap CP$ is the cone of descent direction as shown in figures 6a, 6b. Let $\mathcal{E}$ represent this cone. If $\mathcal{E} = \{Y\}$ then Y is the unique optimal solution. The portion of ∂CP constituting an extreme direction of $\mathcal{E}$ will give the direction of the next movement. If P be any point lying within EP(i-j) the weighted rectilinear distance from P to the demand point having greater weight is less than the weighted rectilinear distance from P to the other point whereas if P is outside EP(i-j) this property is reversed.

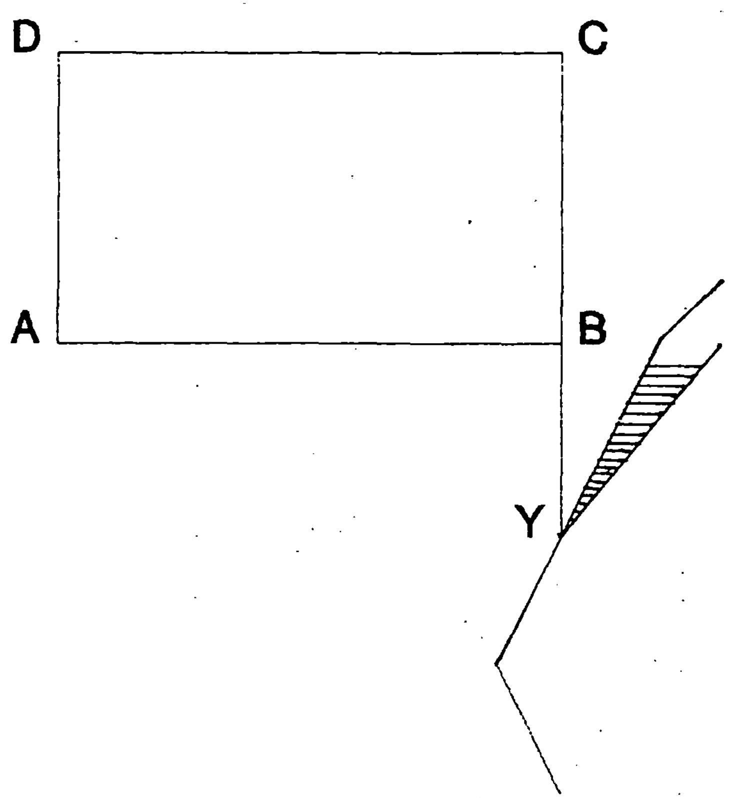


Figure 6a

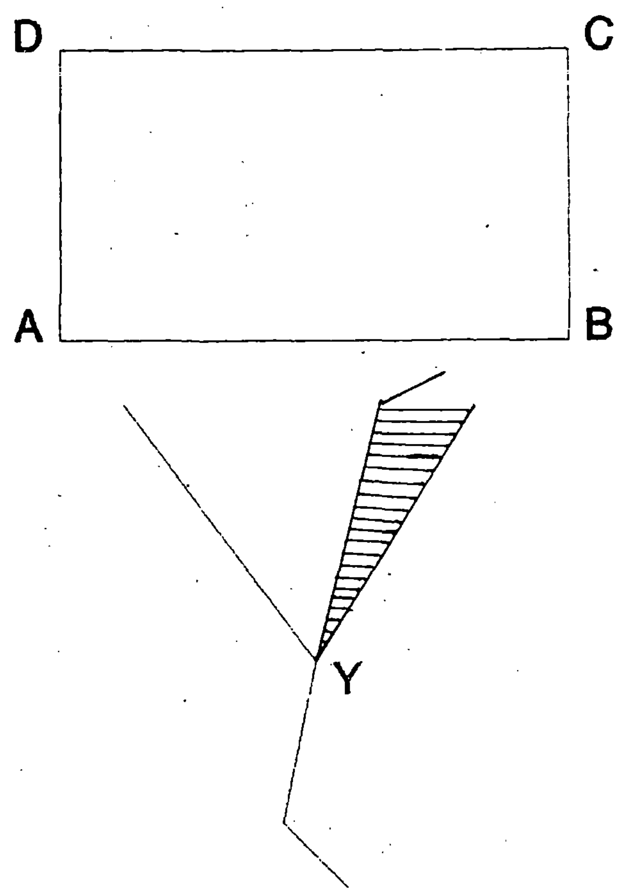


Figure 6b

We, therefore, have the following criterion.

If the direction of movement along the boundary of CP is towards the interior of EP(i-j) then drop the demand point with the greater weight; else drop the point corresponding to the smaller weight. It is worth mentioning in this connection that the interior or exterior of EP(i-j) is determined by comparing the gradients of the concerned edge of the equipolygon and the extreme direction of $\mathscr{E}$.

Moving along an extreme direction of $\mathscr{E}$ any one of the cases enumerated below may become true:

(i) a point P on ∂CP equidistant from at least two demand points is found;

(ii) an extreme point V of ∂CP is attained;

(iii) a point P situated at the intersection of ∂CP and a line through P_i drawn parallel to either coordinate axis is obtained.

In case (i) if the equipolygon with respect to a suitably chosen pair of demand points is directed towards the interior of CP then move along this side of the equipolygon; else get $\mathscr{E}$ afresh and decide on the proper course of action.

In case (ii) if V satisfies the condition of optimality then stop; else drop the currently active edge of ∂CP, select the next edge, obtain $\mathscr{E}$ de novo and repeat the above steps.

In case (iii) determine $\mathscr{E}$ at P with respect to the isoline of P_i, different from the earlier one, and perform the above actions.

CHAPTER 2

Weighted Unconstrained Problems

In this chapter we shall discuss solution procedures of two facility locaton problems in the unconstrained case with the minimax objective. The distance measure chosen for the purpose is rectilinear. Section 2.1.1 deals with the case in which the weights are symmetric and in secton 2.2.1 the weights have been supposed to be asymmetric.

2.1.1 Solution of a weighted one-centre problem under the L_1 norm

The problem considered in this section now follows

$$\min_{P\in R^2} \max_{i\in I} \; w_i d(P, P_i) \qquad (1)$$

where $P(x, y)$ is a variable point in the plane, $I = \{1,2,\ldots,n\}$, $P_i(a_i, b_i)$ are given points in the plane belonging to a finite set S, w_i are positive weights, and $d(P, P_i)$, the rectilinear distance from P to P_i, is given by

$$d(P, P_i) = | x - a_i | + | y - b_i |.$$

To solve (1) we first obtain the maximum weighted rectilinear distance of the set S from an arbitrarily chosen point P in the plane and then move P such that the weighted rectilinear distance of it from another point in S is the same as the distance between P and the weighted farthest point but greater than the distance between P and any other point of S -- a criterion to be called primal feasibility hereafter. This distance is then gradually diminished by moving the point, maintaining primal feasibility all the while, along a path determined by the pair of points of S until either at least a third point of S is encountered such that all these points of S are equidistant from the moving point P or no such third point at all exists. We have developed the algorithm by translating these ideas for which we would require certain basic concepts such as an equipolygon, a well-behaved point etc. defined in Section 2.1.2. In order to accomplish the task of gradually reducing this distance until the minimum value of the objective function is achieved we have adopted a solution procedure based on the methods of two dimensional analytic coordinate geometry. The strategy for solving the problem has been explained in detail in Section 2.1.3.

As regards the scope of application of the minimax criterion we might consider locating a new facility, say, a polyclinic or a fire station in a large metropolitan area where the objective is to minimize the maximum rectilinear travel distance of a potential user, weighted by some importance factor which is any positive number quantifying the nature of interaction between the facility and the category of user.

In the recent past Love, Morris and Wesolowsky [55] have made an in-depth study of models concerning layout and location of facilities. The equiweighted rectilinear metric problem has been investigated by Francis [34] and Elzinga and Hearn [31] using geometrical properties. By transforming into an equivalent linear programming problem Francis and

White [36] have developed a solution procedure via linear programming for the weighted version of the location problem. The solution procedure developed by them requires α_{ij} and β_{ij} (pp 384-389, [36]) to be calculated for all the demand points to arrive at a conclusion whereas our method, being essentially an iterative one, requires no more investigation if the point of intersection of two equipolygons lies on the rectangle formed by any two of the points defining the equipolygons as a result of which the remaining equipolygons are excluded from further consideration. Wendel, Hurter and Lowe [77] have given efficient algorithms for finding the set of efficient locations with the L_1 norm for the single facility planar location model. Drezner and Wesolowsky [24] have also extensively studied one centre L_p -distance minimax locational problems. Francis, McGinnis and White [37] and Hansen, Peeters and Thisse [44] have given a method-oriented selective survey of the literature and provided a comparison of the different computationally efficient algorithms. Morris [64] has presented an efficient algorithm for solving the constrained multifacility location problems. For the multifacility minimax location problem we refer the reader to the excellent works of, among others, Drezner [21], Aneja, Chandrasekaran and Nair [2], Ko, Lee and Chang [49], Dearing and Francis [18] and Wesolowsky [78].

2.1.2 Procedure

At the outset we shall try to obtain the locus of a point in R^2 from which the weighted rectilinear distances of two given location points are equal.

Let $P_1(a_1, b_1)$ and $P_2(a_2, b_2)$ be any two points in the plane with associated weights w_1 and w_2 and (x, y) any point from which the weighted rectilinear distances of P_1 and P_2 are the same. Without any loss of generality we may assume $a_1 \leq a_2$ as otherwise we may always interchange the labels of the points. Furthermore, let us suppose $w_1 < w_2$. Equating the weighted rectilinear distances of P_1 and P_2 from (x, y) we get

$$w_1(|x - a_1| + |y - b_1|) = w_2(|x - a_2| + |y - b_2|) \quad (2)$$

Let us rewrite this equation as $F(y) = G(x)$

$$\text{where} \quad F(y) = w_1|y - b_1| - w_2|y - b_2| \quad (3)$$

$$\text{and} \quad G(x) = w_2|x - a_2| - w_1|x - a_1| \quad (4)$$

For a given $x = x_0$ we want to find an y such that $F(y) = G(x_0)$. In other words, we want to determine the points of intersection of the curves $u = G(x_0)$ and $u = F(t)$.

Let us now investigate the nature of the functions defined by (3) and (4).

If $b_1 < b_2$ then the function $u = F(t)$ is continuous everywhere, strictly increasing in $(-\infty, b_1)$ and (b_1, b_2) and strictly decreasing in (b_2, ∞) and having its maximum positive value of $w_1(b_2 - b_1)$ at $t = b_2$ associated with the greater weight. Also $F(t) \longrightarrow -\infty$ as $t \longrightarrow \pm\infty$.

If, on the other hand, $b_1 > b_2$ then the above function strictly increases in $(-\infty, b_2)$ and decreases in (b_2, b_1) and (b_1, ∞) but still has the maximum positive value of $w_1(b_1 - b_2)$ at $t = b_2$.

Since the function represented by (4) has a form identical to that given by (3), it immediately follows that the function $u = G(t)$ is continuous, strictly decreasing in $(-\infty, a_1)$ and (a_1, a_2) and strictly increasing in (a_2, ∞), having the minimum negative value of $- w_1(a_2 - a_1)$ at $t = a_2$. Also $G(t) \longrightarrow +\infty$ as $t \longrightarrow \pm\infty$.

Note: If $w_1 > w_2$ then the curves represented by (3) and (4) will simply exchange their respective forms.

As $\max F(t) > 0$ and $G(a_2) < 0$ it follows from the nature of the curve $u = F(t)$ that it will intersect $u = G(a_2)$ at two points. Hence we can conclude that the set of (x, y) satisfying (2) is not void.

In a similar manner it can be deduced that the curves $u = G(t)$ and $u = F(b_2)$ will intersect at exactly two points, say α and β $(> \alpha)$, given by

$$\beta = \left[w_2 a_2 - w_1 a_1 + w_1(b_2 - b_1)\right]/(w_2 - w_1) \quad \text{and}$$

$$\alpha = \begin{cases} \left[w_2 a_2 - w_1 a_1 - w_1(b_2 - b_1)\right]/(w_2 - w_1) \text{ if } F(b_2) > G(a_1) \\ \left[w_2 a_2 + w_1 a_1 - w_1(b_2 - b_1)\right]/(w_2 + w_1) \text{ if } F(b_2) \le G(a_1) \end{cases}$$

provided $a_1 \neq a_2$.

If $a_1 = a_2$ then $\beta = a_1 + w_1(b_2 - b_1)/(w_2 - w_1)$

and $\alpha = a_1 - w_1(b_2 - b_1)/(w_2 - w_1)$

If $x < \alpha$ or $x > \beta$ then since $G(x) > F(b_2)$, the curves (3) and

(4) will not intersect at all.

If, on the other hand, we consider any $x_o \in (\alpha, \beta)$ then there exists an y_o such that (x_o, y_o) satisfies (2) which implies that the curve $u = F(t)$ intersects $u = G(x_o)$ at two distinct points. At $x = \alpha$ or $x = \beta$ there exists only one value of y, viz. $y = b_2$, such that $F(b_2) = G(\alpha) = G(\beta)$.

From the above discussions we, therefore, have the following lemma.

Lemma 1. The locus of (x, y) as given by (2) is a closed polygon having within it the point associated with the greater weight.

In what follows we shall call this locus the equipolygon of P_1 and P_2 and denote it by EP(1-2). This equipolygon cuts the line joining P_1 and P_2 both internally and externally in the inverse ratio of the weights at the extremities, their coordinates being given by

$$\left((w_1 a_1 + kw_2 a_2)/(w_1 + kw_2),\ (w_1 b_1 + kw_2 b_2)/(w_1 + kw_2)\right)$$

where k = 1 in the former case and k = -1 in the latter.

We state without proof the following corollary to be required in the sequel.

Corollary 1: The weighted rectilinear distance from any point P in R^2 to the location point corresponding to the greater weight is greater than that to the other point when the point P is outside the equipolygon, a closed contour in this case, and vice versa.

To have a closer look into the nature of the equipolygon we divide the entire plane into 9 regions I through IX depending on the position of (x, y) as follows; see figure 1.

I $x \le a_1$, $y \le b_i$;

II $x \le a_1$, $b_i \le y \le b_j$;

III $x \le a_1$ $y \ge b_j$;

IV $a_1 \le x \le a_2$, $y \le b_i$;

V $a_1 \le x \le a_2$, $b_i \le y \le b_j$;

VI $a_1 \le x \le a_2$, $y \ge b_j$;

VII $x \ge a_2$, $y \le b_i$;

VIII $x \ge a_2$, $b_i \le y \le b_j$;

IX $x \ge a_2$, $y \ge b_j$;

where $b_i \le b_j$; $i, j = 1, 2$ and $i \ne j$.

In each of the above regions if the equipolygon is defined then it may be represented as shown below.

I $x + y = a + b$ where $a = (w_2 a_2 - w_1 a_1)/(w_2 - w_1)$

and $b = (w_2 b_2 - w_1 b_1)/(w_2 - w_1)$

II $(w_2 - w_1)x + k(w_1 + w_2)y = (w_2 a_2 - w_1 a_1) + k(w_1 b_1 + w_2 b_2)$

III $x - y = a - b$

IV $(w_1 + w_2)x + (w_2 - w_1)y = (w_1 a_1 + w_2 a_2) + (w_2 b_2 - w_1 b_1)$

V $x + ky = c + kd$, where $c = (w_2 a_2 + w_1 a_1)/(w_2 + w_1)$

and $d = (w_2 b_2 + w_1 b_1)/(w_2 + w_1)$

VI $(w_1 + w_2)x - (w_2 - w_1)y = (w_1 a_1 + w_2 a_2) - (w_2 b_2 - w_1 b_1)$

VII $x - y = a - b$

VIII $(w_2 - w_1)x - k(w_1 + w_2)y = (w_2 a_2 - w_1 a_1) - k(w_1 b_1 + w_2 b_2)$

IX $x + y = a + b$

$$\text{where } k = \begin{cases} 1, & \text{if } b_2 > b_1 \\ -1, & \text{if } b_2 < b_1 \end{cases}$$

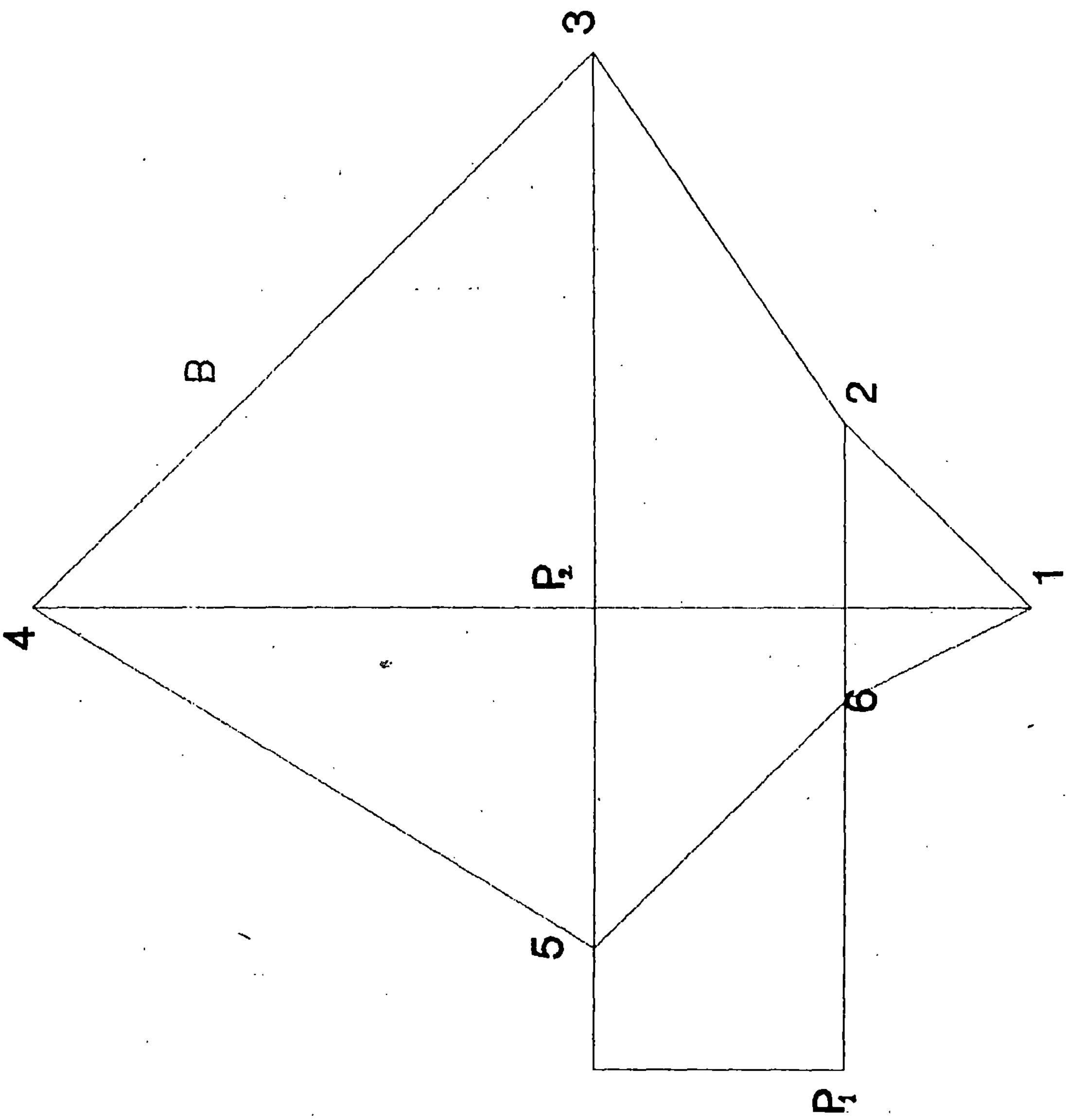


Fig.1

Definition 1. $R\Sigma$ represents the smallest bounding rectangle constructed through the four points of a finite set Σ, having minimum and maximum ordinates and minimum and maximum abscissas respectively, by drawing lines parallel to the x and y axes. When $\Sigma = \left\{A_1, A_2, \ldots, A_k\right\}$ then the smallest bounding rectangle is denoted by $R(A_1, A_2, \ldots, A_k)$.

Let us now make a detailed study of the structure of the equipolygon vis-a-vis the regions of definitions.

Let l amd s denote the lengths of adjacent sides of $R(P_1, P_2)$ where $l = |a_2 - a_1|$ and $s = |b_2 - b_1|$. Let $l > s$. If $w_2 > w_1$ and $(w_2/w_1) < (l/s)$ the locus of (x, y) consists of six straight line segments lying in regions IV through IX joined end to end forming a closed polygon as shown in figure 1, whereas if $w_2 > w_1$ and $(w_2/w_1) \geq (l/s)$ it has four segments in regions V, VI, VIII and IX or IV, V, VII and VIII according as $b_1 <$ or $> b_2$; see figure 2. If, on the other hand, $w_1 > w_2$ then the equipolygon comprises line segments belonging to regions I through VI in the former and I, II, IV and V or II, III, V and VI in the latter cases. If $l \leq s$ the regions within which segments forming the equipolygon will lie may be obtained in an analogous manner.

If $b_1 = b_2$ then the locus is a four-sided equipolygon, the sides being located in regions IV, VI, VII and IX or I, III, IV and VI according as $w_1 <$ or $> w_2$. In a similar manner when $a_1 = a_2$ the equipolygon has sides belonging to regions II, III, VIII and IX or I, II, VII and VIII in the respective cases.

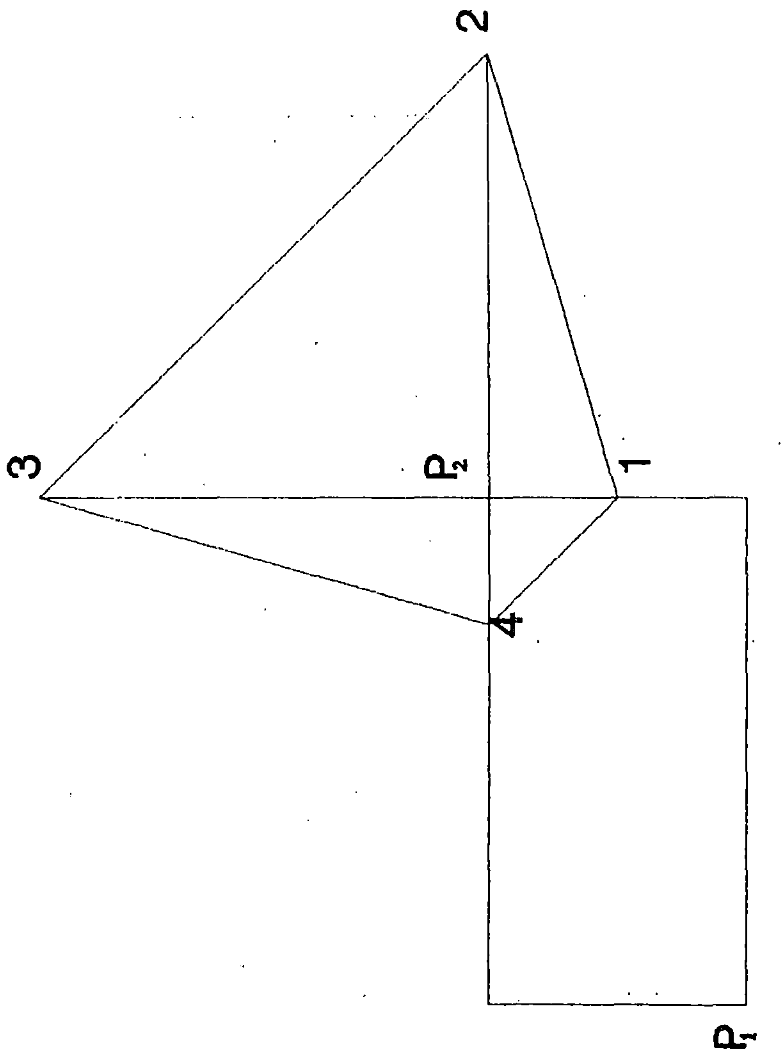


Fig.2

If $w_1 = w_2$ the locus represented by (2) is no longer a closed polygon and consists of a line segment through the centre of the rectangle lying within region V flanked by semi-infinite lines parallel to the y-axis contained in regions IV and VI. This is illustrated in figure 3 in case $l > s$. If $l < s$ then the locus is made up of a line segment in zone V bounded by semi-infinite lines parallel to the x-axis contained in regions II and VIII. When $l = s$ the locus consists of either diagonal together with any two half-rays with vertices at the extremities of this diagonal in regions III and VII or I and IX according as $b_1 <$ or $> b_2$.

In our subsequent discussion we will make use of the following definitions and notations.

Definition 2. With respect to any point $(h, k) \in R^2$ we denote by $L(h, k; a_i, b_i)$ the path consisting of two line segments joining (h, k) to (a_i, k) and (a_i, k) to (a_i, b_i).

Definition 3. Γ_{ij} will represent the portion of EP(i-j) intercepted by the boundaries of $R(P_i, P_j)$.

Definition 4. M_{ij} denotes a portion of EP(i-j) from any point upto the nearest $\partial R(P_i, P_j)$ along the direction of descent. This is illustrated in figure 1 where M_{12} consists of the line segments B4 and 45 of EP(1-2).

Definition 5. Q will be called a well-behaved point with respect to suffixes $i, j \in I$, $i \neq j$, if

$w_i d(Q, P_i) = w_j d(Q, P_j) \geq w_k d(Q, P_k)$, for all $k \in I - \{i, j\}$.

Definition 6. A direction of descent is one by moving along

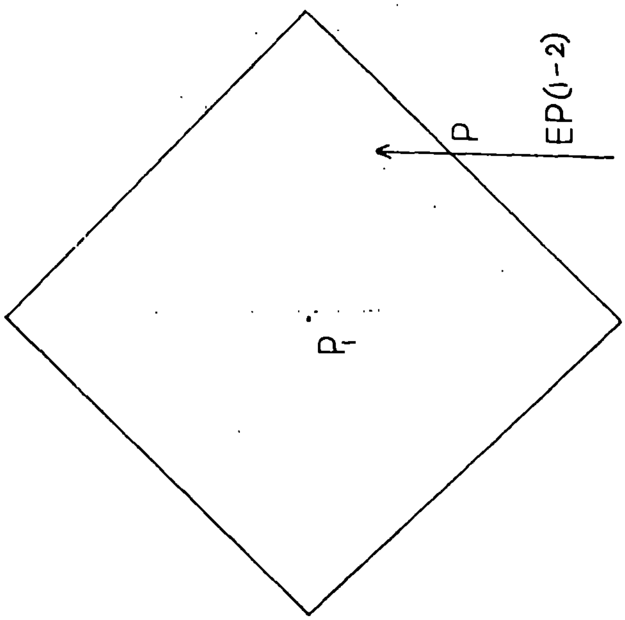


Fig 4

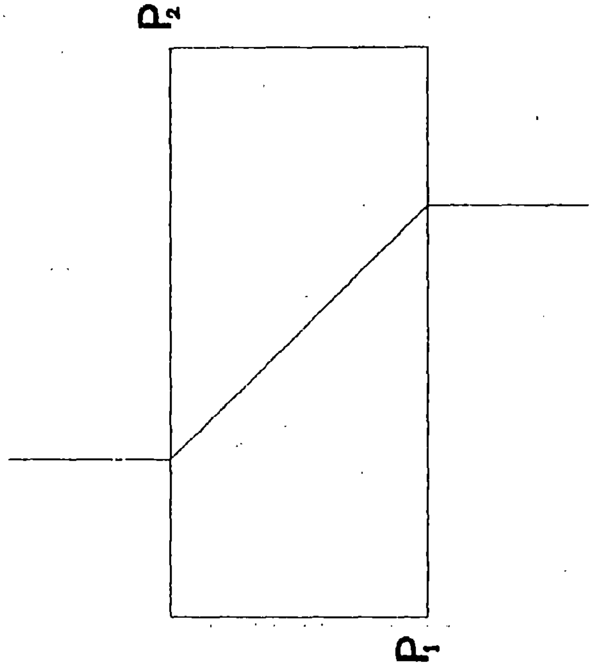


Fig.3

which the value of the objective function does not increase.

To bring the ideas contained in the last definition into a sharp focus let A, B be the consecutive vertices of the edge AB of an equipolygon EP(1-2) and P be any point on AB. Then the direction PA will be called the direction of descent at P with respect to P_1, P_2 if $d(U, P_1) \leq d(P, P_1)$, U being any point on the line segment AP.

2.1.3. Solution of the problem

Our algorithm is based on the movement in the direction of descent along an edge of an equipolygon. It is, therefore, proper now to introduce a rule for its determination.

Determination of the Direction of Descent

Let P be any point on an edge of an equipolygon EP(1-2). Let us construct a diamond [36] through P centred at either of the points P_1, P_2. Then the portion of the edge of EP(1-2) directed towards the diamond gives the direction of descent as shown in Figure 4.

For unequal weights the orientations of the directions of descent of the equipolygon vis - a - vis the different regions are given below. See figure 1 for an illustration.

Region	Orientation
IV	from 1 to 6
VI	from 4 to 5
VII	from 2 to 1
VIII	from 3 to 2
IX	from 3 to 4 or 4 to 3

In case the weights are equal the direction of descent will be towards the longer side of the rectangle $R(P_1, P_2)$.

Let us now discuss the strategy for solving problem (1). The optimal objective value with respect to any two points A and B occurs in R(A,B). Moreover, RS contains all $R(P_i, P_j)$, $P_i, P_j \in S$. Consequently, the minimax solution of problem (1) lies on RS.

At a particular iteration if there exists a well-behaved point $\in R(P_1, P_2)$ on an edge of an equipolygon EP(1-2), the well-behaved point associated with the immediately succeeding iteration may be obtained as follows.

We consider the intersection of the direction of descent of this edge of EP(1-2) with the equipolygons formed by P_1 or P_2 and each of the other (n-2) points and find the one nearest to a well-behaved point. If none exists then the extreme point of the edge of EP(1-2) serves as the next well-behaved point. Mathematically, if the end points of an edge of EP(1-2) through P be denoted by A, B such that $\vec{PA}$ is the direction of descent with respect to P_1, P_2 then $w_1 d(U_i, P_1) = w_i d(U_i, P_i)$, $i \in I - \{1, 2\}$, provided U_i is on PA and such an U_i exists. Let $PU = \min_i \{\|PU_i\|\}$. We choose U as the well-behaved point for the next iteration in case such an U is available; otherwise, we choose A.

We next prove the following lemma to be subsequently required for developing our algorithm.

Lemma 2. Let us consider the two given location points P_i, i = 1, 2 and find the greater of the weighted rectilinear distances of these from any point (g, h). If the path corresponding to the maximum distance be represented by $L(g,h; a_i, b_i)$ then there exists a point $P \in L(g, h; a_i, b_i)$ such that the weighted rectilinear distances of P_i, $i = 1, 2$ from P are equal.

Proof: If the weighted distances of P_1, P_2 from (g, h) be equal nothing remains to be proved. Consider, therefore, the situation in which the two are not the same and the greater distance corresponds to P_1. We define a function $f(x, y)$ as follows :

$$f(x, y) = f_1(x, y) - f_2(x, y) \text{ where}$$

$$f_i(x, y) = w_i(|x - a_i| + |y - b_i|).$$

Clearly, $f(x, y)$ is a continuous function, $f(g, h) > 0$ and $f(a_1, b_1) = - f_2(a_1, b_1) < 0$. Hence the proof of the lemma is complete.

We have the following corollary, the proof of which, being obvious, is left out.

Corollary 2. Every equipolygon $EP(k-i)$, where P_k is the weighted farthest point from (g, h), $i \in I - \{k\}$, intersects $L(g,h; a_k, b_k)$ at least once.

Suppose that we are moving along the direction of descent of the equipolygon $EP(i-j)$, $w_i \geq w_j$ along A_1A and on reaching A, let $w_i d(A, P_i) = w_j d(A, P_j) = w_k d(A, P_k)$. Then since $w_k d(A_1, P_k) < w_i d(A_1, P_i) = w_j d(A_1, P_j)$ we conclude that the equipolygons $EP(i-k)$ and $EP(j-k)$ passing through A will have some portions at least of them on either side of A_1A in the neighbourhood of A. In particular, it is obvious that a portion of the boundary of EP(i-k) will be within EP(i-j) irrespective of the magnitudes of w_i and w_k, and that of EP(j-k) will be outside EP(i-j) as shown in figure 5.

It is to be noted further that if the point A be outside $R(P_i, P_j, P_k)$ then A cannot be a minimax location for, any movement from A towards the nearest $\partial R(P_i, P_j, P_k)$ along the direction perpendicular to the boundary will cause the objective function value to diminish. The direction of movement is given by the following criterion.

Criterion C

The rule of selecting one from three or more equipolygons meeting at a non-optimal point : Let P(g, h) be the point from which the weighted rectilinear distances of $(a_i, b_i) \in S_1 \subseteq S$, the cardinality of S_1 being r ($\geq$ 3), are equal but those of points $\in S \setminus S_1$ are less. We take any three points of S_1. Since P is not an optimal point, one of the following possibilities must be true.

1. All three points lie on the same side of $x = g$ but any two of them lie on one side and the third on the other of $y = h$.
2. All three points lie on the same side of $y = h$ while any two of them lie on one side and the third on the other of $x = g$.

As an example let us assume that each of b_i, b_j, $b_k > h$ and $a_i < g$, a_j, $a_k > g$. We retain (a_i, b_i) and consider the directions of descent of EP(i-j) and EP(i-k) at P. We next construct a diamond with respect to $P_i(a_i, b_i)$ passing through P. Of the two equipolygons EP(i-j) and EP(i-k) the one

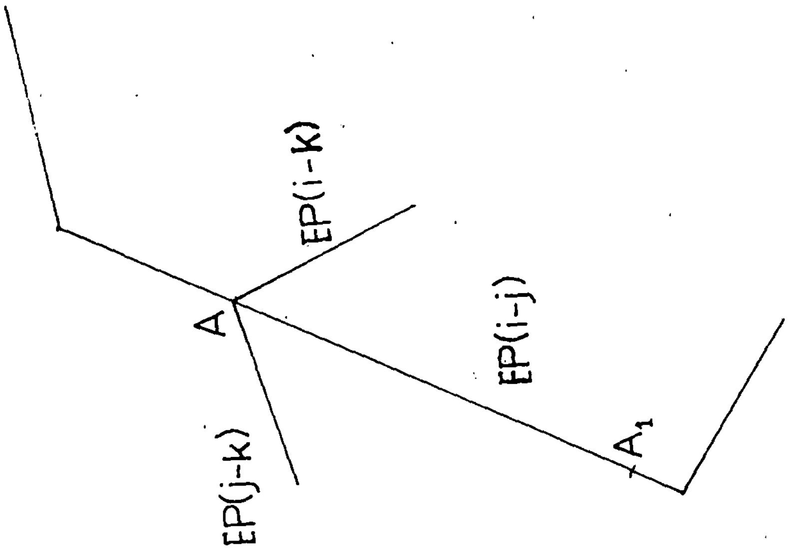


Fig. 5

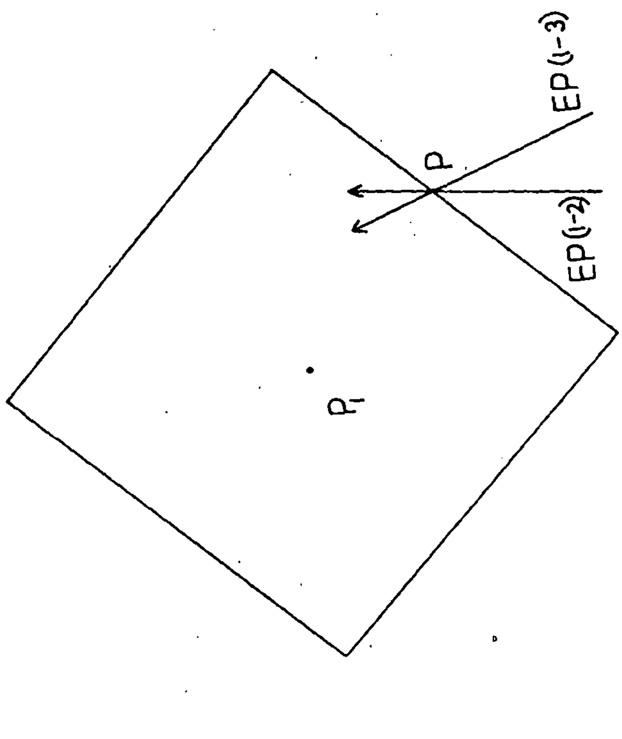


Fig 6

having a smaller angle of inclination with the side of the diamond passing through P will be selected for the subsequent iteration. Figure 6 illustrates the case of EP(1-2) being chosen.

In the algorithm that follows we assume that the movement always takes place along the direction of descent of an equipolygon in such a way that primal feasibility condition is never violated.

2.1.4. Algorithm

Step 0. (Initialisation Step)

Take any extreme point $(\bar{x}, \bar{y}) \in \partial RS$ and calculate the maximum weighted rectilinear distance of it from the set S. Let this maximum occur for i = 1. Now determine a point A on $L(\bar{x}, \bar{y}; a_1, b_1)$ such that

$$\max_{A \in L(\bar{x}, \bar{y} : a_1, b_1)} \left\{ w_1 d(A, P_1) = w_i d(A, P_i) \mid i \in I - \{1\} \right\}$$

holds. Let $P_2 \in S$ be the point satisfying the above; go to step 1.

Step 1. If $A \in \partial R(P_1, P_2)$ or if we can move upto a point $B \in \partial R(P_1, P_2)$ in such a way that any point $P \in M_{12}$ is a well-behaved point with respect to indices 1, 2 in which case $A \longleftarrow B$, then go to step 3(a). Else go to step 2.

Step 2. For a point $P \in M_{12}$ such that $w_j d(P, P_j) = w_1 d(P, P_1)$, $j \in I - \{1, 2\}$ and the distance of P from A measured along M_{12} is a minimum, if $P \in R(P_1, P_2, P_j)$ then go to step 3(b).

Else apply **criterion C** to determine the direction of next movement, A ←— P, rename the corresponding points defining the equipolygon as P_1 and P_2 and go to step 1.

Step 3(a). If any other equipolygon intersects Γ_{12} at P then any point ∈ AP is a possible minimax location. Otherwise, any point ∈ Γ_{12} will be a required location.

Step 3(b). From P we follow the path along Γ_{kl} satisfying **criterion C**, kl being any combination of 1, 2 and j, until we obtain the point of intersection Q of Γ_{kl} and another equipolygon, in which case any point ∈ PQ is a possible minimax location. Otherwise, any point ∈ the stretch of Γ_{kl} from P to the point of intersection of Γ_{kl} and the boundary of $R(P_k,P_l)$ situated along the direction of descent is a required location.

2.1.5. Analysis of time complexity

Let us take two location points P_i and P_j the weighted rectilinear distances of which from some (h, k) are equal. Let the edge of EP(i-j) containing (h, k) meet $R(P_i, P_j)$ at (p, q). We next determine if the point of intersection of the segment joining (h, k) and (p, q), and some other EP(i-l) closest to (h, k), belongs to this segment.

Our algorithm chooses one of the corner points of RS as a starting solution. Taking the right hand bottom corner as the initial choice, (h, k) may initially belong to one of the zones IV, VII or VIII with respect to P_i and P_j. If it is in zone VII, since two equipolygons in this zone, being parallel, cannot intersect each other, it cannot remain there in the following iterations, thus signalling the failure to obtain an (h_1, k_1). If, on the other hand, the non-optimal point (h, k) belongs to zone IV initially, it is then obvious that (h_1, k_1) also belongs to zone IV. By the very definition of zone IV, if (h_1, k_1) be non-optimal then the ordinates of the three points will be greater than k_1 whereas two of them will be on the one side and the third on the other of $x = h_1$. **Criterion C** determines which one of the two points on the same side of $x = h_1$ is to be retained in the next iteration. Thus each of the retained points generating the next iteration has ordinate greater than k_1 while continuing to remain on either side of $x = h_1$ indicating that (h_1, k_1) still belongs to zone IV with respect to the updated points. If more than three points have the same weighted distance from (h_1, k_1) then by repeated application of **Criterion C** one can conclude that if the iteration at some stage be restricted to zone IV it will continue to remain so until optimality is achieved. On the other hand, if no (h_1, k_1) belonging to the line segment terminated by (h, k) and (p, q) exists then any one of the zones I or VII may have to be traversed. By the same token we may draw a parallel between the arguments given for zone IV and those for zone VIII.

The procedure developed by us is based on two equipolygons intersecting at most once in a given zone. If

i_1, i_2 be the indices corresponding to a well-behaved point in a particular non-optimal iteration, then these indices are not required to be considered any further until optimal solution is obtained. Thus in the worst case the algorithm has a $O(n^2)$ time complexity. Had we tried to solve the LP formulation of the present problem via simplex method it wouldn't have been possible for us to conclude beforehand the order of polynomial time complexity.

2.1.6. Numerical Example

Let us find the solution to the problem considered by Francis and White [36] by making all $g_i = 0$. The location points are (3, 3), (3, 6), (6, 3) and (7, 8) with associated weights 2. 3, 4 and 2 respectively. We construct the rectangle RS and take a point $(3, 8) \in \partial RS$. The weighted farthest point is found to be $P_1 = (6, 3)$. We next find the point A to be $(6, 6) \in L(3, 8; 6, 3)$. We designate the point (3, 3) as P_2 following step 0 of the algorithm. Clearly $A \notin R(P_1, P_2)$. Moving along EP(1-2) we reach the point $P = \left(\frac{81}{14}, \frac{75}{14} \right)$ from which the weighted rectilinear distances of P_1, P_2 and $P_3 = (3, 6)$ are equal. Since $P \in R(P_1, P_2, P_3)$ applying **criterion C** we can move along the direction of descent of EP(1-3) upto the point $T = \left(\frac{36}{7}, \frac{33}{7} \right)$ from which the weighted rectilinear distance of each of P_1, P_2 and $P_4 = (7, 8)$ is the same. Hence by step 3(b), any point $\in$ PT is a required minimax location.

2.1.7. Computational Experience

No. of points	Frequency of convergence in		
	1 iteration	2 iterations	3 iterations
500	18	5	2
550	19	5	1
600	17	5	3
650	17	7	1
700	18	6	1
750	19	5	1
800	15	9	1
850	18	6	1
900	18	7	0
950	19	6	0
1000	21	2	2

Table 1

This section deals with the computational test of the algorithm. With this end in view we developed the Pascal code of the algorithm. Three sets of random numbers corresponding to x_i, y_i and w_i were generated 25 times for a given n ($500 \leq n \leq 1000$) over a rectangle RS chosen in advance having unequal adjacent sides by employing standard Turbo Pascal Procedure Randomize and Function Random. n was next allowed to vary and the same procedure repeated . Random data generation technique was resorted to on account of non-availability of actual data required for large size problems. It is interesting to note that in no case the algorithm required more than three iterations to converge. The results of computation are summarised in Table 1 above.

2.1.8 Sensitivity Analysis

At early stages of problem formulation some factors may be overlooked and in many practical situations data may not be known in advance exactly. Sensitivity analysis takes care of these factors and updates the current optimal solution without performing the expensive task of resolving the problem from scratch. Let us now see how these ideas can be implemented to obtain the current optimal solution from the previous solution. Let us consider the following instances.

(i) Introducing a new demand point

(ii) Removing an existing demand point

(iii) Changing the weight associated with a given demand point.

Regarding case (i) if the weighted distance of the recently added point be less than or equal to the optimum objective value calculated at both the extremities of the stretch (prior to insertion of the demand point) constituting the set of optimum solutioms then the solution set remains the same as before; else if this distance be less than the optimum value of the objective function obtained at one extremity only then we have to recalculate the stretch; else we choose either extremity of the stretch as $(\bar{x}, \bar{y})$ and repeat the algorithm described in sec. 2.1.4 after making allowance for this additional point.

As regards case (ii) if the deleted point be not an active demand point then the set of optimal solutions remains unchanged; else we choose any end point of the stretch as the starting point $(\bar{x}, \bar{y})$ and proceed in accordance with the directions indicated in sec. 2.1.4.

In case (iii) if altering the weight associated with an existing demand point destroys primal feasibility at either end point of the stretch then we follow a procedure similar to case (ii) to restore primal feasibility.

As an illustration let us introduce a new demand point at (5, 2) with associated weight 5 in addition to the four already existing ones considered in sec. 2.1.6 to throw light on the observations made above. As the weighted rectilinear distance of the newly added point from both the end points of the stretch is greater than the previous optimal solution violating primal feasibility, we follow the procedure mentioned in (i) to get the new stretch extending from (5.05, 4.24) to (5.00, 4.29). The stretch belongs to the smallest rectangle formed by the fourth and the newly introduced points whereas in the original problem the stretch is the line segment from (5.14, 4.71) to (5.79, 5.36) lying within the smallest rectangle constructed with the second and third demand points.

2.2.1 Solution of an asymmetric rectilinear distance minimax location problem

In this section we consider a minimax location problem using a rectilinear measure of distance lacking symmetry so that with each demand point is associated four different weights corresponding respectively to the main four directions. This lack of symmetry is typical of rush hour traffic where the speeds towards and away from the commercial centre of a metropolitan city are different. There are other practical situations also where distance between two points is not a symmetric function - for an air craft, for example, flying in the presence of steady wind flowing in one direction only the speeds in the direction of current and opposite to it are different. Again, for motion on an inclined plane the upslope speed is different from the downslope speed.

Presently we would discuss briefly how the above mentioned model may be gainfully applied to the tea industry. The northern part of West Bengal abounds in tea gardens. The Terai region of Darjeeling district and the Dooars region of Jalpaiguri district account respectively for approximately 50 and 250 gardens. Tea is one of the chief agricultural produce earning foreign exchange. It is conventionally grown at a place where there is no waterlogging despite abundant rainfall. For this reason it is natural that the sub-Himalayan region of West Bengal should be selected for tea plantations. But, as a matter of fact, most gardens employ conventional methods for growing and plucking of tea. For an increase in the yield as also for the protection and sustenance of the plants, it requires, among other things, pruning, knife cleaning and depilation, chilling, light hoeing, trimming, spraying of pesticides and weed killers, plucking out of creepers, infilling, construction and maintenance of drains etc. As an example, by simply improving on the existing drainage system the Trihanna Tea Estate increased the yield by about 12.5% (the international market value of this extra yield being estimated at $ 0.2 million). To check soil erosion and prevent water from accumulating, each garden develops its own drainage system depending on its topography. The gardens are situated on an inclined plane extending northward from the base. For an incline of less than 1 in 50 the usual practice is to construct north south drains interspersed at regular intervals with east west ones while for inclines exceeding it, contour drains are preferred. The places from which the above operations of plant treatment etc. are being carried out, to be called the facility point hereafter, are not normally located in accordance with the prescriptions suggested by facility layout and location models. The Panighata Tea Estate in the lower Terai offers a case in point (total area 6.17 hectares, area under cultivation 4.25 hectares of an irregular shape and an elevated northern side, maximum width 2.5 kms). The facility point is located at one end of the garden while the other end is 4 kms apart. Our research was motivated by the problem of locating the facility point so that the maximum distance from it to a plant - a demand point - is minimised. Moreover, for a considerable time of the year there is a steady westwind blowing over the gardens. Thus on both counts the minimax criterion involving asymmetric L_1 metric for a single facility is the most appropriate one for the declared objective.

Dykstra et al. [30] consider the cost of log harvesting in which the logs are displaced from 'prebunching sites' by means of helicopters. Hodgson et al. [46] and Chen [13] have proposed solutions to the p-centroid location problem of log harvesting on an inclined plane. Drezner and Wesolowsky [25] have provided an efficient algorithm for an asymmetric rectilinear distance minimax location problem while Chakrabarty and Chaudhuri [10] have given a geometrical solution procedure for the constrained two-dimensional minimax problem using weighted rectilinear norm. Tamir [73] has given a complexity bound improvement for the 1-centre rectilinear asymmetric distance location problem in the plane.

For a survey of selected locational literature we would like to refer to the excellent works of, among others, Francis et al. [37], Hansen et al. [44] and Love et al. [55].

A plethora of publications dealing with the minimax criterion for the unweighted as well as the weighted rectilinear metric is available (Francis [34], Elzinga and Hearn [31], Wesolowsky [78], Morris [64], Francis et al. [38], Hansen et al. [43], Drezner [21] and Batta et al. [4]. But scant little attention has been given to the asymmetric L_1- distance location problems. Ours is an effort to fill this gap. We have developed an iterative technique based on a well-defined stopping criterion.

The solution procedure which we have developed, at first finds a point P from which the weighted rectilinear distance of at least two points of the set S of given location points are equal while the weighted distances of P from the remaining points of S are greater. The point P is now moved until optimality is reached by maintaining the above mentioned property of distance to be called primal feasibility hereafter. All this is accomplished with the help of plane analytic geometry.

Throughout our discussion we shall use a single letter, or a juxtaposed pair of letters, subscripted or otherwise, in bold face Roman type to denote vectors.

2.2.2. Problem formulation

Let us consider the following problem:

$$\underset{(x,\ y) \in R^2}{\text{Min}} \quad \underset{i \in I}{\text{Max}} \quad \mathbf{W}_i^{*} \mathbf{Z}_i \qquad (1)$$

where (x, y), a variable point, denotes the proposed location of the facility point, I = {1, 2,...,n}, $P_i(x_i,\ y_i)$

are the existing location points belonging to the set $S = \{ P_i(x_i, y_i): i \in I \}$,

$Z_i = \begin{bmatrix} |x-x_i| \\ |y-y_i| \end{bmatrix}$ and W_i^* represents the transpose of $W_i = \begin{bmatrix} u_i \\ v_i \end{bmatrix}$,

u_i and v_i being given by

$$u_i = \begin{cases} u_i^- \text{ if } x < x_i \\ u_i^+ \text{ if } x > x_i \end{cases}; \quad v_i = \begin{cases} v_i^- \text{ if } y < y_i \\ v_i^+ \text{ if } y > y_i \end{cases}.$$

We have not attempted to define u_i, v_i when $x_i = x_i$ or $y = y_i$ since they have no contribution towards the objective owing to the fact that $|x - x_i| = 0$ for $x = x_i$ and $|y - y_i| = 0$ for $y = y_i$.

We shall use the following notations throughout our discussion :

$$d(P, P_i) = |x - x_i| + |y - y_i|$$

$$\rho(P, P_i) = W_i^* Z_i = u_i|x - x_i| + v_i|y - y_i|$$

To start with let us assume that $u_i^+ > u_i^-$, $v_i^+ > v_i^-$, although such an assumption is not at all necessary to develop our algorithm.

Using the approach on page 227 of Francis et al. [38] we can translate the above problem into the following linear program (LP):

Minimise z subject to

$$u_i^+ (x - x_i) + v_i^+ (y - y_i) \le z$$

$$u_i^- (x_i - x) + v_i^+ (y - y_i) \le z$$

$$u_i^+ (x - x_i) + v_i^- (y_i - y) \le z$$

$$u_i^- (x_i - x) + v_i^- (y_i - y) \le z$$

for all $i \in I$.

2.2.3. Solution of the problem

To solve problem (1) we have developed a simple geometrical approach. For a problem of moderate size with $n \le 20$, say, (1) may be solved by using the ruler and the compass. But when the problem size is large our method can be easily implemented on a PC.

We shall first obtain the locus of (x, y) satisfying $W_1^* Z_1 = W_2^* Z_2$ which implies

$$u_1 |x - x_1| + v_1 |y - y_1| = u_2 |x - x_2| + v_2 |y - y_2| \qquad (2)$$

or what is the same thing as $F(y) = G(x)$ (3)

where $F(y) = v_1 |y - y_1| - v_2 |y - y_2|$ (4)

and $G(x) = u_2 |x - x_2| - u_1 |x - x_1|$ (5)

Without any loss of generality we may always assume $x_1 < x_2$. For, if otherwise, we might call the point with the smaller x-coordinate (x_1, y_1). For convenience, let us for the time being consider $y_1 < y_2$. It can be easily shown by direct substitution that the point

$$\left(\left(u_2^- x_2 + u_1^+ x_1\right) \Big/ \left(u_2^- + u_1^+\right), \left(v_2^- y_2 + v_1^+ y_1\right) \Big/ \left(v_2^- + v_1^+\right)\right)$$

$\in \left\{(x, y): x_1 \le x \le x_2, y_1 \le y \le y_2\right\}$ lies on the locus represented by equation (2) with $u_1 = u_1^+$, $v_1 = v_1^+$, $u_2 = u_2^-$ and $v_2 = v_2^-$, implying that the equation has a solution. Clearly, W_1 may be greater than, equal to, or less than W_2.

Case $W_1 > W_2$. The function $t = G(x)$ is continuous everywhere, strictly increasing in $(-\infty, x_1)$ and strictly decreasing in (x_1, x_2) and (x_2, ∞), having attained the maximum positive value of $u_2^-(x_2 - x_1)$ at $x = x_1$. Moreover, $G(x) \longrightarrow -\infty$ as $x \longrightarrow \pm\infty$. See figure 1.

Since the functions $F(y)$ and $-G(x)$ have identical forms, it follows immediately that $t = F(y)$ is continuous, decreasing in $(-\infty, y_1)$ and increasing in (y_1, y_2) and (y_2, ∞) with the minimum negative value of $v_2^-(y_1 - y_2)$ at $y=y_1$. Furthermore, $F(y) \longrightarrow \infty$ as $y \longrightarrow \pm\infty$. Refer to figure 2.

If $F(y_1) > G(x_2)$ then there exists no y satisfying $F(y)=G(x)$ for any $x \notin [\alpha_{11}, \alpha_{12}]$ where

$$\alpha_{11} = \left[v_2^-(y_1 - y_2) + u_1^- x_1 - u_2^- x_2\right] \Big/ \left[u_1^- - u_2^-\right]$$

and $\alpha_{12} = \left[v_2^-(y_2 - y_1) + u_1^+ x_1 + u_2^- x_2\right] \Big/ \left[u_1^+ + u_2^-\right]$

whereas if $F(y_1) < G(x_2)$ then there is no y that corresponds to values of $x \notin [\gamma_{11}, \gamma_{12}]$ such that $F(y) = G(x)$ where

$$\gamma_{11} = \alpha_{11}$$

and $\gamma_{12} = \left[v_2^-(y_2 - y_1) + u_1^+ x_1 - u_2^+ x_2\right] \Big/ \left[u_1^+ - u_2^+\right]$

in view of the increasing nature of $G(x)$ in $(-\infty, x_1)$ and its decreasing nature in (x_1, ∞). We can therefore conclude that the locus represented by (2) lies wholly within two straight lines parallel to the y-axis.

On the other hand, if $G(x_1) < F(y_2)$ we cannot get an x such that $G(x) = F(y)$ for values of $y \notin [\beta_{11}, \beta_{12}]$, β_{11} and

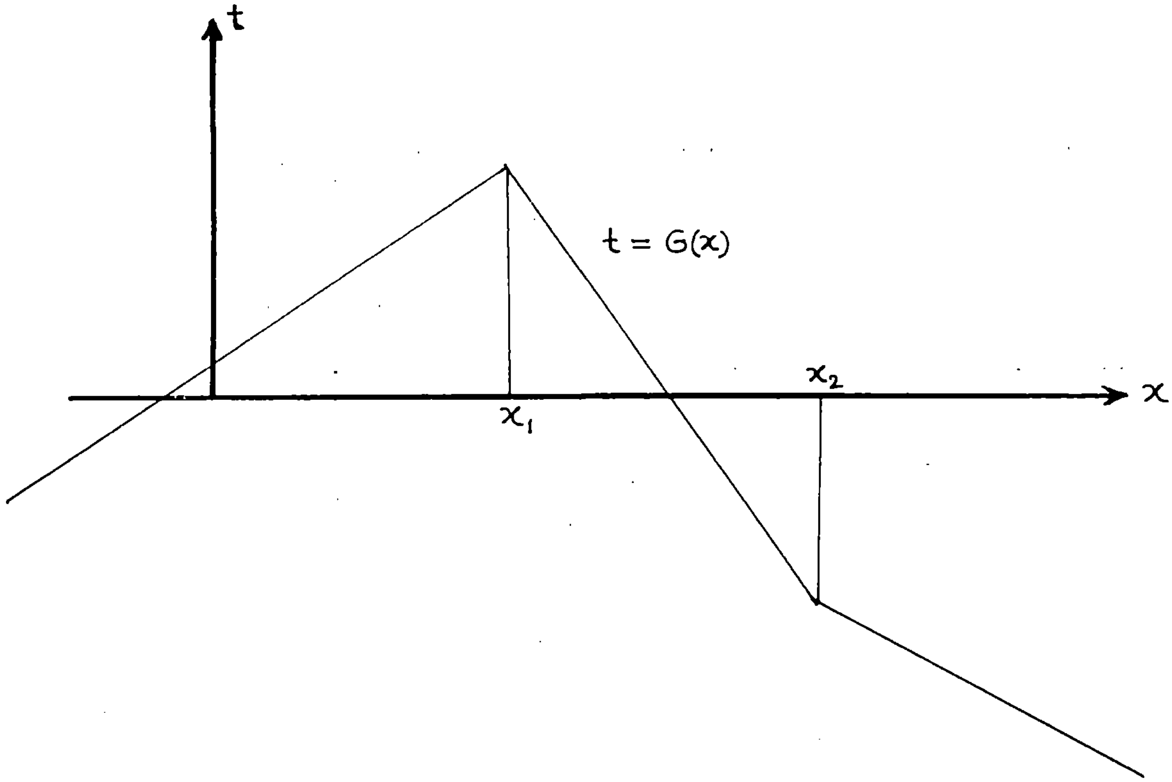


Fig 1

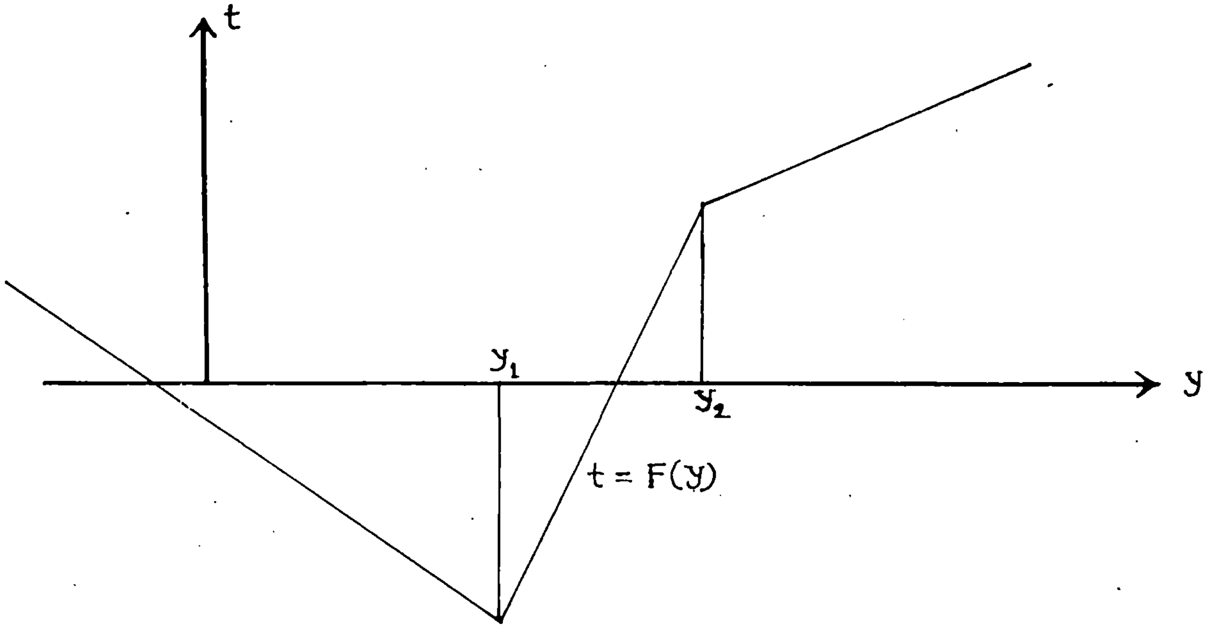


Fig 2

β_{12} being given by

$$\beta_{11} = \left[u_2^-(x_2 - x_1) - v_1^- y_1 + v_2^- y_2\right] \Big/ \left[v_2^- - v_1^-\right]$$

and $\beta_{12} = \left[u_2^-(x_2 - x_1) + v_1^+ y_1 + v_2^- y_2\right] \Big/ \left[v_1^+ + v_2^-\right]$

while if $G(x_1) > F(y_2)$ we could find no x such that $G(x) = F(y)$ holds for any $y \notin [\delta_{11}, \delta_{12}]$, δ_{11} and δ_{12} being given by

$$\delta_{11} = \beta_{11}$$

and $\delta_{12} = \left[u_2^-(x_2 - x_1) + v_1^+ y_1 - v_2^+ y_2\right] \Big/ \left[v_1^+ - v_2^+\right]$

owing to the decreasing nature of $F(y)$ in $(-\infty, y_1)$ and its increasing nature in (y_1, ∞). From the above it readily follows that the locus given by (2) remains wholly within two straight lines parallel to the x-axis.

Since the curve represented by (2) is bounded in both the x and y directions we can immediately conclude that (2) symbolises a bounded curve.

Furthermore, for any $x \in (\alpha_{11}, \alpha_{12})$ or $(\gamma_{11}, \gamma_{12})$ there exists exactly two distinct values of y which, however, coincide in case x is equal to either end point forming the interval. Arguing similarly it can be shown that for any $y \in (\beta_{11}, \beta_{12})$ or $(\delta_{11}, \delta_{12})$ we can have exactly two values of x, which are coincident when y is equal to either end point forming the interval. The above reasoning clearly demonstrates that the locus of (x, y) is a closed curve consisting of several straight line segments described around (x_1, y_1), the point associated with the greater weight. The number of line segments constituting the curve will be given shortly. We shall hereafter call this locus the *equipolygon* of the given pair of points, as shown in figure 3. For simplicity we denote the equipolygon of two points A and B with associated weights W_A and W_B by E_{AB}.

We next state a lemma the proof of which, being obvious, is left out.

Lemma 1. Let $W_A > W_B$. If P lies outside E_{AB} then $\rho(P, A) > \rho(P, B)$ else $\rho(P, A) \le \rho(P, B)$.

In addition to the definitions and notations already given we shall make use of the following in the sequel.

i) R(A, B) denotes the rectangle with A and B as opposite vertices and sides parallel to the axes of coordinates.

ii) By $\partial R(A,B)$ we shall mean the boundary of the region R(A,B).

iii) L(P, A) denotes an L-shaped curve consisting of two line segments — one through P parallel to the x-axis and the other through A parallel to the y-axis — meeting at a point, provided both the coordinates of P and A are different.

N.B. In case x or y coordinates of P and A are equal L(P, A) degenerates into a straight line segment.

iv) Γ_{AB} is the stretch T_1T_2 of E_{AB} cut off by $\partial R(A, B)$ where T_1 is the extremity of Γ_{AB} first of all encountered, maintaining primal feasibility, and T_2 is the other

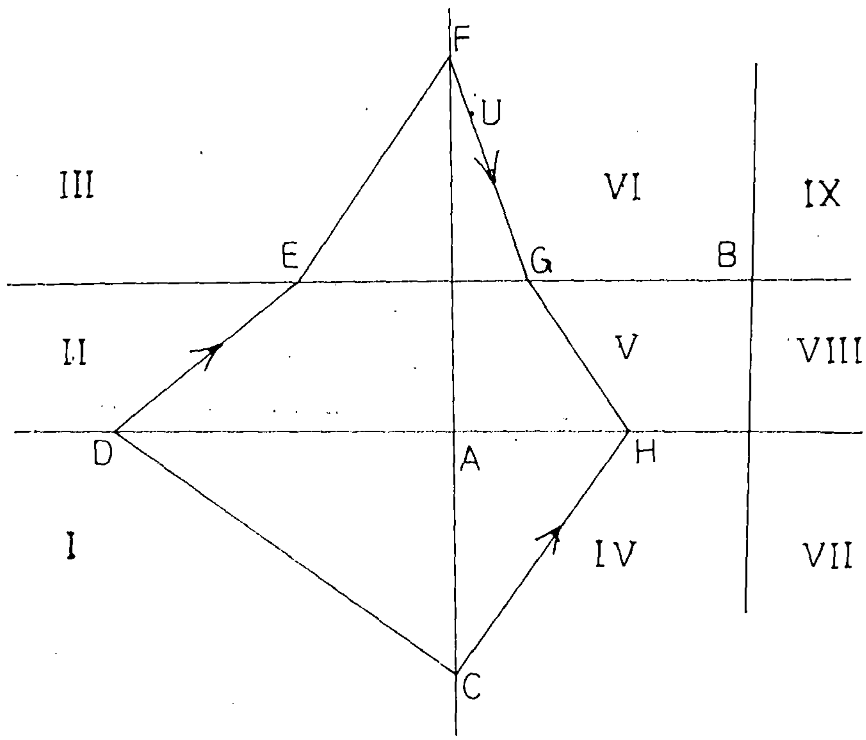


Fig 3

extremity.

v) For any point $Q \in \Gamma_{AB}$ we define $D_{AB}(Q)$ by

$D_{AB}(Q) = u_A^+ v_B^k - u_B^- v_A^l$, where $k = -1$, $l = +1$ in case $y_A \leq y_B$ and $k = +1$, $l = -1$ in case $y_A > y_B$, y_A and y_B standing for the ordinates of A and B respectively.

vi) Let $\rho(Q, P_i) = \rho(Q, P_j) > \rho(Q, P_k)$ where $i, j \in I$ and $k \in I \setminus \{i, j\}$. If Q is not an optimal solution and moving along $E_{P_i P_j}$ does not violate the condition of primal feasibility then the points P_i and P_j will be called the *Dominating points* at Q.

We shall next deliberate upon the number of extreme points the closed and bounded equipolygon representing the locus can have. Let $r = (x_2 - x_1)/(y_2 - y_1)$ denote the ratio of the lengths of the adjacent sides of $R(P_1, P_2)$. If $r \in \left[v_2^- / u_1^+, v_1^+ / u_2^- \right]$ which corresponds to the case $G(x_1) < F(y_2)$ and $F(y_1) > G(x_2)$ the equipolygon consists of four corner points given by (α_{11}, y_1), (x_1, β_{11}), (α_{12}, y_1) and (x_1, β_{12}). If r coincides with either end point then one of the two vertices of the rectangle R not occupied by a location point will be a corner point.

If, on the other hand, $r > v_1^+ / u_2^-$ or $r < v_2^- / u_1^+$ corresponding to $G(x_1) > F(y_2) \rightarrow F(y_1) > G(x_2)$ or $F(y_1) < G(x_2) \rightarrow G(x_1) < F(y_2)$ respectively, then the equipolygon has six extreme points which are (α_{11}, y_1), (x_1, δ_{11}),

(α_{12}, y_1), (γ_2, y_2), (x_1, δ_{12}) and (γ_1, y_2) or (γ_{11}, y_1), (x_1, β_{11}), (x_2, δ_1), (γ_{12}, y_1), (x_2, δ_2) and (x_1, β_{12}) ,as the case may be, where

$$\gamma_1 = \left[v_1^+(y_2 - y_1) + u_1^- x_1 - u_2^- x_2\right] \Big/ \left[u_1^- - u_2^-\right]$$

$$\gamma_2 = \left[v_1^+(y_1 - y_2) + u_1^+ x_1 + u_2^- x_2\right] \Big/ \left[u_1^+ + u_2^-\right]$$

$$\delta_1 = \left[u_1^+(x_2 - x_1) + v_1^- y_1 - v_2^- y_2\right] \Big/ \left[v_1^- - v_2^-\right]$$

and $$\delta_2 = \left[u_1^+(x_1 - x_2) + v_1^+ y_1 + v_2^- y_2\right] \Big/ \left[v_1^+ + v_2^-\right]$$

Note: By taking $y_1 > y_2$ results similar to the above with obvious changes at appropriate places may be obtained.

Case $W_1 < W_2$. The equipolygon, in this case, encloses the point (x_2, y_2) instead and results analogous to the preceding will have been found.

Case $W_1 = W_2$. The equipolygon here degenerates into an open polygon with only two extreme points which are respectively

(1) (α_2, y_2) and (α_1, y_1) when $G(x_1) > F(y_2) \Rightarrow G(x_2) < F(y_1)$

(2) (x_2, β_2) and (x_1, β_1) when $G(x_2) > F(y_1) \Rightarrow G(x_1) < F(y_2)$

(3) (x_1, β_1) and (α_1, y_1) when $G(x_1) < F(y_2)$ and $G(x_2) < F(y_1)$

where

$$\alpha_1 = \left[v_1^-(y_2 - y_1) + u_1^+ x_1 + u_1^- x_2\right] \Big/ \left[u_1^+ + u_1^-\right]$$

$$\alpha_2 = \left[v_1^+(y_1 - y_2) + u_1^+ x_1 + u_1^- x_2\right] \Big/ \left[u_1^+ + u_1^-\right]$$

$$\beta_1 = \left[u_1^-(x_2 - x_1) + v_1^+ y_1 + v_1^- y_2\right] \Big/ \left[v_1^+ + v_1^-\right]$$

and $\beta_2 = \left[u_1^+(x_1 - x_2) + v_1^+ y_1 + v_1^- y_2\right] \Big/ \left[v_1^+ + v_1^-\right]$

and for $G(x_1) = F(y_2)$ an extreme point is (x_1, y_2) while for $G(x_2) = F(y_1)$ the corresponding extreme point is (x_2, y_1).

We construct the *smallest rectangle* - to be called SR hereafter - containing all the points of S by drawing lines parallel to the coordinate axes through four properly chosen points having respectively maximum and minimum abscissas and ordinates.

A point P lying outside SR cannot be the optimal location in the unconstrained case as a movement through P perpendicular to the nearest boundary of SR and towards it will cause the objective function value to diminish. Consequently, we shall have to seek the required solution within SR and with this end in view we shall concentrate on Γ_{AB}. But in the constrained case, besides an active boundary of the constrained region, an edge of the equipolygon lying outside SR may have to be considered in order to determine the optimal location.

The theorem stated below, which serves to find the direction of movement when three or more equipolygons coincide at a non-optimal point, will be used in developing our algorithm.

Theorem 1. While moving along an edge of E_{AB} and maintaining primal feasibility let a point G be obtained such that $\rho(G, A) = \rho(G, B) = \rho(G,C)$, $C \in S\setminus\{A, B\}$. By drawing lines through G parallel to the coordinate axes it is easily seen that if G is non-optimal, the point occupying the same quadrant with respect to G as the latest entrant C, is to be dropped.

It is to be noted further that when the number of equipolygons meeting at a point is more than three, by a repeated application of the above the number of dominating points can always be reduced to two.

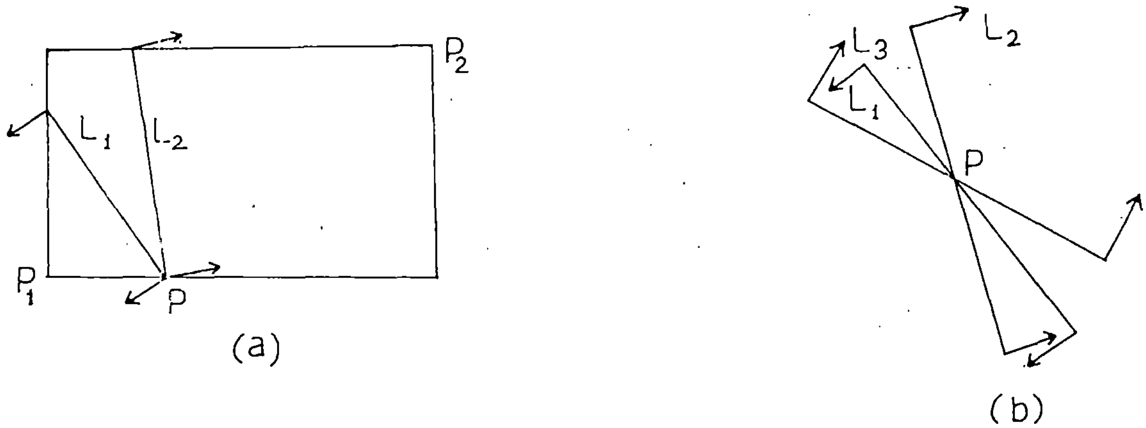
The proof of the theorem follows as a direct consequence of lemma 1.

We now state a criterion which will be useful in determining the optimal solution(s). The proof of the criterion is trivial and is, therefore, omitted.

Stopping criteria (SC)

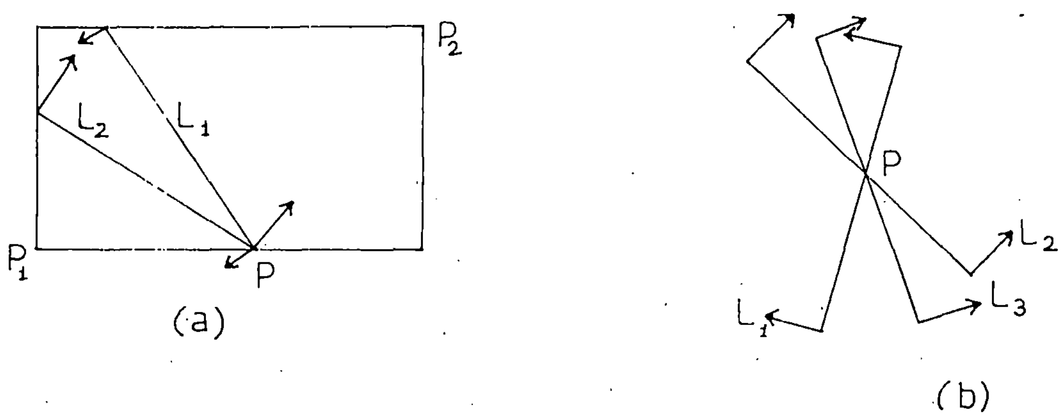
Let L_i be the isoline of P_i through some point P of the equipolygon $E_{P_i P_j}$. Also let H_i be the half-space defined by L_i containing P_i. We define the cone $\mathcal{C}$ by $\mathcal{C} = \bigcap_{1}^{k} H_i$, where k denotes the cardinality of the set of isolines passing through P and having the same level value . Then, clearly, P is the vertex of $\mathcal{C}$. If $\Sigma = \mathcal{C} \cap R(P_i, P_j) = \{P\}$ we have a unique optimum at P as shown in figure 4. Refer to figure 5 for the case when P is not optimal.

If, however, Σ degenerates into a line then we obtain a stretch comprising the set of optimal points one end of which is clearly P.



Unique Optimum at P.

Fig 4



No Optimum at P

Fig 5

Before presenting the algorithm of the current problem we introduce a definition to be required afterwards.

The weighted rectilinear distance from any point Q on E_{AB} to the fixed point A is a positive quantity. Also the distance function, being necessarily continuous, must possess a lower bound which is attained by the function at least once. Let the lower bound (or when there are more than one, that lower bound which, after maintaining primal feasibility, comes first) correspond to the point we call ME_{AB}. There exists a direction from Q along which this distance monotonically decreases as Q approaches ME_{AB}. Or it may so happen that at some point P in between Q and ME_{AB} the condition of primal feasibility may be violated. In this latter case we leave E_{AB} and move from P to U to ME_{AB}, U being the point of intesection of E_{AB} and the isoline having the smaller gradient (vide appendix). The path from Q to ME_{AB} consisting of sides of E_{AB} or a combination of sides of E_{AB} and an isoline, maintaining primal feasibility, will be denoted by $S_{AB}(Q)$.

2.2.4. Algorithm

Step 0. Select any extreme point $P \in \partial SR$ as the starting point and find the location point P_i for which $W_i^* PP_i$ is a maximum. Denote P_i by A and obtain a point $Q \in L(P, A)$ such that

$$\left\{ W_i^* \, QP_i = W_A^* \, QA : P_i \in S \setminus \{A\} \text{ and } d(P, Q) \text{ is minimum} \right\}.$$

Without any loss of generality the point P_i satisfying the above may be denoted by B. Go to step 1.

Step 1.

If a point $Q_1 \in S_{AB}(Q)$ exists such that $\rho(Q_1,A) = \rho(Q_1,P_i)$,

$P_i \in S \setminus \{A,B\}$ then go to step 2

else go to step 3.

Step 2.

If Q_1 satisfies SC then

if $D_{AB}(Q_1) \neq 0$ then Q_1 is the unique optimal solution

else

if $Q \in \Gamma_{AB}$ then the stretch QQ_1 is the set of optimal solutions

else the stretch T_1Q_1 of Γ_{AB} constitutes the set of optimal solutions

else determine the points for the next iteration using theorem 1, rename these points as A and B, $Q \longleftarrow Q_1$ and go to step 1.

Step 3.

If $D_{AB}(T_1) = 0$ then

if $Q \notin \Gamma_{AB}$ then the whole of Γ_{AB} comprises the solution set

else the stretch QT_2 gives the set of optimal solutions

else if T_1 satisfies SC then T_1 is the unique optimal point

else T_2 is the unique optimal point.

2.2.5. Numerical example

We now illustrate the working of the algorithm by means of an example. Let the demand points be all located on a rough inclined plane of inclination 10° having coefficient of friction μ = 0.3, supposed uniform. The x-axis is taken to be horizontal and the y-axis upwards along the line of greatest slope. The forces necessary to overcome the combined effect of gravity and friction on a body of weight W in the upslope and downslope directions are approximately 0.48W and 0.12W respectively.

P_i	x_i	y_i	U_i^-	U_i^+	V_i^-	V_i^+
P_1	8	4	0.8	1.2	0.12	0.48
P_2	3	3	1.6	2.4	0.24	0.96
P_3	9	5	0.6	0.9	0.09	0.36
P_4	4	2	1.2	1.8	0.18	0.72
P_5	6	3	3.2	4.8	0.48	1.92
P_6	5	1	0.4	0.6	0.06	0.24
P_7	3	6	2.4	3.6	0.36	1.44
P_8	5	7	2.8	4.2	0.42	1.68
P_9	7	8	1.6	2.4	0.24	0.96
P_{10}	4	5	2.0	3.0	0.3	1.20

Table 1

We have chosen a model which depends on the velocity of wind in the horizontal direction. Thus if the wind blows steadily from east to west we may take the perturbed values of the forces in the x-increasing and x-decreasing directions to be 1.2W and 0.8W respectively. In the calculations that follow we retain figures correct to 3 places of decimal. Let the position and the weights in the four principal directions - West, East, South and North - associated with each existing location point be given as in table 1 above.

Let us take $P = (3, 8) \in \partial SR$ as the starting point. Using step 0 of the algorithm we can easily find $A = (6, 3)$, $B = (3, 6)$ and $Q = (5.400, 8.000)$. By step 1 it immediately follows that $P_i = (3, 3)$ and $Q_1 = (5.280, 6.300)$. By step 2, since Q_1 does not satisfy SC, $Q \leftarrow Q_1$, drop the point A, and $A \leftarrow P_i$. By two successive iterations of steps 1 and 2, we finally obtain $Q_1 = (4.254, 3.003)$. As Q_1 clearly satisfies SC and, moreover, $D_{AB}(Q_1) \neq 0$, Q_1 is the unique optimal point and the corresponding objective value = 5.597.

Computational experience

To develop the Pascal code of the algorithm we have randomly generated six vectors with n components each - two for position and four for associated directional weights - by means of standard Pascal procedure **Randomize** and function **Random**. This has been repeated for values of n between 500 and 1000 over a preselected rectangular region with unequal contiguous sides, giving rise to 500 random samples of varying sizes. Interestingly enough, in all cases the algorithm required at most three iterations to converge. As actual data for problems with an n of the stated size is not readily available, we had to be content with random data.

2.2.6. Operation count

Since the objective value strictly decreases with each iteration the same pair of points, once excluded at a particular iteration for a given zone, will never recur and consequently, prevention of cycling is guaranteed. Furthermore, we are in a position to employ the information, currently generated, for future use. Our method consists in moving along an equipolygon maintaining primal feasibility. In so moving we shall either obtain the optimal solution or attain a point G such that $\rho(G, A) = \rho(G, B) = \rho(G,C)$, $C \in S\setminus\{A, B\}$. From theorem 1 we know that a path different from the current one is to be chosen at G. Moreover, an edge of an equipolygon can not intersect that of another more than once in a particular zone and there remain (n-2) other equipolygons. Thus if a point is excluded at a particular iteration it will cease to be required if the iteration is restricted to the same zone. Thus at most (n-2) pairs of linear equations need to be solved for a particular iteration in a given zone. Again, inasmuch as the number of sides of an equipolygon is at most six the number of operations is $O(n^2)$ in the worst case.

2.2.7 Appendix

Choosing the direction of movement: Let E_{AB} be an equipolygon enclosing A where $x_A < x_B$. We divide the whole xy-plane into nine zones - I through IX - by drawing lines parallel to the coordinate axes through A and B. Let us assume for the preseent $y_A < y_B$. E_{AB} has generally six edges lying in zones I to VI as shown in figure 3. In II, IV and VI the directions of movement are from D to E, C to H and F to G respectively. In zone I the movement is along CD or DC according as $u_1^- / v_1^- >$ or $< u_2^- / v_2^-$. For zone III the direction is from E to F or the other way round depending on whether $u_1^- / v_1^+ <$ or $> u_2^- / v_2^+$. If it is along FE no movement along E_{AB} is possible without violating primal feasibility. Hence, in order to reduce the objective further we make a detour via the isoline having a smaller gradient until the point $U \in FG$ in VI is encountered whence movement will be along UG towards G. In zone V the direction of movement is governed by the **Stopping Criteria**. If $y_A > y_B$ the direction of movement in II is from E to D and the roles of I and III discussed above will simply be interchanged. Similar remarks hold when E_{AB} encloses B instead.

CHAPTER 3

Constrained Problems

In this chapter we propose to study two rectilinear distance constrained problems with the minimax objective. Sec. 3.1.1 discusses the equiweighted case while sec. 3.2.1 solves both the symmetric as well as asymmetric weighted problem.

3.1.1 Geometric solution of a constrained rectilinear distance minimax location problem *

In a wide variety of situations persons interested in facility layout and location are confronted with the problem of locating a new facility in the midst of existing facilities (referred to as the one-centre problem in the literature) in such a way that the maximal distance from the new facility to the existing locations is minimised. The equal-weighted unconstrained single facility minimax location problem under the L_1 metric has been well studied by Elzinga and Hearn [31], Francis and White [36], Wesolowsky [78] et al. The weighted and constrained minimax problem under the L_2 metric in n-dimensional space has been solved by Scott et al. [69] using the concept of conjugate duality. Dutta and Chaudhuri [29] have given an elegant method of obtaining an exact solution to an equi-weighted planar constrained problem under the L_2 or Euclidean norm. Hansen et al. [43], Francis et al. [37] Hansen et al. [44], Drezner and Shelah [22] and Drezner [21] have also discussed Facility Location Models at length. Morris [64] has also

---

* This paper was published in APJOR vol 7 (1990) pp 163-171.

considered constrained multifacility minimax location problem under L_1 metric. Our objective in this section is to present a geometric solution to the constrained version of the problem in the plane in which the distances are rectilinear. As a practical application of the problem addressed in this section one might consider locating an emergency service facility, for example - a health clinic in a rural area or a fire station in a large metropolitan city - where the facility is restricted to lie within a given region under the assumption that travel is allowed on a grid only.

3.1.2 Problem statement

Assume that a set D is defined by

$D = \{ P_i : i = 1, 2, \ldots, n \}$ where $P_i(a_i, b_i)$ are the n existing location points in the plane E_2. Also assume that a new facility is to be located at $T(x, y)$ in such a way that the maximum rectilinear distance bet ween the new facility and the n given locations is minimised subject to the restriction that T is constrained to lie in a convex polytope R. The rectilinear or rectangular distance between T and P_i is given by

$$d_1(T, P_i) = |x - a_i| + |y - b_i|, \; P_i \in D.$$

The minimax problem is

$$\min_{T \in R} \; \max_{1 \le i \le n} \; d_1(T, P_i) \qquad (1)$$

where $R \subset E_2$ is given by

$$R = \left\{ X : AX \le \mathbf{b} \ ; \ A = \begin{bmatrix} a_{11} & a_{21} & \dots & a_{m1} \\ a_{12} & a_{22} & & a_{m2} \end{bmatrix}^t ; \right.$$

$$\left. \mathbf{b} = [b_1, b_2, \dots, b_m]^t \ ; \quad X = [\ x,\ y\ ]^t \right\}$$

To solve problem (1) we make use of certain elementary properties of rectilinear distance function in a plane, the proofs of which being obvious will be omitted.

As may seen in Figure 1, all the n given location points P_i may be covered by a rectangle $A_1A_2A_3A_4$ by drawing a pair of parallel lines inclined at 135° with the x-axis through two of these points P_i farthest apart and another pair of parallel lines inclined at 45° through two similar points, i.e.,points farthest from each other.

The Unconstrained Case: As long as point T lies to the right of the vertical A_3Q through A_3 the rectilinear distance from T to any point on the side A_2A_3 is greater than the same to any point on the side A_3A_4. On A_3Q itself these distances are equal. In the absence of any constraint this latter distance continually diminishes as one moves upwards along the vertical reaching its minimum value at Q, the point of intersection of the perpendicular bisector KQ of the longest side of the rectangle and A_3Q and maintains this value upto the point K, the other point of intersection of KQ and the vertical A_1K through A_1. Any point on KQ is a possible minimax location [31, 36].

We shall use the following definitions and notations in our subsequent discussion.

Dominating side: The side or sides of the rectangle having the maximum contribution towards the rectilinear distance function at a point will be called the dominating side at the point.

Cone of Descent Direction: Let f be the objective function and x any point in R . The cone of descent direction of f at x is defined by

$$D_f (x) = \{ d : \exists \bar{\alpha} > 0 \text{ with } f (x + \alpha d) < f (x) \quad \forall 0 < \alpha \le \bar{\alpha} \} .$$

Let a line l be drawn through $x \in R$ parallel to the dominating side at x .

Also let H^+ denote the closed half-space containing the dominating side defined by l. Then the cone of descent direction in the present case will be given by $N_\delta(x) \cap H^+$ or $R \cap N_\delta(x) \cap H^+$ according as x is an interior or a boundary point of R, $N_\delta(x)$ being a neighbourhood of x.

By ∂R we shall denote the **boundary of the region R.**

The Constrained Case The direction of movement is determined by the direction of descent which in turn is related to the cone of descent direction. The cone of descent direction as defined above is obtained from the dominating side of the rectangle.

Let C be the intersection of the cone of descent direction and the region R. The value of the objective function (i.e., the maximum rectilinear distance from T) does not increase along any ray lying within C - rather, it has a gradually diminishing value along ∂R constituting an extreme edge of C provided ∂R is not parallel to a dominating side or C is not degenerate.

Either of the following instances determines a stopping criterion.

I. The intersection of the two cones of descent direction reduces to their common vertex, which is the point of intersection of an active boundary, with either the vertical through A_1 or A_3, or the horizontal through A_2 or A_4.

II. The cone of descent direction degenerates into a point coinciding with an extreme point of an active boundary.

The algorithm of the present problem, which follows shortly, is justified by the lemma given below, the proof of which is given in sec. 3.1.7.

Lemma: For any point other than the one obtained by using either of the stopping criteria the rectilinear distance will be greater.

3.1.3. The Algorithm

Define the quantities c_1 through c_4 as follows:

$$c_1 = \min_{1 \le i \le n} (a_i + b_i); \quad c_2 = \max_{1 \le i \le n} (a_i + b_i);$$

$$c_3 = \min_{1 \le i \le n} (-a_i + b_i); \quad c_4 = \max_{1 \le i \le n} (-a_i + b_i).$$

The points P_i define a rectangle S the sides of which are given by

l_1: $x+y = c_1$; l_2: $x+y = c_2$; l_3: $-x+y = c_3$; l_4: $-x+y = c_4$.

Label $l_1 \cap l_3$, $l_1 \cap l_4$, $l_2 \cap l_4$, $l_2 \cap l_3$ as A_1, A_2, A_3 and A_4

respectively.

Denote the segment of perpendicular bisector of one of the longer sides of S intercepted by lines through A_2 and A_4 drawn parallel to the x-axis (or through A_1 and A_3 drawn parallel to the y-axis) by KQ, K having a greater ordinate than Q. Any point on KQ is a possible minimax location in the unconstrained case.

Without any loss of generality choose lines through A_2 and A_3 parallel to the axes as the coordinate axes and change the equations of constraints and the coordinates of the vertices of S accordingly.

Step 0 (Initialization Step)

If R contains the whole or any part of KQ then go to step 4. Starting from any point $P(X,Y) \in \text{int } R$ move towards Oy, the y-axis along the line l: $y = Y$. If the point $l \cap Oy = (0,Y) \in R$ then move towards the origin along the y-axis till the point $M = \partial R \cap Oy$ is reached and go to step 1(a). Else denote the point $l \cap \partial R$ by M and go to step 3.

Step 1(a). If M satisfies a stopping criterion then stop. M is the required point. Else go to step 1(b).

Step 1(b). Move along the extreme edge of the direction of descent until any one of the following four possibilities materializes:

(i) an extreme point, say V, is reached : go to step 2(a)

(ii) the point of intersection of the direction of descent and the line through A_1 parallel to the y-axis is attained : call this point M and go to step 1(a).

(iii) the point, say N, of intersection of the direction of descent and the x-axis is arrived at: go to step 2(b), or

(iv) the point of intersection of the direction of descent and the line through A_4 parallel to the x-axis is attained : call this point N and go to step 2(b).

Step 2(a). If V satisfies a stopping criterion then stop. V is the required location. Else obtain the adjacent edge through V, drop the current edge and go to step 1(b).

Step 2(b). If a stopping criterion holds good for the point N then stop. N is the sought after location point. Else go to step 1(b).

Step 3. Obtain the extreme edge of the direction of descent at M and move along it. If it meets the y-axis then call this point M and go to step 1(a). Else go to step 1(b).

Step 4. If $R \cap KQ \neq \Phi$ then any point $T \in R \cap KQ$ is a minimax location.

Note: When an active constraint is parallel to a dominating side of the rectangle S then the required facility location will be any point belonging to the whole stretch of the active boundary included between the axes or between lines parallel to them.

3.1.4. Analysis of Time Complexity

For constructing the rectangle S we are required to determine real numbers c_1 through c_4. Again calculating any of these four quantities necessesitates (n-1) comparisons. Hence S can be obtained in O(n) time. Then m given linear constraints defining the region R determine the extreme points of a convex hull. To get the convex hull we have to find the slopes of all the constraints requiring m multiplications and sort the angles of inclination in O(mlogm) time. The extreme points are then obtained by solving each pair of consecutive constraints arranged in sorted angular order employing method of Gaussian elimination for which a total of 5m multiplications is necessary. A few additional multiplications are needed for obtaining the point of intersection of the line y = Y with an active boundary or the y-axis as the case may be, as well as that of the direction of descent with either or both the axes of coordinates and / or lines parallel to them. Hence the worst case time complexity of the algorithm is max{O(n),O(mlogm)}.

3.1.5. Sensitivity Analysis

Introduction or removal of a location point If inserting an additional location point or deleting an existing one does not alter the configuration of the rectangle S or alters only any of its non-dominating sides then the current solution also remains unaltered. If, on the other hand, the above procedure affects a vertex of S with respect to which the current location point was obtained, then we regard the present solution as the starting solution, apply the algorithm and obtain the new optimal solution.

Introduction or removal of a constraint

If we remove a non-binding constraint or introduce a constraint which, besides altering the region R, does not have any impact on the currently active boundary with respect to which the optimum solution was obtained, modification of the solution is necessary. On the contrary, imposition of an additional constraint may result in either of the following cases.

I. The current optimal point T may now lie within the modified region R. P $\longleftarrow$ T and go to step 0.

II. The current optimal point T may lie outside the modified region R : denote the newly introduced constraint by $L \leq 0$. If the slope of $L = 0$ is non-zero then move parallel to the x-axis until the point of intersection, say N, of $L = 0$ and $y = 0$ is reached. If $N \in \partial R$ then go to step 2(b); else move towards an extreme point V along $L = 0$ and go to step 2(a). If, on the contrary, the slope of $L = 0$ is zero then move parallel to the y-axis and reach M, the point of intersection of $L = 0$ and $x = 0$. If $M \in \partial R$ then go to step 1(a); else move towards V along $L = 0$ and go to step 2(a).

Again removal of a binding constraint renders the present facility point either an interior point or a boundary point. In the former case go to step 0; in the latter go to step 1(b).

3.1.6. Numerical Solutions

We demonstrate the algorithm given above by means of the following examples :

Problem 1.

Let the convex polytope be defined by $R = \{ X : AX \leq b \}$

where

$$A = \begin{bmatrix} 2 & 5 & 2 & 0 & -1 & -1 & -5 \\ -3 & -1 & 1 & 1 & 1 & 0 & -4 \end{bmatrix}^t ;$$

$X = [x, y]^t$; $b = [-6, 4.5, 10, 11, 15, 7, 20]$

and suppose that the set D consists of the following points, their respective coordinates being noted alongside each :

P_1 = (2.00,10.00); P_2 = (0.00,12.50); P_3 = (-0.25,12.50);
P_4 = (7.00,9.00); P_5 = (3.00,13.00); P_6 = (3.60,10.45);
P_7 = (4.50,11.50); P_8 = (5.00,12.25); P_9 = (7.00,12.00);
P_{10} = (6.25,8.75); P_{11} = (7.00,10.65); P_{12} = (7.35,9.80);
P_{13} = (8.30,10.55); P_{14} = (3.25,15.45); P_{15} = (3.80,14.15);
P_{16} = (1.00,14.00); P_{17} = (1.20,13.85); P_{18} = (3.95,14.60);
P_{19} = (5.15,12.45); P_{20} = (6.30,12.20).

Let P be the point (-3.00, 1.50) $\in$ int R. Here KQ is given by the line segment joining K = (5.00, 12.50) and Q = (3.00, 10.50). Since R $\cap$ KQ = Φ, M is calculated to be (-0.75, 1.50) by step 0. Now move along the descent direction of the active constraint given by the equation 2x - 3y = -6 until the extreme point V = (1.50, 3.00) is reached. Drop this edge by step 2(a) and proceed along the next boundary given by the equation 5x - y = 4.5. After successively executing three iterations attain the required facility point N = (-0.25,10.50), which, in this case, is the point of intersection of the binding constraint having equation 2x + y = 10 and the line through A_4 = (8.50, 10.50) parallel to the x-axis.

Problem 2.

Let us consider the effect of introducing an additional constraint defined by $L : -x + 0.5 \leq 0$ in Problem 1. The current facility point N now belongs to int R. Since the slope of $L = 0$ is non-zero by section 5 move parallel to the x-axis and reach the point M = (0.50, 10.50) of intersection of NA_4 and $L = 0$. Since this point $M \notin \partial R$ move along $L = 0$ towards the extreme point V = (0.50,9.00) and following step 2(a) of the algorithm V is the required optimal location.

3.1.7. Appendix

Proof of Lemma 1. Let us suppose that the point N of intersection of an active boundary with the horizontal through A_2 is the sought for minimax location. Then the maximum rectilinear distance of N from the set D is given by NA_2. For any other point $U \in R$ or ∂R lying above or on NA_2 the maximum rectilinear distance being given by the horizontal line segment included between U and the dominating side A_1A_2, produced if necessary, is clearly greater than or equal to NA_2. Suppose U lies below NA_2 and on the same side of the vertical through A_3 as N. Then the maximum rectilinear distance of U from A_3A_2 ,the dominating side, measured horizontally is also easily seen to be greater than NA_2. Furthermore, when U and N are on opposite sides of the vertical through A_3 the maximum rectilinear distance of U from the set D (being given by the horizontal distance from the dominating side A_3A_4) is no less than the maximum rectilinear distance of U from A_3A_2 which is greater than or equal to NA_2. Hence for all positions of the point U the rectilinear distance of U is no less than NA_2. Thus NA_2 is the minimum rectilinear distance. The same holds when the facility point is the point of intersection of a boundary and the horizontal through A_4.

It can be shown in a similar way that the above conclusion is valid when the optimal location is either the point M of intersection of an active boundary of R with the vertical through A_3 or A_1, or is an extreme point V.

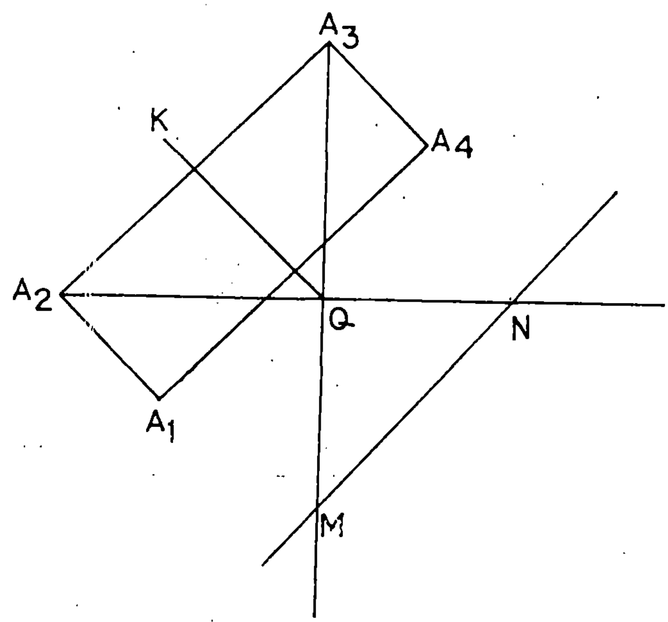


Fig 1

3.2.1 Geometric solution to some planar constrained minimax problems involving the weighted rectilinear metric*

In most practical situations we have to locate a facility point within a constrained, rather than an unconstrained, region owing to the nature of space available for the purpose of locating a facility. By incorporating constraints the facility location problem more closely resembles a real life situation. The rectilinear metric is ideally suited to urban applications in view of the assumption that travel is usually restricted to a rectangular grid. In this section we have studied the one-centre problem, also known as the single facility minimax location problem, in the presence of linear constraints forming a convex polytope in E_2 wherein we assume that besides being rectilinear the underlying distance metric is weighted by some importance factor.
When the distance between two points is a symmetric function of their positions we assign a positive weight to each demand point commensurate with the intensity of demand at that point. But for peak-hour traffic and many similar situations, when the distance is not a symmetric function, we associate four weights along the four principal directions - horizontally left or right and vertically up or down - with respect to the demand point.

Our method of approach in this problem consists first

---

* This paper appeared in APJOR vol 9 (1992) pp 135-144

in finding the maximum weighted rectilinear distance of the set of given locations from a point, arbitrarily selected within the constrained region, and subsequently reducing this distance gradually so that primal feasibility is never violated. With a view to achieving this we have appealed to methods of plane analytic geometry.

Although the solution technique has been developed for polyhedral sets, it is applicable to any convex set.

3.2.2 Formulation of the problem

Suppose that $P_i(x_i, y_i)$, $i \in I = \{1, 2, \ldots, n\}$ are the given demand points comprising the set S, (x, y) is the location of the facility (to be determined), $W_i = \begin{pmatrix} U_i \\ V_i \end{pmatrix}$ is the weight associated with the ith demand point, where

$$U_i = \begin{cases} U_i^- & \text{for } x < x_i \\ U_i^+ & \text{for } x \geq x_i \end{cases} \quad \text{and} \quad V_i = \begin{cases} V_i^- & \text{for } y < y_i \\ V_i^+ & \text{for } y \geq y_i \end{cases}$$

Let $F(x, y) = \max_i \left\{ U_i |x - x_i| + V_i |y - y_i| \right\}$. Our problem may then be stated as follows.

$$\underset{(x, y) \in E_2}{\text{minimize}} \quad F(x, y) \qquad (1)$$

subject to $a_j x + b_j y \leq c_j$, $j \in \{1, 2, \ldots, m\}$, a_j, b_j and c_j being constants defining the linear constraints, forming the convex polyhedron R. For the symmetric distance case, $U_i^- = U_i^+ = V_i^- = V_i^+$ for all i.

The unweighted 1-centre problem in the absence of constraints has been elegantly dealt with by Wesolowsky [78] and Elzinga and Hearn [31] using the rectilinear norm. Megiddo [57, 58] has proposed a linear time algorithm that solves the weighted but unconstrained problem. Francis and White [36] have given an extensive treatment to, inter alia, single and multiple facility unconstrained minimax location problems with or without weights. The equiweighted problem corresponding to (1) has been solved by Chakrabarty and Chaudhuri [7]. The method of solution required four pairs of lines each pair being drawn parallel to the axes of coordinates through a vertex of the smallest rectangle enclosing all the demand points by two lines inclined at angles of 45° and 135° respectively with the x-axis. These four lines are actually the equipolygon with respect to a pair of dominating points associated with the objective function. These lines together with the boundary were sufficient to determine the optimal solution whereas in the present case ${}^{n}C_{2}$ equipolygons in addition to the boundary need to be considered. The unconstrained version of the present problem has been studied using the symmetric distance rectilinear metric by Chakrabarty and Chaudhuri [8]. Batta et. al. [4] presented an algorithm to solve the locational problem involving weights using L_1-norm with barriers of arbitrary shape and forbidden convex regions. The solution to the asymmetric distance minimax problem has been obtained by Dykstra et al. [30], Hodgson et al. [46], Drezner and Wesolowsky [25] and Chakrabarty and Chaudhuri [9]. Furthermore, Drezner [19] has presented an algorithm for solving, among other cases, the weighted minimax problem subject to planar constraints.

3.2.3 Preliminaries

We now provide requsite definitions and notations for developing our algorithm.

1) R_{ij} denotes the rectangle having sides parallel to the coordinate axes through diagonally opposite vertices P_i and P_j.

2) P_{ij} denotes the locus of points at which the weighted rectilinear distances of P_i and P_j are equal. This locus will be hereafter called the *equipolygon* of P_i and P_j. Refer to figure 1 for the diagram of the equipolygon P_{ij}.

3) $L(P_i, P)$ represents an "L-shaped" curve having a straight line segment through P_i parallel to y-axis joined end-to-end to another segment through P parallel to the x-axis.

4) Let P_iX and P_iY be lines drawn through P_i parallel to the axes of coordinates. The *cone of descent direction* at (α, β) for the objective function at (x_i, y_i) is then defined by the set

$$H = \Big\{(x, y) : U_i|x - x_i| + V_i|y - y_i| \le U_i|\alpha - x_i| + V_i|\beta - y_i| \text{ and bounded by } P_iX \text{ and } P_iY \text{ with respect to which the points } (x,y) \text{ and } (\alpha, \beta) \text{ lie in the same quadrant}\Big\}.$$

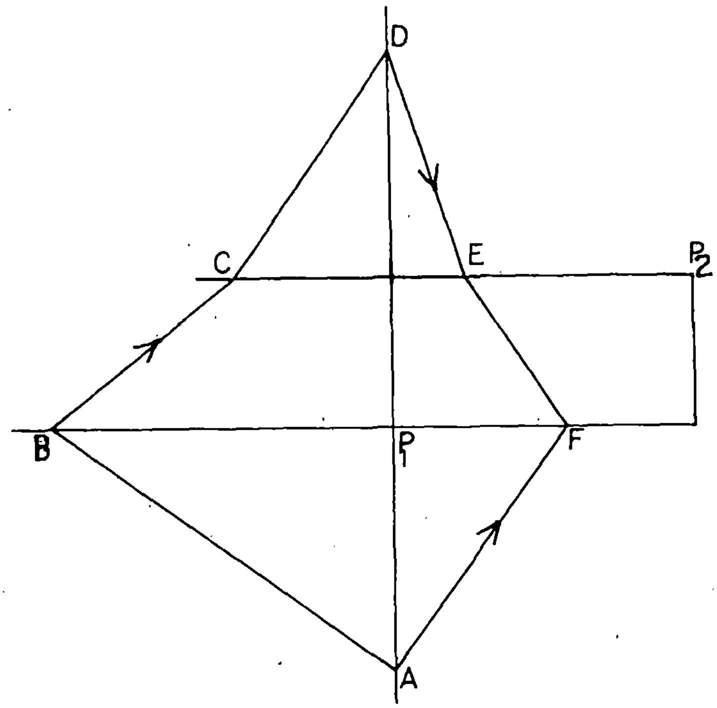


Fig 1

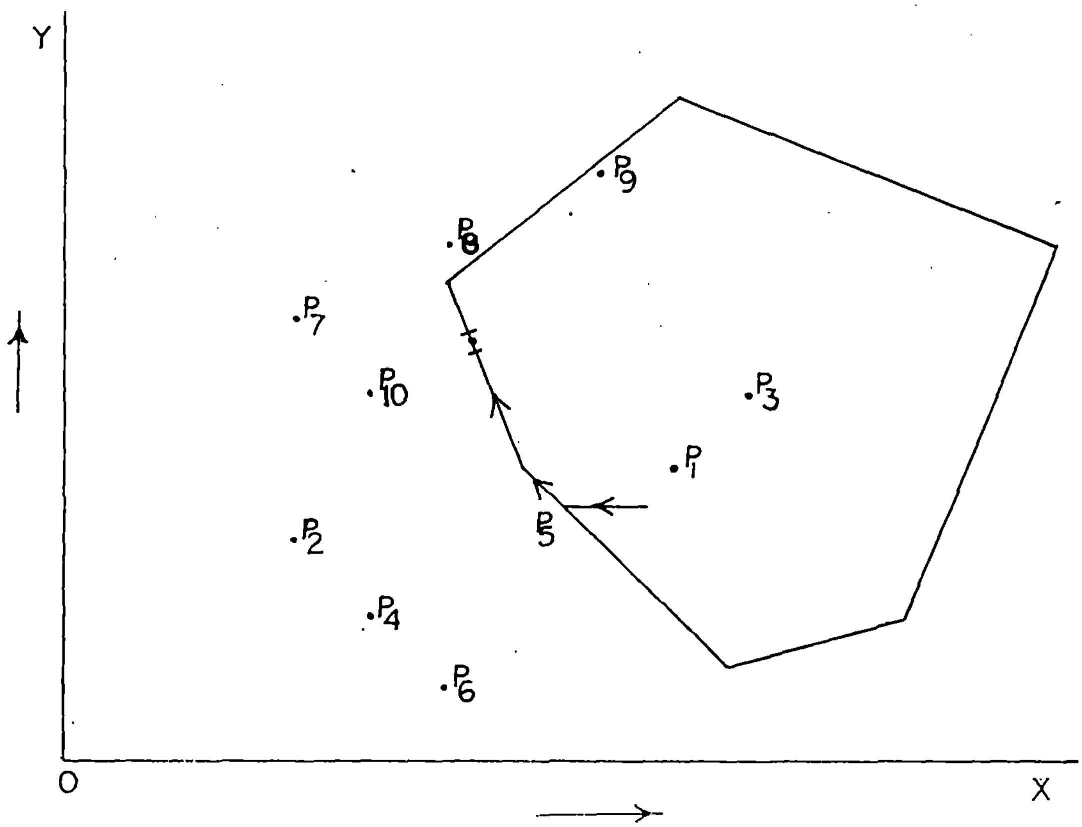


Fig 2

5) $D_{ij} = U_i^{+} V_j^{k} - U_j^{-} V_i^{l}$, where $k = -1$, $l = 1$ or $k = 1$, $l = -1$ according as $y_i < y_j$ or $y_i > y_j$, with $x_i < x_j$.

6) Γ_{ij} is the portion of P_{ij} lying within R_{ij}.

7) ∂R defines the boundary of R.

8) d(A, B) indicates the rectilinear distance between A and B and $\rho(A, P_i)$ denotes the weighted rectilinear distance between A and P_i.

9) Let the index set $J \subset I$ be such that $\rho(Q, P_i) = \rho(Q, P_j) > \rho(Q, P_k)$ where $i, j \in J$ and $k \in I \setminus J$. If Q is not an optimal solution and moving along the direction of descent of P_{ij} does not violate the condition of primal feasibility then the points P_i, P_j are to be called the *Dominating points* at Q. For its determination we shall make use of a criterion given in sec. 3.2.6.

Stopping Criteria (SC)

I Let Q be a point such that $\rho(Q, P_i) = \rho(Q, P_j) = \rho(Q, P_k) \geq \rho(Q, P_l)$ where $i, j, k \in I$ and $l \in I - \{i, j, k\}$. If a movement away from Q in the direction of descent of any equipolygon violates primal feasibility then Q is optimal.

II(a) When moving away from Q defined in I in the direction of descent of P_{ij} and maintaining primal feasibility, if no Q_1 exists such that $\rho(Q_1, P_i) = \rho(Q_1, P_j) = \rho(Q_1, P_k)$, $k \in I - \{i, j\}$ then an extreme point of Γ_{ij} is a unique optimum if

$D_{ij} \neq 0$ (sec. 3.2.6). If, however, $D_{ij} = 0$ then the whole of Γ_{ij} or the portion extending from Q to the extreme point beyond which primal feasibility ceases to be valid, forms the set of optimal points according as $Q \notin$ or $\in R_{ij}$.

(b) If $D_{ij} = 0$ and the objectives at Q, $Q_1 \in \Gamma_{ij}$ are equal then any point on the line segment QQ_1 is optimal; else, if $Q_1 \in \Gamma_{ij}$ and $Q \notin \Gamma_{ij}$ then the part of Γ_{ij} from its extreme point, encountered first, along the descent direction up to Q_1, forms the set of optimal points (sec. 3.2.6).

III (For a Boundary Point)

(a) Let (x_1, y_1) be the weighted farthest point from $(\alpha, \beta) \in \partial R$ with respect to the chosen norm. If $\partial R \cap H = \{(\alpha, \beta)\}$ then we have a unique optimum thereat. But if $\partial R \cap H$ consists of a segment of ∂R then any point contained in this segment is optimal.

(b) If the direction of descent of P_{ij} at $Q = P_{ij} \cap \partial R$ is outside R and, moreover, if any movement along a direction defined by Q and ∂R destroys primal feasibility, then Q is the unique optimum. We shall be making use of the above criteria while developing our algorithm which now follows.

3.2.4 Algorithm

Step 0 Take any $P \in \text{Int } R$ as the starting point. Find $P_i \in S$ so that $\rho(P, P_i)$ is a maximum. Let $1 \longleftarrow i$. Determine a point $Q \in L(P_1, P)$ such that $\rho(Q, P_1) = \rho(Q, P_j)$, $j \in I -$

(1) and d(Q, P) is minimum. If $Q \notin R$ then $Q \longleftarrow \partial R \cap L(P_1, P)$ and go to step 1; else $P_2 \longleftarrow P_j$. If $Q \in \partial R$ then go to step 2(a); else go to step 3.

Step 1 If Q satisfies SC-III(a) then stop. Q is the required location; else move along the direction of descent of ∂R until the first occurrence of any one of the following:

(i) an extreme point B of the currently active boundary is attained : replace the boundary with the succeeding one, $Q \longleftarrow B$ and go to step 1;

(ii) a point $B \in \partial R$, from which the weighted distances of three or more points are equal, is reached: $Q \longleftarrow B$ and go to step 2(a).

(iii) a point $B \in \partial R$, from which the weighted distances of two location points - call them P_1 and P_2 - are equal, is obtained: $Q \longleftarrow B$ and go to step 2(b).

Step 2(a) If a single equipolygon passes through Q then call it P_{12} and go to step 2(b). For three or more equipolygons meeting at Q apply SC-I. If Q satisfies it then stop; Q is optimal; else, choose the two dominating points from among them (sec. 3.2.6) required for the next iteration (call them P_1 and P_2) and go to step 2(b).

Step 2(b) If SC-III(b) applies at Q then it is the optimal point; else, if one of the points is non-dominating, then drop it and go to step 1; else go to step 3.

Step 3 Moving in the direction of descent of P_{12} at Q and maintaining primal feasibility, if a point $Q_1 \notin R$ can be found such that $\rho(Q_1, P_1) = \rho(Q_1, P_2) = \rho(Q_1, P_k)$, $k \in I - \{1, 2\}$ then $Q \longleftarrow \partial R \cap P_{12}$ and go to step 2(b); else, if $Q_1 \in R$ then apply SC-I and, if necessary, SC-II(b) (when $D_{12} = 0$), to Q_1. If Q_1 satisfies the former or both depending upon the value of D_{12} then stop, with the optimal solution; else, if Q_1 satisfies neither, then $Q \longleftarrow Q_1$, choose an equipolygon by dropping a non-dominating point from among P_1, P_2 and P_k, call it P_{12}, and repeat step 3; else, if no such Q_1 exists, then go to step 4.

Step 4 Apply SC-II(a) to obtain the optimal solution or solutions as the case may be.

N.B. 1. For the symmetric distance rectilinear metric $D_{ij} = 0$ always. 2. The algorithm has been designed in such a way that the movement always takes place along the direction of descent of an equipolygon having a finite number of sides (sec. 3.2.6) or an edge of ∂R - the number of edges that that make R is also finite - maintaining all the while primal feasibility. This, therefore, ensures that cycling never occurs.

3.2.5 Numerical Examples

We shall consider two examples - the first one involving asymmetric distance and the other employing symmetric distance- to illustrate how the algorithm works. In the computation of the optimal solution in both these cases we shall retain figures correct to 3 decimal places.

1. Asymmetric distance case: Let us assume that the location points and the associated weights in the four principal directions are as shown in Table 1 below. Also, let the convex polyhedral region R, within or on the boundary of which the required facility point is to be located, be given by

$-x - y \le -10$ (1) $\quad$ $-5x - 2y \le -38$ (2)

$-5x + 6y \le 14$ (3) $\quad$ $2x + 5y \le 61$ (4)

$5x - 2y \le 31$ (5) $\quad$ $2x - 7y \le 8$ (6)

P_i	x_i	y_i	U_i^-	U_i^+	V_i^-	V_i^+
P_1	8	4	0.8	1.2	0.12	0.48
P_2	3	3	1.6	2.4	0.24	0.96
P_3	9	5	0.6	0.9	0.09	0.36
P_4	4	2	1.2	1.8	0.18	0.72
P_5	6	3	3.2	4.8	0.48	1.92
P_6	5	1	0.4	0.6	0.06	0.24
P_7	3	6	2.4	3.6	0.36	1.44
P_8	5	7	2.8	4.2	0.42	1.68
P_9	7	8	1.6	2.4	0.24	0.96
P_{10}	4	5	2.0	3.0	0.30	1.20

Table 1

Let $P = (8, 3.5) \in \text{Int } R$ be the initial point. Following step 0 of the algorithm the weighted farthest point $P_1 = (3,$

6), P_2 = (6, 3), and Q = (4.421, 3.500). As Q $\notin$ R, Q $\longleftarrow$ $\partial R \cap L(P_1, P)$ = (6.500, 3.500). By step 1, we move along the descent direction of ∂R until the extreme point B = (6, 4) is obtained. We next replace the currently active boundary of (1) with the succeeding one corresponding to (2) and Q $\longleftarrow$ B. After two consecutive iterations of step 1 and another of step 2(b), we finally obtain the optimal location to be any point $\in$ the segment of ∂R from (5.253, 5.867) to (5.220, 5.950) with the corresponding optimum objective = 8.160. Refer to figure 2, which clearly explains how the optimal solution of problem 1 converges relative to the constrained set.

2. Symmetric distance case: We next consider a problem in which the coordinates of the location points and the constraints defining the polygonal region R are the same as in problem 1, but the weights associated with the respective location points are fixed constants (shown alongside each entry in a separate column in Table 2 below) independent of the directions in order to demonstrate how symmetry in the distance norm affects the optimal location.

Let P = (8, 3.5) $\in$ int R as before. By step 0 of the algorithm, P_1 = (5, 7), Q = (7.500, 3.500) and P_2 = (3, 6). As Q $\in$ int R we obtain following step 3, Q_1 = (4.833, 3.167) $\notin$ R. Q $\longleftarrow$ $\partial R \cap P_{12}$ = (7.000, 3.000). After two more iterations of steps 1 and 2(b) the optimal is found to be any point $\in$ the portion of Γ_{12} from (5.551, 5.122) to (5.786, 5.357) and the corresponding optimal objective = 10.286.

P_i	x_i	y_i	W_i
P_1	8	4	1.00
P_2	3	3	2.00
P_3	9	5	0.75
P_4	4	2	1.50
P_5	6	3	4.00
P_6	5	1	0.50
P_7	3	6	3.00
P_8	5	7	3.50
P_9	7	8	2.00
P_{10}	4	5	2.50

Table 2

The above example clearly validates the observation made regarding the simplifications inherent in the second problem.

3.2.6 Appendix

In order to develop our algorithm we have made use of the following results given in [9].

1. The optimal objective value with respect to any two location points P_i and P_j occurs within R_{ij} in the absence of constraints.

2. The equipolygon of two points P_i and P_j with unequal

weights is a closed figure having 4 or 6 vertices enclosing the point associated with the greater weight, as shown in figure 1. In case the weights are equal it consists of a straight line segment lying within R_{ij} flanked by two semi-infinite straight lines perpendicular to the sides of R_{ij}.

3. The weighted rectilinear distance from any point outside the equipolygon to the location point with greater weight is greater than that to the other point and vice versa.

CHAPTER 4

Computer Codes

Our purpose in this chapter is to offer two complete programs which will help considerably in exploring the methodology of our work. In sec. 4.1.1 we have given the pseudo code of the computer program for the weighted unconstrained case, whose Pascal program appears in sec. 4.1.2 whereas sec. 4.1.3 describes the program in Pascal language in the constrained equiweighted case.

4.1.1 Pseudo code of the weighted minimax problem

We now present the pseudo code of our algorithm of the MinimumSpanningDiamond in the unconstrained case. We assume the availability of a subroutine InitialStep having as its input the coordinates of the demand points belonging to the set S together with the weights associated with them. This procedure outputs a point P whose weighted rectilinear distances from A, B $\in$ S are equal. We further assume the existence of another subroutine Stretch which calculates the set of alternative optimal solutions constituting a portion of an edge of EP(A,B). ALGORITHM MSD processes the set S of demand points and yields the minimum spanning diamond of S. UNDONE, a Boolean variable, being initially set to TRUE, becomes FALSE with the attainment of the solution in which case the algorithm terminates.

Algorithm MSD

```
Let Γ be the portion of EP(A,B) extending from P to ∂R(A,B)
Call InitialStep
UNDONE = TRUE
WHILE UNDONE DO
   IF P ∈ R(A,B) THEN UNDONE = FALSE
   ELSE
    S1 = S \ {A,B}
    S2 = {C ∈ S1| the weighted distance from P1 ∈ Γ, different
      from P, is the same as that of A or B where PP1 is least}
    IF |S2| = 0 THEN
      P = EP(A,B) ∩ ∂R(A,B)
      UNDONE = FALSE
    ELSE
     IF |S2| = 1 THEN
       IF P1 ∈ R(A,B) or R(B,C) or R(A,C) THEN UNDONE = FALSE
       ELSE drop A or B and assign C to the point dropped
     ENDIF
    ELSE
     select two points from S2 ∪ {A,B}, rename them A,B
     and P ←— P1
   ENDIF
ENDWHILE
Call Stretch
output the minimum spanning diamond
END
```

4.1.2 Pascal program for the weighted unconstrained location problem

```
Program Minimum_spanning_diamond;
uses  crt,dos;
type list=array[1..n] of real;
 var infile,outfile:text;
     x,y,w:list;
     i,i1,i2,i3,l1,m1,l2,m2,ic:integer;
     d,ds,x1,x2,y1,y2,h,k,u,v,dif,lambda,w1:real;
     flag,aflag,bflag:boolean;
     hh,mm,ss,hs:word;
 Procedure Maxdist(var ii:integer);
 {Proc to find point i1 at a max wtd rect dist from (x2,y1)}
  var d,maxd:real;
  begin
   maxd:=-maxint;
   for i:=1 to n do
    begin
     d:=w[i]*(x2-x[i]+y[i]-y1);
     if maxd<d then
      begin
      maxd:=d;
       ii:=i
      end
    end
  end;                          {End Maxdist}
 Procedure Diff(var df:real; j1,j:integer; c,d:real);
 {This proc calculates the difference of rect distances of}
 {2 points j1,j from a given point (c,d)}
 var f1,f2:real;
 begin
  f1:=w[j1]*(abs(x[j1]-c)+abs(y[j1]-d));
  f2:=w[j]*(abs(x[j]-c)+abs(y[j]-d));
  df:=f1-f2
 end;                           {End Diff}
 Procedure Product(a,b,c,d:real);
 {This proc finds the sign of the product of rect dist of}
 {points i1,i2 from the points (a,b) and (c,d)}
  var dif1,dif2:real;
  begin
   bflag:=true;
   diff(dif1,i1,i2,a,b);
   diff(dif2,i1,i2,c,d);
   if dif1*dif2 <0 then bflag:=false
  end;                          {End Product}
 Procedure Quadrant(var p,q:integer; a,b,c,d:real);
 {Proc to find the quadrant in which (b,d) lies w.r.t.(a,c)}
  begin
   p:=0;q:=0;
   if a>b then p:=-1;
   if a<b then p:=1;
   if c>d then q:=-1;
   if c<d then q:=1
  end;                          {End Quadrant}
```

```
Procedure Min_max(var min,max:real; r,s:real);
{Proc to find the max and min of 2 real numbers r and s}
 begin
  min:=r;
  if min>s then
   begin
    max:=min;
    min:=s
   end
  else max:=s
 end;                          {End Min_max}
Procedure Rectangle(a,b,f1,g1,f2,g2:real; var fl:boolean);
{Proc to test if (a,b) is within or on the rectangle with}
{(f1,g1),(f2,g2) as opposite vertices}
 begin
  fl:=true;
  min_max(x1,x2,f1,f2);
  min_max(y1,y2,g1,g2);
  if ((a>=x1) and (a<=x2)) and ((b>=y1) and (b<=y2))
   then fl:=false
 end;                          {End Rectangle}
Procedure Select;
{This proc selects 2 points needed for the next iteration}
 begin
  Quadrant(l1,m1,x[i1],h,y[i1],k);
  Quadrant(l2,m2,x[i2],h,y[i2],k);
  if (l1=l2) and (m1=m2) then     {I,III,VII,IX w.r.t. i1,i2}
   if w[i1]<w[i2] then i2:=i3 else i1:=i3
  else                            {II,IV,VI,VIII w.r.t. i1,i2}
   begin
    Quadrant(l2,m2,x[i3],h,y[i3],k);
    if (l1=l2) and (m1=m2)        {I,III,VII,IX w.r.t. i1,i3}
     then i1:=i3 else i2:=i3
   end
 end;                          {End select}
Procedure Interchange(var j1,j2:integer);
{This is a proc for swapping 2 integers}
 var t:integer;
 begin
  t:=j1; j1:=j2; j2:=t
 end;                          {End Interchange}
Procedure Point(var k1:real;k2:real;p1,p2,q1,q2:integer);
{Proc to find the point of intersection of the equipolygon}
{and the boundary of the rectangle}
 var a:real;
 begin
  a:=w[i1]*(l1*x[i1]+m1*y[i1])-w[i2]*(l2*x[i2]+m2*y[i2]);
  k1:=(a-k2*(w[i1]*p1-w[i2]*p2))/(w[i1]*q1-w[i2]*q2);
 end;                          {End Point}
Procedure Boundary;
var l,m:integer;
{Proc to obtain the point of intersection of the edge of }
{the equipolygon and the boundary of the rectangle moving}
{along the direction of descent}
 begin
```

```
min_max(x1,x2,x[i1],x[i2]);
min_max(y1,y2,y[i1],y[i2]);
Quadrant(l1,m1,x[i1],h,y[i1],k);
Quadrant(l2,m2,x[i2],h,y[i2],k);
l:=l1+l2; m:=m1+m2; u:=maxint; v:=maxint;
if (l=0) or (m=0) then  {zone IV,VI or VIII,II}
 if (l=0) then      {zone IV or VI}
  begin
   if (m=-2) then v:=y1 else v:=y2;
   product(x1,v,x2,v);
   if bflag=true then
    begin
     v:=maxint;
     if w[i1]>w[i2] then u:=x[i2] else u:=x[i1]
    end
  end
 else  {m=0}
  begin               {zone VIII or II}
   if (l=2) then u:=x2 else u:=x1;
   product(u,y1,u,y2);
   if bflag=true then
    begin
     u:=maxint;
     if w[i1]>w[i2] then v:=y[i2] else v:=y[i1]
    end
  end;
if (l1=l2) and (m1=m2) then {zone VII or I or III or IX}
 if (m=-2) then                  {zone VII or I}
  begin
   v:=y1; product(x1,v,x2,v);
   if bflag=false then
    begin
     v:=maxint;
     if (l=2) then u:=x2 else u:=x1
    end
  end
 else         {m=2}
  begin       {zone III or IX}
   v:=y2; product(x1,v,x2,v);
   if bflag=false then
    begin
     v:=maxint;
     if (l=2) then u:=x2 else u:=x1
    end
  end;
if (l1=0) or (m1=0) or (l2=0) or (m2=0) then
{Lines of separation of all zones in pairs except V}
 begin
  if (m=2) or (m=-2) then
   begin
    if (m=2) then v:=y2 else v:=y1;
    product(x1,v,x2,v);
    if bflag=false then
     if (l1<>0) then l2:=-l1 else l1:=-l2
    else
```

```
        if (l1<>0) then l2:=l1 else l1:=l2
      end;
     if (l=2) or (l=-2) then
      begin
       if (l=2) then u:=x2 else u:=x1;
       product(u,y1,u,y2);
       if bflag=false then
        if (m1<>0) then m2:=-m1 else m1:=-m2
       else
        if (m1<>0) then m2:=m1 else m1:=m2
      end
    end;
  if (u=maxint) then point(u,v,m1,m2,l1,l2)
                 else point(v,u,l1,l2,m1,m2)
 end;                          {End Boundary}
Procedure Test(a1,b1,a2,b2:real);
{Proc to find euclidean dist of (h,k) lying on a given }
{equipolygon from the point of intersection of this and}
{another equipolygon}
 var xm,ym,z:real;
 begin
  xm:=(a1+a2)/2; ym:=(b1+b2)/2;
  Quadrant(l2,m2,x[i],xm,y[i],ym);
  z:=w[i1]*(l1*(x[i1]-h)+m1*(y[i1]-k))-w[i]*(l2*(x[i]-h)+
     m2*(y[i]-k));
  z:=z/(w[i1]*(l1*(u-h)+m1*(v-k))-w[i]*(l2*(u-h)+m2*(v-k)));
  if (z>0) then
   begin
    lambda:=z;
    i3:=i
   end
 end;                          {End Test}
Procedure Convex(var lambda1:real; a,b,c:real);
{Proc to test if a real number lies in a given interval}
 begin
  if (b<>a) then
   begin
    lambda1:=(c-a)/(b-a);
    if (lambda1<0) or (lambda1>1) then lambda1:=maxint
   end
  else lambda1:=maxint
 end;                          {End Convex}
Procedure Update;
{Proc to update a given point (h,k) for the next iteration}
 var xm,ym,r,s:real;
 Procedure value(var f:real; lamda,a,b:real);
  begin
   f:=lamda*b+(1-lamda)*a
  end;
 begin
  lambda:=maxint; i3:=maxint;
  xm:=(u+h)/2;ym:=(v+k)/2;
  quadrant(l1,m1,x[i1],xm,y[i1],ym);
  for i:=1 to n do
   begin
```

```
if (i<>i1) and (i<>i2) then
 begin
  diff(dif,i1,i,u,v);
  if (dif<=0 ) then
   begin
    convex(r,h,u,x[i]);
    convex(s,k,v,y[i]);
    if (r<>maxint) and (s<>maxint) then
     begin
      if (r<s) then
       begin
        value(ym,r,k,v);
        diff(dif,i1,i,x[i],ym);
        if (dif<=0) then test(h,k,x[i],ym)
        else         {dif>0}
         begin
          value(xm,s,h,u);
          diff(dif,i1,i,xm,y[i]);
          if (dif<=0) then test(x[i],ym,xm,y[i])
          else       {dif>0}
           begin
            diff(dif,i1,i,u,v);
            if (dif<=0) then test(xm,y[i],u,v)
           end
         end
       end
      else             {r>=s}
       begin
        value(xm,s,h,u);
        diff(dif,i1,i,xm,y[i]);
        if (dif<=0) then test(h,k,xm,y[i])
        else  {dif>0}
         begin
          value(ym,r,k,v);
          diff(dif,i1,i,x[i],ym);
          if (dif<=0) then test(xm,y[i],x[i],ym)
          else       {dif>0}
           begin
            diff(dif,i1,i,u,v);
            if (dif<=0) then test(x[i],ym,u,v)
           end
         end
       end
     end
    else
     if (r<>maxint) then
      begin
       value(ym,r,k,v);
       diff(dif,i1,i,x[i],ym);
       if (dif<=0) then test(h,k,x[i],ym)
       else         {dif>0}
        begin
         diff(dif,i1,i,u,v);
         if (dif<=0) then test(x[i],ym,u,v)
        end
```

```
          end
        else
         if (s<>maxint) then
          begin
           value(xm,s,h,u);
           diff(dif,i1,i,xm,y[i]);
           if (dif<=0) then test(h,k,xm,y[i])
           else            {dif>0}
            begin
             diff(dif,i1,i,u,v);
             if (dif<=0) then test(xm,y[i],u,v)
            end
          end
        else test(h,k,u,v);
        u:=h*(1-lambda)+u*lambda;
        v:=k*(1-lambda)+v*lambda
       end   {dif<=0}
     end      {i<>i1,i2}
   end;                    {end of for}
  h:=u; k:=v
 end;                        {End Update}
Procedure Initial_step;
{Proc to find (h,k) equidistant from at least 2 points}
 var d1,d11:real;
 begin
  x2:=x[1]; y1:=y[1];
  for i:=2 to n do
   begin
    if x2<x[i] then x2:=x[i];
    if y1>y[i] then y1:=y[i]
   end;
  Maxdist(i1);
  i2:=0;k:=y1;h:=x2;u:=x[i1];v:=y1;
  update;
  if i3=maxint then
   begin
    h:=x[i1];u:=x[i1];v:=y[i1];
    update;
    k:=v
   end
  else h:=u;
  i2:=i3
 end;                        {End Initial_step}
Procedure Stretch;
{Proc to find the line segment giving the optimal soln set}
 var l,m:integer;
     f1,f2,f3,f4:real;
 Procedure One(b,c:real;p1,p2,q1,q2,s:integer);
  begin
   point(b,c,p1,p2,q1,q2);
   if s=2 then diff(dif,i1,i3,c,b)
          else diff(dif,i1,i3,b,c);
   if dif>0 then
    begin
     if s=2 then begin u:=c; v:=b end
```

```
            else begin u:=b; v:=c end;
    update
   end
  else u:=maxint
 end;
begin
 Min_max(x1,x2,x[i1],x[i2]);
 Min_max(y1,y2,y[i1],y[i2]);
 Quadrant(l1,m1,x[i1],h,y[i1],k);
 Quadrant(l2,m2,x[i2],h,y[i2],k);
 l:=l1+l2;m:=m1+m2;u:=maxint;
 diff(f1,i1,i2,x1,y1); diff(f2,i1,i2,x2,y1);
 diff(f3,i1,i2,x2,y2); diff(f4,i1,i2,x1,y2);
 if (l=0) and (m=0) then      {(h,k) inside the rectangle}
  begin
   if (f1*f2<0) then one(u,y1,m1,m2,l1,l2,1);
   if (f2*f3<0) and (u=maxint) then
            one(v,x2,l1,l2,m1,m2,2);
   if (f3*f4<0) and (u=maxint) then
            one(u,y2,m1,m2,l1,l2,3);
   if u=maxint then
         begin
          u:=x1; point(v,u,l1,l2,m1,m2)
         end
  end                         {(h,k) on boundary of rectangle}
 else
  begin
   if (m=-1) then
    begin
     if (f2*f3<0) then u:=x2
     else
      if (f3*f4<0) then v:=y2 else u:=x1
    end
   else
    if (l=1) then
     begin
     if (f1*f2<0) then v:=y1
     else
      if (f3*f4<0) then v:=y2 else u:=x1
    end
   else
    if (m=1) then
     begin
     if (f1*f2<0) then v:=y1
     else
      if (f2*f3<0) then u:=x2 else u:=x1
    end
   else
    begin
     if (f1*f2<0) then v:=y1
     else
      if (f2*f3<0) then u:=x2 else v:=y2
    end;
   if u=maxint then point(u,v,m1,m2,l1,l2)
                else point(v,u,l1,l2,m1,m2);
```

```
      update
     end
   end;                          {End Stretch}
 begin                          {main action statements}
  clrscr;
  flag:=true;
  assign(infile,'file.dat');
  reset(infile);
  assign(outfile,'out.put');
  append(outfile);
  writeln('supply the number of data points');
  readln(n);
  for i:=1 to n do
   readln(infile,x[i],y[i],w[i]);
  gettime(hh,mm,ss,hs);
  write(outfile,hh,':',mm,':',ss,'.',hs:2);
  initial_step;
  rectangle(h,k,x[i1],y[i1],x[i2],y[i2],aflag);
  if aflag=false then flag:=false else Boundary;
  ic:=0;
  while flag do
   begin
    update;
    if (i3=maxint) then flag:=false
    else
     begin
      rectangle(h,k,x[i1],y[i1],x[i3],y[i3],aflag);
      if aflag=false then
       begin
        flag:=false;
        interchange(i2,i3)
       end
      else
       begin
        rectangle(h,k,x[i2],y[i2],x[i3],y[i3],aflag);
        if aflag=false then
         begin
          flag:=false;
          interchange(i1,i3)
         end
        else select
       end
     end;                        {end i3<>maxint}
    Boundary;
    ic:=ic+1
   end;                          {end while}
  writeln('total no. of iterations=',ic);
  write('The stretch extends from (',h:5:2,',',k:5:2);
  Stretch;
  writeln(') to (',u:5:2,',',v:5:2,')');
  gettime(hh,mm,ss,hs);
  write(outfile,'      ',hh,':',mm,':',ss,'.',hs:2);
  writeln(outfile,'              ',ic);
  close(outfile);
  close(infile);
end.
```

4.1.3 Pascal program of the equiweighted constrained minimax location problem

```
program Constrained;
uses   crt,dos;
const  inf = 1.0E+35;
type   list  =array[1..100] of real;
       list1 =array[1..500] of real;
       row    =array[1..4] of real;
       col   =array[1..2] of real;
       index =set of 1..100;
var    infil,outfil:text;
       a,b,c,s,a1,b1,c1,x1,y1:list;
       x,y:list1;
       ax,ay,cc,dd,m1:row;
       z:col;
       s1,s2:index;
       m,n,i,i1,i2,i3,j,jj,l1,l2,p,choice,count:integer;
       u1,u2,v1,v2,h1,k1,h2,k2,h,k,x0,y0,c11,m11:real;
       m12,alpha:real;
       flag,done:boolean;
 procedure interchange(var d1,e1,f1,d2,e2,f2:real);
 { Swaps two sets of variables}
 var t1,t2,t3:real;
 begin
   t1:=d1;d1:=d2;d2:=t1;
   t2:=e1;e1:=e2;e2:=t2;
   t3:=f1;f1:=f2;f2:=t3
 end;                                  { End interchange }
 procedure Diamond(var x1,x2,y1,y2:real);
 {Finds the stretch KQ in the unconstrained case}
 var u,v:real;
  function FindMax(t1,t2:list1;p,q:integer):real;
  {Constructs the smallest rectangle containing all demand
   points having sides parallel to x+y=0, -x+y=0}
  var  z,max:real;
  begin
   for i:=1 to n do
    begin
     z:= p*t1[i] + q*t2[i];
       if (i=1) or (z>max) then max:=z;
    end;
   FindMax := max
  end;
 begin
  cc[2]:= FindMax(x,y,1,1);
  cc[1]:= - FindMax(x,y,-1,-1);
  cc[4]:= FindMax(x,y,-1,1);
  cc[3]:= - FindMax(x,y,1,-1);
  {Finding coordinates of K(x1,y1) and Q(x2,y2)}
  u:=(cc[2]-cc[1]);  v:=(cc[4]-cc[3]);
  x1:= (cc[1]-cc[3])/2;  x2:= (cc[2]-cc[4])/2;
  if u >= v then
   begin
```

```
    y1 := (cc[2]+cc[3])/2; y2 := (cc[1]+cc[4])/2
   end
  else
   begin
    y1 := (cc[1]+cc[4])/2; y2 := (cc[2]+cc[3])/2
   end
 end;                            { End Diamond }
 Procedure FindSegment(var f1,g1,f2,g2:real;
                        var found:boolean);
 {Finds the optimal solution set in case KQ has a non-void
  intersection with the constrained region R}
 var  d,e,ub,lb,vmax,vmin:real;
 begin
  found := true;
  ub:=1; lb:=0; i :=1;
  while (i<=m) and found do
   begin
    d:=a1[i]*(u1-u2) + b1[i]*(v1-v2);
    e:=c1[i] - a1[i]*u2 - b1[i]*v2;
    if d>0 then
     begin
      vmax:= e/d;
      if ub > vmax then ub := vmax
     end
    else
     if d<0 then
      begin
       vmin:= e/d;
       if lb < vmin then lb := vmin
      end
    else
     if e<0 then found := false;
    i := i + 1
   end;
  if (ub < lb) then found := false
  else
   begin
    f1 := ub*u1 + (1-ub)*u2;
    g1 := ub*v1 + (1-ub)*v2;
    f2 := lb*u1 + (1-lb)*u2;
    g2 := lb*v1 + (1-lb)*v2
   end
 end;                            { End FindSegment }
 procedure FindDominate(var u,v:real; var j1,j2:integer);
 {Finds dominating side(s) at a point}
 var  d:row; max:real;
 begin
  j1 := 0; j2 := 0;
  d[1]:=abs(u+v-cc[1]);  d[2]:=abs(u+v-cc[2]);
  d[3]:=abs(-u+v-cc[3]); d[4]:=abs(-u+v-cc[4]);
  max := 0;
  for i:= 1 to 4 do
   begin
    if d[i]>max then
```

```
    begin
     max := d[i];
     j1 := i
    end
   else
    if d[i]=max then j2 := i
  end
end;                              { End FindDominate }
procedure Gauss(var a1,b1,c1,a2,b2,c2,u,v:real);
{Finds the point of intersection of two linear equations}
var k1:real;
begin
 if a1<>0 then
  begin
   k1:=a2/a1;
   if (b2-k1*b1 <>0) then
    begin
     v:=(c2-k1*c1)/(b2-k1*b1);
     u:=(c1-v*b1)/a1
    end
   else writeln('the solution does not exist')
  end
 else
  begin
   v:=c1/b1;
   if a2<>0 then u:=(c2-v*b2)/a2 else writeln('parallel')
  end
end;                              { End Gauss }
procedure counterclock;
{Arranges the constraints to form the convex polyhedron}
var j1,j2:integer;
 procedure partition;
 {Partitions the constraints in sets s1,s2
  according as b[i] < or > 0}
 begin
  s1 := []; s2 := [];
  p := 0;
  for i:= 1 to m do
   begin
    if (b[i] < 0) or ((b[i] = 0) and (a[i] < 0)) then
     begin
      p := p + 1;
      s1 := s1 + [i]
     end
    else s2 :: s2 + [i]
   end
 end;
 procedure sort(i1:integer);
 {Sorts the constraints in ascending order of slopes}
 var t:real;
 begin
  for i := 1 to i1-1 do
   for j := i+1 to i1 do
    if s[i] > s[j] then
```

```
      begin
       interchange(a1[i],b1[i],c1[i],a1[j],b1[j],c1[j]);
       t := s[i]; s[i] := s[j]; s[j] := t
      end
 end;
begin
 partition;
 j1 := 1; j2 := p + 1;
 for i := 1 to m do
  if i in s1 then
   begin
    a1[j1] := a[i];
    b1[j1] := b[i];
    c1[j1] := c[i];
    j1 := j1 + 1
   end
  else
   begin
    a1[j2] := a[i];
    b1[j2] := b[i];
    c1[j2] := c[i];
    j2 := j2 + 1
   end;
 for i := 1 to m do
  if (b1[i]<>0) then s[i] := (-a1[i]/b1[i])
  else
   if (a1[i] < 0) then s[i] := -inf  else s[i] := inf;
 sort(p);
 p := m-p;
 sort(p);
end;                              { End Counterclock }
procedure Vertices(r,s,t:list);
{Obtains the vertices of the polyhedron}
begin
 for i := 1 to m do
  if (i<>m) then Gauss(r[i],s[i],t[i],r[i+1],s[i+1],t[i+1],
      x1[i+1],y1[i+1])
  else Gauss(r[m],s[m],t[m],r[1],s[1],t[1],x1[1],y1[1])
end;                              { End Vertices }
procedure InitialChoice(var f,g:real);
{Assigns starting values to (h,k) }
begin
 f := (x1[1]+x1[2]+x1[3])/3;
 g := (y1[1]+y1[2]+y1[3])/3
end;                              { End InitialChoice }
procedure Convex(c,d:real; var u,lambda:real;
                   var found:boolean);
{Finds if a real number u lies between 2 real numbers c,d}
begin
 found := false;
 if (c<>d) then lambda := (u-d)/(c-d) else lambda := 2;
 if (lambda>=0) and (lambda<=1) then found := true
end;                              { End Convex }
```

```
procedure NextMovement(v:real; l,s:list);
{Finds the direction of movement at a nonoptimal point}
var lamda,zz:real; aflag:boolean;
begin
 i1 := 0; i2 := 0; i3 := 0;
 for i := 1 to m do
  begin
   convex(l[i],l[i+1],v,lamda,aflag);
   if (aflag=true) then
    if (i1=0) then
     begin
      i1 := i; z[1] := lamda*s[i] + (1 - lamda)*s[i+1]
     end
    else              {  i1 <> 0  }
     begin
      if (i2=0) then
       begin
        i2 := i; z[2] := lamda*s[i] + (1 - lamda)*s[i+1]
       end
      else            {  i1,i2 <> 0  }
       if (i3=0) then
        begin
         i3 := i; zz := lamda*s[i] + (1 - lamda)*s[i+1]
        end
     end;
   if (i2=i1+1) then
    begin
     z[2] := zz; i2 := i3
    end
  end
end;                          { End NextMovement }
procedure DominatingPair;
{Assigns coordinates of a vertex to (x0,y0) when two
 two dominating sides exist}
begin
 if (l1=1) and (l2=3) then choice := 1;
 if (l1=1) and (l2=4) then choice := 2;
 if (l1=2) and (l2=4) then choice := 3;
 if (l1=2) and (l2=3) then choice := 4;
 case choice of
 1 : begin x0:=ax[1]; y0:=ay[1]; count:=1 end;
 2 : begin y0:=ay[2]; x0:=ax[2]; count:=2 end;
 3 : begin x0:=ax[3]; y0:=ay[3]; count:=1 end;
 4 : begin y0:=ay[4]; x0:=ax[4]; count:=2 end
 end
end;                          { End DominatingPair }
procedure SingleSide;
{Assigns coordinates of a vertex to (x0,y0) when two
 one dominating sides exists}
begin
 if (l1=1) or (l1=3) then
  begin
   x0 := ax[1];
   if (l1=1) then y0 := ay[2] else y0 := ay[4];
  end
```

```
 else          { (l1=2) or (l1=4) }
  begin
   x0 := ax[3];
   if (l1=2) then y0 := ay[4] else y0 := ay[2];
  end
end;                              { End SingleSide }
Procedure Update(var f,g:real);
{Selects one of the two points of intersection of x=h or
 y=k with the boundary}
begin
if abs(f-z[1]) < abs(f-z[2]) then
 begin
  g := z[1];
  i := i1
 end
else
 begin
  g := z[2];
  i := i2
 end
end;                              { End Update }
procedure InitialStep;
{Reaches an active boundary from an interior point
 maintaining primal feasibility}
var alpha:real; aflag:boolean;
begin
 aflag := true;
 vertices(a1,b1,c1);
 x1[m+1] := x1[1];
 y1[m+1] := y1[1];
 InitialChoice(h,k);
 FindDominate(h,k,l1,l2);
 if (l2=0) then
  begin
   SingleSide;
   NextMovement(k,y1,x1);
   Convex(z[1],z[2],x0,alpha,aflag);
   if (aflag=true) then       {y=k meets x=x0 inside R}
    begin
     h := x0;
     FindDominate(h,k,l1,l2);
    end
   else Update(x0,h)          {y=k meets the boundary}
  end;
 if (l2<>0) then
  begin
   DominatingPair;
   if (count=1) then          {(choice=1) or (choice=3)}
    begin
     NextMovement(h,x1,y1);
     Update(y0,k);
    end
   else                       {(choice=2) or (choice=4)}
```

```
    begin
     NextMovement(k,y1,x1);
     Update(x0,h);
    end
  end
end;                         { End InitialStep }
procedure Replace(t,u:list; f:real);
{Updates boundary if SC is not satisfied at extreme point}
begin
 m11 := s[i];
 if abs(t[i+1]-f) < abs(t[i]-f) then
  begin
   i := i + 1;
   h := t[i]; k := u[i]
  end
 else
  begin
   i := i - 1;
   h := t[i]; k := u[i]
  end;
 m12 := s[i]
 if (m1[l1]=1) then
  begin
   if (m1[l1]*m11<0) then
    begin if (m11*m12<0) and (m12<1) then done := true end
   else
    if (((m11>1) and (m12<1)) or ((m11<1) and (m12>1)))
     then done := true
  end
 else                        { m1[l1]=-1}
  begin
   if (m1[l1]*m11<0) then
    begin if (m11*m12<0) and (m12>-1) then done:=true end
   else
    if (((m11<-1) and (m12>-1)) or
    ((m11>-1) and ((m12<-1) or (m12>0)))) then done:=true
  end
end;                         { End Replace }
procedure One(var f,g:real; u:real; t,w:list);
{Course to be taken when s[i]*m1[l1]>0 }
var lamda:real; aflag:boolean;
begin
 Convex(t[i],t[i+1],u,lamda,aflag);
 if (aflag=true) then
  begin
   f := u; g := lamda*w[i] + (1-lamda)*w[i+1]; done := true
  end
 else replace(x1,y1,u)
end;                         { End One }
procedure Two;
{Course of action when (h,k) is on x=x0 or y=y0 }
var aflag:boolean;
begin
 if (count=1) then
```

```
  begin
   if (abs(s[i])<1) then done := true   { M is optimal }
   else                          { abs(s[i])>=1 }
    begin
     if (s[i]<0) then y0 := ay[2] else y0 := ay[4];
     Convex(y1[i],y1[i+1],y0,alpha,aflag);
     if (aflag=true) then
      begin
       done := true;
       k := y0; h := alpha*x1[i] + (1-alpha)*x1[i+1]
      end
     else Replace(y1,x1,y0)
    end
  end
 else                            { (count=2) }
  begin
   if (abs(s[i])>1) then done := true   { N is optimal }
   else                          { abs(s[i])<=1 }
   begin
    if (s[i]<0) then x0 := ax[1] else x0 := ax[3];
    Convex(x1[i],x1[i+1],x0,alpha,aflag);
    if (aflag=true) then
     begin
      done := true;
      h := x0; k := alpha*y1[i] + (1-alpha)*y1[i+1]
     end
    else Replace(x1,y1,x0)
   end
  end
end;                             { End Two }
procedure Four;
{Course to be adopted when s[i]*m1[l1]<0}
var aflag:boolean;
begin
 if (b1[i]<>0) then
  begin
   c11 := (c1[i]-a1[i]*x0)/b1[i];
   Convex(k,y0,c11,alpha,aflag);
   if (aflag=true) then
    begin
     Convex(x1[i],x1[i+1],x0,alpha,aflag);
     if (aflag=true) then
      begin
       FindDominate(h,k,l1,l2);
       DominatingPair;
       h := x0;
       NextMovement(h,x1,y1);
       Update(y0,k);
       if (abs(s[i])<1) then done := true   { M is optimal }
       else                        { abs(s[i])>=1 }
        begin
         Convex(y1[i],y1[i+1],y0,alpha,aflag);
         if (aflag=true) then
          begin
```

```
            done := true;
            k := y0; h := alpha*x1[i] + (1-alpha)*x1[i+1]
           end
          else Replace(y1,x1,y0);              { aflag=false }
          aflag := true
         end
       end
      else Replace(x1,y1,x0);                  { aflag=false }
      aflag := true
     end
   end;
  if (b1[i]=0) or (aflag=false) then
   begin
    Convex(y1[i],y1[i+1],y0,alpha,aflag);
    if (aflag=true) then
     begin
      if (abs(s[i])>1) then
       begin
        done := true;                               { M is optimal }
        k := y0; h := alpha*x1[i] + (1-alpha)*x1[i+1]
       end
      else
       begin
        jj := l1;
        FindDominate(h,k,l1,l2);
        if (l1=jj) then l1 := l2;
        SingleSide;
        Convex(x1[i],x1[i+1],x0,alpha,aflag);
        if (aflag=true) then
         begin
          done := true;
          h := x0; k := alpha*x1[i] + (1-alpha)*x1[i+1]
         end
        else Replace(y1,x1,y0)
       end
     end
    else Replace(x1,y1,x0)
   end
 end;                              { End Four }
begin                              { main action statements }
 clrscr;
 assign(infil,'input file');
 {The coordinates of the demand points and the coefficients
  of the linear constraints to be read from the input file}
 reset(infil);
 assign(outfil,'output file');
 rewrite(outfil);
 writeln('supply no. of constraints m and no. of points n');
 readln(m,n);
 for i := 1 to n do readln(infil,x[i],y[i]);
 for i := 1 to m do readln(infil,a[i],b[i],c[i]);
 Diamond(u1,u2,v1,v2);
 ax[1]:=(cc[1]-cc[3])/2; ay[1]:=(cc[1]+cc[3])/2;
 ax[2]:=(cc[1]-cc[4])/2; ay[2]:=(cc[1]+cc[4])/2;
```

```
  ax[3]:=(cc[2]-cc[4])/2; ay[3]:=(cc[2]+cc[4])/2;
  ax[4]:=(cc[2]-cc[3])/2; ay[4]:=(cc[2]+cc[3])/2;
  for i := 1 to 4 do
   if (i=1) or (i=2) then m1[i] := -1 else m1[i] := +1;
  Counterclock;
  FindSegment(h1,k1,h2,k2,flag);
  if flag=true then
   begin
    write('The segment from (',h1,', ',k1,')','to (',h2);
    writeln(', ',k2,') constitutes the required solution')
   end
  else                                { flag=false }
   begin
    done := false;
    InitialStep;
    while not(flag) do
     begin
      if (l2<>0) then
       begin
        Two;
        if (done=true) then flag := true
       end
      else                            { l2=0 }
       begin
        if (s[i]*m1[l1]>0) then
         begin
          if (abs(s[i])<1) then One(h,k,x0,x1,y1)
                                else One(k,h,y0,y1,x1);
          if (done=true) then flag := true
         end
        else                          { (s[i]*m1[l1]<0) }
         begin
          Four;
          if (done=true) then flag := true
         end
       end
     end                              { end while }
   end;
  writeln(outfil,'h= ',h,'k= ',k);
  close(outfil);
  close(infil)
end.
```

BIBLIOGRAPHY

1. Aly, A. A., D. C. Kay and D. W. Litwhiler. Location dominance on spherical surfaces. Opns. Res. 27 (1979) 972-981.

2. Aneja, Y. P., R. Chandrasekaran, and K. P. K. Nair. A note on the m-center problem with rectilinear distances. Eur. J. Opnl. Res. 35 (1988) 118-123.

3. Aykin T. and G. F. Brown. Interacting new facilities and location-allocation problems. Trans. Sci. 26 (1992) 212-222.

4. Batta, R., A. Ghose, and U. S. Palekar. Locating Facilities on the Manhattan Metric with Arbitrarily Shaped Barriers and Convex Forbidden Regions. Trans. Sci. 23 (1989) 26-36.

5. Blumenthal, L. M., and G. E. Wahlin. On the spherical surface of smallest radius enclosing a bounded subset of n-dimensional Euclidean space. Am. Math. Soc. Bull. 47 (1941) 771-777.

6. Castells, C., and R. C. Melville. An unusual algorithm for finding the minimum spanning circle of a convex polygon. Technical Report 17, Department of Electrical Engineering and Computer Science, John Hopkins University, Baltimore. (1983).

7. Chakrabarty N. R., and P. K. Chaudhuri. Geometric solution of a constrained rectilinear distance minimax location problem. APJOR. 7 (1990) 163-171.

8. Chakrabarty N. R., and P. K. Chaudhuri. Solution of a Weighted One-Centre Problem under the L_1 Norm. Communicated.

9. Chakrabarty N. R., and P. K. Chaudhuri. Solution of an asymmetric rectilinear distance minimax location problem. Communicated.

10. Chakrabarty N. R., and P. K. Chaudhuri. Geometric solution to some planar constrained minimax problems involving the weighted rectilinear metric. APJOR 9 (1992) 135-144.

11. Chakrabarty R. K., and P. K. Chaudhuri. Note on geometrical solutions for some minimax location problems. Trans. Sci. 15 (1981) 164-166.

12. Chatelon, J. A., D. W. Hearn and T. J. Lowe. A subgradient algorithm for certain minimax and minisum location problems. Math. Prog. 15 (1978) 130-145.

13. Chen, R. An improved method for the solution of the problem of location on an inclined plane. Recherche operationnelle 25 (1991) 45-53.

14. Christofides, N. Graph Theory : An algorithmic approach. Academic Press, INC. (1986).

15. Chrystal, G. On the problem to construct the minimum circle enclosing n given points in a plane. Proc. Edinburg Math Soc. 3 (1885) 30-33.

16. Courant, R., and H.Robbins. What is Mathematics? Oxford University Press, New York (1941).

17. Dasarathy, B. and L. J. White. A maxmin location problem. Opns. Res. 28 (1980) 1385-1401.

18. Dearing P. M., and R. L. Francis. A Network Flow Solution to a Multifacility Minimax Location Problem Involving Rectilinear Distances. Trans. Sci. 8 (1974) 126-141.

19. Drezner, Z. Constrained location problems in the plane and on a sphere. IIE Trans. 15 (1983) 300-304.

20. Drezner, Z. A solution to the Weber location problem on the sphere. J. Opnl. Res. Soc. 36 (1985) 333-334.

21. Drezner, Z. On the Rectangular p-Centre Problem. Nav. Res. Log. 34 (1987) 229-234.

22. Drezner, Z. and S. Shelah. On the Complexity of the Elzinga-Hearn Algorithm for the 1-Centre Problem. Math. Oper. Res. 12 (1987) 255-261.

23. Drezner, Z. and G. O. Wesolowsky. Optimal location on a sphere. J. Opnl. Res. Soc. 29 (1978) 997-1004.

24. Drezner, Z., and G. O. Wesolowsky. Single Facility L_p-distance Minimax Location. SIAM J. Alg. Discr. Meth. 1 (1981) 315-321.

25. Drezner, Z., and G. O. Wesolowsky. The asymmetric distance location problem. Trans. Sci. 23 (1989) 201-207.

26. Drezner, Z. and G. O. Wesolowsky. A maximin location problem with maximum distance constraints. AIIE Trans. 12 (1980) 249-252.

27. Drezner, Z. and G. O. Wesolowsky. The location of an obnoxious facility with rectangular distances. J. Reg. Sci. **23** (1983) 241-248.

28. Drezner, Z. and G. O. Wesolowsky. Location of an obnoxious route. J. Opl. Res. Soc. **40** (1989) 1011-1018.

29. Dutta, D. and P. K. Chaudhuri. Geometric Solution for a Constrained Minimax Location Problem. APJOR **6** (1989) 148-157.

30. Dykstra, D. P., D. E. Auelrich and J. R. Henshaw. Prebunching to reduce helicopter logging costs. J. Forest **76** (1978) 362-364.

31. Elzinga, J. and D. W. Hearn. Geometrical Solutions for Some Minimax Location Problems. Trans. Sci. **6** (1972a) 379-394.

32. Elzinga, D. J. and D. W. Hearn. The minimum covering sphere problem. Mgmt. Sci. **19** (1972b) 96-104.

33. Elzinga, J. and D. W. Hearn. Minimax multifacility location with Euclidean distances. Presented at the 41st National Meeting of ORSA, New Orleans (1972)

34. Francis, R. L. A Geometrical Solution Procedure for a Rectilinear Distance Minimax Location Problem. AIIE Trans. **4** (1972) 328-332.

35. Francis, R. L. Some aspects of a minimax location problem. Opns. res. **15** (1967) 1163-1169.

36. Francis, R. L., and J. A. White. Facility Layout and Location :An Analytic Approach, Prentice Hall, Englewood Cliffs, N. J. (1974)

37. Francis, R. L., L. F. McGinnis and J. A. White. Locational Analysis. Eur. J. Opnl. Res. **12** (1983) 220-252.

38. Francis, R. L., L. F. McGinnis and J. A. White. Facility Layout and Location :An Analytic Approach, Second Edition, Prentice Hall, Englewood Cliffs, N. J. (1992)

39. Frank, H. Optimum locations on a graph with probilistic demands Opns. Res. **14** (1966) 409.

40. Frank, H. Optimum locations on a graph with correlated normal demands, Opns. Res. **15** (1967) 552.

41. Goldman, A. J. Minimax location of a facility in a network. Presented at the 41st National Meeting of ORSA, New Orleans (1972).

42. Hakimi, S. L. Optimum location of switching centres and the absolute centres and medians of a graph, Opns. Res. **12** (1964) 450-459.

43. Hansen, P., D. Peeters, D. Richard and J. F. Thisse. The Minisum and Minimax Location Problems Revisited. Opns. Res. **33** (1985) 1251-1264.

44. Hansen, P., D. Peeters and J. F. Thisse. Public Facility Location Models: A Selective Survey, in Locational Analysis of Public Facilities, J. F. Thisse and H. G. Zoller (eds), North Holland (1983).

45. Hearn D. W. and J. Vijay. Efficient algorithms for the (weighted) minimum circle problem. Opns. Res. **30** (1982) 777-795.

46. Hodgson, M.J., R.T. Wong and J. Honsaker. The p-centroid problem on an inclined plane. Opns.Res. **35** (1987) 221-233.

47. Hwang, R. Z., R. C. T. Lee and R. Chang. The slab dividing approach to solve the Euclidean p-centre problem. Algorithmica, **9** (1993) 1-22.

48. Jacobsen, S. K. An algorithm for the minimax Weber problem, Eur. J. Opnl. Res. **6** (1981) 144-148.

49. Ko, M. T., R. C. T. Lee and J. S. Chang. Rectilinear m-Center Problem. Nav. Res. Log. **37** (1990) 419-427.

50. Kuhn, H. W. Locational problems and Mathematical Programming. Separatum-colloquium on the application of Mathematics to Economics. (1963) 235-242.

51. Kuhn, H. W. A note on Fermat's problem. Math. Prog. **4** (1973) 98-107.

52. Kuhn, H. W. and R. E. Kuenne. An efficient algorithm for the numerical solution of the generalised Weber problem in spatial Economics. J. Reg. Sci. **4** (1962) 21-33.

53. Lawson, C. L. The smallest covering cone or sphere. SIAM Rev. **7** (1965) 415-417.

54. Love R. F. and J. G. Morris. Solving constrained multi-facility location problems involving l_p distances using convex programming. Operations Research Technical report,

Graduate School of Business, University of Wisconsin, Madison (1975).

55. Love, R. F., J. G. Morris and G. O.Wesolowsky. Facilities Location, Models and Methods, North Holland (1988).

56. Love, R. F., G. O. Wesolowsky and S. A. Kraemer. A multi-facility minimax location method for Euclidean distances. Int. J. Prod. Res. **11** (1973) 37.

57. Megiddo, N. Linear-time algorithms for linear programming in R^3 and related problems. SIAM J. Comput. **12** (1983a) 759-776.

58. Megiddo, N. The weighted Euclidean 1-center problem. Math. Oper. Res. **8** (1983b) 498-504.

59. Megiddo, N. Linear time algorithms for linear programming in $R \times R \times R$ and related problems. Proceeding of the 23rd Annual Symposium on the foundations of computer science. (1982) 151-162.

60. Mehrez, A., Z. Sinuany-Stern and A. Stulman. An enhancement of the Drezner-Wesolowsky algorithm for single-facility location with maximin of rectilinear distance. J. Opl. Res. Soc. **37** (1986) 971-977.

61. Melachrinoudis, E. The MAXIMIN location problem using a Euclidean metric. Ph. D. dissertation, Department of Industrial Engineering and Operations Research,University of Massachusetts (1980).

62. Melachrinoudis, E. An efficient computational procedure for the rectilinear MAXIMIN location problem. Trans. Sci. 22 (1988) 217-223.

63. Melachrinoudis, E. and P. Cullinane. Locating an undesirable facility with a minimax criterion. Eur. J. Opnl. Res. 24 (1986) 239-246.

64. Morris, J. G. A Linear Programming Approach to the Solution of Constrained Multi-Facility Minimax Location Problems where Distances are Rectilinear. Opnl. Res. Quart. 24 (1973) 419-435.

65. Nair K. P. K. and R. Chandrasekaran. Optimal location of a single service centre of certain types. Nav. Res. Log. Quart. 18 (1971) 503-510.

66. Oommen, B. J. An efficient geometric solution to the minimum spanning circle problem. Opns. Res. 35 (1987) 80-86.

67. Preparata, F. P. and M. I. Shamos. Computational Geometry: An Introduction. Springer-Verlag, New York (1990).

68. Rademacher, H. and O. Toeplitz. The enjoyment of Mathematics. Princeton University Press, Princeton, N. J. (1957).

69. Scott, C. H., B. A. Murtagh and E. Sirri. Solution of Constrained Minimax Location with Euclidean Distances via Conjugate Duality. NZOR 13 (1985) 61-67.

70. Shamos, I. M. Computational Geometry. Ph.D. thesis, Yale

University (1978).

71. Sylvester, J. J. A question in the geometry of situation. Quart. J. Pur. Appl. Math. **1** (1857) 79.

72. Sylvester, J. J. On Poncelet's approximate linear valuation of surd forms. Phil. Mag. Ser. 4, **20** (1860) 203-222.

73. Tamir A. On the complexity of some classes of location problems. Trans. Sci. **26** (1992) 352-354.

74. Toregas, C.. R. Swain, C. Revelle and L. Bergman. The location of emergency service facilities. Opns. Res. **19** (1971) 1363-1373.

75. Vijay, J. An algorithm for the p-centre problem in the plane. Trans. Sci. **19** (1985) 235-245..

76. Weber, A. Uber den standort der Industrien (1909). Translated as Alfred Weber's Theory of the location of Industries by C. J. Friedrich, Chicago (1929).

77. Wendell, R. E. A. P. Hurter and R. J. Lowe. Efficient Points in Location Problems. AIIE Trans. **9** (1977) 238-246.

78. Wesolowsky, G. O. Rectilinear Distance Location under the Minimax Optimality Criterion. Trans. Sci. **6** (1972) 103-113.

79. Wesolowsky, G. O. Location problem on the sphere. Reg. Sci. Urban Econ. **12** (1983) 495-508.

80. Zhukhovitsky S. I. and L. I. Avdeyeva. Linear and Convex Programming. Saunders, Philadelphia (1966).