ANALYSIS OF MULTIMEDIAN PROBLEMS ON TIME DEPENDENT NETWORKS

A THESIS
SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING
AND THE INSTITUTE OF ENGINEERING AND SCIENCE
OF BİLKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
F. Sibel Salman
July, 1994

ANALYSIS OF MULTIMEDIAN PROBLEMS ON TIME DEPENDENT NETWORKS

A THESIS

SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

F. Sibel Salman

July, 1994

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Assoc. Prof. Barbaros Ç. Tansel (Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Assoc. Prof. Mustafa Akgül

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Assoc. Prof. Cemal Dinçer

Approved for the Institute of Engineering and Science:

[illegible]

Prof. Mehmet Baray
Director of the Institute

ABSTRACT

ANALYSIS OF MULTIMEDIAN PROBLEMS ON TIME DEPENDENT NETWORKS

F. Sibel Salman
M.S. in Industrial Engineering
Advisor: Assoc. Prof. Barbaros Ç. Tansel
July, 1994

Time dependency arises in transportation and computer-communication networks due to factors such as time varying demand, traffic intensity, and road conditions. This necessitates a locational decision to be based on an analysis involving a time horizon. In this study, we analyze multi-median problems with linear demand functions on both tree and cyclic networks in a continuous time domain. The trajectory of the optimal solution is a piecewise linear concave function. We develop an algorithm that constructs the trajectory by solving $O(\tilde{q})$ static problems, where $\tilde{q}$ is the number of linear pieces in the trajectory. The properties of the optimal solution over the time horizon are also analyzed for various randomly generated problem instances.

Keywords: Dynamic Multimedian Problem, Parametric Analysis, Trajectory Construction.

ÖZET

ZAMANA BAĞIMLI SERİMLERDE ÇOK TESİSLİ YER SEÇİMİ PROBLEMİNİN ANALİZİ

F. Sibel Salman
Endüstri Mühendisliği, Yüksek Lisans
Danışman: Doç. Dr. Barbaros Ç. Tansel
Temmuz 1994

Zamanla değişen talep, trafik yoğunluğu, yol durumu gibi faktörler ulaşım ve haberleşme serimlerini zamana bağımlı kılabilir. Bu durumda tesis yer seçimi kararının bir zaman sürecini içeren analize dayanması gerekir. Bu çalışmada, ağaç ve genel serimlerde çok tesisli yer seçimi probleminde talepler zamana bağımlı doğrusal fonksiyonlar olarak alınmıştır. Optimal çözümün izdüşümü parçalı doğrusal bir fonksiyondur. Parça sayısı $\tilde{q}$ ise, $O(\tilde{q})$ statik problem çözerek izdüşümü hesaplayan bir algoritma geliştirilip, optimal çözümün değişim özellikleri analiz edilmiştir.

Anahtar Sözcükler: Zamana Bağımlı Çok Tesisli Yer Seçimi Problemi, Parametrik Analiz, Optimal Çözümün İzdüşümü.

ACKNOWLEDGMENTS

This thesis would not have been possible without the sympathetic aid given and the deep interest shown by my supervisor, Assoc. Prof. Barbaros Ç. Tansel. His invaluable instruction in the field of operations research, especially in location theory has been my constant guidance and source of inspiration.

I would like to express my thanks to both members of my thesis committee, Assoc. Prof. Mustafa Akgül and Assoc. Prof. Cemal Dinçer for their valuable comments.

Special thanks are due to other professors and fellow students, who helped me in the preparation of this thesis, especially Selçuk Avcı and Hakan Özaktaş for providing me with their codings and Alper Atamtürk for his kind help.

I cannot fully express my gratitude to Ayşe Cankara and Gürkan Salk for their invaluble help and support.

My thanks also go to my former and current officemates, Bahar Kara and Muhittin H. Demir, who supported and encouraged me throughout my studies.

I would like to express my deepest thanks to my family for making it possible.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The importance of facility location decisions can be emphasized by the fact that over 250 billion dollars is annually spent in the U.S. alone on establishing new facilities and modifying them later on. Incorrect decisions are too costly that any decision must be well justified after a thorough analysis. In accordance with these considerations, a lot of research has focused on facility location and theories have been developed for various facility location problems for over 50 years. An overview of research on facility location can be found in [7] and [51],[52].

Traditional approaches have considered the location problem as a static one and assumed that the location determined according to present conditions would sustain its superiority throughout the lifetime of the facility. However, as an examination of changes in number, location, size and product variety of plants of a sample of 64 small and medium sized multiplant enterprises over 6 years by Healey [34] shows, facilities are subject to various changes among which locational changes have high frequency. Facilities operate in a dynamic environment influenced by technological innovations, cycles of economy, changes in tastes and environmental factors. Demand patterns not only change in the long run due to factors like population shifts, changing income levels, development of competing products, failing to keep customer satisfaction, but also in the short run due to seasonality and promotion campaigns. In addition to demand changes, costs relevant to the location decision like transportation costs, labor cost, taxes, rental costs, are subject to changes that can usually be successfully predicted by forecasting methods. Thus, the dynamics of the problem have to be incorporated into a location decision to prevent a myopic decision that could cause the locations to be unattractive by time. As different locations become optimal at different times, the initial locations may fall to be suboptimal by time. Then, an analysis involving an appropriate time horizon, such as the forecasted life-time of the facility, is required. The decision of either relocating the facilities throughout the time horizon, following the path of the optimal locations, or choosing permanent locations at the beginning of the time horizon which will be close to the optimal locations should be made. This decision depends on the tradeoffs between the costs of relocating the facilities, including non-monetary terms like the resistance of the organizations, and the savings from operating costs as a result of the relocations. If costs appear to overweigh, permanent locations would be chosen and it is worthwhile to try to turn the locations into favorable ones by promotional activities. In some cases, relocations can easily be justified, as in the case of mobile or portable service facilities where the optimal locations could be followed. Thus, knowing exactly what locations will be best in what time intervals is an asset in both situations.

Let us consider mobile facilities serving demand centers scattered in a region over a time horizon, for example, a military base in a war situation in coordination with quarters that might change their locations frequently due to strategic reasons or a tourism information booth that serves demand which shows seasonal variations. It is reasonable then to determine how optimal solutions change in response to the dynamic factors and to follow the optimal path.

Another situation in which relocating the facility over time may be preferred and easily accomplished is that of determining which machines will perform special tasks like control of information routing, storage etc., depending on the changing traffic intensity on a computer and communication network. In this case, relocations are ignorably incostly and extremely manageable.

Even though relocating the facilities could be too costly and problematic, for a decision that considers an appropriate time horizon, knowledge about the optimal locations of the facilities throughout the planning horizon would be helpful. For example, consider warehouses from which non-durable goods are distributed to retail stores in a city. The "rush-hour" behavior of traffic that causes travel times to vary within a day can be fairly well predicted. Obviously, relocating warehouses within a day is out of consideration. However, knowing the optimal trajectory of the locations throughout the day may be useful in choosing a location that would be better off for normal traffic intensity and would not be undesired for rush-hour periods.

1.2 Dynamic Location Problems in the Literature

Against static facility location problems which depict a single-period or static situation, dynamic facility location models which reflect a multiperiod or time dependent situation were posed about a quarter century ago.

Dynamic facility location problems raise the question of *when* to establish and if necessary to replace a productive capacity due to changing conditions in addition to the classical questions of *where* and *what size.*

A dynamic location problem was first studied by Ballou [1] who considered multiperiod warehouse location problem with capacity constraints (1968). He gave an approximate solution by dynamic programming and later Sweeney and Tatham [48] solved the problem to optimality by dynamic programming with an extended state space. Wesolowsky [53] examined multiperiod single facility location-relocation problem. Relocation of the facility is permitted only at the beginning of the periods with the assumption that amount of demand, number of demand points, distribution and relocation costs are constant within a period but may change between periods. Relocation cost is assumed to be independent of the location of the facility before and after relocation, which is actually unrealistic. Wesolowsky gave an incomplete dynamic programming algorithm that finds the optimal locations in each of the r periods by solving $O(r^2)$ static problems. Later, Wesolowsky and Truscott [54] extended this model to a multi-facility location-allocation problem and presented mixed integer programming as well as dynamic programming solutions.

Afterwards, there appeared numerous studies in the literature about discrete time location models with a multiperiod situation where demand varies between time periods, usually with an increasing trend. Thus, capacity expansion was also considered in addition to location and relocation aspects. Such problems are discussed in detail in Chapter 4, "Multiperiod Capacitated Location Models", of *Discrete Location Theory* [42]. A general multiperiod capacitated location model whose solution requires the minimization of a nonlinear, concave function over a polyhedron, is presented by Jacobsen in this chapter. The solution methods are gradient search algorithms and dynamic programming.

Erlenkotter [25] studied uncapacitated dynamic location problems and formulated problems in the private, public and quasi-public sectors, where demands are influenced by prices at various locations so that pricing and location decisions are determined simultaneously. He showed that these problems may be transformed to the fixed-demand location model of Efroymson and Ray which may be solved by available methods. Later, Van Roy and Erlenkotter [45] gave a dynamic uncapacitated facility location model that minimizes total discounted costs for meeting demand over time with the flexibility of opening new facilities and closing existing ones by time. A branch and bound procedure which incorporates a dual ascent method is presented and is claimed to be superior to previous methods. Erlenkotter also gave a continuous time formulation in [24] and suggested a discrete-time approximation heuristic solution.

One recent discrete-time model is Dutta and Lim's [20] multiperiod capacity planning model for computer and communication networks. The model allows traffic among existing nodes to increase, new nodes to be added to the network and its topology to change over time. The model is formulated as an IP and a Lagrangian relaxation based solution method is proposed. Shulman [47] also gave an algorithm based on Lagrangian relaxation technique for solving dynamic capacitated plant location problems with discrete expansion sizes.

Chand [11] examined dynamic location problem from a different aspect and addressed the problem of finding decision and forecast horizons for a single facility dynamic location problem. He defines a *decision horizon* to be the number of periods for which decisions have to be made now. A *forecast horizon* is defined to be the number of periods of forecast data required to ascertain optimality of the initial decisions over the decision horizon and gives a forward procedure to find an optimal initial decision. Bastian and Volkmer [2] claimed that Chand's algorithm is not "perfect" in the sense that it does not determine an optimal initial decision using only data for the minimum forecast horizon. They presented a more general relocation model and a perfect forward algorithm to solve it.

Heuristic methods for multiperiod location problems are discussed in Jacobsen [35] and Erlenkotter [27] compares the performance of 7 approximate methods for locating new capacity over time to minimize the total discounted costs of meeting growing demand at several locations on two real-world problems.

More information on discrete time dynamic location problems can be found in the bibliography prepared by Erlenkotter [23] and a survey paper by Luss [40]. A recent study by Hakimi, Labbé and Schmeichel [31] not only reviews dynamic location problems but also discusses some interesting modeling aspects.

Dynamic facility location models in a continuous time domain are researched by Orda and Rom [43], Drezner and Wesolowsky [19], Campbell [10] and Tansel [50].

Orda and Rom [43] address a single-facility dynamic network location problem within the context of computer and communication networks. As a modeling convenience, the location space is restricted to the nodes of the network and the edge lengths are defined to be positive functions of time. The objective function to be minimized consists of the cost of locating the facility at certain nodes and the cost of switching the facility, if any occurs, throughout the time horizon. General costs are used in the model. The location cost in a time interval in which the facility remains fixed in its location is the integration of the instantaneous cost of having the facility at its location over that time interval, where instantaneous cost is defined by any performance measure. Switching cost depends on the locations before and after the switch as well as the time in which it takes place. The cost is assumed to be non-decreasing with the duration of switch and goes to infinity in the limiting case of zero duration. It is also assumed to be a continuous or piecewise continuous function of time.

Discrete time version of the model is solved by a shortest path algorithm on a network that has as many levels of the original nodes as the number of periods, k. Each node in a level is connected to each node in the next level with appropriate costs determined by location and switching cost functions at this period. A source node with no cost of edges is connected to each node in level k-1 with only appropriate location costs.

For the continuous time model they give an algorithm which first computes the cost of the optimal locations throughout the planning horizon and then calculates the locations. The algorithm is like a limiting shortest path algorithm on a network with levels for each time period. The algorithm stops after a finite number of iterations when there is no change in the cost functions between iterations. No time bound is given for this exact algorithm.

Drezner and Wesolowsky [19] also investigate facility location when demand varies with continuous time in a predictable way. They study single facility minisum problem on a planar location space with weights being nonnegative time functions. The location of the facility can be changed during the time horizon but each relocation incurs a fixed cost. The decision variables are time points at which the facility changes location, i.e. breakpoints and the location of the facility in each time interval between breakpoints. For the special case of rectilinear distances, linear weights and the restriction that only one breakpoint is allowed, the problem is solved using the property that the static problem decomposes into two one-dimensional problems, each with a solution on a demand point, as well as the monotonicity of the optimal location due to linear weights. The algorithm goes as follows. First, the solution for time zero is found and the smallest breakpoint b' among the breakpoints of switching to a pair of adjacent coordinates is identified. Once the locations before and after time point b' are known, $F(b)$, the objective value with these locations and a breakpoint at b, where b is taken as a parameter, is computed. Beginning with the time b_0, which minimizes $F(b)$ in the interval $[0, b']$, the process is repeated until the next smallest breakpoint is beyond the time horizon or does not exist. Then, among b_0's, calculated during the process, the one with minimum $F(b_0)$ is the optimal breakpoint and $F(b_0)$ is the optimal objective value.

When more than one breakpoint is allowed, they propose a heuristic based on a univariate search with breakpoints. They initially divide the time horizon into k segments. Repeatedly, one-breakpoint problem is solved by keeping all but one breakpoint fixed. The procedure stops when no significant changes in breakpoints occur.

The authors also give a minimax version of the weight parametric model. Two algorithms which find the solution to the problem with k relocations is given. Then, a search on k is suggested. The algorithms can be applied to a weight parametric p_center problem on a network with the assumption that a fixed relocation cost is incurred at breakpoints whatever the number of centers that are to be relocated.

Another interesting relocation model in a continuous time domain is given by Campbell [10]. The model incorporates locating transportation terminals in a service region with an expanding demand. Shipments are sent from origins to destinations scattered over this region via one or more terminals since shipments between terminals are less costly. The tradeoffs between savings in transportation costs and terminal relocation and establishment costs determine when and where terminals should be added and relocated.

Campbell analyzes the performance of three strategies. One is relocating terminals as many times as required by the minimization of only transportation and terminal establishment costs. The other two are allowing no relocations and relocation of only one existing terminal when a new terminal is added. Comparison of the strategies in terms of difference from a lower bound suggests that optimal solutions can be reached by limiting the number of relocations when relocation costs are low. With high relocation costs, making no relocations can be preferred, however the solution may not be very close to optimal.

Tansel [50] considers the time dependent version of the single/multi median problem for which weights or distances are functions of time. Various node optimality results are given, a penalty minimization problem is defined and solved. The multi-relocation problem is analyzed by transforming the problem into a rectilinear path problem on time coordinates. By an intermediate value optimality theorem, finitely many points on the plane through which the optimal path must go are identified. Then, the problem becomes that of finding a shortest path on a network whose nodes correspond to the identified points.

Another perspective in the analysis of dynamic facility location problems is parametric analysis which involves determining the optimal locations for certain values of a parameter inherent in the model. While Brandeau and Chiu [6] analyzed single-median problems with an L_p-norm based objective function with parameterization on p, Erkut and Öncü [21] analyzed single obnoxious facility location problem with parametric weights. Erkut and Tansel [22] addressed and analyzed time dependent single-median problems with parametric weights or distances, where the parameter is time. Brandeau and Chiu [9] summarized recent results on parametric analysis of a single facility location for various location problems.

Brandeau and Chiu [6] analyze the optimal location of a single facility on a tree network with the objective of minimizing the sum of weighted *Lp*_norm distances from each node to the facility, as the parameter p changes in the interval [1,∞). As p increases, the cost becomes more sensitive to distance. Thus, parameterization is on customer disutility for travel. For $p = 1$, we have the median problem and for $p = \infty$, the problem becomes unweighted center problem. For fixed p, optimality conditions are given using directional derivatives and the trajectory of the optimal location is constructed by a method that hinges on the convexity of the objective function. It is shown that the optimal trajectory is continuous and need not be monotonic between the median and center locations. The method of finding the trajectory fails for a cyclic network since the objective function need not be convex anymore. It is also shown that nonconvexity can lead to a discontinuous, disconnected trajectory. The conditions for having a continuous trajectory of the optimal location to single facility parametric location problems are given in a separate note by the authors [5].

Erkut and Öncü [21] introduce a parametric version of the weighted 1_maximin problem in a convex polygon, where parameterization on weights ($w_i^{\frac{1}{q}}, 1 \leq q \leq \infty$) creates a range of problems with different importance of weights, from ordinary weighted one to the unweighted version. Optimal trajectory is constructed for two example problems and it is concluded that the trajectory may be discontinuous and radically different optimal locations can be found for different values of the parameter.

Erkut and Tansel [22] analyze the 1_median problem on a tree network with parametric node weights for the cases of weights being linear and nonlinear functions of time. A "connectedness" theorem stating that the trajectory of optimal solutions over the time horizon is connected and is in the form of a subtree, is given for the general case of nonlinear weights. For the problem with linear weights, this subtree reduces to a path between the optimal locations to the problems at the end points of the time horizon. Using the half-sum property, the trajectory of the optimal solution is constructed in O(n) time, n being the number of nodes in the tree. The connectedness result also facilitates the construction of the optimal trajectory of the problem with nonlinear weights. However, the efficiency of the construction method depends on the effort required to solve nonlinear equations and the total number of breakpoints in the trajectory.

Brandeau and Chiu [9] summarize recent results on parametric analysis of the optimal location of a single facility on a network or a planar region. The parametric problems addressed are cent-dian problem introduced by Halpern [32] [33] whose objective function is a linear combination of center and median objectives, problems that use an *Lp*_norm based cost function introduced by Shier and Dearing [46], stochastic location problems with parameterization on the customer call rate [3], [4], [8], [15], [16] and dynamic facility location problems where demands or distances may change over time [19], [21].

With the assumption of unique optimal solution at a fixed value of the parameter and continuity of the trajectory of the optimal location, a method for determining the trajectory is presented by the authors. For the median problem with demands being time functions uniqueness does not hold, thus the method presented is not applicable. Although parametric 1_median problem is analyzed by Erkut and Tansel [22] as explained above, there is no study in the literature for the parametric multimedian problem.

1.3 Scope of the Thesis

In this study we focus on parametric multimedian problems. We examine p_median and p_median with mutual communication problems with weights being linear functions of time both on tree and cyclic networks. We assume that changes in demand or costs over time are predictable. Since linear regression is frequently used for short and medium term forecasts, it may be acceptable to represent future weights as linear functions. For long term forecasts, nonlinear demands can be approximated by piecewise linear functions. Our analysis for the linear case can easily be extended to the piecewise linear case by partitioning the time horizon into intervals each with linear demands.

We construct the trajectory of the optimal solution over the time horizon [0,u), by an algorithm that requires the solution of $O(\tilde{q})$ static p_median problems, where $\tilde{q}$ is the number of linear pieces in the trajectory. In this study, the term "*static problem*", will be taken to mean the relevant problem for a fixed value of the parameter time. When we use the term "*trajectory*", we mean the time path of the objective value of the optimal solution to the problem at a time point, over the time horizon. By "*time horizon*", we mean the continuous time interval for which the optimal locations of the facility are intended to be determined. The term "*breakpoints*" will be used to mean time points at which the optimal locations change by at least one location in the trajectory.

We have implemented the algorithm that constructs the optimal trajectory and analyzed the behavior of some randomly generated problem instances. Trajectories of p_median and p_median with mutual communication problems were analyzed for special type trees like line, star as well as arbitrary trees and cyclic networks with various edge density factors. In the analysis, the emphasis is on tree networks since this study originated to analyze the problems on tree networks and then extended to general networks as well.

The general parametric multimedian problem is introduced in Chapter 2. Chapter 3 examines the case with linear weights. The algorithm to construct the trajectory is also presented in this chapter. Chapter 4 summarizes the design of experiments and results from their analysis. Finally, we conclude and give future research directions in Chapter 5.

Chapter 2

Multimedian Problems on Time Dependent Networks

With the goal of reaching optimal locational decisions, many prescriptive location models have been developed for the last three decades. The literature is huge and rapidly growing. Yet, among numerous formulations, p_median, p_center, uncapacitated facility location and quadratic assignment problems are considered to be basic model types. Among these problems, we concentrate on p_median problem and present its time dependent version in this chapter.

2.1 The Classical p_Median Problem

For a given value of p, the *p_Median Problem* is to find locations of p facilities to be established and to supply each client from a subset of facilities such that the demands of clients are fully met and the costs incurred are minimized. Facilities are *uncapacitated* in the sense that they can serve any number of clients and supply any amount of demand. Nothing is gained from serving a client from more than a single facility, so the problem has the *Single-Assignment Property* (Dominance Property). Other assumptions are that demand is fixed and independent of server location, a single client is served by a single trip and all facilities are identical in the sense that they are operated by one firm or are non-competing. With these assumptions finding only the locations of facilities suffices. Each client is served by the least costly facility.

A general formulation of the p_median problem by a bipartite network is as follows.

Let I denote the set of clients, I = $\{1, \ldots, m\}$

and J denote the set of potential facility sites, J = $\{1, \ldots, n\}$

The bipartite network N_B = (V,E) = (I ∪ J,E) has the sets I and J as its set of nodes and E as the set of edges. Each arc connects a node in I with a node in J.

Let c be a real valued cost vector associated with the edges. If facility j cannot supply client i, $c_{ij} = \infty$. Given the data instance m, n, p and $c = \{c_{ij}\}$, the p_median problem is,

$$\text{p\_MP}: \quad \min_{Q \subseteq J, |Q|=p} \left(\sum_{i \in I} \min_{j \in Q} c_{ij} \right)$$

The p_median problem was first formalized by Weber (1909). His model was a minisum location problem on a *plane*. The term " p_median" was introduced by Hakimi [30] within the context of *network* location. Hakimi assumed that the cost of serving a client from a facility is proportional to deterministic distance between them. His p_median problem is as follows.

Suppose we are given a network N = (V,E) with node set V = $\{v_1,\ldots,v_n\}$ and edge set E. Each edge e in E has a positive length $l(e)$. We denote by L the set of all points of the network. L is our location space. A point x in L is either a node or a point in some edge $e = [v_p, v_q]$ such that it divides the edge into two subedges $[v_p, x]$ and $[x, v_q]$ satisfying $l(\ [v_p, v_q]\ ) = l(\ [v_p, x]\ ) + l(\ [x, v_q]\ )$. The length function l induces a distance d on L which assigns a real value $d(x, y)$ to every pair of points x, y in L. Here, $d(x, y)$ is the length of a shortest path connecting x and y. Note that d defines a distance metric so that it satisfies the following properties for all $x, y \in L$.

1) Nonnegativity : $d(x, y) \geq 0$, $d(x, y) = 0$ iff $x = y$

2) Symmetry : $d(x, y) = d(y, x)$

3) Triangle Inequality : $d(x, u) + d(u, y) \geq d(x, y), \forall u \in L$

Let w_i be a nonnegative weight associated with node v_i, i.e. w_i represents the demand at node v_i. Then, the p_median problem is:

$$\text{p\_M}: \qquad \min_X \left(\sum_{i=1}^{m} w_i \min\{d(x_j, v_i) : x_j \in X\} \right),$$

where $X = \{ x_1, x_2, \ldots, x_p \}$ is a set of p points in L , where facilities are to be established, i.e. a p_median. Actually, Hakimi differentiated between absolute p_medians and vertex p_medians. He called the set X an ***absolute p_median*** when the points are allowed to be any point in L and a *vertex p_median* when the points are chosen among vertices. Then, he showed by his " Node Optimality Theorem " that there exists an absolute p_median whose elements are all vertices. Therefore the distinction is unimportant. We refer to the above problem as the classical p_median problem and to its optimal solution as a p_median.

The p_median problem was discussed in detail by Kariv and Hakimi [36] who showed that the problem of finding a p_median of a general network is NP_hard. However, they gave a polynomial time algorithm to find the p_median when the network is a tree. Generalizations of the p_median problem are discussed in detail in a survey paper by Tansel et. al. [51] and in the 2nd Chapter of *Discrete Location Theory* [42]. At this point we suffice with introducing the problem and delay the discussion of the solution methods until Chapter 4.

2.2 The p_Median with Mutual Communication Problem

The *p_Median with Mutual Communication Problem* is to find the location of p new facilities which are in interaction with each other and the existing n facilities, to minimize the sum of the costs of interaction. The costs of interaction are represented as weighted distances between pairs of new and existing facilities, and pairs of new facilities. Facilities are distinguishable. While existing facilities are distinctly located, new facilities may well be located at the same site. An existing facility can be in interaction with more than one new facility, therefore the single-assignment property is not applicable in this scenario.

The problem was first posed with a *planar* location space by Francis [28]. The solution methods under different distance metrics are discussed in Chapter 5 "Multifacility Location Problems" of *Facility Layout and Location* by Francis, McGinnis and White [29]. The problem on a *network* is defined by Dearing, Francis and Lowe [17] in the presence of distance constraints. The authors show that the problem is a convex optimization problem for all data choices if and only if the network is a tree. For the case of a general network, there exists an optimal solution on the vertices of the network. The problem on a network is formulated as follows:

Suppose our location space L is a network N = (V,E), where the set of vertices V represents the locations of the existing facilities. The function $d(.,.)$, as defined above for the classical p_median problem, denotes the distance between two points on the network which is induced by the edge lengths. We represent nonnegative weight associated with the interaction between the existing facility at node v_i and the new facility at location x_j by w_{ji}, and that of between the new facilities at locations x_j and x_k by v_{jk}. The problem is:

$$\text{p\_MM :} \qquad \min_X \left(\sum_{j=1}^p \sum_{i=1}^n w_{ji} d(x_j, v_i) + \sum_{j=1}^p \sum_{k=j+1}^p v_{jk} d(x_j, x_k) \right)$$

Here, $X = (\ x_1, x_2, \ldots, x_p\ )$ denotes p points in L , where new facilities are to be located. X is a vector since the new facilities are distinguishable. It is possible, for example, to have $x_i = x_j$ for $i \neq j$ so that facilities i and j are located at the same point.

The problem on a *general* network is shown to be NP_hard by Kolen [39], while polynomial time algorithms have been given by Picard and Ratliff [44], and Kolen [38] on a *tree* network. Furthermore, Tamir [49] has shown that the problem is NP_hard even on a tree, if we restrict the location of each new facility to a proper subset of vertices. Although efficient algorithms have been presented for the problem on a tree network, there has not been any efforts on solving the problem on a general network, with the exception of Chhajed and Lowe [13] who gave a polynomial time algorithm when the communication between new facilities can be represented by a series-parallel graph. The solution methods will be discussed in Chapter 4 as the experimental studies are presented.

2.3 Time Dependent p_Median Problems

Time dependency can arise either by weights or edge lengths being functions of time, or both. We first define a general time dependent p_median problem with both weights and edge lengths being functions of time. Then, we focus on the weight parametric case.

The time dependent p_median problem is to find the locations of p facilities to be established so that the costs incurred from supplying changing demands of all clients with changing distances to each other over the time horizon $[0,u]$ is minimized at all time points in $[0,u]$.

Suppose that node weights (w_i(t)), i.e. client demands and edge lengths (l(e,t)) are nonnegative continuous functions of time such that the ratio of a subedge length to the entire edge length is constant for t $\in$ [0,u]. The length function l(.,t) induces a time dependent distance function $d(x,y,t)$ which assigns a real value to every pair of points x,y in L at a time point t. In accordance with the static case, $d(x,y,t)$ is the length of a shortest path between x and y when edge and subedge lengths are fixed at time t. The function $d(.,.,t)$ satisfies nonnegativity, symmetry and triangle inequality properties of a distance function for each fixed t. Thus, $d(.,.,t)$ is a well defined distance function. For $X = \{ x_1, x_2, \ldots, x_p \}$, let $f(X,t)$ be the cost function defined as,

$$f(X,t) = \sum_{i=1}^{n} w_i(t) \min\{d(x_j, v_i, t) : x_j \in X\}$$

With these definitions, the time dependent problem at a time point t is ,

$$\text{p\_M(t)}: \qquad \text{z(t)} = \min_X f(X,t)$$

If we denote the set of p_medians that correspond to the optimal objective value z(t) at time point t by X_p(t), then the time dependent p_median problem is to find X_{opt}, the set of all optimal solutions throughout the time horizon [0,u] , which is defined as

$X_{opt} = \{ X_p(t), t \in [0,u] \}$

Here, z(t) , $\forall t \in [0,u]$ is the trajectory of the optimal objective value.

Tansel [50] has shown that node optimality holds for the time dependent case. The problem can then be restated as

p_M(t) : $z(t) = \min_{1 \leq r \leq q} f(X^r, t)$

where X^r denotes the r-th choice of p nodes and q is the number of all possible choices of p nodes. Construction of z(t) for $t \in [0,u]$ as the pointwise minimum of q functions identifies X_v, the subset of X_{opt} with elements that consist of vertices only, i.e. $X_v = \{ X_p(t) \cap V, t \in [0,u] \}$

We will be focusing on the weight parametric case from now on, with the motivation of modeling situations in which demands are time varying in a predictable way. The two problems that we examine are *Weight Parametric p_Median Problem* and *Weight Parametric p_Median with Mutual Communication Problem.*

The weight parametric p_median problem is the only-weight parametric version of the classical p_median problem and represents situations in which demand varies predictably by time. The formulation of the problem is as follows.

$$f(X,t) = \sum_{i=1}^{n} w_i(t) \min\{d(x_j, v_i) : x_j \in X\}$$

p_MP(t) : $z(t) = \min_{X} f(X,t)$

Using the node optimality property, the problem can be restated as,

p_MP(t) : $z(t) = \min_{1 \leq r \leq q} f(X^r, t) = \sum_{i=1}^{n} w_i(t) \min\{d(x_j^r, v_i) : x_j^r \in X^r\}$

Then, the problem is to find the minimum of q functions where, $q = \binom{n}{p}$. The definitions of $X_p(t)$, X_{opt}, X_v and z(t) are still valid.

Weight parametric p_median with mutual communication problem models the cases in which there is a time varying traffic among the new facilities in addition to the time varying traffic among new and existing facilities. If we denote the nonnegative continuous weight between new facility j and existing facility i by $w_{ji}(t)$ and that of between new facilities j and k by $v_{jk}(t)$, then the problem is :

$$f(X,t) = \sum_{j=1}^{p} \sum_{i=1}^{n} w_{ji}(t) d(x_j, v_i) + \sum_{j=1}^{p} \sum_{k=j+1}^{p} v_{jk}(t) d(x_j, x_k)$$

p_MMP(t) : $\quad z(t) = \min_{X} f(X,t)$

In this problem, we denote the set of optimal location vectors at time t by $X_p(t)$. X_{opt} is the set of all optimal vector solutions throughout [0,u]. If we restate the problem using node optimality property, then the formulation will be,

$$f(X^r,t) = \sum_{j=1}^{p} \sum_{i=1}^{n} w_{ji}(t) d(x_j^r, v_i) + \sum_{j=1}^{p} \sum_{k=j+1}^{p} v_{jk}(t) d(x_j^r, x_k^r)$$

p_MMP(t) : $\quad z(t) = \min_{1 \leq r \leq q} f(X^r,t)$

Then, z(t) is the minimum of q functions, where $q = n^p$ and X_v is the subset of X_{opt} that contains all elements that are vectors of p vertices.

Chapter 3

Medians with Linear Weights : Trajectory Construction

In this chapter we focus on p_median problems with linear weights. As mentioned in the Introduction Chapter, short and medium range forecasts usually show a linear trend. Furthermore, long range forecasts can adequately be approximated by piecewise linear functions. Thus, solving the linear weights case could facilitate locational decisions in more general time dependent settings.

Within the frame of definitions made in the previous chapter, we define the two problems that we will be analyzing in this chapter, *p_median problem with linear weights* and *p_median with mutual communication problem with linear weights*. Development of an efficient divide_and_conquer type algorithm to construct the trajectory is presented step by step.

3.1 p_Median Problems with Linear Weights

The formulations of the weight parametric p_median problems were presented in the previous chapter. We obtain p_median problems with linear weights by inserting linear weights into these formulations.

For the p_median problem with linear weights (p_ML(t)), we take weights as linear functions of the form

$w_i(t) = a_i + b_i * t$, $\forall v_i \in$ V and t $\in$ [0,u] ,

where a_i's and b_i's are given constants which specify a problem instance together with n, p and the network structure. We assume $a_i \geq 0$ and $a_i + b_i u \geq 0$, $\forall i$. It follows then, for $X = \{x_1, x_2, \ldots, x_p\} \subseteq$ L, we have

$$f(X,t) = \textstyle\sum_{i=1}^{n}(a_i + b_i t)\min\{d(x_j, v_i) : x_j \in X\}$$

$$\text{p\_ML(t)} : \qquad \min_X f(X,t), \quad t\in[0,u].$$

Since we know that there exists a choice of p vertices which minimizes $f(X,t)$ at a time point t, we can restrict our solution space to all possible choices of p vertices. The set of all choices of p vertices among n vertices is finite with cardinality $q = \binom{n}{p}$. Suppose that all possible choices are indexed arbitrarily from 1 to q. We denote the set of these indices by Q. With X^r denoting the r-th choice of node restricted p_medians, we have

$$f(X^r, t) = A_r + B_r * t, \text{ where}$$

$$A_r \triangleq \textstyle\sum_{i=1}^{n} a_i D(X^r, v_i)$$

$$B_r \triangleq \textstyle\sum_{i=1}^{n} b_i D(X^r, v_i)$$

$$\text{with,} \quad D(X^r, v_i) \triangleq \min_{1\leq j \leq p} d(x_j^r, v_i)$$

For the p_median with mutual communication problem with linear weights (p_MML(t)), we take weights between pairs of new and existing facilities as linear functions of the form

$$w_{ji}(t) = a_{ji} + b_{ji} * t \text{ , for } (j,i) \in I_1$$

$$\text{with } I_1 \subseteq \{ (j,i) : 1 \leq j \leq p , 1 \leq i \leq n \}$$

and weights between pairs of new facilities as

$$v_{jk}(t) = \alpha_{jk} + \beta_{jk} * t \text{ , for } (j,k) \in I_2$$

$$\text{with } I_2 \subseteq \{ (j,k) : 1 \leq j < k \leq p \}.$$

In this case, the problem data is specified by the constants a_{ji}, b_{ji}, $(j,i) \in I_1$ and α_{jk}, β_{jk}, $(j,k) \in I_2$. Assume $a_{ji} \geq 0$, $a_{ji} + b_{ji}u \geq 0$, $\max\{a_{ji}, a_{ji} + b_{ji}u\} > 0$

$\forall(j,i) \in I_1$ and $\alpha_{jk} \geq 0$, $\alpha_{jk} + \beta_{jk}$u≥ 0, $\max\{\alpha_{jk}, \alpha_{jk} + \beta_{jk}\text{u}\} > 0$ $\forall(j,k) \in I_2$. For $X = (x_1, x_2, \ldots, x_p) \in L^m$, we have

$$f(X,t) = \sum_{(j,i)\in I_1} (a_{ji} + b_{ji}t)d(x_j, v_i) + \sum_{(j,k)\in I_2} (\alpha_{jk} + \beta_{jk}t)d(x_j, x_k)$$

There exists a solution vector on vertices so that we can restrict the solution space to all vectors of size p whose elements are vertices. There are $q = n^p$ such vectors. Suppose that all such vectors are indexed arbitrarily from 1 to q. We denote the set of these indices by Q. With X^r denoting the r-th vector, we have

$$f(X^r, t) = A_r + B_r * \text{t}$$

$$\text{p\_ML(t)} : \qquad \min_X f(X,t), \quad \text{t}\in[0,\text{u}], \text{ where}$$

$$A_r \triangleq \textstyle\sum_{(j,i)\in I_1} a_{ji}d(x^r_j, v_i) + \sum_{(j,k)\in I_2} \alpha_{jk}d(x^r_j, x^r_k)$$

$$B_r \triangleq \textstyle\sum_{(j,i)\in I_1} b_{ji}d(x^r_j, v_i) + \sum_{(j,k)\in I_2} \beta_{jk}d(x^r_j, x^r_k).$$

Then, multimedian network location problems with linear weights can be represented as

$$\text{PL(t)} : \qquad \text{z(t)} = \min_{r\in Q} f(X^r, t) = A_r + B_r * \text{t}.$$

For simplicity of notation, we define

$F_r(\text{t}) \triangleq f(X^r, t)$. Thus, $\text{z(t)} = \min_{1\leq r\leq q} F_r(\text{t})$.

With z(t) being the lower envelope (pointwise minimum) of q linear functions over the interval [0,u], we have the following lemma.

Lemma 1 : z(t) is piecewise linear concave function of t with at most q pieces.

Let $\tilde{q}$ be the number of pieces of z(t) and $\tilde{Q}$ be the set of $i \in Q$ for which $F_i(\text{t})$ is a nondegenerate piece of z(t) in [0,u]. Clearly, $\mid \tilde{Q} \mid = \tilde{q}$ and $\tilde{q} \leq q$. Let $B = \{b_1, b_2, \ldots, b_{\tilde{q}-1}\}$ be the set of breakpoints of z(t) and B' be the extended set which is the union of set B, breakpoints, and the end points of the interval

[0,u]; that is,

$B' = \{b_0, b_1, \ldots, b_{\tilde{q}-1}, b_{\tilde{q}}\}$ with $b_0 = 0$ and $b_{\tilde{q}}$ =u. We assume that the indexing is done in such a way that $b_0 < b_1 < b_2 < \ldots < b_{\tilde{q}-1} < b_{\tilde{q}}$.

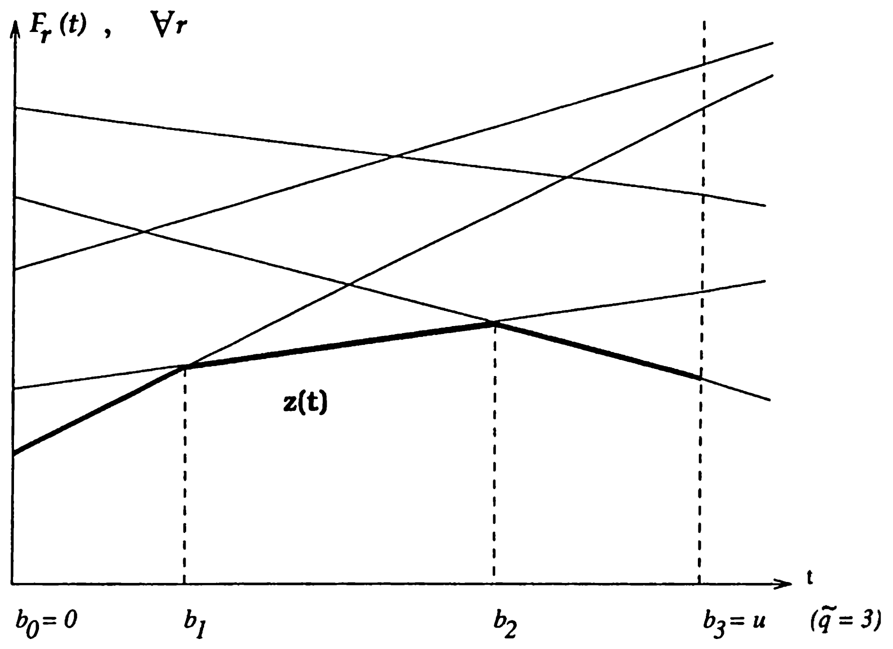


Figure 3.1. An example to the trajectory, z(t)

In general, the lower envelope of q linear functions can easily be constructed in $O(q^2)$ time. It may also be possible to do the same task with better time complexity, such as $O(q \log q)$. This requires knowing the two parameters of the linear functions, the intercepts and the slopes. In our case, the parameters are A_i, B_i, $\forall i \in Q$. Since $q = \binom{n}{p}$ or n^p, the computation of all A_i, B_i and the construction of the lower envelope z(t) via known methods takes *at least* $O(\binom{n}{p} \log \binom{n}{p})$ or $O(n^p \log n^p)$ time.

We propose, however, a much more efficient method that avoids *a priori* computation of the parameters A_i, B_i which are exponential in number. We construct the lower envelope by solving $O(\tilde{q})$ static problems and compute only $O(\tilde{q})$ of the parameters A_i, B_i on a need basis during the construction process. If the static problem can be solved in $O(g(n,p))$ time, then the proposed method constructs the trajectory of z(t) in $O(\tilde{q}g(n,p))$ time. It is evident from our computational studies presented in Chapter 4, that $\tilde{q}$ is typically much

much smaller than q for many instances. (e.g. among the 1350 randomly generated p_MML(t) problems, maximum $\tilde{q}$ turned out to be 15 for the problem with $n = 60$ and $p = 30$, where $q = 60^{30} = 2.21\text{e}{+}53$.) Therefore, trajectory can be constructed in reasonable time for problems of large sizes, even though q is extremely large. (e.g. 10 minutes for p_MML(t) problems with $n = 100$ and $p = 95$ and 1 hour for p_ML(t) problems with the same values of n,p on tree networks.)

At this point two questions arise. One is, whether $\tilde{q}$ ever equals q or there is a bound on $\tilde{q}$ which is significantly smaller than q which also explains the immense gap between q and $\tilde{q}$ in our experimental studies. We present two bounds on $\tilde{q}$ after discussing our trajectory construction algorithm. However, there is still a huge gap between these bounds and realizable values of $\tilde{q}$ in practice, the reasons of which remain to be explored. The other question is, what the least order of constructing the trajectory could be. To explore the answer to the second question, let us define the sets, Z, Z_B, and S as follows.

$$\begin{aligned}
Z &\triangleq \{\, (t, z(t)) \;:\; t \in [0,u] \,\} \\
Z_B &\triangleq \{\, (b_i, z(b_i)) \;:\; b_i \in B' \,\}\text{and} \\
S &\triangleq \{\, (t, X(t)) \;:\; f(X(t),t) \;=\; z(t),\; t \in [0,u] \,\}.
\end{aligned}$$

In our context, what we mean by constructing trajectory is finding the sets Z and S. Clearly, it suffices to know Z_B to generate Z. Observe that finding each z(b_i) requires solving a static problem, since z(b_i) is defined as

$$z(b_i) = \min_X f(X, b_i), \quad \text{for } 0 \le i \le \tilde{q}.$$

Hence, construction of Z_B requires solving $\tilde{q} + 1$ static problems. Knowing the solutions to these static problems, i.e. $X(b_i)$'s, is not sufficient to construct the set S since a solution $X(b_i)$ may not be optimal at any time point different than b_i. To find a solution that is optimal between two breakpoints b_{i-1} and b_i, a static problem solution is required at a time point t $\in (b_{i-1}, b_i)$. Thus, the least number of static problems to be solved to construct the trajectory is $2\tilde{q} + 1$. The method we propose in this thesis constructs the trajectory by solving at most $4\tilde{q}$-2 ($O(\tilde{q})$) static problems. Therefore, we may conclude that our construction algorithm is a best order algorithm unless there exists a more efficient method of identifying optimal solutions at a time point than solving a static problem.

3.2 Trajectory Construction

We start this section by developing some preliminary information that leads to an efficient algorithm to construct the trajectory.

3.2.1 Preliminaries

We could construct the trajectory by solving $2\tilde{q}+1$ static problems as explained above, if we knew the set of breakpoints, B. However, the set B is *a priori* unknown. A breakpoint is identified as the intersection point of two adjacent linear pieces of z(t), i.e. the intersection point of $F_i(t)$ and $F_j(t)$, for some $i, j \in \tilde{Q}$. The intersection point of the functions $F_i(t)$ and $Fj_(t)$ can be identified as, $t = \frac{A_i - A_j}{B_j - B_i}$ whenever $B_i \neq B_j$. Let $T = \{ t : t \in$ (0,u) and $\exists i, j \in Q$ such that $B_i \neq B_j$ and $t = \frac{A_i - A_j}{B_j - B_i} \}$. With this definition, T is the set of time points in (0,u) at which two (or more) functions intersect. We call T the set of intersection points. T has at most $q(q-1)/2$ elements. Observe that each breakpoint is supplied from T; that is, $B \subseteq T$.

Our trajectory construction algorithm solves static problems at points which are those elements of T that are candidates to be breakpoints. Even though T has $O(q^2)$ elements, the algorithm correctly selects $O(\tilde{q})$ elements as candidates thereby eliminating a large number of intersection points from candidacy.

At an intersection point, more than two linear functions may intersect. If the objective value induced by intersecting functions happens to be optimal, then this point is a breakpoint. We need to find the functions that are optimal to the left and right of that breakpoint to correctly identify the solutions in [0,u]$-B$. Figure 3.2 depicts the situation.

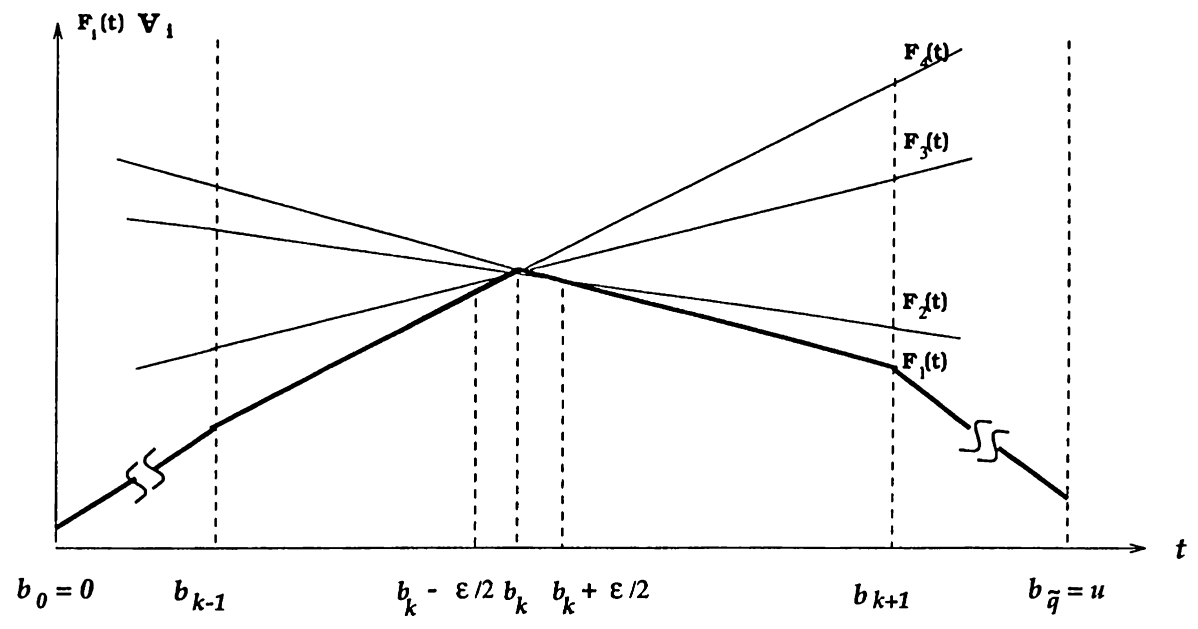


Figure 3.2. Alternate solutions at a breakpoint

In Figure 3.2, among all alternate solutions at t $= b_k$, we want the one that remains to be optimal at t $= b_k + \varepsilon/2$ for t $> b_k$ and at t $= b_k - \varepsilon/2$ for t $< b_k$, where ε is a small positive real number. We should use such an ε that there should not be any breakpoints in the intervals $[b_k - \varepsilon/2, b_k)$ and $(b_k, b_k + \varepsilon/2]$.

Let $\ell = \min_{2 \leq i \leq \tilde{q}} (b_i - b_{i-1})$ and $\ell' = \min_{\tau,\tau' \in T, \tau < \tau'} (\tau' - \tau)$. Thus, ℓ and ℓ' specify, respectively, the minimum distance between any two distinct members of B and T. Note that $B \subseteq T$ implies $0 < \ell' \leq \ell <$u. Since the breakpoints and intersection points are computed only when needed, the sets B and T are not available at the beginning. Hence, ℓ and ℓ' are not a priori computable. However, it is possible to calculate a lower bound on ℓ and ℓ' in terms of the problem data. The computation of such a lower bound will be given at the end of this chapter.

Lemma 2 : Let LB be a lower bound such that $0 < LB \leq \ell' \leq \ell$. If we choose ε from (0,LB),

i) there can be at most one breakpoint, i.e. an element of B, in a time interval of length ε.

ii) there can be at most one intersection point, i.e. an element of T, in a time interval of length ε.

Statement i) follows from $LB \leq \ell$ and ii) from $LB \leq \ell'$. Using Lemma 2, we can easily identify solutions that remain optimal on both sides of a breakpoint. To avoid repeating the long expression, we call the solution that remains to be optimal to the immediate left (right) of a breakpoint as "left (right) optimal solution" and use the term "left_right optimal solutions" when we refer to both of such solutions.

Another lemma that we will be using in our algorithm is a consequence of Lemma 1.

Lemma 3 : Let $X(l)$ and $X(u)$ be the optimal solutions to the static problems at the end points of the interval $[l,u]$. If $f(X(l),u) = f(X(u),u)$, then z(t) = $f(X(l),t)$ for all t $\in [l,u]$. Similarly, if $f(X(l),l) = f(X(u),l)$, then z(t) = $f(X(u),t)$ for all t $\in [l,u]$.

Proof: The fact that for all $X \subset$ V, $\mid X \mid$ = p,

$$f(X(l),l) \leq f(X,l)$$

$$f(X(l),u) \leq f(X,u)$$

implies, together with linearity of f in t for fixed X that

$$f(X(l),t) \leq f(X,t), \ \forall t \in [l,u]. \ \square$$

We had stated that to construct the trajectory we solve static p_median problems at certain intersection points $t \in T$. Let i,j be distinct indices in Q such that $t = t_{ij} \triangleq \frac{A_i - A_j}{B_j - B_i}$ with $B_i \neq B_j$ and $0 < t_{ij}$ <u. When we solve the p_median problem at an intersection point t_{ij}, we may have one of the following possibilities.

Possibility I : t_{ij} is a breakpoint

Two subcases arise depending on if $F_i(t_{ij}) = F_j(t_{ij}) = z(t_{ij})$ or not.

Ia) X^i and X^j are optimal at t = t_{ij}

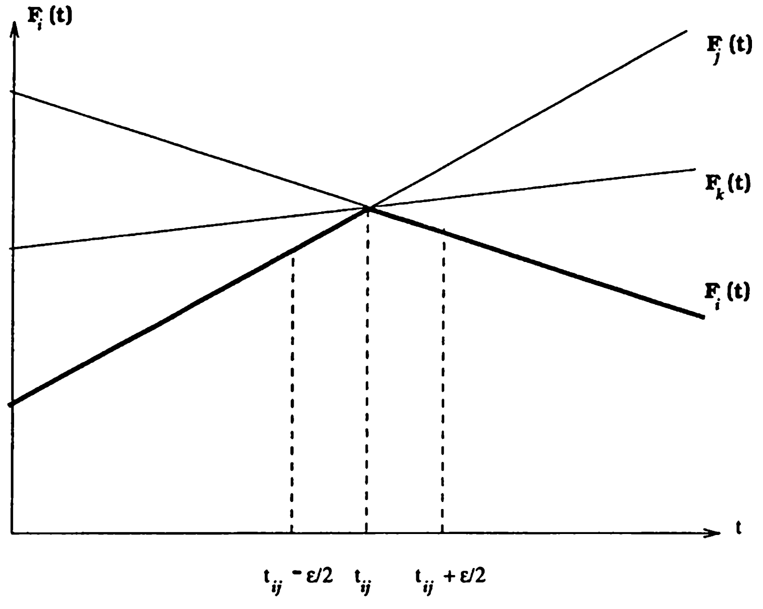


Figure 3.3. Possibility Ia

X^i and X^j are optimal at t = t_{ij}. F_i(t) is not equal to F_j(t) for t different than t_{ij} so that t_{ij} is a breakpoint of the optimal trajectory.

Since there can be only one breakpoint in an interval of length ε, we know that there cannot be any other breakpoints in the interval [t_{ij} - $\varepsilon/2$,t_{ij} + $\varepsilon/2$]. The solution at t = t_{ij} - $\varepsilon/2$ is the left optimal solution at t = t_{ij}, and correspondingly, the solution at t = t_{ij} + $\varepsilon/2$ is the right optimal solution at t = t_{ij} and they are both optimal at t = t_{ij}. Note that the left_right optimal solutions may or may not be supplied from $\{X^i, X^j\}$. (In Figure 3.3, they are).

Ib)X^i **and** X^j **are not optimal at t =** t_{ij}

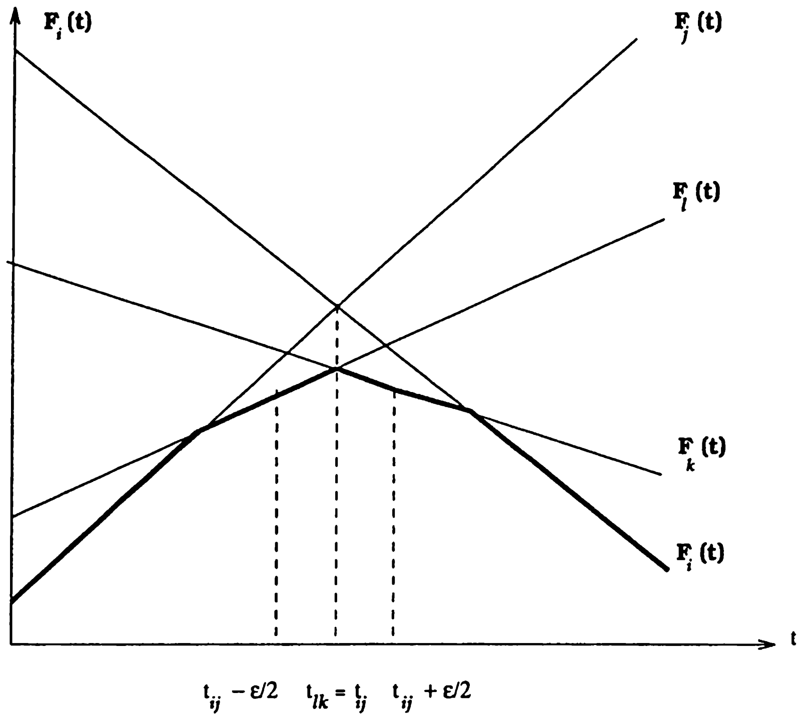


Figure 3.4. Possibility Ib

X^i and X^j are *not* optimal at t = t_{ij}. However, there is a breakpoint of the optimal trajectory corresponding to optimal solutions at t = t_{ij}. $F_i(t_{ij}) = F_j(t_{ij}) >$ z(t_{ij}) implies that there exist indices l, k with $\{i,j\} \cap \{l,k\} = \emptyset$ so that the optimal solutions at t = $t_{ij} = t_{lk}$ are X^l, X^k. In this case, the solutions at t = t_{ij} - $\varepsilon/2$ and t = $t_{ij} + \varepsilon/2$ are the left_right optimal solutions at t = t_{ij}.

Possibility II : t_{ij} is *not* a breakpoint

Two subcases arise depending on if there is a breakpoint within an $\varepsilon/2$ neighborhood of t_{ij} or not.

IIa) No breakpoint in the $\varepsilon/2$ neighborhood of t_{ij}

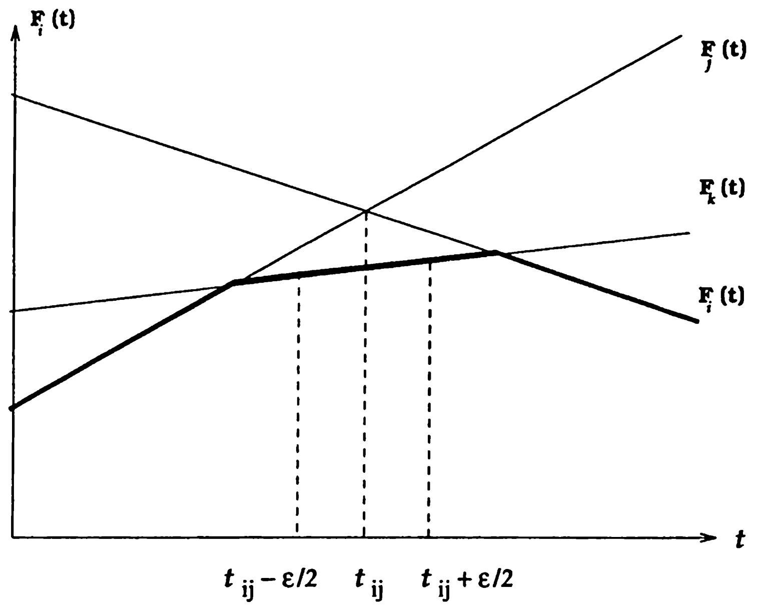


Figure 3.5. Possibility IIa

X^j and X^i are *not* optimal at t = t_{ij}. Neither do we have a breakpoint of the optimal trajectory in the interval [t_{ij} - $\varepsilon/2$,t_{ij} + $\varepsilon/2$].

The optimal solution at t = t_{ij}, namely X^k is optimal both at t = t_{ij} - $\varepsilon/2$ and t = t_{ij} + $\varepsilon/2$. Due to Lemma 3, we can conclude that X^k is optimal in the interval [t_{ij} - $\varepsilon/2$,t_{ij} + $\varepsilon/2$] and we do not have any breakpoints in this interval.

IIb) There is a breakpoint in the $\varepsilon/2$ neighborhood of t_{ij}

X^j and X^i are *not* optimal at t = t_{ij}. We have X^k as the optimal solution. However, this solution is not optimal throughout $[t_{ij}$ - $\varepsilon/2$,t_{ij} + $\varepsilon/2]$. Therefore, there is a breakpoint either on the left or on the right of t_{ij} but not both since we cannot have two breakpoints in an interval of length ε (Lemma 2).

IIbi : There is a breakpoint to the right of t_{ij}

In this subcase the solution X^k is still optimal at t = t_{ij} - $\varepsilon/2$ but another solution, X^l in Figure 3.6, is optimal at t = t_{ij} + $\varepsilon/2$. So, there is a breakpoint in the interval $(t_{ij}, t_{ij} + \varepsilon/2)$.

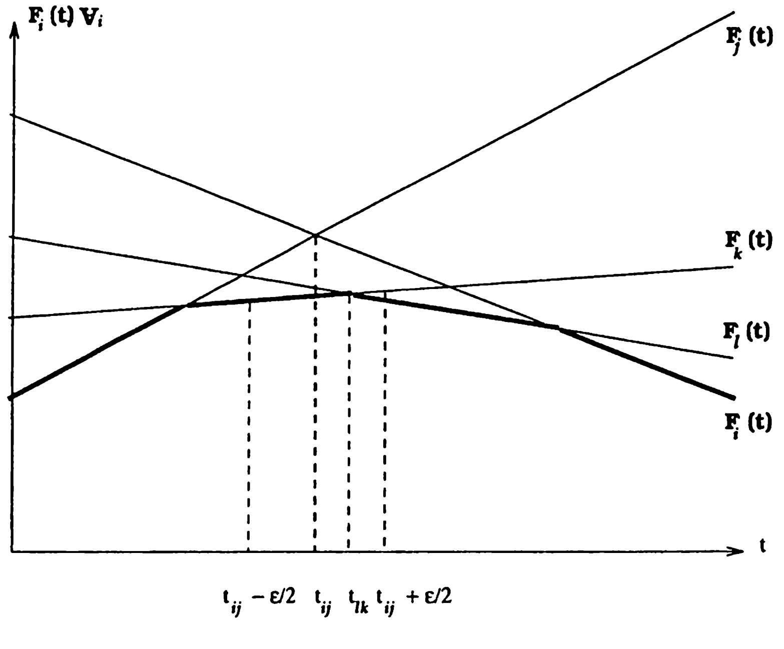


Figure 3.6. Possibility IIbi

IIbii : There is a breakpoint to the left of t_{ij}

In this subcase the solution X^k at t = $t_{ij} + \varepsilon/2$ is optimal but another solution, X^l in Figure 3.7, is optimal at t = t_{ij} - $\varepsilon/2$. So, there is a breakpoint in the interval $(t_{ij} - \varepsilon/2, t_{ij})$.

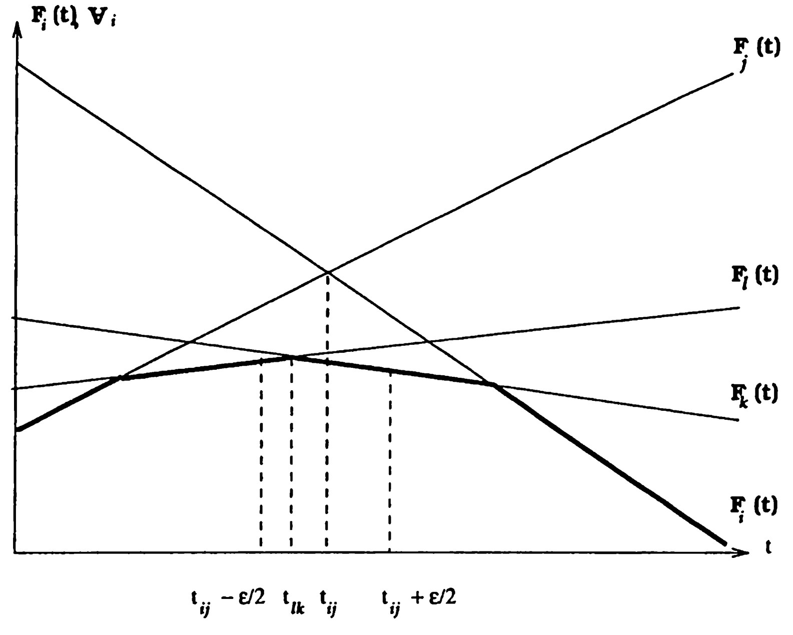


Figure 3.7. Possibility IIbii

In both subcases of IIb, we have exactly one breakpoint in the interval (t_{ij} - $\varepsilon/2$,t_{ij} + $\varepsilon/2$).

In all of the above explained possibilities, we can decompose the problem into two by dividing the time interval [0,u] into two subintervals: $[l, t_{ij}]$ and $[t_{ij}, u]$.

All of the above possibilities may arise if we are using an ε from (0,ℓ). However, we will not have Possibility II b) if ε is chosen from (0,ℓ') or $(0, LB)$ because there cannot be more than one intersection point, i.e. an element of T, in an interval of length ε when $\varepsilon < LB \leq \ell \leq \ell'$. If we exclude Possibility II b) for a moment, then for all of the remaining cases the course of action is the same. We find the left solution X^- and $f(X^-, t)$ at t = $t_{ij} - \varepsilon/2$ and the right solution X^+ and $f(X^+, t)$ at t = $t_{ij} + \varepsilon/2$. Then we divide the current interval [l,u] into two subintervals $[l, t_{ij}]$, $[t_{ij}, u]$ and continue with the newly generated intervals in like manner.

If we use $\varepsilon \in (0, \ell)$, we would have Possibility II b). In Possibility II b) i, we find the intersection point of F_l(t) and F_k(t), namely t_{lk}. Then, we find the solution at t $= t_{ij} + (t_{lk} - t_{ij})/2 = (t_{lk} + t_{ij})/2$ to determine the left_right optimal solution at t $= t_{ij}$. Using Lemma 2 and the fact that $(t_{lk} - t_{ij}) < \varepsilon/2$, we can conclude that the solution at t $= (t_{lk} + t_{ij})/2$ is optimal in $[t_{ij}, t_{lk}]$. We divide the problem into two corresponding to intervals $[l, t_{ij}]$ and $[t_{lk}, u]$.

Similarly, in Possibility II b) ii, we find the breakpoint of F_l(t) and F_k(t), namely t_{lk}, find the solution at t $= t_{lk} + (t_{ij} - t_{lk})/2 = (t_{ij} + t_{lk})/2$ and conclude that this solution is optimal in the interval $[t_{lk}, t_{ij}]$. We divide the problem into two as that of finding the trajectory in intervals $[l, t_{lk}]$ and $[t_{ij}, u]$.

There only remains the course of action to be taken at the end points of the interval [0,u] to be investigated. Let X^j be an optimal solution found at 0 (or u). When we are solving a static problem at the end points, we have one of the three possibilities shown in Figure 3.8.

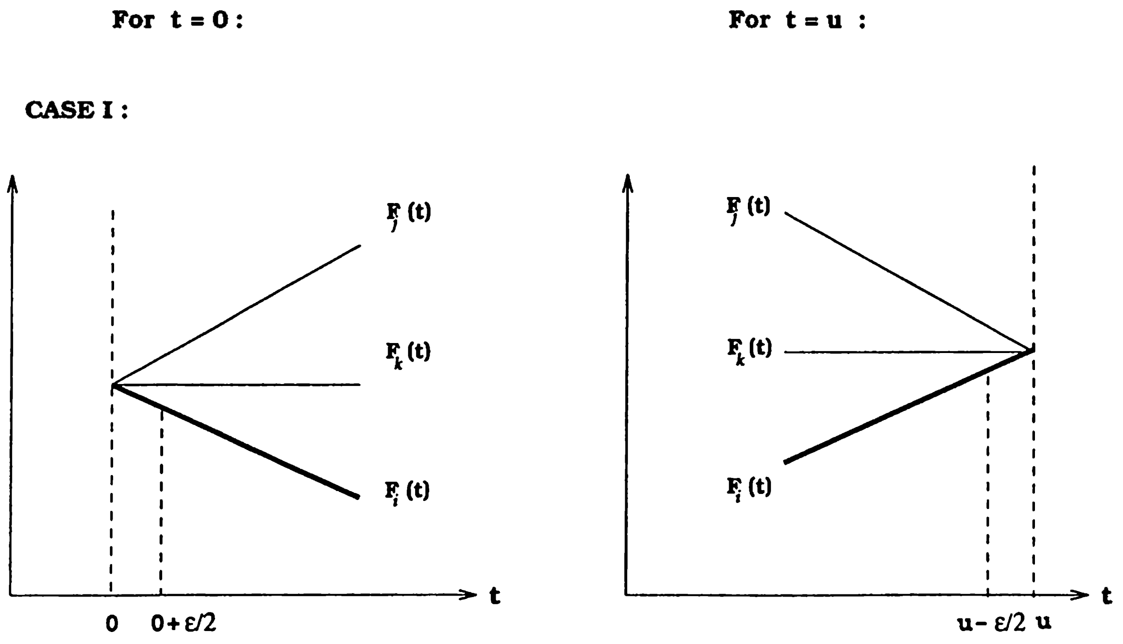


Figure 3.8. Possibilities at end points (continues on the next page)

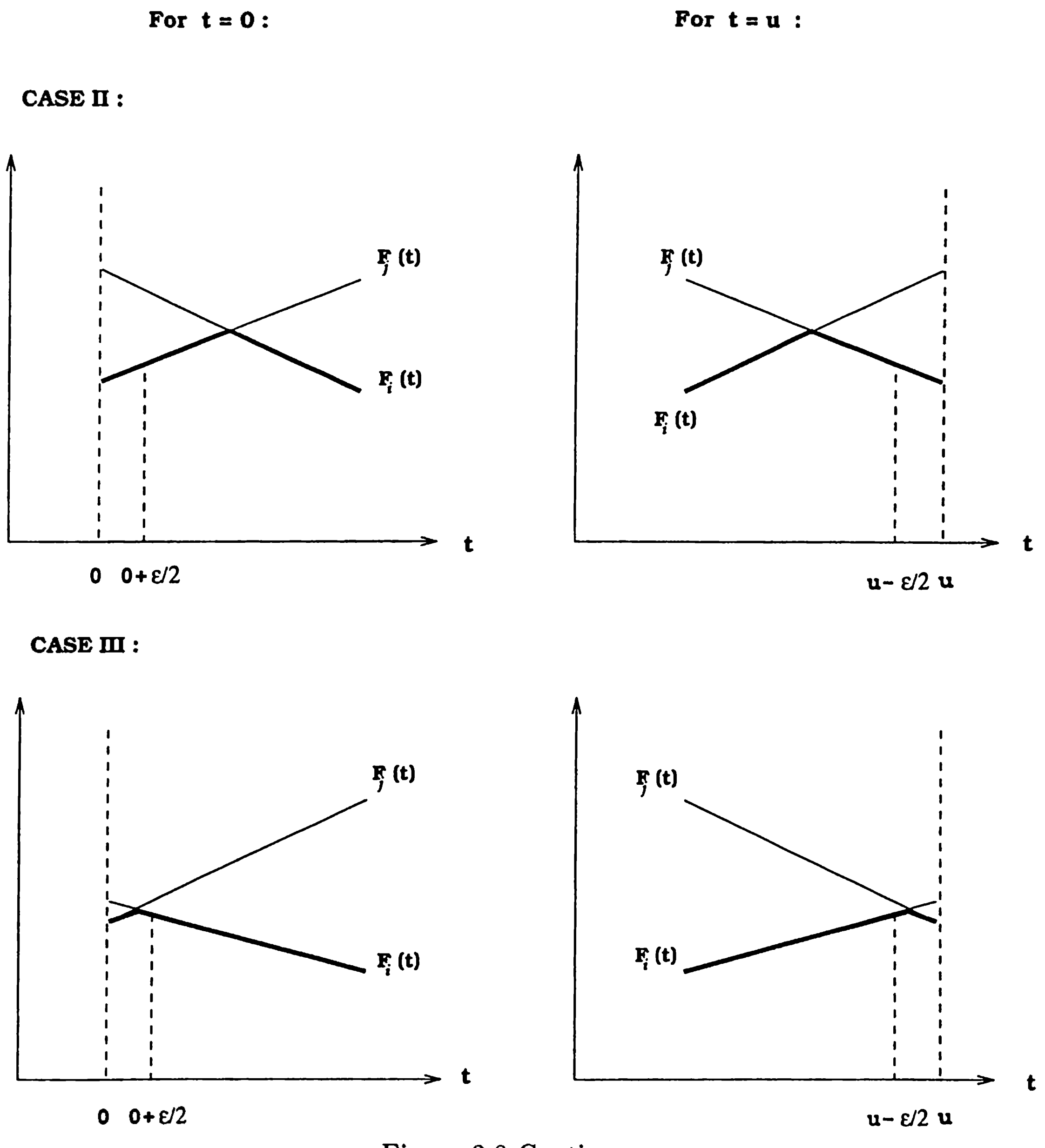


Figure 3.8 Continues

As is evident from the figures, we need to compare the objective values of solutions at t = 0 and t = $\varepsilon/2$ or t = u and t = u-$\varepsilon/2$. If they are the same, then the solution at t = $\varepsilon/2$ is the right optimal solution at t = 0 or the solution at t = u-$\varepsilon/2$ is the left optimal solution at t = u (Cases I and II). If they are different, we take the course of action that we would take for Possibility IIb) i at t = 0 and Possibility II b) ii at t = u (Case III). All three cases are possible regardless of how we choose ε (from (0,LB) or (0,ℓ), as there is no bound on the length between a breakpoint and an end point of the time interval [0,u]).

Based on the above analysis, the proposed algorithm constructs the trajectory of p_median problems with linear weights by solving O($\tilde{q}$) static multimedian problems. The algorithm is general in the sense that we could find the trajectory of p_ML(t) and p_MML(t) problems both on trees and cyclic networks with the algorithm. Furthermore, if we can find a lower bound to the length between two adjacent breakpoints and if we have an algorithm to find the minimum function at a fixed time point in O(r) time (r is a function of the problem input), we could find the lower envelope of exponentially many induced linear functions in O(r$\tilde{q}$) time using this algorithm.

3.2.2 The Algorithm

We will first explain the algorithm on a small example and then give the formal statement. Let us consider a p_MML(t) problem on a tree network with n = 9 existing facilities and p = 4 new facilities to be located over the interval [0,100].

Weights between new facilities, v_{jk}(t) = $\alpha_{jk} + \beta_{jk}$* t :

j\k	1	2	3	4
1	0+0t	0+3t	1+3t	0+0t
2	0+3t	0+0t	5+t	2+2t
3	1+3t	5+t	0+0t	4+t
4	0+0t	2+2t	4+t	0+0t

I_2 = { (1,2),(1,3),(2,3),(2,4),(3,4)}

Weights between new and existing facilities, w_{ij}(t) = $a_{ij} + b_{ij}$* t :

j\i	1	2	3	4	5	6	7	8	9
1	6+t	0+0t	5+2t	3+2t	0+0t	0+4t	0+0t	0+0t	0+0t
2	0+0t	3+t	0+0t	3+2t	5+t	0+3t	0+0t	0+0t	0+0t
3	0+0t	0+0t	0+0t	0+0t	2+2t	0+0t	1+t	0+3t	2+3t
4	0+0t	0+0t	0+0t	0+0t	0+0t	2+t	5+2t	0+0t	3+2t

I_1 = { (1,1),(1,3),(1,4),(1,6),(2,2),(2,4),(2,5),(2,6),

(3,5),(3,7),(3,8),(3,9),(4,6),(4,7),(4,9)}

The tree network on which facilities are to be located is as follows with numbers on edges denoting edge lengths.

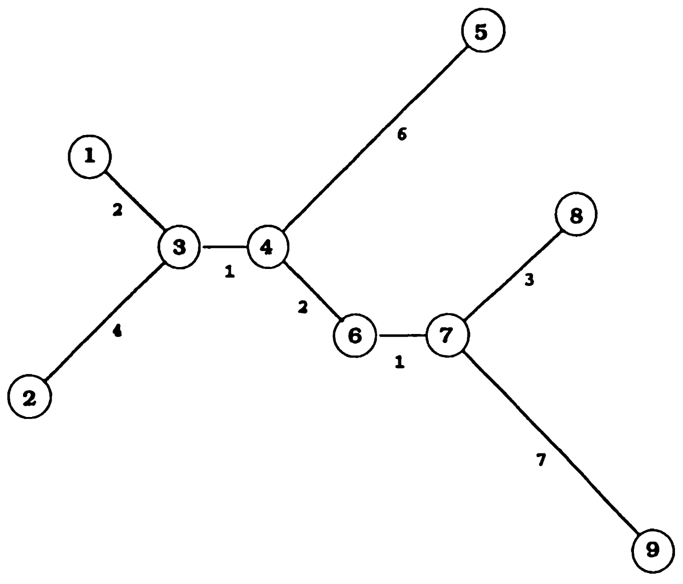


Figure 3.9. Tree structure of the example problem

To find the solution at a particular time point, we solve the static problem that arises when we calculate the weights at that time point. We use Kolen's algorithm for p_median with mutual communication problem on tree networks [38] to solve the static problems.

The trajectory construction algorithm goes as follows:

As preprocessing, we compute distances and LB, and choose $\varepsilon = 1.0596 * 10^{-7}$ from (0,LB).

We find the solutions at t = l = 0 and t = 0 + $\varepsilon/2$: $X(0)$ = (3,4,4,7). Since the solutions at t = 0 and t = 0 + $\varepsilon/2$ are the same, we can conclude that the right optimal solution at t = 0 is $X(0)$. We compute the time dependent objective value corresponding to $X(0)$, i.e. $F(X(0),t)$ = 137 + 126*t. By $X(t)$, we denote the left \ right optimal solution at t.

Then, we find the solutions at t = u = 100 and t = 100 - $\varepsilon/2$. The two solutions turn out to be $X(100)$ = (6,6,6,7). After calculating $F(X(100),t)$ = 181 + 103*t, we compute the time point at which the two objective functions, $F(X(0),t)$ and $F(X(100),t)$ intersect. We denote this point as t_{lu}. (Figure 3.10)

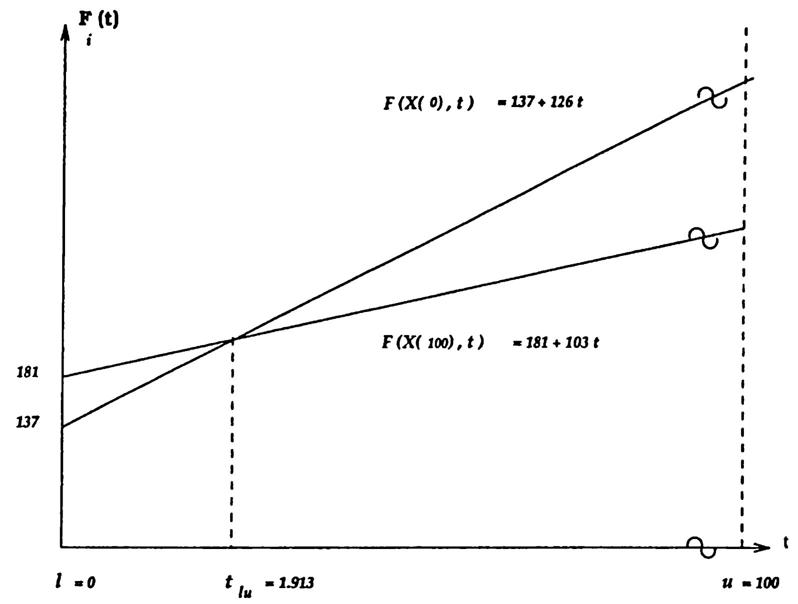


Figure 3.10. Example problem, step 1

Next, we concentrate on the interval [0,t_{lu}] = [0,1.913]. Since we already know the solution at t = 0, we only find the solution at t = u - $\varepsilon/2$ = 1.913 - $\varepsilon/2$: $X(1.91)$ = (4,4,7,7). $F(X(1.91),t)$ = 147 + 111*t. Then, we find the intersection point of $F(X(0),t)$ and $F(X(1.91),t)$, t_{lu} = 0.667 (Figure 3.11).

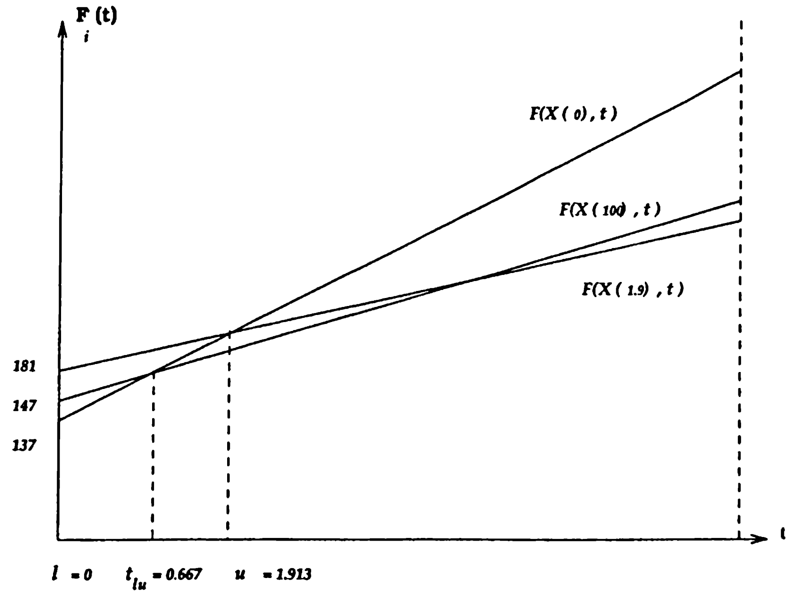


Figure 3.11. Example problem, step 2

Now, we concentrate on the interval [0,0.667]. The solution at t = 0.667 - $\varepsilon/2$ is $X(0.67)$ = (3,4,7,7). $F(X(0.677),t)$ = 140 + 120*t. The intersection point of $F(X(0),t)$ and $F(X(0.67),t)$ is t_{lu} = 0.500. Next we restrict ourselves to the interval [0,0.5]. At t = 0.5 - $\varepsilon/2$, we find the solution $X(0.5)$ = (3,4,4,7) with $F(X(0.5),t)$ = 137 + 126*t. This solution is the same as $X(0)$. Using Lemma 3 , we conclude that $X(0)$ is optimal in [0,0.5] (Figure 3.12).

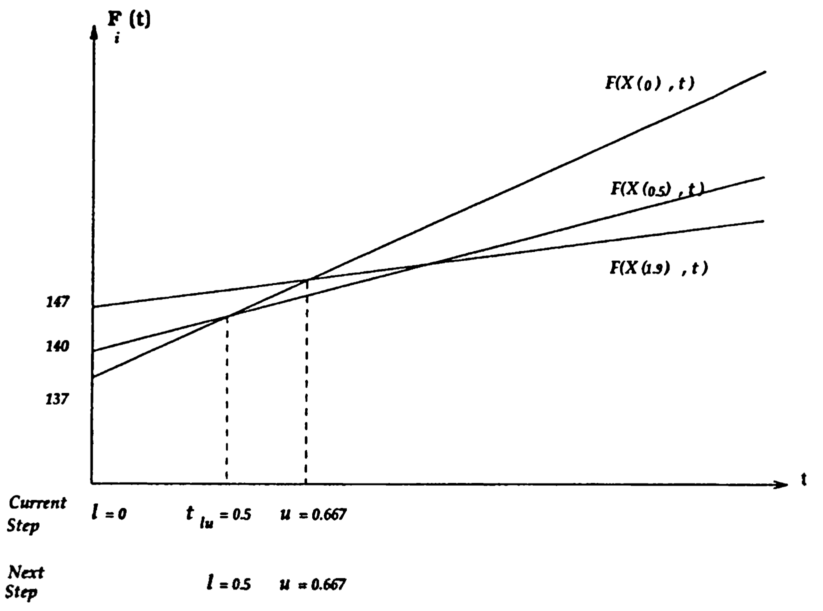


Figure 3.12. Example problem, steps 3 and 4

Now, we consider the interval [0.5,0.667]. We compute the solution at t = 0.5 + $\varepsilon/2$: (3,4,7,7). This solution is the same as $X(0.67)$, therefore we conclude that $X(0.67)$ is optimal in the interval [0.5,0.667].

The next interval is [0.667,1.913]. We find the solution at t = 0.667 + $\varepsilon/2$: (3,4,7,7). We find the intersection point of $F(X(0.67),t)$ and $F(X(1.91),t)$, t_{lu} = 0.778.

Our next interval is [0.667,0.778]. We find the solution at t = 0.778 - $\varepsilon/2$: (3,4,7,7). This solution is the same as $X(0.67)$, thus we conclude that $X(0.67)$ is optimal also in the interval [0.667,0.778].

Next we process [0.778,1.913] and find that $X(1.91)$ is optimal in this interval. Then, we proceed with the interval [1.913, 100] and find the breakpoint t = 4.25. The algorithm goes on like this and stops after processing the interval

[4.25,100] since no more intersection points are found and the end point of the time horizon is reached. The optimal trajectory in the time horizon [0,100] is shown in Figure 3.14.

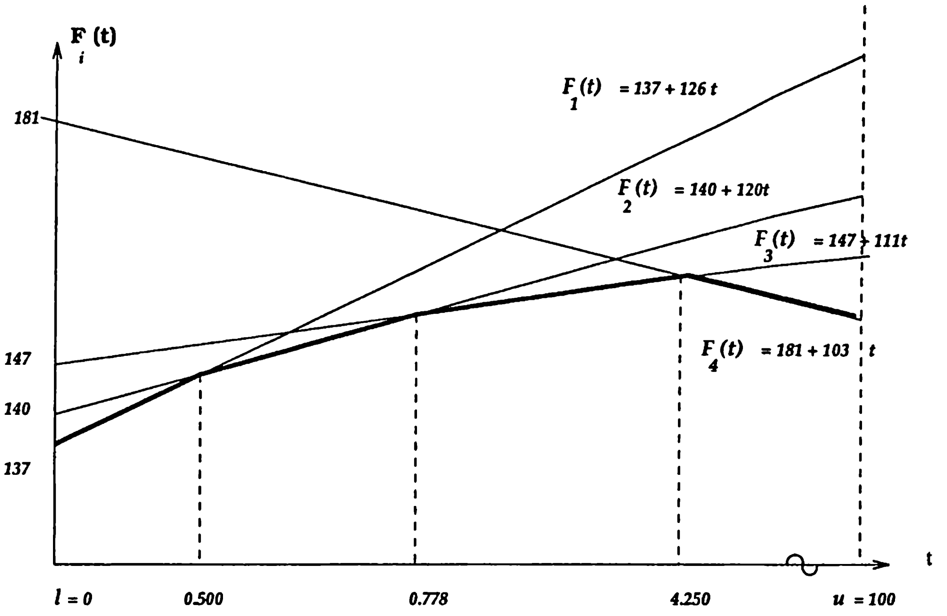


Figure 3.13. Example problem, trajectory

The figure is not accurate as it is not drawn to scale, but it roughly describes the trajectory which is composed of 4 linear pieces and three breakpoints t = 0.5, t = 0.778 and t = 4.25. The solution can be represented on the network as follows.

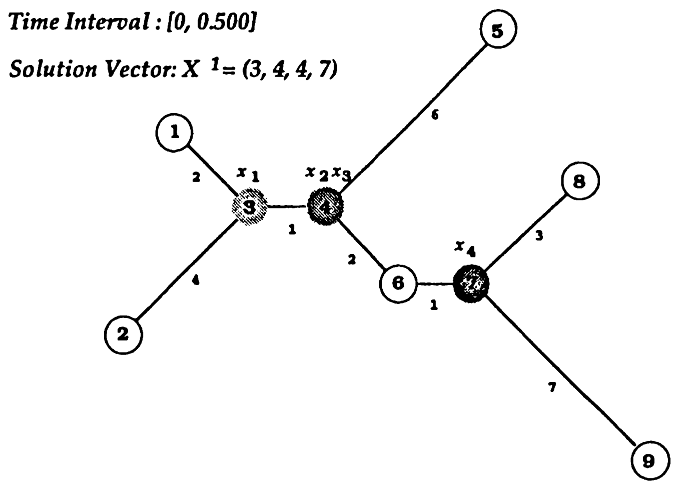
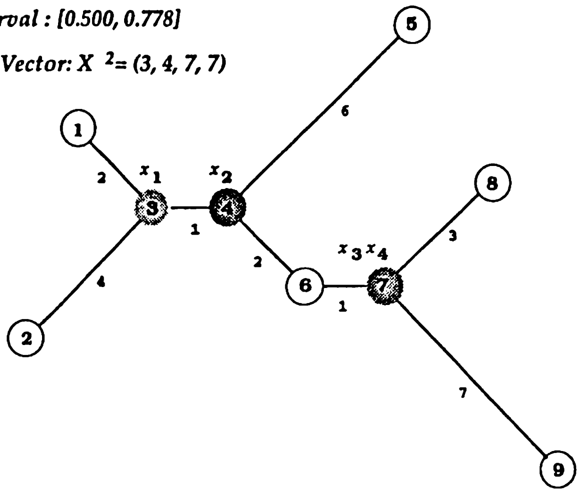

Time Interval : [0, 0.500]
Solution Vector: $X^1 = (3, 4, 4, 7)$
1
2
3
4
5
6
7
8
9
x_1
$x_2 x_3$
x_4
2
4
1
6
2
1
3
7

Figure 3.14. Solution on the network (continues on the next page)

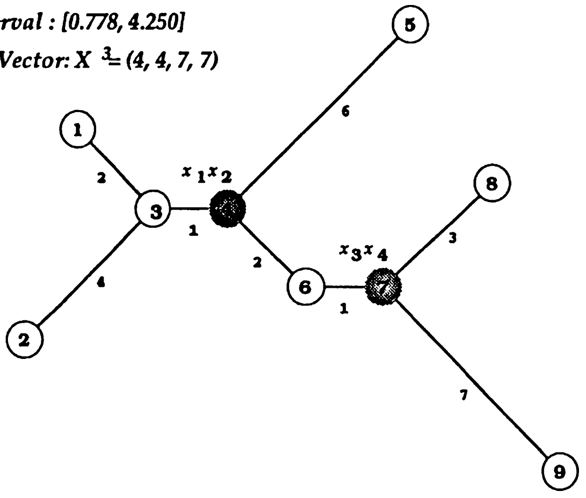
Time Interval : [0.500, 0.778]

Solution Vector: X ^{2}= (3, 4, 7, 7)



Time Interval : [0.778, 4.250]

Solution Vector: X ^{3}= (4, 4, 7, 7)



Time Interval : [4.250, 100]

Solution Vector: X ^{4}= (6, 6, 7, 7)

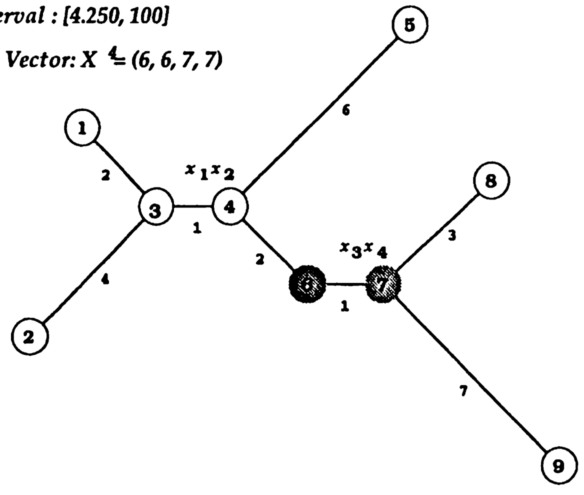


Figure 3.14 Continues

We can show the flow of control of the algorithm by a binary tree. We first find the left_right optimal solutions at t = l and t = u and the corresponding objective functions. If the objective functions are the same, we stop concluding that these solutions are optimal in $[l, u]$ (Lemma 3). If not, we find their intersection point t_{lu} and by splitting the interval $[l, u]$ into two parts $[l, t_{lu}]$ and $[t_{lu}, u]$, we branch into two. We choose the left branch and go on depth_first until the solutions at t = l and t = u have the same objective functions. At this point, we stop branching and conclude that the solutions $X(l)$ and $X(u)$ are optimal in $[l, u]$. We go on with the right branch of the previous level. Figure 3.15 depicts the flow tree corresponding to our example.

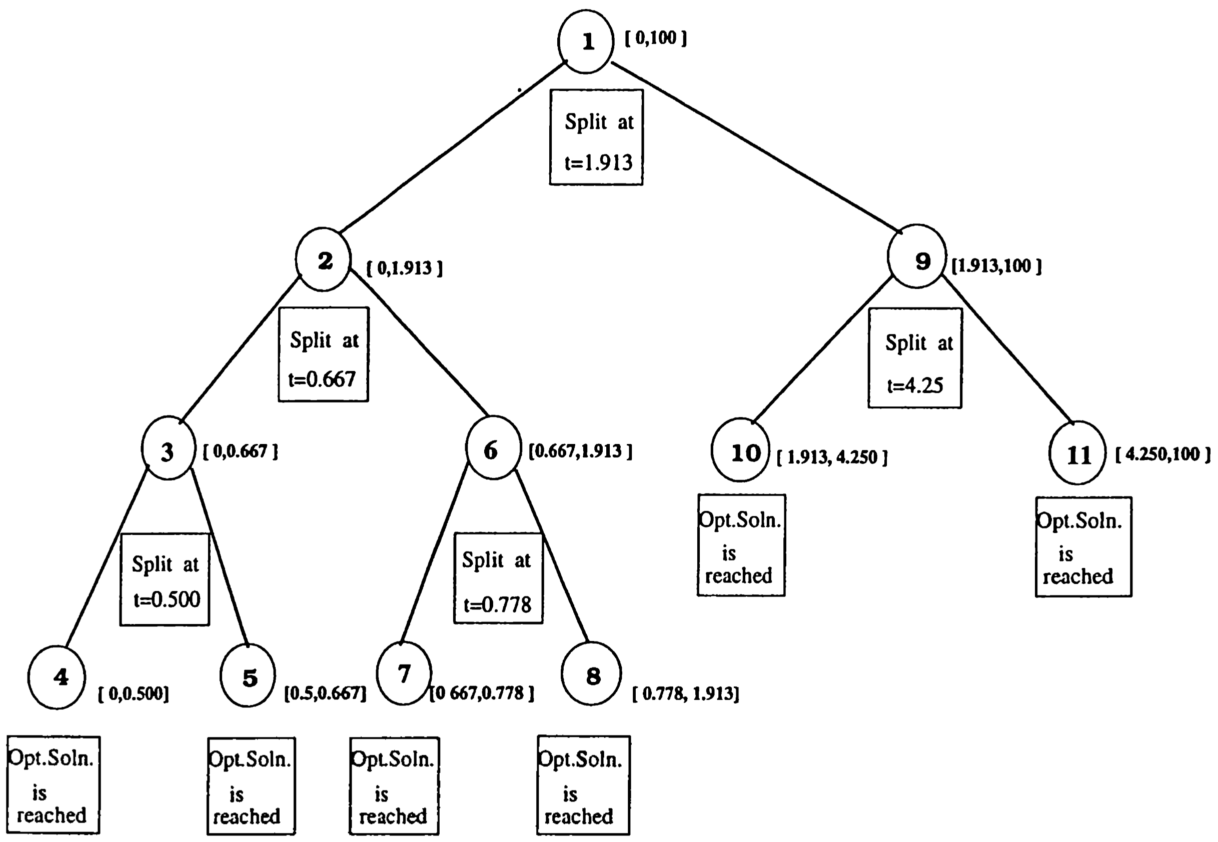


Figure 3.15. The flow tree of the algorithm

In essence, the algorithm is of divide_and_conquer type and can be coded easily by a recursive procedure. We give the pseudo_code of two different forms of the algorithm corresponding to the two cases when ε is chosen from $(0,\ell)$ or $(0, LB)$. To prevent any possible confusion , we denote ε as "ε_ℓ" if it is chosen from $(0,\ell)$ and as "ε_{LB}" if it is chosen from $(0, LB)$. The algorithm is more efficient when ε_{LB} is used. However, in solving large problems on computer, LB

may turn out to be very small and may cause numerical instability. Therefore, using a larger ε like ε_ℓ may be necessary. That is why we give the version of the algorithm when ε_ℓ is used, even though we neither know ℓ a priori nor have a bound LB' on it such that $LB < LB' \leq \ell$. We first present the algorithm using ε_ℓ and then give the simplified one with ε_{LB}.

The Algorithm LOCATEε_ℓ

```
Compute distances and $LB'$, choose $\varepsilon_\ell$ from (0,$LB'$).

Optimum_at_l ← not found

Optimum_at_u ← not found

SOLVE_IN($l, u$)
```

The Recursive Procedure SOLVE_IN

```
If Optimum_at_l is not found, then
    Find z($l$)   ( Optimal objective value at t = $l$ )
    Find $X(l+\varepsilon_\ell/2)$ and $F(X(l+\varepsilon_\ell/2),t) = A_{l+} + B_{l+}$ * t.
    Calculate $z_+ = A_{l_+} + B_{l_+}$ * $l$.
    If $z_+ > z(l)$, then   ( There is a breakpoint in the vicinity of $l$ )
        Find $F(X(l),t) = A_l + B_l$ * t
        Find $t_l = \frac{A_l - A_{l+}}{B_{l+} - B_l}$ , $X(\frac{t_l+l}{2})$   ( Right optimal solution at t = $l$ )
        $X(\frac{t_l+l}{2})$ is optimal in $[l, t_l]$   ( Possibility II b) i )
        SOLVE_IN($t_l, u$)
    Optimum_at_l ← found
If Optimum_at_u is not found, then
    Find z($u$)   ( Optimal objective value at t = $u$ )
```

```
        Find $X(u-\varepsilon_\ell/2)$ and $F(X(u-\varepsilon_\ell/2),t) = A_{u_-} + B_{u_-} * \text{t}$.
        Calculate $\text{z}_- = A_{u_-} + B_{u_-} * u$.
        If $\text{z}_- > \text{z}(u)$, then   ( There is a breakpoint in the vicinity of $u$ )
            Find $F(X(u),t) = A_u + B_u * \text{t}$
            Find $t_u = \frac{A_u - A_{u_-}}{B_{u_-} - B_u}$ , $X(\frac{t_u+u}{2})$   ( Left optimal solution at $u$ )
            $X(\frac{t_u+u}{2})$ is optimal in $[t_u, u]$   ( Possibility II b) ii )
            SOLVE_IN$(l, t_u)$
        Optimum_at_$u$ ← found
    If $A_{l_+} = A_{u_-}$ and $B_{l_+} = B_{u_-}$, then   ( Same solution at $l$ and $u$)
        $X(l+\varepsilon_\ell/2)$ is optimal in $[l,u]$
    Else,
        Calculate $t_{lu} = \frac{A_{l_+} - A_{u_-}}{B_{u_-} - B_{l_+}}$   (The problem is split at $t_{lu}$)
        Optimum_at_$l$ ← not found
        SOLVE_IN$(l, t_{lu})$
        Optimum_at_$u$ ← not found
        SOLVE_IN$(t_{lu}, u)$
```

The pseudo_code of the algorithm using ε_{LB} is as follows.

The Algorithm LOCATEε_{LB}

```
Compute distances and $LB$, choose $\varepsilon_{LB}$ from (0,$LB$).
Optimum_at_$l$ ← not found
Optimum_at_$u$ ← not found
$l$_equals_0 ← true
```

```
$u$_equals_u $\leftarrow$ true
SOLVE_IN($l, u$)
```

<u>The Recursive Procedure SOLVE_IN</u>

```
If Optimum_at_$l$ is not found, then
    If $l$_equals_0, then
        Find z($l$)   ( Optimal objective value at t = $l$ )
        Find $X(l + \varepsilon_{LB}/2)$ and $F(X(l + \varepsilon_{LB}/2), t) = A_{l+} + B_{l+}$ * t.
        Calculate $z_+ = A_{l_+} + B_{l_+} * l$.
        If $z_+ > z(l)$, then   ( There is a breakpoint in the vicinity of $l$ )
            Find $F(X(l), t) = A_l + B_l$ * t
            Find $t_l = \frac{A_l - A_{l+}}{B_{l+} - B_l}$ , $X(\frac{t_l + l}{2})$
            $X(\frac{t_l + l}{2})$ is optimal in $[l, t_l]$
            SOLVE_IN($t_l, u$)
        $l$_equals_0 $\leftarrow$ false
    Else,
        Find $X(l + \varepsilon_{LB}/2)$ and $F(X(l + \varepsilon_{LB}/2), t) = A_{l+} + B_{l+}$ * t.
    Optimum_at_$l$ $\leftarrow$ found
If Optimum_at_$u$ is not found, then
    If $u$_equals_u, then
        Find z($u$)   ( Optimal objective value at t = $u$ )
        Find $X(u - \varepsilon_{LB}/2)$ and $F(X(u - \varepsilon_{LB}/2), t) = A_{u-} + B_{u-}$ * t.
        Calculate z_ = $A_{u_-} + B_{u_-} * u$.
        If z_ > z($u$), then    ( There is a breakpoint in the vicinity of $u$ )
            Find $F(X(u),t) = A_u + B_u$ * t
            Find $t_u = \frac{A_u - A_{u_-}}{B_{u_-} - B_u}$ , $X(\frac{t_u+u}{2})$
            $X(\frac{t_u+u}{2})$ is optimal in $[t_u, u]$
            SOLVE_IN($l, t_u$)
        $u$_equals_u ← false
    Else,
        Find $X(u - \varepsilon_{LB}/2)$ and $F(X(u - \varepsilon_{LB}/2), t) = A_{u_-} + B_{u_-}$ * t.
    Optimum_at_$u$ ← found
If $A_{l_+} = A_{u_-}$ and $B_{l_+} = B_{u_-}$, then
    $X(l + \varepsilon_{LB}/2)$ is optimal in $[l, u]$
Else,
    Calculate $t_{lu} = \frac{A_{l_+} - A_{u_-}}{B_{u_-} - B_{l_+}}$
    Optimum_at_$l$ ← not found
    SOLVE_IN($l, t_{lu}$)
    Optimum_at_$u$ ← not found
    SOLVE_IN($t_{lu}, u$)
```

3.2.3 Time Complexity of the Algorithm

The time complexity of the algorithm depends on the time required to solve the static problems and the number of pieces in the trajectory. The algorithm solves static problems which are in linear order of the number of pieces in the trajectory.

At each end point of the planning horizon [0,u], we solve *two* static problems:

At t = 0, we solve one problem for t = 0 and one for t = $0 + \varepsilon/2$.

At t = u, we solve one problem for t = u and one for t = $u - \varepsilon/2$.

At an intersection point t_{ij}, if we use ε_ℓ, we solve *three* static problems: at t = t_{ij} , t = $t_{ij} - \varepsilon_\ell/2$ and t = $t_{ij} + \varepsilon_\ell/2$. If we use ε_{LB}, we solve *two* static problems: one at t = $t_{ij} - \varepsilon_{LB}/2$ and the other one at t = $t_{ij} + \varepsilon_{LB}/2$.

Lemma 4 : For each linear piece of the trajectory, we solve static problems at most *once* at an intersection point that is not a breakpoint of the trajectory. This corresponds to Possibility II a) and b).

In Possibility I a) and b), we directly detect a breakpoint of the trajectory, so we do not solve any static problems at non_breakpoint time points.

In Possibility II a) and b), once we find the left_right optimal solutions and corresponding time dependent functions at an intersection point t_{ij}, we find the intersection points of this function with the functions that are optimal at the end points of the current interval. These intersection points are either the end points of this piece or are at other pieces. In this case we may solve static problems only *once* at a non_breakpoint time point for one linear piece of the trajectory.

Thus, we can say that we solve at most once static problems at an intersection point that is not a breakpoint of the trajectory. The number of static problems solved is only a function of the number of linear pieces in the trajectory, $\tilde{q}$.

Lemma 5 : We solve at most **$6\tilde{q}$ - 5** static problems to find the trajectory using ε_ℓ, and **$4\tilde{q}$ - 2** static problems using ε_{LB}.

Let us explain this result on an example. Consider a problem instance for which $\tilde{q} = 4$.

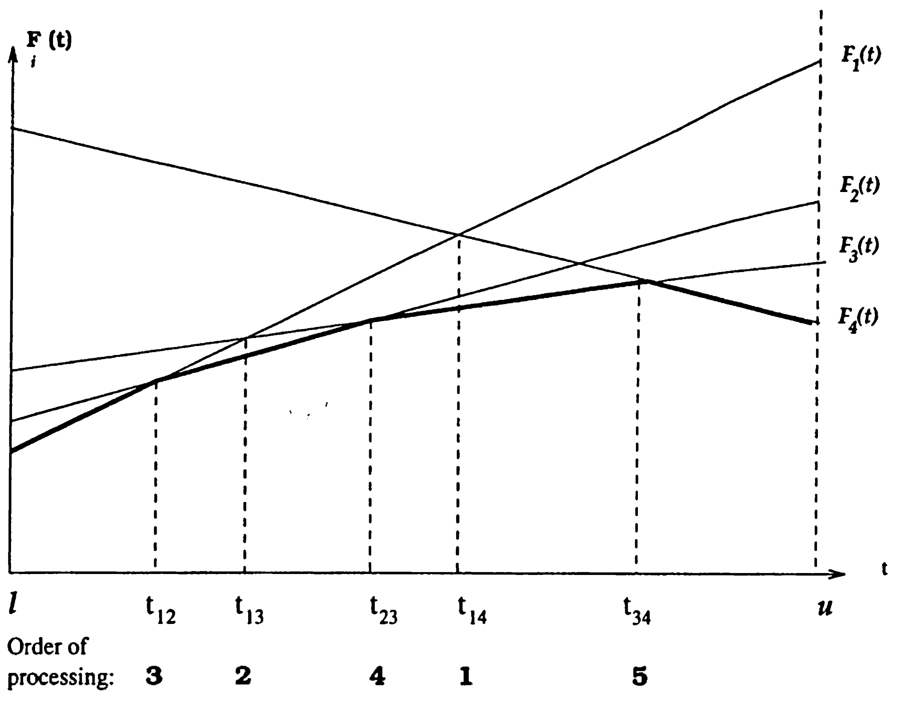


Figure 3.16. Points at which static problems are solved

There are 5 t_{ij}'s for which we solve static problems. We solve 4 more static problems at the end points, 0 and u.

It is clearly seen that in general, we solve at most once static problems at a non_breakpoint for each piece that is not adjacent to the end points of [0,u]. Actually, in almost all instances we solve these three or two problems for each non_adjacent piece since Possibility I b) rarely occurs. For pieces that are adjacent to the end points of the time horizon, we do not solve any problems at a non_breakpoint intersection point. If we have $\tilde{q}$ pieces in the trajectory, we have $\tilde{q}$ - 1 breakpoints and $\tilde{q}$ - 2 non_adjacent pieces. If we denote the number of static problems solved at a non_breakpoint by k, then we can represent the number of static problems that we solve during the trajectory construction as

$$\underbrace{k * (\tilde{q} - 2)}_{non\_breakpoints} + \underbrace{k * (\tilde{q} - 1)}_{breakpoints} + \underbrace{4}_{endpoints} = k * (2\tilde{q} - 3) + 4$$

for $\tilde{q} > 1$.

When we use ε_ℓ, k = 3 and for ε_{LB}, k = 2. Thus, Number of static problems solved = $6\tilde{q}$ - 5 using ε_ℓ for $\tilde{q} > 1$

Number of static problems solved = $4\tilde{q}$ - 2 using ε_{LB} for $\tilde{q} > 1$

If $\tilde{q} = 1$, we do not have any breakpoints. We find the single piece in the trajectory by only solving 4 static problems at the end points of the time horizon using either ε_ℓ or ε_{LB}.

3.3 Bounding Time Between Adjacent Breakpoints

We present the calculation of LB, a lower bound to the length between two adjacent intersection points in this section. As was explained earlier, LB also bounds the length between any two breakpoints of z(t). We derive two bounds for each of the problems p_ML(t) and p_MML(t).

Let $\tau', \tau \in T$ with $\tau' < \tau$ and $\ell' = \tau - \tau'$. Let $\{i, j\}$ and $\{k, l\}$ be index pairs such that $\tau = t_{ij}$ and $\tau' = t_{kl}$. Observe that $\tau' \neq \tau$ implies $\{i, j\} \neq \{k, l\}$; that is, at least three of the indices are distinct. Figure 3.17 depicts the case with $j = l$.

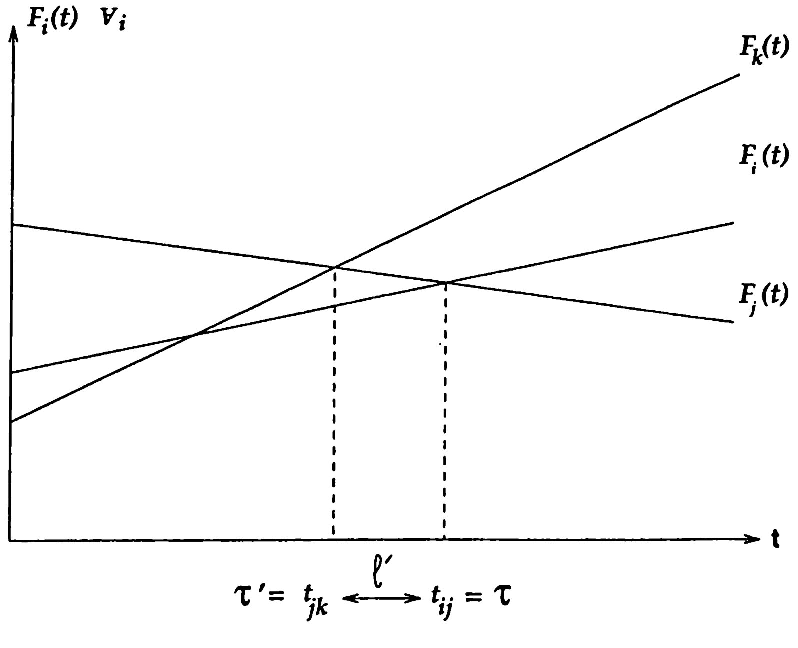


Figure 3.17. Length between nearest intersection points

Now let us analyze (t_{ij} - t_{kl}) to derive a lower bound. Since $t_{ij}, t_{kl} \in$ (0,u), from the definitions of t_{ij} and t_{kl} we have

t_{ij} - t_{kl} = $\frac{A_i - A_j}{B_j - B_i}$ - $\frac{A_k - A_l}{B_l - B_k}$, and both ratios are positive. Hence, the following term is well defined:

$$t_{ij} - t_{kl} = \frac{(A_i-A_j)(B_l-B_k)-(A_k-A_l)(B_j-B_i)}{(B_j-B_i)(B_l-B_k)}$$

Since $\tau' < \tau$, we have (t_{ij} - t_{kl}) > 0. We may assume without loss of generality that $B_j > B_i$ and $B_l > B_k$. (If not, rename the indices.) Thus, both the numerator and the denominator in the above term are positive. Therefore, with integer data (a_i, b_i's for p_ML(t) and $a_{ji}, b_{ji}, \alpha_{kl}, \beta_{kl}$'s for p_MML(t)), minimum value of

$(A_i - A_j)(B_l - B_k) - (A_k - A_l)(B_j - B_i)$ is 1. Hence,

$$\frac{1}{(B_j-B_i)(B_l-B_k)} \leq (t_{ij} - t_{kl}) = \ell'.$$

Our aim is now to find a lower bound LB to the LHS in the above inequality, i.e. find an LB such that

$$LB \leq \frac{1}{(B_j-B_i)(B_l-B_k)} \leq \ell' ,$$

3.3.1 Calculating LB for the p_ML(t) Problem

For ease of notation let $LB \triangleq \frac{1}{UB}$. Then, we are looking for,

$$UB \geq (B_j - B_i)(B_l - B_k)$$

For the p_ML(t) problem, the slope B_i is defined as

$B_i = \sum_{j=1}^{n} b_j D(X^i, v_j)$. Then,

$$(B_j - B_i) = \sum_{l=1}^{n} b_l D(X^j, v_l) - \sum_{l=1}^{n} b_l D(X^i, v_l)$$

$$= \sum_{l=1}^{n} b_l \left[D(X^j, v_l) - D(X^i, v_l)\right]$$

Let us define for a node v_l the minimum and maximum distance between this node and all nodes of the network,

$mindist_l = \min_{j \in V} d(v_j, v_l) = 0$ (minimum occurs when $j = l$)

$maxdist_l = \max_{j \in V} d(v_j, v_l)$

Then, we can bound the term [$D(X^j, v_l)$ - $D(X^i, v_l)$] as follows:

$mindist_l$ - $maxdist_l \leq [\ D(X^j, v_l) - D(X^i, v_l)\ ] \leq maxdist_l$ - $mindist_l$

which is equivalent to (since $mindist_l = 0$)

$$- maxdist_l \leq [\ D(X^j, v_l) - D(X^i, v_l)\ ] \leq maxdist_l.$$

If $b_l \geq 0$, then $b_l[\ D(X^j, v_l) - D(X^i, v_l)\ ] \leq b_l\, maxdist_l$ and

if $b_l < 0$, then $b_l[\ D(X^j, v_l) - D(X^i, v_l)\ ] \leq b_l\ (-\ maxdist_l)$.

Thus, $(B_j - B_i) \leq \sum_{l=1}^{n} \mid b_l \mid maxdist_l = U$

Note that U is independent of j and i and is an upper bound for $(B_j - B_i)$ for all pairs of j and i. So,

$(B_j - B_i)(B_l - B_k) \leq U(B_l - B_k) \leq U^2 = UB.$

We have found the upper bound UB that we were looking for :

$UB = (\sum_{l=1}^{n} \mid b_l \mid maxdist_l)^2$. Thus,

$LB = \frac{1}{UB} = \frac{1}{(\sum_{l=1}^{n} |b_l| maxdist_l)^2} \leq \frac{1}{(B_j - B_i)(B_l - B_k)} \leq (t_{ij} - t_{kl}) = \ell'.$

Another bound which is looser but somehow easier to calculate compared to the above bound is $\widehat{UB}$, which is calculated as follows.

Let b_range denote the maximum difference between the slopes of all weights and md denote the maximum distance between the nodes of the network.

$b\_range = \max_{1 \leq l \leq n} b_l$ - $\min_{1 \leq l \leq n} b_l$

$md = \max_{j,l \in \{1,\ldots,n\}} d(v_j, v_l)$

Then, $(B_j - B_i) \leq n * b\_range * md = \hat{U}$

Again, $\hat{U}$ is independent of i, j. So,

$(B_j - B_i)(B_l - B_k) \leq \hat{U}(B_l - B_k) \leq \hat{U}^2 = \widehat{UB}$

$\widehat{LB} = \frac{1}{\widehat{UB}} = \frac{1}{(n*b\_range*md)^2} \leq \ell'$.

Note that $\widehat{LB} = \frac{1}{(n*b\_range*md)^2} \leq \frac{1}{(\sum_{l=1}^{n} |b_l| maxdist_l)^2} = LB$

As a result, we could choose ε from the interval $(0, \frac{1}{(\sum_{l=1}^{n} |b_l| maxdist_l)^2})$.

3.3.2 Calculating LB for the p_MML(t) Problem

For the p_MML(t) problem, the slope of the objective function for the r-th choice of p_nodes is defined as,

$B_r = \sum_{(j,i) \in I_1} b_{ji}\, d(x_j^r, v_i) + \sum_{(j,k) \in I_2} \beta_{kl}\, d(x_j^r, x_k^r)$. Then,

$(B_r - B_s) = \sum_{(j,i) \in I_1} b_{ji}\, d(x_j^r, v_i) + \sum_{(j,k) \in I_2} \beta_{kl}\, d(x_j^r, x_k^r)$

$- \sum_{(j,i) \in I_1} b_{ji}\, d(x_j^s, v_i) - \sum_{(j,k) \in I_2} \beta_{kl}\, d(x_j^s, x_k^s)$

$(B_r - B_s) = \sum_{(j,i) \in I_1} b_{ji}\, [\ d(x_j^r, v_i) - d(x_j^s, v_i)\ ]$

$+ \sum_{(j,k) \in I_2} \beta_{kl}\, [\ d(x_j^r, x_k^r) - d(x_j^s, x_k^s)\ ]$

With the same definitions of $mindist_l$, $maxdist_l$ and md for the p_ML(t) problem, we have

$\max_{r<s \in Q}(B_r - B_s) \leq \sum_{(j,i) \in I_1} \mid b_{ji} \mid maxdist_i + md \sum_{(j,k) \in I_2} \mid \beta_{kl} \mid = U$

Again U is independent of indices r and s. So,

$(B_j - B_i)\ (B_l - B_k) \leq U^2 = UB$ and

$LB = \frac{1}{UB} = \frac{1}{(\sum_{(j,i) \in I_1} |b_{ji}| maxdist_i + md \sum_{(j,k) \in I_2} |\beta_{kl}|)^2} \leq \ell'$.

As a result, we could choose ε from the interval $(0, LB)$.

Using the second approach for the p_ML(t) problem, we can find another bound for the p_MML(t) problem, too.

Let us define *b_range* as the maximum difference between the slopes of all weights between pairs of new and existing facilities and *β_range* as the maximum difference between the slopes of all weights between pairs of new facilities.

$$b\_range = \max_{(j,i)\in I_1} b_{ji} - \min_{(j,i)\in I_1} b_{ji}$$

$$\beta\_range = \max_{(j,k)\in I_2} \beta_{kl} - \min_{(j,k)\in I_2} \beta_{kl}$$

Then, $(B_j - B_i) \leq \mid I_1 \mid * b\_range * md + \mid I_2 \mid * \beta\_range * md = \hat{U}$

where, $\mid I_1 \mid$ and $\mid I_2 \mid$ denote the cardinalities of the sets I_1 and I_2.

$(B_j - B_i)(B_l - B_k) \leq \hat{U}(B_l - B_k) \leq \hat{U}^2 = \widehat{UB}$ and

$\widehat{LB} = \frac{1}{\widehat{UB}} = \frac{1}{(|I_1|*b\_range*md+|I_2|*\beta\_range*md)^2}$.

Note that $\widehat{LB} \leq LB$ so that $\widehat{LB}$ is a looser bound.

3.4 Bounding Number of Pieces in the Trajectory

At the beginning of the chapter, we had posed the question of *whether $\tilde{q}$ ever equals q* and delayed the answer until this point. Now that we have presented a bound on the length between two adjacent breakpoints of the trajectory, we may well bound the number of pieces in the trajectory using this bound.

Since the length between two adjacent breakpoints is greater than or equal to LB, in an interval of length u we may have at most u$/LB$ pieces in the trajectory. Thus, $\tilde{q} \leq$ u$/LB$. This bound depends on the data of the problem. Examining the problems generated in our experimental studies, we observed that this bound still happens to be much larger than $\tilde{q}$ even though it tends to be significantly smaller than the bound q as the size of the problem increases. The following table gives some idea about the magnitudes of $\tilde{q}$, q and u$/LB$.

For p_MML(t) problems,

n	p	observed $\max \tilde{q}$	q	u/LB
20	10	5	3.2×10^{13}	1.8×10^{9}
100	95	1	1.0×10^{190}	1.2×10^{15}

For p_ML(t) problems,

n	p	observed $\max \tilde{q}$	q	u/LB
20	10	5	1.8×10^{5}	1.4×10^{8}
100	50	1	1.0×10^{29}	1.7×10^{10}

Another bound to $\tilde{q}$ in terms of problem data could be found using the idea that data ranges could put a limit on the number of *distinct* linear functions $F_i(\mathrm{t}) = A_i + B_i$. Let us define the ranges of values that the linear functions could take at points 0 and u as

$$A\_range \triangleq \max_{i \in Q} A_i - \min_{i \in Q} A_i \text{ and}$$

$$U\_range \triangleq \max_{i \in Q}(A_i + \mathrm{u}B_i) - \min_{i \in Q}(A_i + \mathrm{u}B_i).$$

If we use integer data, A_i's and B_i's also turn out to be integral. Then, we may have at most A_range number of different A_i's and U_range number of different $A_i + uB_i$'s. Then, there can be at most $A\_range * U\_range$ distinct linear functions. We can find the exact values of A_range and U_range by solving a minisum and a maxisum problem at $t = 0$ and $t =$u or easily find bounds to A_range and U_range using a method similar to that of calculating LB. Say the exact values are A and U. Then, $\tilde{q}$ is bounded above by $A * U$. Unfortunately, experiments revealed that $A * U$ is a worse bound compared to u/LB. Thus, the question of whether there is a much smaller and realizable bound on $\tilde{q}$ is still open to research. Examining the behavior of locational changes at breakpoints could somehow give clues about the reasons of having such small number of breakpoints.

Chapter 4

Experimental Analysis of Trajectory

Empirical studies may sometimes light up an obscure point and may well lead to theories. With the aim of putting a light on the behavior of optimal facility locations under a time varying demand pattern, we observed the optimal trajectory of a sample of problem instances. The aim was solely to observe the behavior of optimal locations in response to various parameters and special network topology, rather than testing a hypothesis statistically.

We constructed and analyzed trajectories of randomly generated P_ML(t) and P_MML(t) problems on trees and cyclic networks by implementing the algorithm presented in the previous chapter. In this chapter, we first explain the static problem solution methods that we have used during implementation. Then, we explain the experimental conditions and finally give some interesting results extracted from the experimental analysis.

4.1 Static Problem Solution Methods

As we have explained in the previous chapter, our trajectory construction algorithm solves static problems at potential breakpoints. The efficiency of static problem solution methods directly affects the running time of the algorithm. For p_median problems on tree networks, polynomial time algorithms which exploit the network structure, exist in the literature. However, p_median problems on cyclic networks are shown to be NP_hard and only small size problems can be solved by various branch and bound methods.

4.1.1 Solving p_Median with Mutual Communication Problem

p_Median with mutual communication problem can be efficiently solved on tree networks. In a forthcoming paper, Tamir [49] gives complexity results for the problem. Picard and Ratliff [44] and Kolen [38] showed that the problem decomposes into $n-1$ minimum cut problems based on necessary and sufficient optimality conditions, hence an $O(np^3)$ algorithm exists. Actually, the algorithm can have better time bounds due to more efficient minimum cut algorithms. Tamir applies a centroid decomposition to the tree and improves the $O(np^3)$ complexity bound to $O(p^3 n \log n + pn + n \log n)$. An $O(np)$ algorithm is given by Chhajed and Lowe [14] for a special case of the problem where the graph which characterizes the interaction between new facilities is series_parallel.

In our implementation we applied the minimum cut based algorithm due to Picard et. al. and Kolen. In implementing this algorithm, we used Dinic's maximum flow_minimum cut algorithm which has a time complexity of $O(p^2 m)$, m being the number of edges in the network on which we solve minimum cut problem [18]. While there are minimum cut algorithms that have better time complexities like Karzanov's of $O(p^3)$ [37], Cheriyan et.al.'s of $O(p^3/\log_2 p)$ [12], Dinic's algorithm is prefered for its ease of application.

There is an optimal solution to the p_median with mutual communication problem on vertices of the tree. The minimum cut based algorithm due to Picard et. al. and Kolen seeks a solution on vertices which satisfies the necessary and sufficient condition for optimality. The optimality condition is that a vertex solution is optimal if and only if moving any subset of new facilities to an adjacent vertex cannot reduce the objective value. Based on this condition, the following algorithm is developed.

We choose a tip vertex and tentatively place all new facilities at this vertex. We compute the subset of new facilities which when moved to the adjacent vertex decreases the objective value the most by solving a minimum cut problem. We move this subset of new facilities to the adjacent vertex and locate the remaining facilities at the tip vertex. We update the weights of the yet unlocated facilities by adding the weights of the located ones, trim the edge between the tip and the adjacent vertex and repeat the process until all facilities are located. Since we solve at most n - 1 minimum cut problems for each edge of the tree, the time complexity of the algorithm that we implemented to solve static problems is $O(np^2m)$.

In spite of the fact that the problem is efficiently solved on tree networks, the general problem on a cyclic network is shown to be NP_hard by Tamir [49]. We are not aware of any study on solving the general problem on a cyclic network in the literature, except for a polynomialy solvable special case analyzed by Chhajed and Lowe [13]. However, the integer programming formulation of the problem has a quadratic objective function and is very similar to the Quadratic Assignment Problem (QAP), for which solution approaches have been suggested in the literature. There are three groups of exact algorithms for solving QAP: branch_and_bound algorithms, cutting plane methods and dynamic programming. Branch_and_bound codes can solve up to instances with $n = 15$ facilities within reasonable time (1 hour).

The integer programming formulation of the p_median with mutual communication problem is as follows.

$$\begin{array}{lll}
\text{Min} & 1/2\sum_{i=1}^{n}\sum_{l=1}^{n}\sum_{j=1}^{p}\sum_{k=1}^{p} v_{jk}d(v_i,v_l)x_{ji}x_{kl} & \text{(Quadratic term)} \\
& +\sum_{i=1}^{n}\sum_{l=1}^{n}\sum_{j=1}^{p} w_{jl}d(v_i,v_l)x_{ji} & \text{(Linear term)} \\
s.t. & & \\
& \sum_{i=1}^{n} x_{ji} = 1 & j = \{1,\dots,p\} \\
& x_{ji} \in \{0,1\} & \forall i,j
\end{array}$$

where the decision variables are,

$$x_{ji} = \begin{cases} 1 & \text{if new facility } j \text{ is located at site } i \\ 0 & \text{otherwise} \end{cases}$$

For simplicity of notation, let us define,

$v_{ijkl} = v_{jk}d(v_i, v_l)$

$w_{ijl} = w_{jl}d(v_i, v_l)$

If we put the linear term into quadratic form, the problem is very similar to the QAP, the only differences being that $\sum_{j=1}^{n} x_{ji} = 1$ constraints in QAP are relaxed in this problem and j, k run from 1 to p.

Lawler (1963) linearized QAP with $O(n^4)$ binary variables. Bazaraa and Sherali (1980) constructed a MIP with n^2 binary variables, $n^2(n-1)^2/2$ real variables and $2n^2$ linear constraints. The smallest size linearization known so far is due to Kaufman and Broeckx (1978). They have shown that QAP is equivalent to a MIP with n^2 real and n^2 integer variables and $O(n^2)$ constraints. The application of their MIP formulation to our problem yields np real and np integer variables and $O(np)$ constraints. The formulation is as follows.

$$
\begin{array}{lll}
\text{Min} & 1/2\sum_{i=1}^{n}\sum_{j=1}^{p} r_{ji} + \sum_{i=1}^{n}\sum_{l=1}^{n}\sum_{j=1}^{p} w_{ijl}x_{ji} & \\
s.t. & & \\
 & \sum_{i=1}^{n} x_{ji} = 1 & j = \{1, \ldots, p\} \\
 & (\sum_{l=1}^{n}\sum_{k=1}^{p} v_{ijkl})x_{ji} + \sum_{l=1}^{n}\sum_{k=1}^{p} v_{ijkl}x_{kl} & \\
 & \qquad -r_{ji} \leq (\sum_{l=1}^{n}\sum_{k=1}^{p} v_{ijkl}) & \forall i, j \\
 & x_{ji} \in \{0, 1\} & \forall i, j \\
 & r_{ji} \geq 0 & \forall i, j
\end{array}
$$

where the real variables,

$r_{ji} = x_{ji}(\sum_{l=1}^{n}\sum_{k=1}^{p} v_{ijkl}x_{kl})$ are introduced to maintain linearity.

We solved the above mixed integer program by a branch_and_bound method with the software *Cplex* to get the static problem solutions.

4.1.2 Solving p_Median Problem

While the general problem of finding a p_median on a cyclic network is shown to be NP_hard by Kariv and Hakimi [36], the p_median problem on a tree network is tractable. Matula and Kolde [41] gave an $O(n^3p^2)$ algorithm, Kariv and Hakimi [36] gave an $O(n^2p^2)$ algorithm for finding a p_median of a **tree** where $p > 1$. Based on the node optimality property, this algorithm finds an optimal solution on the vertices of the tree by dynamic programming.

In our implementation, we used Kariv_Hakimi's $O(n^2p^2)$ algorithm for solving static p_median problems on a tree. The algorithm consists of two phases. In the first phase, the optimal objective value is calculated by traversing the rooted tree upward from tips to the root in stages. In the second phase, the p_median of the tree is found using the calculations made in the first phase. In essence, the algorithm uses the idea that a p_median partitions the tree into p connected subtrees. If we know that a facility at a certain vertex v_f serves a client at another vertex v_c in the optimal solution, then the clients at the vertices on the path connecting v_f and v_c are also served by v_f.

The approaches for solving the p_median problem on a **general** network are primal and dual based integer programming. The LP relaxation of the integer programming formulation quite frequently gives an integer solution. However, integer solutions cannot be guaranteed. Thus, researchers have recently focused on dual based techniques which proved to be very efficient. One of the most efficient techniques available is a modification of Erlenkotter's DUALOC method for uncapacitated facility location problem [26], to solve the pmedian problem. The lower bounds generated by solving the dual of a relaxed problem are used in a branch_and_bound procedure.

In implementing our trajectory construction algorithm we solved the integer programming problem with the software *Cplex* , which actually uses a branch_and_bound procedure. The integer programming formulation of the p_median problem is as follows.

$$
\begin{array}{lrcll}
Min & \multicolumn{4}{l}{\sum_{i=1}^{n}\sum_{j=1}^{n} \; c_{ij}x_{ij}} \\
s.t. & & & & \\
 & \sum_{j=1}^{n} x_{ij} & = & 1 & i \in I \\
 & x_{ij} & \leq & y_j & i \in I, j \in J \\
 & \sum_{j=1}^{n} y_j & = & p & \\
 & x_{ij} & \in & \{0,1\} & \forall i,j \\
 & y_j & \in & \{0,1\} & \forall j
\end{array}
$$

where,

$I = \{1, \ldots, n\}$ denotes the set of clients and

$J = \{1, \ldots, n\}$ denotes the set of potential facility sites

The decision variables are,

$$y_j = \begin{cases} 1 & \text{if a facility is established at site } j \\ 0 & \text{otherwise} \end{cases}$$

$$x_{ij} = \begin{cases} 1 & \text{if client } i \text{ is served by facility at site } j \\ 0 & \text{otherwise} \end{cases}$$

and the cost of serving client i by a facility at site j at a time point t_0 is $c_{ij} = d(v_i, v_j) * w_i(t_0)$.

Note that minimization of the objective function assures that each demand center is served by the closest facility.

4.2 Experimental Conditions

We have analyzed the behavior of the optimal solution in a time interval for a sample of p_median with mutual communication problems with linear weights, P_MML(t) and p_median problems with linear weights P_ML(t), both on tree and cyclic networks.

The network types that we analyzed are *line*, *star*, *arbitrary trees* and *cyclic networks* with 25%, 50% and 75% edge density. For the problem with mut ual communication, we investigated three different levels of communication between new and existing facilities relative to the communication between new facilities. These dif ferent levels are reflected by a factor k that shifts the range of the interval in wh ich the weights between new and existing facilities are generated from. Thus, $k = \{ 0.5, 1, 5\}$ corresponds to the situations in which the communication between new and exis ting facilities has *half*, *equal* and *five times* importance compared t o the communication between new facilities.

We randomly generated 10 instances for each problem type that is a combination of the above stated 6 network types, three levels of relative communication for the problem with mutual communication and the following problem sizes, n and p.

For P_MML(t) and P_ML(t) on Trees (Line, Star and Arbitrary):

n	20			40			60			80			100		
p	5	10	15	5	20	35	5	30	55	5	40	75	5	50	95

For P_ML(t) on General Networks (25%, 50% and 75% Edge Density):

n	10			20			30			40		
p	2	5	8	5	10	15	5	15	25	5	20	35

For P_MML(t) on General Networks (25%, 50% and 75% Edge Density):

n	10			20	
p	2	5	8	5	10

We coded our algorithm in C language and used static problem solution algorithms as subroutines. These subroutines are C codes of Picard and Ratliff, and Kolen's algorithm for p_median with mutual communication problem on trees, Kariv and Hakimi's algorithm for p_median problem on trees and MIP solutions to the problems on general networks using *Cplex* Mixed Integer Library Routines. The programs were run on SPARC Stations under SunOS 4.1.3.

4.2.1 Generation of Data

We generate integer data. All weights are in the form $a + b$ t and are nonnegative throughout the time horizon $[0,u]$. The intercept a comes from a discrete uniform distribution between 0 and 200, while the slope b comes from a discrete uniform distribution between -10 and 10. Thus, we allow *increasing* or *decreasing* linear weights but avoid nonnegative weights by discarding and regenerating weights until we get a weight that is nonnegative throughout the time horizon.

For P_MML(t), we multiply intercept and the slope of the randomly generated weight between new and existing facilities by the factor k to get different levels of relative communication as explained above.

Edge lengths are also integral and come from a discrete uniform distribution between 1 and 12.

4.2.2 Generation of Network Structure

Number of nodes in the network, n is taken as input. Number of nodes alone determines the structure of line and star type trees.

For arbitrary trees, a rooted tree such that each node has a random number of children is generated. Number of children of a node comes from a truncated normal distribution with mean 2 and variance 1. Negative values are casted into 1. We start constructing the tree from the root and add random number of children to it. Children generated at a level are kept in a list as the set of potential fathers of the next level. If the number of nodes generated does not exceed n, we go on to the next level and add children to each potential father in the list until n nodes are generated.

To generate a general network, we first generate a tree. We calculate the number of edges in the network, m by multiplying a density factor with the number of edges in a complete graph. The density factors are 0.25, 0.5 and 0.75 . If m is greater than $n - 1$, we add edges to the tree until we reach m edges. We choose the next edge to be generated randomly among the potential edges.

4.3 Results of Experiments

Results from the analysis of trajectory of P_MML(t) and P_ML(t)problems which are randomly generated as explained in the previous section, are presented in this section. We first give results related to the running time of the algorithm and the number of pieces in the trajectory for both problems on line, star, arbitrary trees and cyclic networks. Then, the analysis of locational changes in different network structures is presented.

4.3.1 Run Time of the Algorithm

While explaining the time complexity of our trajectory construction algorithm, we had discussed that the efficiency of the algorithm depends on the number of pieces in the trajectory, $\tilde{q}$, and the efficiency of the static problem solution methods. In the instances that we solved, the number of pieces in the trajectory was extremely low compared to the number of linear functions. We have efficient solution algorithms to solve static problems on trees. Therefore, the trajectory could be constructed in about 10 minutes for instances of P_MML(t) problems with $n = 100$ and $p = 95$ and in about 1 hour for that of P_ML(t) problems. Run times of the problems on cyclic networks are comparatively high since we use branch_and_bound methods for static problem solutions. Thus, we could construct the trajectory in about 2 minutes for instances of P_MML(t) problems with $n = 20$ and $p = 10$ and 3 minutes for that of P_ML(t)problems with $n = 40$ and $p = 35$.

The average run times of 10 randomly generated instances of the problems on line, star, arbitrary trees and cyclic networks can be found on the following tables. In solving problems on cyclic networks, *Cplex* gave erroneous solutions in some of the instances. Thus, in the last column of tables for general networks, we give the number of instances correctly solved (denoted by I) out of samples of 10 instances. On these tables the column d denotes the % density of the general networks generated.

n	p	k	LINE	STAR	ARBITRARY
		0.5	0.551	1.083	0.275
	5	1	0.930	1.248	0.340
		5	2.528	1.275	0.523
		0.5	1.733	4.300	0.806
20	10	1	1.748	4.726	0.856
		5	12.790	4.713	2.486
		0.5	10.975	10.993	1.850
	15	1	4.561	10.531	1.758
		5	25.435	15.321	4.455
		0.5	2.258	2.923	0.596
	5	1	5.255	2.788	0.635
		5	8.351	2.976	1.178
		0.5	31.450	47.171	6.133
40	20	1	33.070	36.855	6.210
		5	467.860	33.148	8.951
		0.5	72.770	49.185	104.106
	35	1	41.396	85.640	93.838
		5	212.788	87.883	141.925
		0.5	5.215	4.325	1.925
	5	1	5.141	4.893	2.406
		5	17.388	5.008	3.518
		0.5	99.171	66.606	22.611
60	30	1	57.176	50.108	22.063
		5	920.126	55.471	138.708
		0.5	97.948	401.495	183.126
	55	1	106.958	178.371	142.221
		5	401.413	162.448	172.766
		0.5	5.316	6.791	2.143
	5	1	23.518	4.141	4.041
		5	37.460	7.350	16.540
		0.5	160.599	295.639	209.851
80	40	1	92.825	108.406	50.260
		5	579.101	135.509	158.568
		0.5	33.518	401.510	252.306
	75	1	384.625	321.595	251.586
		5	428.575	391.894	270.956
		0.5	17.228	10.496	4.613
	5	1	22.673	6.835	28.708
		5	51.794	12.610	42.146
		0.5	10.808	263.066	127.038
100	50	1	664.651	197.413	125.715
		5	1290.030	255.176	69.755
		0.5	639.791	722.343	729.933
	95	1	104.440	706.989	564.923
		5	113.375	779.663	583.381

Table 4.1. Average run times for p_MML(t) problems on tree networks in seconds

GENERAL NETWORKS						
n	p	k	d	Cpu Times		I
			25		1.118″	10
		0.5	50		2.403″	10
			75		1.495″	10
			25		1.833″	10
	2	1	50		3.156″	10
			75		2.180″	10
			25		2.183″	10
		5	50		2.730″	10
			75		2.221″	10
			25		5.461″	10
		0.5	50		6.086″	10
			75		9.183″	10
			25		9.591″	10
10	5	1	50		7.918″	10
			75		14.946″	10
			25		10.975″	10
		5	50		32.296″	10
			75		38.968″	10
			25		16.245″	10
		0.5	50		18.126″	10
			75		38.223″	10
			25		12.461″	10
	8	1	50		21.645″	10
			75		35.956″	10
			25		28.453″	10
		5	50	1′	30.950″	10
			75	3′	44.835″	10

Table 4.2. Average run times for p_MML(t) problems on general networks in minutes and seconds, pg 1

GENERAL NETWORKS						
n	p	k	d	Cpu Times		I
			25		32.280″	10
		0.5	50		25.571″	10
			75		29.880″	10
			25		33.761″	10
20	5	1	50		30.628″	10
			75	1′	1.593″	10
			25	3′	29.956″	10
		5	50	2′	4.303″	10
			75	1′	40.985″	10
			25	1′	38.581″	8
		0.5	50	2′	21.795″	8
			75	1′	34.513″	10
			25	1′	47.866″	10
20	10	1	50	2′	34.269″	8
			75	4′	0.365″	10
			25	23′	26.811″	5
		5	50	92′	27.500″	7
			75	126′	44.950″	4

Table 4.3. Average run times for p_MML(t) problems on general networks in minutes and seconds, pg 2

GENERAL NETWORKS					
n	p	d	Cpu Times		I
		0.25		3.231″	10
	2	0.5		4.141″	10
		0.75		4.493″	10
		0.25		4.436″	10
10	5	0.5		4.340″	10
		0.75		6.135″	10
		0.25		3.783″	10
	8	0.5		3.558″	10
		0.75		4.561″	10
		0.25		25.560″	10
	5	0.5		28.860″	10
		0.75		26.641″	10
		0.25		27.066″	10
20	10	0.5		26.058″	10
		0.75		28.650″	10
		0.25		24.165″	10
	15	0.5		21.621″	10
		0.75		22.525″	10
		0.25	1′	13.498″	10
	5	0.5	1′	9.453″	10
		0.75		36.933″	6
		0.25	1′	20.350″	4
30	15	0.5	1′	21.688″	10
		0.75	1′	42.601″	10
		0.25	1′	11.707″	8
	25	0.5	1′	8.158″	10
		0.75	1′	5.168″	10
		0.25	1′	53.550″	9
	5	0.5	1′	10.975″	10
		0.75	2′	3.433″	6
		0.25	4′	22.783″	7
40	20	0.5	5′	38.716″	9
		0.75	4′	14.550″	7
		0.25	2′	43.806″	9
	35	0.5	2′	1.691″	9
		0.75	1′	50.835″	9

Table 4.4. Average run times for p_ML(t) problems on general networks in minutes and seconds

n	p	LINE	STAR	ARBITRARY
20	5	2.808″	2.183″	3.996″
	10	7.226″	4.328″	6.930″
	15	7.311″	3.923″	11.076″
40	5	38.918″	13.075″	15.600″
	20	2′ 21.810″	1′ 36.295″	2′ 2.056″
	35	3′ 26.978″	1′ 4.573″	1′ 34.116″
60	5	1′ 57.291″	50.403″	2′ 21.468″
	30	19′ 45.734″	2′ 45.923″	2′ 40.333″
	55	13′ 8.213″	4′ 36.833″	5′ 46.268″
80	5	2′ 40.608″	1′ 27.920″	1′ 34.831″
	40	13′ 7.993″	4′ 52.123″	30′ 48.949″
	75	11′ 11.926″	12′ 16.273″	17′ 57.331″
100	5	3′ 29.149″	2′ 38.831″	3′ 29.883″
	50	214′ 54.400″	74′ 20.916″	79′ 42.126″
	95	111′ 3.654″	23′ 38.054″	43′ 50.705″

Table 4.5. Average run times for p_ML(t) problems on tree networks in minutes and seconds

4.3.2 Number of Pieces in the Trajectory

Compared to the number of linear functions that are candidates to form a piece of the trajectory, q, the number of pieces of the trajectory, $\tilde{q}$ is extremely small. Among the instances of P_MML(t) problems generated, maximum $\tilde{q}$ turned out to be 15 for the problem with $n = 60$ and $p = 30$, while for a problem of this size $q = n^p = 2.21e+53$. Among the instances of P_ML(t) problems generated, maximum $\tilde{q}$ turned out to be 33 for the problem with $n = 100$ and $p = 50$, while for a problem of this size $q = \binom{n}{p} = 1.01e+29$.

We observed more pieces in trajectories of P_MML(t) problems compared to P_ML(t) problems. This can be explained by the fact that the mutual communication between new facilities tends to locate them at the same node. This effect, which increases as the number of new facilities increases, is so strong that a new facility is rarely located separately. As a result, the facilities cannot easily change their locations by time. However, in P_ML(t) problems, the medians partition the network into p pieces such that clients within a partition are served from the same facility. Both relocation of the facility within a partition and a repartitioning of the network can lead to lower costs by time.

In instances of P_MML(t) problems with high number of new facilities and dense communication between new facilities, all facilities are located at the same node. If they change their locations, they move altogether to a new node at a breakpoint of the trajectory. As the mutual communication between new facilities decreases, facilities tend to be separately located, so that more breakpoints are likely to occur. While we observed up to 9 pieces in trajectories of P_MML(t) problems with $n = 80$, $p = 40$ and the relative importance factor $k = 5$, we observed only *one* piece in the trajectory of the same sized problem with $k = 0.5$ and $k = 1$. We can see the decrease in number of pieces with an increase in p by the example that while we have at most 12 pieces in the trajectory of the problem with $n = 80$, $p = 5$ and $k = 5$, we observed at most 2 pieces when p increased to 75. These properties are valid whatever the network structure is, with the exception of problems on star networks whose trajectories consist of one piece corresponding to a very robust solution.

The network structure affects the number of pieces in the trajectory. For both P_MML(t) and P_ML(t) problems we have more pieces, if the network is line type than those of arbitrary or star trees. In all of the 750 P_MML(t) problems on star networks we have *only one* piece in the trajectory. Due to the special network topology , in all of the problems all new facilities are located at the central node (the hub).

For P_ML(t) problems with equal q values, i.e. instances with $n = n_1$, $p = p_1$ and $n = n_1$, $p = n - p_1$, we do not have similar number of pieces. Contrary to intuition, the number of pieces is more for instances with larger p even though the number of potential pieces is the same.

The maximum and most frequently observed $\tilde{q}$ in samples of P_MML(t) and P_ML(t) problems are given in Tables 4.6, 4.7,4.8, 4.9, 4.10 .

4.3.3 Locational Changes

Changes in the facility locations by time show properties pertinent to the network structure, while there are some properties that are valid no matter what the network structure is. We will first state properties that are more general,

				LINE		ARBITRARY		STAR	
n	p	k	q	FREQ. $\tilde{q}$	MAX $\tilde{q}$	FREQ. $\tilde{q}$	MAX $\tilde{q}$	FREQ. $\tilde{q}$	MAX $\tilde{q}$
		0.5	$3.20 * 10^6$	1	2	1	3	1	1
	5	1	$3.20 * 10^6$	1-2	2	1	3	1	1
		5	$3.20 * 10^6$	4-5	5	1	5	1	1
		0.5	$1.02 * 10^{13}$	1	2	1	1	1	1
20	10	1	$1.02 * 10^{13}$	1	2	1	1	1	1
		5	$1.02 * 10^{13}$	4	8	1	8	1	1
		0.5	$3.28 * 10^{19}$	1	2	1	2	1	1
	15	1	$3.28 * 10^{19}$	1	2	1	1	1	1
		5	$3.28 * 10^{19}$	5	5	1	5	1	1
		0.5	$1.02 * 10^8$	1-2	2	1	2	1	1
	5	1	$1.02 * 10^8$	2	7	1	2	1	1
		5	$1.02 * 10^8$	3-8	9	1	6	1	1
		0.5	$1.10 * 10^{32}$	1	2	1	1	1	1
40	20	1	$1.10 * 10^{32}$	1	2	1	1	1	1
		5	$1.10 * 10^{32}$	6-9-10	11	1	5	1	1
		0.5	$1.18 * 10^{56}$	1	1	1	1	1	1
	35	1	$1.18 * 10^{56}$	1	1	1	2	1	1
		5	$1.18 * 10^{56}$	1-3	5	1	3	1	1
		0.5	$7.78 * 10^8$	1-2	5	1	3	1	1
	5	1	$7.78 * 10^8$	2	6	1	6	1	1
		5	$7.78 * 10^8$	9	11	1	7	1	1
		0.5	$2.21 * 10^{53}$	1	2	1	1	1	1
60	30	1	$2.21 * 10^{53}$	1	2	1	1	1	1
		5	$2.21 * 10^{53}$	6-7	15	1	6	1	1
		0.5	$6.29 * 10^{97}$	1	1	1	2	1	1
	55	1	$6.29 * 10^{97}$	1	1	1	2	1	1
		5	$6.29 * 10^{97}$	1	4	1	2	1	1
		0.5	$3.28 * 10^9$	3	3	1	6	1	1
	5	1	$3.28 * 10^9$	5	7	1	7	1	1
		5	$3.28 * 10^9$	7	12	1	7	1	1
		0.5	$1.33 * 10^{76}$	1	1	1	1	1	1
80	40	1	$1.33 * 10^{76}$	1	1	1	1	1	1
		5	$1.33 * 10^{76}$	6	9	1	8	1	1
		0.5	$5.40 * 10^{142}$	1	2	1	1	1	1
	75	1	$5.40 * 10^{142}$	1	2	1	1	1	1
		5	$5.40 * 10^{142}$	1	2	1	1	1	1
		0.5	$1 * 10^{10}$	4	4	1	1	1	1
	5	1	$1 * 10^{10}$	3-7	10	1	7	1	1
		5	$1 * 10^{10}$	9	10	1	4	1	1
		0.5	$1 * 10^{100}$	1	2	1	1	1	1
100	50	1	$1 * 10^{100}$	1	2	1	1	1	1
		5	$1 * 10^{100}$	3	8	1	4	1	1
		0.5	$1 * 10^{190}$	1	2	1	1	1	1
	95	1	$1 * 10^{190}$	1	2	1	1	1	1
		5	$1 * 10^{190}$	1	2	1	1	1	1

Table 4.6. Maximum and most frequently observed $\tilde{q}$ in p_MML(t) problems on tree networks

p_MML(t) ON GENERAL NETWORKS							
n	p	k	d	q	FREQ. $\tilde{q}$	MAX $\tilde{q}$	I
		0.5	25	100	1	1	10
			50	100	2	2	10
			75	100	1	2	10
	2	1	25	100	1	2	10
			50	100	1	4	10
			75	100	1	3	10
		5	25	100	1	3	10
			50	100	1	3	10
			75	100	1	3	10
		0.5	25	$1*10^5$	1	2	10
			50	$1*10^5$	1	1	10
			75	$1*10^5$	1	2	10
10	5	1	25	$1*10^5$	1	2	10
			50	$1*10^5$	1	2	10
			75	$1*10^5$	1	3	10
		5	25	$1*10^5$	1	3	10
			50	$1*10^5$	2	6	10
			75	$1*10^5$	1	4	10
		0.5	25	$1*10^8$	1	1	10
			50	$1*10^8$	1	1	9
			75	$1*10^8$	1	2	10
	8	1	25	$1*10^8$	1	1	10
			50	$1*10^8$	1	1	10
			75	$1*10^8$	2	4	10
		5	25	$1*10^8$	1	3	10
			50	$1*10^8$	1	2	9
			75	$1*10^8$	1-2	6	9

Table 4.7. Maximum and most frequently observed $\tilde{q}$ in p_MML(t) problems on general networks, pg 1

p_MML(t) ON GENERAL NETWORKS							
n	p	k	d	q	FREQ. $\tilde{q}$	MAX $\tilde{q}$	I
20	5	0.5	25	$3.2 * 10^6$	1	2	10
			50	$3.2 * 10^6$	1	2	10
			75	$3.2 * 10^6$	1	2	10
		1	25	$3.2 * 10^6$	1	2	10
			50	$3.2 * 10^6$	1	2	10
			75	$3.2 * 10^6$	1	3	10
		5	25	$3.2 * 10^6$	5	5	10
			50	$3.2 * 10^6$	1-2	5	10
			75	$3.2 * 10^6$	3	4	10
20	10	0.5	25	$1.02 * 10^{13}$	1	1	8
			50	$1.02 * 10^{13}$	1	1	8
			75	$1.02 * 10^{13}$	1	1	10
		1	25	$1.02 * 10^{13}$	1	1	10
			50	$1.02 * 10^{13}$	1	2	8
			75	$1.02 * 10^{13}$	1	1	10
		5	25	$1.02 * 10^{13}$	1	4	5
			50	$1.02 * 10^{13}$	3	4	7
			75	$1.02 * 10^{13}$	2	5	4

Table 4.8. Maximum and most frequently observed $\tilde{q}$ in p_MML(t) problems on general networks, pg 2

p_ML(t) ON GENERAL NETWORKS						
n	p	d	q	FREQ. $\tilde{q}$	MAX $\tilde{q}$	I
		25	45	1	3	10
	2	5	45	1-3	3	10
		75	45	2	4	10
		25	252	2	5	10
10	5	5	252	3	5	10
		75	252	4	5	10
		25	45	2	5	10
	8	5	45	3	4	10
		75	45	4	4	10
		25	$1.55 * 10^4$	3	7	10
	5	50	$1.55 * 10^4$	4	5	10
		75	$1.55 * 10^4$	2-4	6	10
		25	$1.85 * 10^5$	4-7	7	10
20	10	50	$1.85 * 10^5$	4	7	10
		75	$1.85 * 10^5$	4	8	10
		25	$1.55 * 10^4$	5-6	7	10
	15	50	$1.55 * 10^4$	4	8	10
		75	$1.55 * 10^4$	4-5-6	8	10
		25	$1.42 * 10^5$	1	5	10
	5	50	$1.42 * 10^5$	1	4	10
		75	$1.42 * 10^5$	1	2	6
		25	$1.55 * 10^8$	6	11	4
30	15	50	$1.55 * 10^8$	7	9	10
		75	$1.55 * 10^8$	7	11	10
		25	$1.42 * 10^5$	7	10	8
	25	50	$1.42 * 10^5$	6	9	10
		75	$1.42 * 10^5$	10	10	10
		25	$6.58 * 10^5$	1	5	9
	5	50	$6.58 * 10^5$	6	9	10
		75	$6.58 * 10^5$	1	4	6
		25	$1.38 * 10^{11}$	1-9	11	7
40	20	50	$1.38 * 10^{11}$	9	12	9
		75	$1.38 * 10^{11}$	8	12	7
		25	$6.58 * 10^5$	8-9	10	9
	35	50	$6.58 * 10^5$	9	9	9
		75	$6.58 * 10^5$	5-7	8	9

Table 4.9. Maximum and most frequently observed $\tilde{q}$ in p_ML(t) problems on general networks

			LINE		ARBITRARY		STAR	
e n	p	q	FREQ. $\tilde{q}$	MAX $\tilde{q}$	FREQ. $\tilde{q}$	MAX $\tilde{q}$	FREQ. $\tilde{q}$	MAX $\tilde{q}$
	5	$1.55 * 10^4$	4	5	3	7	3	6
20	10	$1.85 * 10^5$	5	8	5-6	7	4-5	7
	15	$1.55 * 10^4$	4-6	8	7	8	4	6
	5	$6.58 * 10^5$	5	7	3-4	5	5	5
40	20	$1.38 * 10^{11}$	10	15	10-11	12	9	11
	35	$6.58 * 10^5$	8	11	6-7	10	6-8	10
	5	$5.46 * 10^6$	5	9	5	9	4	6
60	30	$1.18 * 10^{17}$	16	20	14	17	9	16
	55	$5.46 * 10^6$	9	11	7	13	6-8-9-10	12
	5	$2.40 * 10^7$	5	10	2	8	2-3-5-6-7	7
80	40	$1.08 * 10^{23}$	9	11	18	25	17	27
	75	$2.40 * 10^7$	9	12	7-9-11	14	9	11
	5	$7.53 * 10^7$	6	8	3-4	5	3	6
100	50	$1.01 * 10^{29}$	23	33	21	30	22	31
	95	$7.53 * 10^7$	4-6	8	9-10	13	8	14

Table 4.10. Maximum and most frequently observed $\tilde{q}$ in p_ML(t) problems on tree networks

and then elaborate them for each network type.

The analysis of facility locations in *p_MML(t) problems* reveals that *facilities tend to be located at central nodes of the network.* This tendency is valid for star and line networks, thus for arbitrary trees, which is a compromise between the two, as well as general networks. This property is observed very strongly for star networks. All facilities are located at the central node throughout the time horizon in all of the problem instances generated. The solution is very robust to the problem data. Only by manipulating weights in an extraordinarily unbalanced way could we find a problem instance in which a facility is separated from the others and located at a tip node. Facilities are located at central nodes in line networks too. When the number of nodes is small, facilities tend to change location around the central nodes by time. As the number of nodes gets higher such as 80, 100, facilities are located at the center of the line network and rarely leave their locations. In all of the problem instances on line networks no facility is ever located at a tip vertex. However, in *p_ML(t) problems*, the pmedian tends to be as *dispersed* on the network as possible. Thus, a median is frequently located at a tip vertex. Usually, a median at a tip vertex serves only the client at this vertex. This property is observed in all of the problems on star trees, while it is very rare on line trees (Figure 4.1).

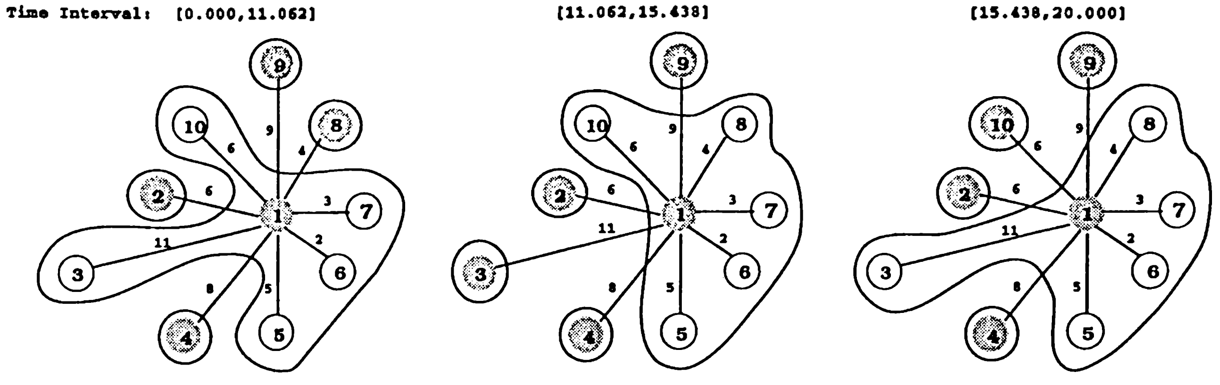


Figure 4.1. An example p_ML(t) problem on a star network

At a breakpoint, more than one new facility may change its location in p_MML(t) problems and the p_median may change in more than one element in p_ML(t) problems. In p_ML(t) problems, at some breakpoints *the partition of the network is preserved* and the medians are relocated only within a partition. But there are also breakpoints at which a totally different picture arises due to a *repartitioning of the network.* In the following examples, one on a line network, and the other on a cyclic network, we have both of such breakpoints.

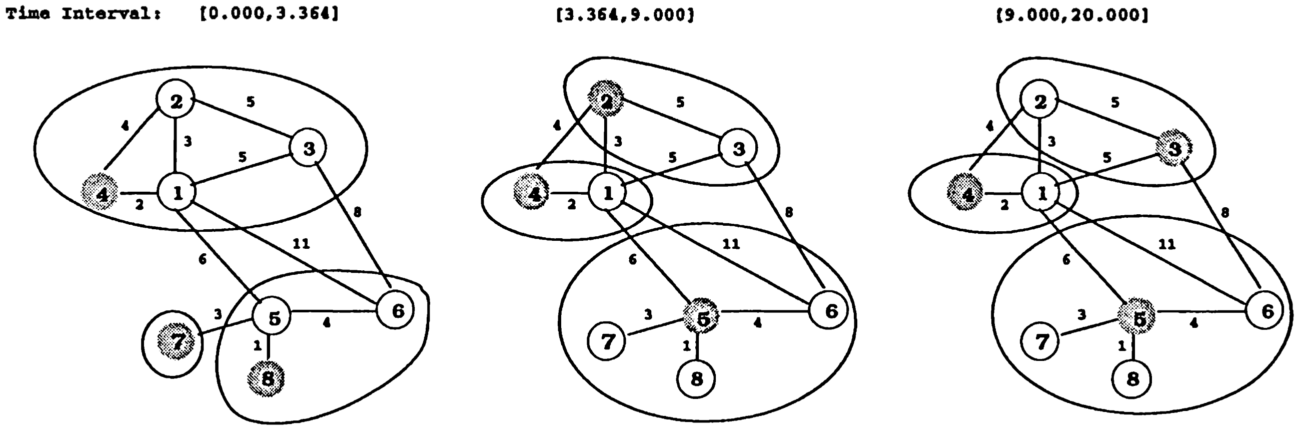


Figure 4.2. An example p_ML(t) problem on a general network

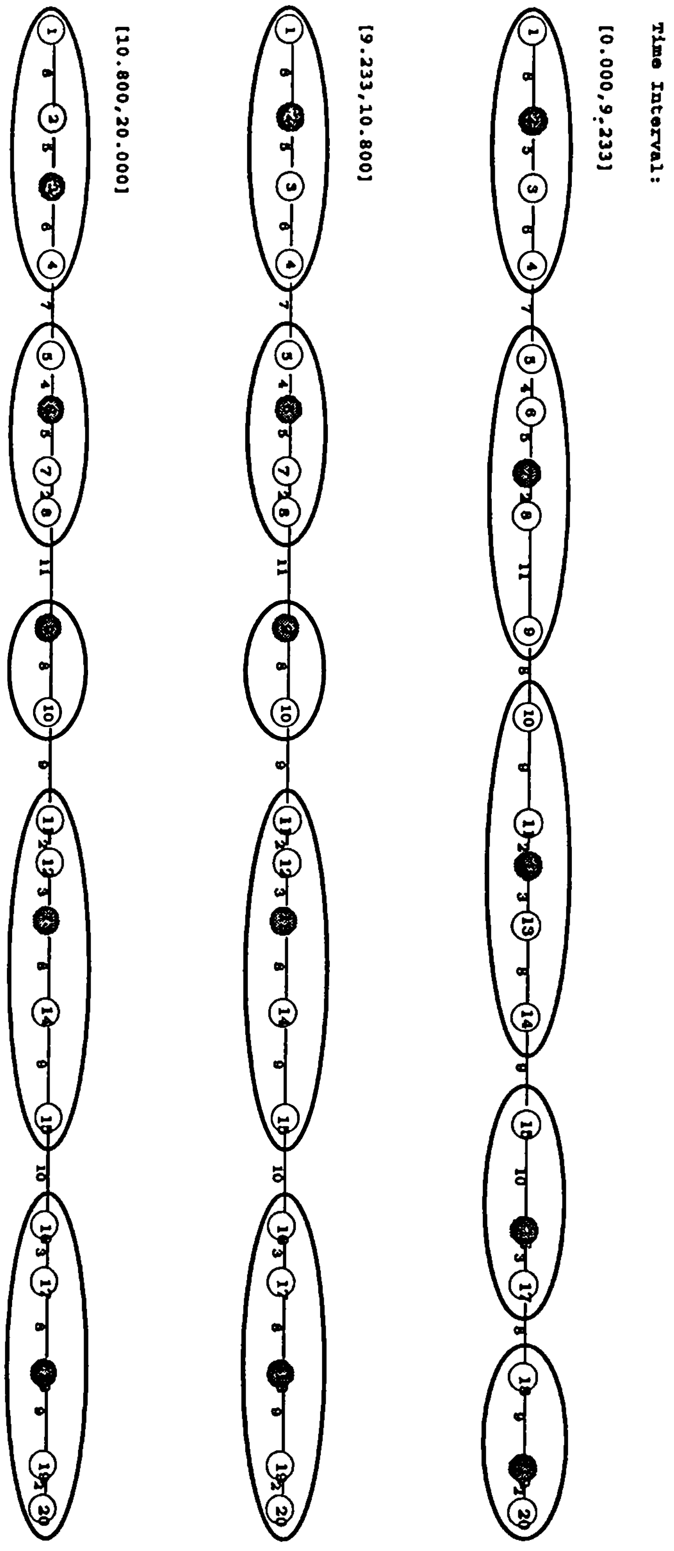


Figure 4.3. An example p_ML(t) problem on a line network

In *p_MML(t) problems*, a facility may *jump to a non-adjacent vertex* at a breakpoint. Thus, the "Connectedness Theorem" of Erkut and Tansel [22] for parametric single median problem on a tree network *does not* apply to the multimedian problem on tree and general networks when the changes in location are analyzed for each facility alone. The following figures are examples to this case on line, arbitrary and general networks.

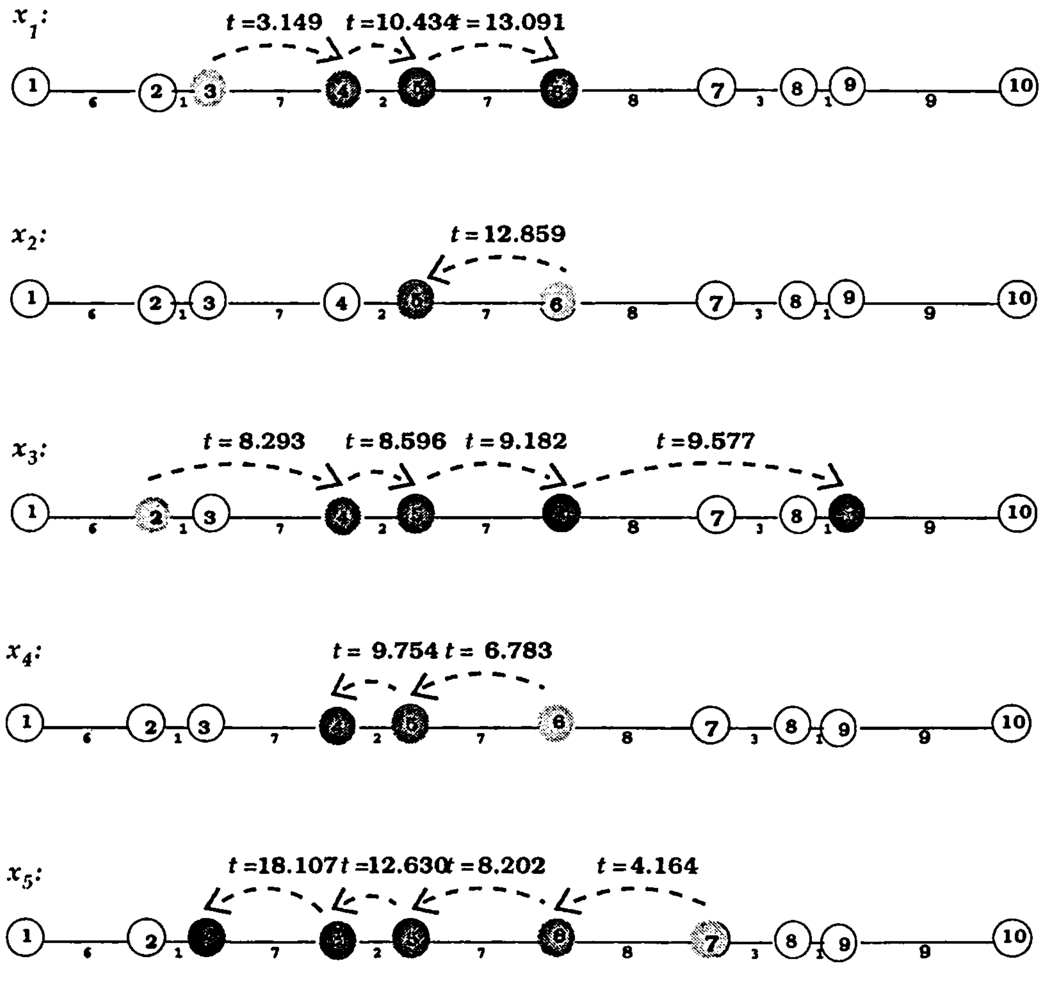

x_1:
t =3.149 t =10.434 t = 13.091
x_2:
t = 12.859
x_3:
t = 8.293 t = 8.596 t = 9.182 t =9.577
x_4:
t = 9.754 t = 6.783
x_5:
t =18.107 t =12.630 t =8.202 t =4.164
1 2 3 4 5 6 7 8 9 10
6 1 7 2 7 8 3 1 9

Figure 4.4. An example to discontinuity in individual facility movements in a p_MML(t) problem on a line network (There are jumps in the time path of facility 3)

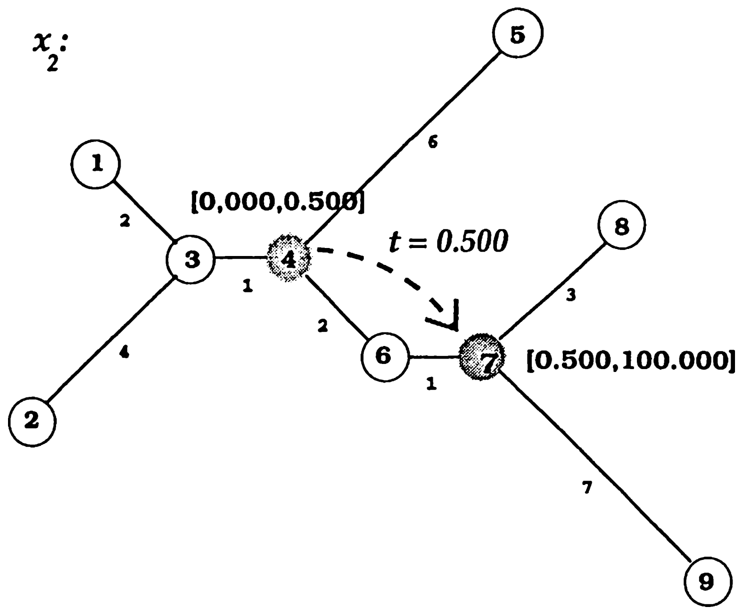


Figure 4.5. An example to discontinuity in individual facility movements in a p_MML(t) problem on an arbitrary tree

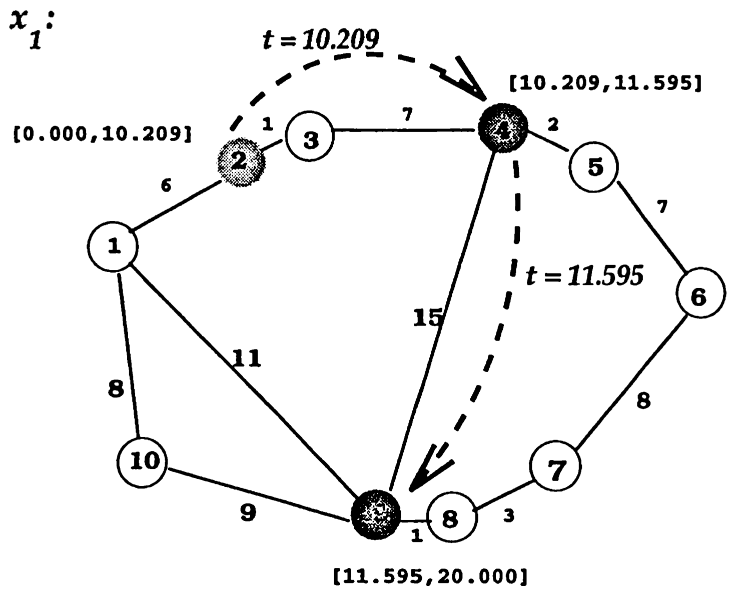


Figure 4.6. An example to discontinuity in individual facility movements in a p_MML(t) problem on a general network

Furthermore, a facility may return to its previous location by time, i.e. backtracking may occur. Thus, *monotonicity of the optimal locations is not guaranteed.* The following example depicts such a situation on a general network, where new facility 2 changes location back and forth between vertices 9 and 4.

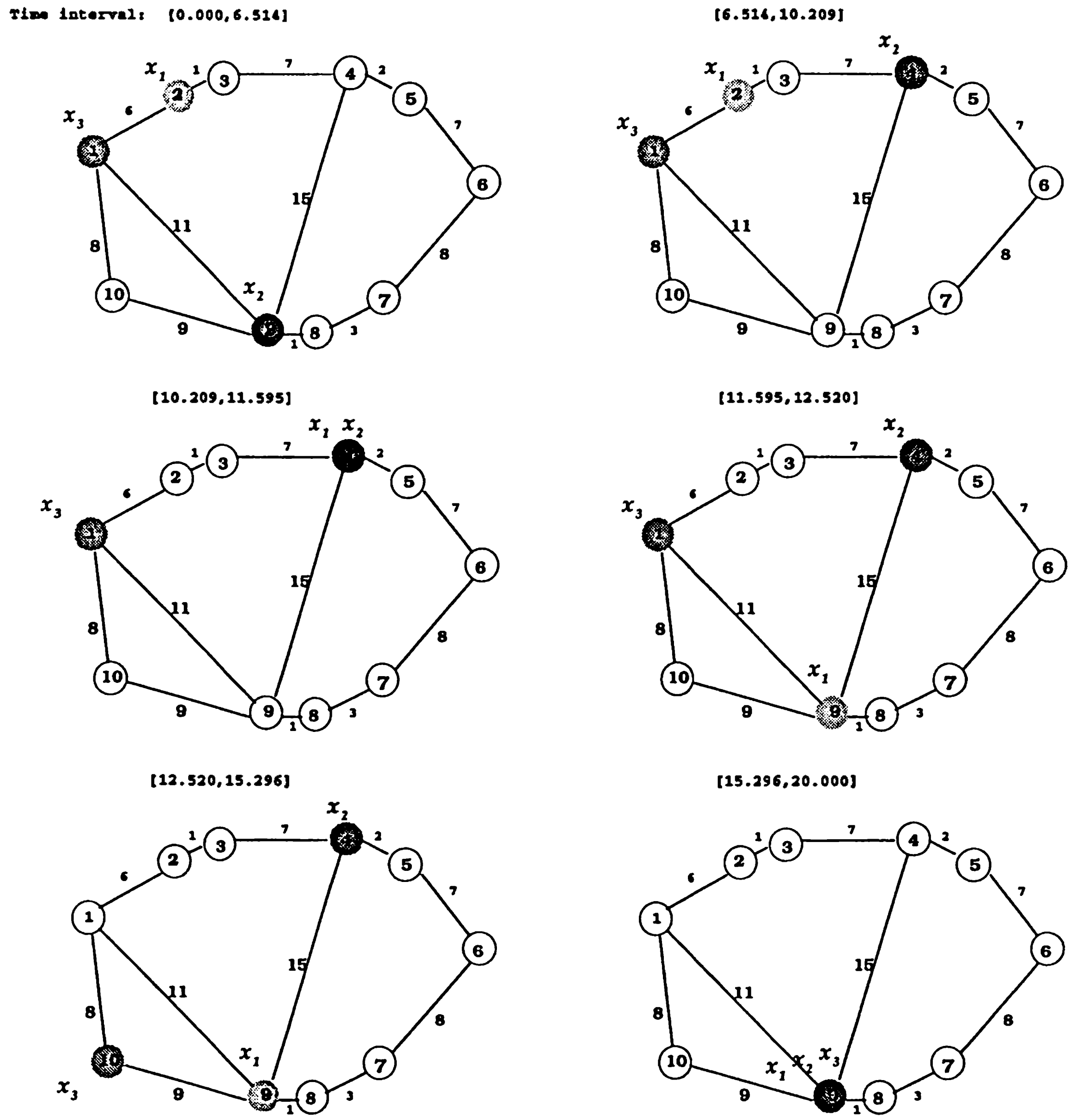


Figure 4.7. An example to backtracking in a p_MML(t) problem on a general network

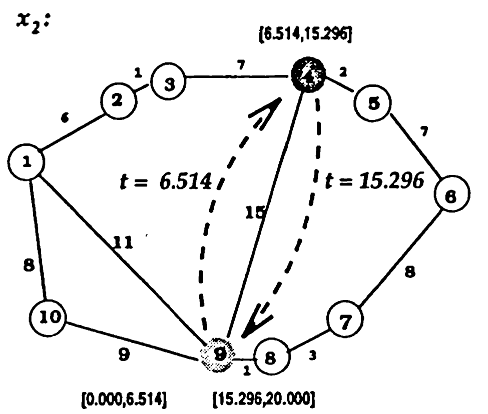


Figure 4.7 Continues. Backtracking in the time path of facility 2

From the above analysis, we see that the changes of locations at a breakpoint are erratic and a priori unidentifiable in the sense that properties that might help us find new locations based on previous ones at a breakpoint, such as having the optimal solution change in only one element to an adjacent node, do not exist. When such properties exist, they considerably reduce the effort to solve parametric problems. For example, in the single median problem with weights being continuous functions of time on trees, the time path of the optimal location of the facility is connected and monotonic. Using this property, Erkut and Tansel [22] have developed a very efficient algorithm to construct the trajectory. Drezner and Wesolowsky [19] also give a polynomial time algorithm to the single facility relocation problem on a plane using the monotonicity of the optimal location for the special case of rectilinear distances, linear weights and the restriction that only one breakpoint is allowed. If we had observed, for example, that new facilities move only to adjacent vertices or change locations in a monotonic way for p_MML(t) problems or that partitions remain unchanged or change in a certain pattern at a breakpoint for p_ML(t) problems, a large set of location vectors/sets could have been dominated at a breakpoint. Thus, we could have easily constructed the trajectory by finding the solution at the beginning of the time interval and then finding the time point at which the solution changes, by a search on the undominated solutions so that solving static problems could have been avoided. Unfortunately, p_median problems with linear weights do not seem to have such properties. Therefore, it seems that solving static problems to identify each solution at a linear piece of the trajectory is unavoidable, which justifies the argument that the trajectory construction algorithm presented is a best order one.

Chapter 5

Conclusion

In this study we have focused on multimedian location problems on time dependent networks. With the aim of facilitating locational decisions under circumstances where demand or transportation costs vary over time in a predictable way, we have introduced and analyzed p_median and p_median with mutual communication problems with weights being linear functions of time.

In making a location decision that involves a changing situation over a time horizon during which facilities will be operating, knowing the trajectory of the optimal solution throughout the time horizon is an asset. We have presented an algorithm that efficiently constructs the trajectory of the optimal solution for p_median and p_median with mutual communication problems with linear weights on both tree and cyclic networks. The trajectory is a piecewise linear concave function. If $\tilde{q}$ is the number of linear pieces in the trajectory, the algorithm constructs the trajectory by solving $O(\tilde{q})$ static p_median problems.Thus, the efficiency of the algorithm depends on the efficiency of static problem solutions.

The algorithm presented uses a lower bound to the length between breakpoints of the trajectory in terms of problem data with the assumption of integrality. The lower bound calculated in this study also bounds the length between intersection points of linear functions whose lower envelope determines the trajectory. With such a bound the trajectory can be more efficiently calculated. We have discussed the algorithm when either of the two bounds are used and given time complexity results.

Maybe the most essential merit of the algorithm is that it is general and can be applied to areas beyond facility location where the problem of finding the lower envelope of exponentially many linear functions arises. If we have a lower bound to the length between breakpoints in the lower envelope and an algorithm to find the minimum function at a time point in O(r) time, we could find the lower envelope in $O(r\tilde{q})$ time, where $\tilde{q}$ is the number of linear pieces in the lower envelope, using this algorithm.

We have implemented the algorithm that constructs the optimal trajectory and analyzed the behavior of p_median and p_median with mutual communication problems with linear weights on both tree and cyclic networks for some randomly generated problem instances. Trajectories were analyzed for special tree types like line and star as well as arbitrary trees and cyclic networks with a range of edge density. From the experimental studies, it is observed that time path of facility locations is neither monotonic nor connected. The number of breakpoints is extremely small compared to its upper bound. Facilities do not change their locations frequently, especially when there is mutual communication among them which forces them to be gathered in a central node.

As can be seen from the experimental studies, the trajectory of the optimal solution carries a lot of information about the system dynamics. Therefore, the trajectory is quite helpful in the decision making process. Knowing the special characteristics of the location network may reveal information about the tendency of facility locations over time based on a priori experimental studies. For example, if our network is a star, for P_MML(t), locating facilities at the central node, i.e. the hub, is well justified without solving the problem. The trajectory also facilitates relocation decisions. In situations where relocating the facility can be easily accomplished, following the path of the optimal locations would be profitable. In situations where relocating the facility is too costly and painful, knowing the optimal pattern could help to keep close to the optimal as far as possible. A recent study by Tansel on multifacility relocation problem with continuous time domain [50] is an evidence to the decision support aspect of the trajectory in relocation decisions. Tansel has shown that the points at which the necessity to relocate at least one facility arises, are closely related to the breakpoints of the trajectory. The breakpoints of the trajectory are input to the construction of a phase diagram which is used in the solution of the relocation problem.

Our analysis for the linear case can easily be extended to the piecewise linear case by partitioning the time horizon into intervals each with linear demands. Thus, nonlinear demands can also be analyzed by approximating non_linear functions with piecewise linear functions. However, there still remains p_median problems with non_linear weights to be analyzed for future work. Another direction for future work is incorporating time dependency to other problem elements as time dependent distances or changes in network topology by time.

Bibliography

[1] R.H. Ballou. Dynamic warehouse location analysis. *Journal of Marketing Research*, 5(2):271 - 276, 1968.

[2] M. Bastian and M. Volker. A perfect forward procedure for a single facility dynamic location/relocation problem. *Operations Research Letters*, 12(1):11 - 16, 1992.

[3] O. Berman, R.C. Larson, and S.S. Chiu. Optimal server location on a network operating as an m/g/1 queue. *Operations Research*, 33:746 - 770, 1985.

[4] M.L. Brandeau and S.S. Chiu. A center location problem with congestion. Working Paper, Department of Industrial Engineering and Engineering Management, Stanford University, 1988.

[5] M.L. Brandeau and S.S. Chiu. Establishing continuity of certain optimal parametric facility location trajectories. *Transportation Science*, 22(3):224 - 225, 1988. Technical Note.

[6] M.L. Brandeau and S.S. Chiu. Parametric facility location on a tree network with an lp-norm cost function. *Transportation Science*, 22(1):59 - 69, 1988.

[7] M.L. Brandeau and S.S. Chiu. An overview of representative problems in location research. *Management Science*, 35(6):645 - 674, 1989.

[8] M.L. Brandeau and S.S. Chiu. A unified family of queueing-location models. *Operations Research*, 38, 1990.

[9] M.L. Brandeau and S.S. Chiu. Parametric analysis of optimal facility locations. *Networks*, 21:223 - 243, 1991.

[10] J.F. Campbell. Locating transportation terminals to serve an expanding demand. *Transportation Research*, 24B(3):173 - 192, 1990.

[11] S. Chand. Decision/forecast horizon results for a single facility dynamic location/relocation problem. *Operations Research Letters*, 7(5):247 - 251, 1988.

[12] J. Cheriyan, T. Hagerup, and K. Mehlhorn. Can a maximum flow be computed in o(nm) time? Technical Report A 90/07, FB. Informatik, Univ. des Saarlandes, Saarbrucken, 1990.

[13] D. Chhajed and T.J. Lowe. M-median and m-center problems with mutual communication: Solvable special cases. *Operations Research*, 40:56 - 66, 1992.

[14] D. Chhajed and T.J. Lowe. An o(nm) algorithm for a special case of the multimedian location problem on a tree. *European Journal of Operations Research*, 63:222 - 230, 1992.

[15] S.S. Chiu. Optimal trajectory of the stochastic queue median as the demand rate varies. Working Paper, Engineering-Economic Systems Department, Stanford University, 1987.

[16] S.S. Chiu, O.Berman, and R.C. Larson. Locating a mobile server queueing facility on a tree network. *Management Science*, 31:764 - 772, 1985.

[17] P.M. Dearing, R.L. Francis, and T.J. Lowe. Convex location problems on tree networks. *Operations Research*, 24:628 - 642, 1976.

[18] E.A. Dinic. Algorithms for solution of a problem of maximum flow in networks with power estimation. *Soviet Mathematical Doklady*, 11:1277 - 1280, 1970.

[19] Z. Drezner and G.O. Wesolowsky. Facility location when demand is time dependent. *Naval Research Logistics*, 38:263 - 277, 1991.

[20] A. Dutta and J.I. Lim. A multiperiod capacity planning-model for backbone computer-communication networks. *Operations Research*, 40(4):689 - 705, 1992.

[21] E. Erkut and T.S. Õncü. A parametric 1-maximin location problem. *Journal of Operational Research Society*, 42(1):49 - 55, 1991.

[22] E. Erkut and B.C. Tansel. On parametric medians of trees. *Transportation Science*, 26(2):149 - 156, May 1992.

[23] D. Erlenkotter. Bibliography on dynamic location models. Discussion Paper No. 55. Management Science Study Center, Graduate School of Management, University of California, Los Angeles.

[24] D. Erlenkotter. Capacity planning for large multilocation systems: Approximate and incomplete dynamic programming approaches. *Management Science*, 22:274 - 285, 1975.

[25] D. Erlenkotter. Facility location with price sensitive demands: Private, public, quasi-public. *Management Science*, 24(4):378 - 386, 1977.

[26] D. Erlenkotter. A dual-based procedure for uncapacitated facility location. *Operations Research*, 26:992 - 1009, 1978.

[27] D. Erlenkotter. A comparative study of approaches to dynamic facility location problems. *European Journal of Operations Research*, 6:133 - 143, 1981.

[28] R.L. Francis. On the location of multiple new facilities with respect to existing facilities. *The Journal of Industrial Engineering*, 15(2):106 - 107, 1964.

[29] R.L. Francis, L.F. McGinnis, and J.A. White. *Facility Layout and Location.* 2nd Ed. Prentice-Hall, 1992.

[30] S.L. Hakimi. Optimum locations of switching centers and the absolute centers and medians of a graph. *Operations Research*, 12:450 - 459, 1964.

[31] S.L. Hakimi, M. Labbe, and E.F. Schmeichel. Locations on time-varying networks. pages 145 - 147. ISOLDE VI Proceedings, June 1993.

[32] J. Halpern. The location of a center-median convex combination on an undirected tree. *Journal of Regional Science*, 16:237 - 245, 1976.

[33] J. Halpern. Finding minimal center-median convex combination (centdian) of a graph. *Management Science*, 24:535 - 544, 1978.

[34] M. Healey. Components of locational change in multi-plant enterprises. *Urban Studies*, 20:327 - 341, 1983.

[35] S.K. Jacobsen. Heuristics for the capacitated plant location model. *European Journal of Operations Research*, 12(3):253 - 261, 1983.

[36] O. Kariv and S.L. Hakimi. An algorithmic approach to network location problems ii: The p-medians. *SIAM Journal of Applied MATH*, 37(3):539 - 560, December 1979.

[37] A.V. Karzanov. Determining the maximum flow in a network by the method of preflows. *Soviet Mathematical Doklady*, 15:434 - 437, 1974.

[38] A.J.W. Kolen. Equivalence between the direct search approach and the cut approach to the rectilinear distance location problem. *Operations Research*, 29(3):617 - 621, 1981.

[39] A.J.W. Kolen. Location problems on trees and in the rectilinear plane. Stitchting Mathematisch centrum, Kruislaan 413, 1098, SJ Amsterdam, The Netherlands, 1982.

[40] H. Luss. Operations research and capacity expansion problems: A survey. *Operations Research*, 30(5):907 - 947, 1982.

[41] D.W. Matula and R. Kolde. Efficient multi-median location in acyclic networks. Presented at ORSA-TIMS meeting, November 1976.

[42] P.B. Mirchandani and R.L. Francis. *Discrete Location Theory*. John Wiley and Sons,Inc., 1990. edited.

[43] A. Orda and R. Rom. Location of central nodes in time varying computer networks. *Operations Research Letters*, 10:143 - 152, 1991.

[44] J.C. Picard and H.D. Ratliff. A cut approach to the rectilinear distance facility location problem. *Operations Research*, 26(3):422 - 433, 1978.

[45] T. Van Roy and D. Erlenkotter. A dual based procedure for dynamic facility location. *Management Science*, 10:1091 - 1105, 1982.

[46] D.R. Shier and P.M. Dearing. Optimal locations for a class of nonlinear location problems on a network. *Operations Research*, 31:292 - 303, 1983.

[47] A. Shulman. An algorithm for solving dynamic capacitated plant location problems with discrete expansion sizes. *Operations Research*, 39(3):423 - 436, 1991.

[48] D.J. Sweeney and R.L. Tatham. An improved long run model for multiple warehouse location. *Management Science*, 22(4):748 - 758, 1976.

[49] A. Tamir. Complexity results for the pmedian problem with mutual communication. To appear in Operations Research Letters, May 1993.

[50] B.C. Tansel. Median location on time dependent networks. pages 265 - 268. ISOLDE VI Proceedings, June 1993.

[51] B.C. Tansel, R.L. Francis, and T.J. Lowe. Location on networks: A survey. part i: The p-center and p-median problems. *Management Science*, 29(4):482 - 497, 1983.

[52] B.C. Tansel, R.L. Francis, and T.J. Lowe. Location on networks: A survey. part ii: Exploiting tree network structure. *Management Science*, 29(4):498 - 511, 1983.

[53] G.O. Wesolowsky. Dynamic facility location. *Management Science*, 19(11):1241 - 1248, 1973.

[54] G.O. Wesolowsky and W.G. Truscott. The multiperiod location-allocation of facilities. *Management Science*, 22(1):57 - 65, 1975.