A SPANNING TREE APPROACH TO SOLVING THE ABSOLUTE p-CENTER PROBLEM

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

By
Burçin Bozkaya
May, 1995

A SPANNING TREE APPROACH TO SOLVING THE ABSOLUTE p-CENTER PROBLEM

A THESIS

SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL

ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCES

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Burçin Bozkaya

May, 1995

Burçin Bozkaya
tarafından [illegible]

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.

Assoc. Prof. Barbaros 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.

Assoc. Prof. Osman Oğuz

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.

Assoc. Prof. İhsan Sabuncuoğlu

Approved for the Institute of Engineering and Sciences:

Prof. Mehmet Baray
Director of Institute of Engineering and Sciences

To my parents and to my Gökçe

ABSTRACT

A SPANNING TREE APPROACH TO SOLVING THE ABSOLUTE p-CENTER PROBLEM

Burçin Bozkaya
M.S. in Industrial Engineering
Supervisor: Assoc. Prof. Barbaros Tansel
May, 1995

The p-center problem on a network is a model to locate p new facilities that will serve n existing demand points on that network. The objective is to minimize the maximum of the weighted distances between each demand point and its nearest new facility. This type of problem usually arises in the location of emergency facilities like hospitals, police and fire stations. The problem is known to be $\mathcal{NP}$-Hard on a cyclic network, but polynomial-time solvable on a tree network. In this study, a spanning tree approach to solving the problem on a cyclic network is discussed. First, the existence of an *optimal* spanning tree that gives the network optimal solution, is proved. Then, two specific types of spanning trees are introduced and experimentally tested whether they contain the optimal tree or not. Also, some properties of such an optimal tree are discussed and some special cases for which the optimal tree can be determined in polynomial time, are identified.

Keywords : p-center, covering.

ÖZET

p-MERKEZ PROBLEMİ ÇÖZÜMÜNE KAPSARAĞAÇ YAKLAŞIMI

Burçin Bozkaya
Endüstri Mühendisliği Yüksek Lisans
Tez Yöneticisi: Doç. Dr. Barbaros Tansel
Nisan, 1995

p-Merkez problemi, bir serim üzerinde yeralan n talep noktasına hizmet verecek p merkezin serim üzerine yerleştirilmesini kapsamaktadır. Amaç, talep noktaları ile hizmet aldıkları en yakın merkezler arasındaki en büyük ağırlıklı uzaklığı enküçüklemektir. Uygulamada bu probleme, hastane, karakol, itfaiye gibi, acil hizmet gerektiren birimlerin yerleştirilmesi örnek gösterilebilir. Problemin çözümünün, genel serimlerde $\mathcal{NP}$-Zor, ağaç serimlerde ise polinom zamanlı olduğu bilinmektedir. Bu çalışmada, problemi genel serimlerde çözmeye yönelik, o serimin kapsarağaçlarının kullanıldığı bir yaklaşım önerilmektedir. Öncelikle, serim eniyi çözümünü veren bir *eniyi* ağacın varlığı gösterilmiş, daha sonra iki ayrı özel ağaç tipinin serim eniyi çözümünü verip vermediği deneysel olarak incelenmiştir. Ayrıca, eniyi ağacın ne gibi özelliklere sahip olduğu araştırılmış, belli özel durumlar için eniyi ağacın polinom zamanda bulunabileceği gösterilmiştir.

Anahtar sözcükler : p-merkez, kaplama.

ACKNOWLEDGEMENT

I am very grateful to my supervisor, Assoc. Prof. Barbaros Tansel whose guidance and support was the primary source of my motivation in this study. His valuable instruction will continue to be the key reference in my future academic career.

I would like to thank to Assoc. Prof. İhsan Sabuncuoğlu and Assoc. Prof. Osman Oğuz for their keen interest and valuable comments on the subject matter.

Special thanks are for my dear Gökçe, for her endless patience and support throughout this study.

I would especially like to thank to my parents for their contributions on me and my educational background.

Finally, I would like to thank to my roommates Dilek Yılmaz and Samir Elhedhli, and all the others who have contributed to this study in some way.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

A significant part of the literature on the formulation and analysis of facility location problems is on locating new facilities on a network. We refer to this related set of problems as "Network Location Problems". In particular, the network under consideration might be a transportation network or a road network which contains demand points or customers located on the network. One typical case is when the demand points on the network are taken to be the network's vertices and new facilities provide service through shortest paths reaching these demand points. Usually, there is a cost of providing material or service from a service facility to a demand point and the objective is to locate the new facilities so that some function of the costs is optimized.

We can distinguish between three major types of service facilities that are to be located : center-type (emergency-type, minimax) facilities which have to be located in such a way that they can respond to service calls as soon as possible (like hospitals, fire stations, police stations, etc.) ; median-type facilities (minisum) which provide service to minimize the total cost ; and obnoxious-type (undesired, maximin or maxisum) facilities that are to be located as far as possible from a number of given points. The absolute p-center problem is of the first type ; namely, p new facilities are to be located with an objective to minimize the maximum response time to service requests made by

demand points. When the response time is a linear function of distances between demand points and service facilities, the objective reduces to minimizing the maximum of the 'weighted' distances between each demand point and its closest new facility.

To give the formal statement of the problem, let $N = (V, E)$ be a transportation network where

- $V = \{v_1, v_2, \ldots, v_n\}$ is the set of vertices and
- $E = \{e_{ij} = (v_i, v_j)\}$ is the set of edges of N.

We take each edge as a rectifiable arc of known positive length and by a point of the network N, we mean any point along any edge. A point may be interior to an edge or may be one of the endpoints of the edge. The following notation will be used throughout the remainder of the thesis :

$P_N(x, y)$		A path joining two points x and y on N.
$\|P_N(x, y)\|$	:	Length of $P_N(x, y)$.
$\|e_{ij}\|$		Length of edge $e_{ij} = (v_i, v_j)$.
$d_N(x, y)$	:	Length of a shortest path between x and y on N.
$X_N = \{x_1, x_2, \ldots, x_p\}$		The set of locations of p new facilities to be located on N.

The assumptions that are imposed on the problem are as follows :

A1. N is a simple and connected network.

A2. Demand points are taken to be vertices of N.

A3. Any point of N is a feasible location for each x_j, $j = 1, \ldots, p$.

A4. All edges of N satisfy $|e_{ij}| = d_N(v_i, v_j)$ (the ones that violate this equality are called *redundant edges* and are deleted from N without causing suboptimality (Kariv and Hakimi 1979 [25])).

A5. A nonnegative weight w_i is associated with each v_i, $i = 1, \ldots, n$ (usually interpreted as the frequency of service request or the relative importance

of the demand points). A weight of zero on a vertex implies that vertex is not actually a demand point but is viewed as one, as a modeling convenience.

A6. Each vertex receives service from the nearest center (ties are broken in favor of the smallest indexed center).

To see how these assumptions are applicable to real life, consider the road network of a city where the intersections and the streets correspond to vertices and edges of N, respectively. In practice, the intersections (vertices of N) might be quite probable locations for accidents and taken to be the demand points (w_i being the probability of accident on the intersection v_i). In case of an accident, the nearest health service facility is in charge, in order to provide the quickest response (assumption A6).

Now, let the shortest path distance on N between a vertex v_i and a nearest center be denoted by $D_N(v_i, X_N)$, where

$$D_N(v_i, X_N) = \min_{j=1,\ldots,p} d_N(v_i, x_j) \tag{1.1}$$

Then the statement of the problem is the following :

$$z_p^*(N) = f_N(X_N^*) = \min_{X_N} f_N(X_N) \tag{1.2}$$

where,

$$f_N(X_N) = \max_{v_i \in V} w_i \cdot D_N(v_i, X_N) \tag{1.3}$$

Any optimal set of locations X_N^* is referred to as an *absolute p-center* of N and $z_p^*(N)$ is referred to as the *absolute p-radius* of N. Throughout the remainder of the thesis, "p-center" and "p-radius" will be used, respectively, instead of these terms.

The problem is called *unweighted* (versus weighted) when $w_i = 1$, for all $i = 1, \ldots, n$. It is *vertex restricted* (versus absolute) when each x_j is restricted to vertices of N. It becomes *continuous* when not only the vertices but all the points of N are demand points.

There is a considerable amount of research on this problem. A detailed literature review will be provided in Section 2.1. One main characteristic of the problem is that it is $\mathcal{NP}$-Hard when the network is a general one (e.g. one containing cycles) and polynomial-time solvable when it is a tree network (Kariv and Hakimi 1979 [25]). Hence, for a cyclic network N, one might be interested in solving the problem on a spanning tree T of N instead of solving on the network itself and still expect to find a p-center of N (assume that T has the same vertex weights as N). Note that a spanning tree $T = (V', E')$ of N contains all the vertices of N (i.e. $V' = V$) but many edges are deleted. Denoting by $d_T(x, y)$ the length of a shortest path between $x, y \in T$ when paths are restricted to edges of T, we have

$$d_T(x, y) \geq d_N(x, y)$$

since there are additional edges on N that are not on T and a shortest path on N between x and y may use some of these edges. By a similar argument, the p-radius of T is greater than or equal to the p-radius of N because the additional edges may provide shorter shortest paths between vertices and centers so that the distance between a vertex and a nearest center may be smaller on N. In addition, the additional edges on N contain additional feasible locations for x_j's which may contain better locations than those of T. Hence T and N satisfy

$$z_p^*(T) \geq z_p^*(N)$$

So, the following major question arises as a consequence : does there exist a spanning tree $\bar{T}$ (with the same vertex weights), what we will call an *optimal tree*, of N which has the same p-radius as N. In other words, is there a spanning tree T of N which satisfies

$$z_p^*(T) = z_p^*(N) \ ?$$

Observe that if equality holds, then a p-center of T is also a p-center of N. If the existence of such an optimal tree is proved, the next question is whether or not such a tree can be found in polynomial time. If this is the case, the absolute p-center problem can be efficiently solved by identifying the optimal

tree $\bar{T}$ and solving the problem on $\bar{T}$, meaning that $\mathcal{P} = \mathcal{NP}$. Therefore, to find an optimal tree (if exists) in polynomial time must be as hard as the absolute p-center problem itself. However, if the existence of an optimal tree $\bar{T}$ is known for a particular network N and some properties of $\bar{T}$ can be identified, these properties might be used to eliminate some of the spanning trees of N and to search $\bar{T}$ in the remaining set. Even if the remaining set does not contain a polynomial number of spanning trees, a search on a polynomial size subset of the remaining set (that is expected to contain trees similar to $\bar{T}$) might be worth considering.

In this study, the answers to the above questions are investigated. First, the existence of such an optimal tree, for an arbitrary cyclic network N, is proved. Note that such a tree cannot be found in polynomial time simply by enumerating all the spanning trees of N, since the number of these trees is not generally polynomial in n. Next, two types of spanning trees are introduced in order to limit the search for the optimal tree. These types of trees are, then, experimentally tested on random cyclic networks, to see whether they contain the optimal tree or not. Also, some properties of the optimal tree (that help to identify its edges) are discussed and some special cases for which the optimal tree can be identified in polynomial time are given.

The organization of the thesis is as follows : in Chapter 2, a literature review regarding the absolute p-center problem and some of its versions is given and the algorithms that are implemented in the experimental part of this study are discussed. In Chapter 3, the spanning tree approach to the problem is described. In Chapter 4, an experimental study on the two types of spanning trees introduced in Chapter 3 is carried and the results are discussed. Finally, Chapter 5 is the conclusion of the study where, also, the future work is discussed.

Chapter 2

The Literature Review and The Algorithms

2.1 The Literature Review

The absolute p-center problem is one of the fundamental problems in the network location theory. Different versions of the problem may be identified with respect to being weighted or unweighted, absolute or vertex-restricted, or being continuous. For this reason, the related literature will be grouped and reviewed under the following major categories :

- Absolute/Vertex Restricted 1-Center Problem
- Absolute/Vertex Restricted p-Center Problem, $p > 1$
- Heuristics

Tansel, Francis and Lowe 1983 [35] provide a survey on the literature regarding network location problems. The above outline is similar to theirs in the classification of different versions of the problem except the exclusion of the p-median problem and the additional review of heuristics.

2.1.1 Absolute/Vertex Restricted 1-Center Problem

The weighted absolute 1-center problem was first defined and solved by Hakimi 1964 [12]. In his study, he looks at the function

$$f_N(X_N) = \max_{i=1,\ldots,n} w_i \cdot D_N(v_i, X_N)$$

on each edge of the network, finds the local minimum center, and selects the best among all such $O(|E|)$ points as the absolute 1-center. His method requires a computational effort of $O(|E| \cdot n^2 \cdot \log n)$ (as shown by Hakimi, Schmeichel and Pierce 1978 [14]) which becomes $O(|E| \cdot n \cdot \log n)$ for the unweighted case. Kariv and Hakimi 1979 [25] provide a refinement to this procedure and propose $O(|E| \cdot n \cdot \log n)$ and $O(|E| \cdot n)$ algorithms for the weighted and unweighted cases, respectively. Minieka 1981 [30] gives an $O(n^3)$ algorithm for the unweighted case.

Minieka 1980 [30] considers the "conditional" 1-center problem, where new facilities are located not only with respect to the existing facilities but with respect to themselves as well. He shows that this conditional problem can be reduced to an unconditional one.

Hooker 1986 [20] gives an algorithm that solves the version of the problem with a nonlinear convex cost function. He provides a unified approach to all versions of the problem for which particular solution procedures already exist. The procedure is based on solving convex subproblems on what Hooker calls 'treelike segments'. He uses the findings of Dearing, Francis and Lowe 1976 [6] and proposes a reasonably efficient algorithm.

The above procedures are devised for general networks, and also applicable to tree networks. However, when the network is a tree, which is a simpler structure, there are more efficient methods. Handler 1973 [15] proposes an $O(n)$ algorithm for the unweighted case. His method is based on locating the 1-center at the midpoint of any longest path of N. For the weighted case in general, Dearing and Francis 1974 [5] show that $z_1^*(N)$ is bounded below by

$\beta_{\max} = \max_{1 \le i < j \le n} \beta_{ij}$ where

$$\beta_{ij} = \frac{d_N(v_i, v_j)}{\frac{1}{w_i} + \frac{1}{w_j}}$$

Further, they show that this lower bound is always realized for tree networks. In this case, $z_1^*(N)$ is equal to β_{st} for some vertices $v_s, v_t \in V$ and the 1-center is located uniquely on the path joining these two 'critical' vertices. Hakimi, Schmeichel and Pierce 1978 [14] reduce the computational effort for computing $\beta_{\max}$. Related to this, Kariv and Hakimi 1979 [25] propose an $O(n \cdot \log n)$ algorithm for solving the absolute 1-center problem on a tree.

Dearing 1977 [4] and Francis 1977 [9] provide a similar bound for the nonlinear version. Shier and Dearing 1983 [33] suggest some necessary and sufficient conditions for the local optimum of the problem, for the same version. Further, they unify the known results for tree networks to a single procedure.

The research on the vertex restricted 1-center problem goes as early as 1869. Jordan 1869 [24] refers to this problem as a graph theoretic problem. Hakimi 1964 [12] gives an $O(n^3)$ algorithm based on Jordan's findings. However, Hedetniemi, Cockayne and Hedetniemi 1981 [18] propose $O(n)$ algorithms for the unweighted version, which are the most efficient ones to date.

For the weighted version, Rosenthal, Hersey, Pino and Coulter [32] present a generalized algorithm that solves a number of problems on trees one of which is the 1-center problem.

The versions of the 1-center problem are well-studied and there exist very efficient algorithms for this class of problems. More recent research on absolute p-center problem is mostly focused on exploring more general versions like the weighted absolute p-center problem with $p > 1$. There are also heuristics available in the literature. These will be discussed in the following two sections.

2.1.2 Absolute/Vertex Restricted p-Center Problem, $p > 1$

The absolute p-center problem was first formulated by Hakimi 1965 [13]. The solution procedures proposed by Hakimi basically rely on solving a sequence of set covering problems.

The *set covering problem* and the *covering problem* are closely related with the p-center problem. The set covering problem is the following : Let A be an $m \times n$ matrix of zeros and ones. The matrix A is such that whenever $a_{ij} = 1$ for some $1 \leq i \leq m$ and $1 \leq j \leq n$, a demand point of row i is *covered* by the supply point of column j. The objective is to select a minimum number of columns so that each row is covered by at least one column. The covering problem $C(z)$ is similar to this problem. The objective is to locate the minimum number of centers on a network N, so that $w_i \cdot D_N(v_i, X_N) \leq z$ is satisfied for all vertices $v_i \in V$, i.e. each vertex v_i is covered by a center within a distance of z/w_i. If the minimum value to the covering problem $C(z)$ is denoted by $n^*(z)$, then $z_p^*(N)$ satisfies

$$n^*(z_p^*(N)) \leq p \quad \text{and} \quad n^*(z) > p \quad \text{for all} \quad z < z_p^*(N).$$

The covering problem can be solved by solving an associated set covering problem. And most of the solution procedures for solving the absolute p-center problem rely on solving a sequence of covering problems. However, both the set covering and covering problems are known to be $\mathcal{NP}$-Hard (Kariv and Hakimi 1979 [25]). Still, there exist finite algorithms for solving the absolute p-center problem.

For the unweighted case, Minieka 1970 [28] shows that the absolute p-center is restricted to the set $P = P' \cup V$ where the set P' contains those points x (intersection points) such that $d_N(v_i, x) = d_N(v_j, x)$ is satisfied for some distinct $v_i, v_j \in V$ on some edge e. The procedure proposed by Minieka is then based on solving a finite number of set covering problems. Garfinkel, Neebe and Rao 1977 [11] provide some reduction rules in the number of columns and

Handler 1979 [17] proposes similar rules for reduction in both the number of rows and columns.

For the weighted case, Christofides and Viola 1971 [3] suggest a solution procedure that solves a sequence of covering problems where, also, the solutions for $n-1, n-2, \ldots, p+1$ centers are obtained. Kariv and Hakimi 1979 [25] show the $\mathcal{NP}$-Hardness of the weighted absolute p-center problem, where they suggest a finite procedure of $O(|E|^p \cdot (n^{2p-1})(\log n)/(p-1)!)$. The complexity is reduced to $O(|E|^p(n^{2p-1})/(p-1)!)$ for the unweighted case.

One important information about the p-center problem is that of finding an approximate solution to either the vertex restricted or the absolute version whose value is within 100% or 50% of $z_p^*(N)$ is $\mathcal{NP}$-Hard (Hsu and Nemhauser 1979 [22]). This means that the discovery of any polynomial heuristic that provides a worst-case bound of δ times optimal value, $\delta < 2$, will lead to $\mathcal{P} = \mathcal{NP}$. Hence, the best polynomial heuristic turns out to give 2 times optimal in the worst case (assuming $\mathcal{P} \neq \mathcal{NP}$). In fact, such a heuristic exists and will be discussed in the following section.

For the vertex restricted p-center problem, Toregas, Swain, ReVelle and Bergman 1971 [38] propose a solution procedure again based on solving a sequence of covering problems. Hooker 1989 [21] gives an algorithm that solves the multi-facility nonlinear problem efficiently for small p but admits higher values of p when some facilities are fixed beforehand.

Since the problem is $\mathcal{NP}$-Hard, there only exist **finite** algorithms some of which perform well on large scale problems. However, when the underlying network is a tree, polynomial-time algorithms for the problem exist.

Handler 1978 [16] considers a special case of the absolute p-center problem, for $p = 2$, and proposes two $O(n)$ algorithms. His method is based on splitting the tree into two at an absolute 1-center and then solving 1-center on each part. Kariv and Hakimi 1979 [25] describe an $O(n^2 \cdot \log n)$ algorithm. They show that $z_p^*(N)$ is one of the β_{ij} values (see Section 2.1.1) and solve the p-center problem by identifying all such values and performing a binary search on this

set of values. For each value picked, the covering problem is solved in $O(n)$ time.

Tansel, Francis, Lowe and Chen 1982 [37] discuss the nonlinear p-center problem which also includes distance constraints that impose upper bounds on $D_N(v_i, X_N)$, $\forall v_i \in V$.

Chandrasekaran and Daughety 1981 [1] describe a method for solving the *continuous p-center problem.* They suggest an $O(n)$ algorithm for solving the related covering problem and then an $O((n \cdot \log p)^2)$ algorithm that solves the continuous p-center problem. Chandrasekaran and Tamir 1980 [2] propose a different method that uses the concept of intersection graph. The intersection graph contains those edges (v_i, v_j) with $v_i, v_j \in V$, such that v_i and v_j can both be covered by a single center within a distance of z/w_i and z/w_j, respectively. A clique cover found on this intersection graph provides a solution to the problem $C(z)$. This procedure is polynomial in n and p. For the same problem, Tamir 1985 [34] shows that the p-radius is a rational number p_1/p_2 where $p_i = O(n^5 \log d + n^6 \log p)$ with d being the sum of edge lengths and uses this result to develop a finite algorithm. Megiddo, Tamir, Zemel and Chandrasekaran 1981 [27] give an $O(n \cdot \log^2 n)$ algorithm to find the k-th longest path in a tree and using this algorithm, they develop an $O(n \cdot \min(p \cdot \log^2 n, n \cdot \log p))$ algorithm for the continuous p-center problem.

2.1.3 Heuristics

The $\mathcal{NP}$-Hardness of the absolute p-center problem is proved by Kariv and Hakimi 1979 [25], by reducing the covering problem (whose $\mathcal{NP}$-Hardness they also prove) to the absolute p-center problem. Consequently, some research has been made on finding approximate solutions to the problem, i.e. developing heuristics. Dyer and Frieze 1985 [7] suggest one for the vertex restricted version, which has a computational effort $O(np)$ and a worst-case bound of $\min(3, 1+\alpha)$ times $z_p^*(N)$ where $\alpha = \max_i w_i / \min_i w_i$. The procedure starts with locating the first center at the vertex with the maximum weight and carries on locating,

at each iteration, a center at the vertex that has the maximum current weighted distance (among all the vertices) to its nearest center.

Hochbaum and Shmoys 1985 [19] propose an algorithm that gives approximate solutions to the vertex restricted p-center problem that are no more than **twice** the optimal solution. Their algorithm assumes the triangular inequality on edge lengths, i.e. $|e_{ij}| \leq |e_{ik}| + |e_{kj}|$ for all triple, $v_i, v_j, v_k \in V$. The procedure uses the concepts of *square of a graph* and the *dominating set problem.* The dominating set problem is analogous to the covering problem for the vertex restricted case. The algorithm basically identifies a dominating set on the square of N which is shown to give an objective function value no more than twice the $z_p^*(N)$. The algorithm is $O(|E| \cdot \log |E|)$.

For the vertex restricted case, Martinich 1988 [26] propose two algorithms of computational complexity $O(|E| \cdot \log |E|)$ and $O(|E|^2)$. He proves that both of the algorithms converge to optimum for special cases and perform very well for relatively large values of p/n.

The best heuristic in terms of the worst-case bound is given by Plesník 1987 [31]. He generalizes Hochbaum and Shmoys' algorithm to the absolute p-center problem with a polynomial time algorithm. The algorithm is the best possible one because due to Hsu and Nemhauser 1979 [22], finding an approximate solution to the problem within less than 100% times optimal is $\mathcal{NP}$-Hard.

2.2 The Algorithms

In this section, the algorithms that are used in the experimental part of the study are discussed. Since the problem is $\mathcal{NP}$-Hard on cyclic networks, the corresponding algorithm is therefore not polynomial. Both algorithms rely on solving a sequence of *covering* problems, hence the covering problem has to be introduced before presenting the algorithms for the absolute p-center problem. Here is the formulation of the covering problem $C(z)$, defined on N with the

same assumptions A1-A6 stated in Chapter 1 :

$$\begin{aligned} n^*(z) = \quad & \min |X_N| \\ & s.t. \\ & w_i \cdot D_N(v_i, X_N) \leq z \quad (i = 1, \ldots, n) \end{aligned} \tag{2.1}$$

In the covering problem, we want to locate a minimum number of centers on N such that for every vertex $v_i, i = 1, \ldots, n$, a center is located within a distance of z/w_i units, for fixed z. The relationship between this problem and the p-center problem is quite straightforward : $z_p^*(N)$ satisfies

$$n^*(z_p^*(N)) \leq p \quad \text{and} \quad n^*(z) > p \quad \text{for all} \quad z < z_p^*(N)$$

That is, we cannot reduce $z_p^*(N)$ anymore unless we are allowed to locate additional center(s).

The covering problem is $\mathcal{NP}$-Hard on cyclic networks and polynomial time solvable on tree networks (Kariv and Hakimi 1979 [25]). These authors suggest an $O(n)$ algorithm for the tree case. The cyclic case can be solved in

$$O\left(\begin{pmatrix} |E| \cdot n^2 \\ p \end{pmatrix}\right) = O\left(\frac{|E|^p \cdot n^{2p}}{p!}\right) \tag{2.2}$$

time by solving an associated set covering problem with an $O(|E| \cdot n^2)$ number of columns, once all the intersection points of N are computed (see Section 2.2.1 for the definition of an intersection point).

2.2.1 On a Cyclic Network

Most of the algorithms for solving the absolute p-center problem on cyclic networks rely on solving a sequence of covering problems. In this study, for the purpose of solving the problem in the experimental part, the combination of the algorithm by Minieka 1970 [28] and the findings of Kariv and Hakimi 1979 [25] is used.

Kariv and Hakimi show that the optimal p-center is restricted to some set $P = P' \cup V$. This is an extension of the results of Minieka 1970 [28] where he showed the same thing for the unweighted case. The set P' contains those points $x \in e_{ij}$ for some edge $e_{ij} \in E$, such that there exist two distinct vertices $v_p, v_q \in V$ that satisfy

$$w_p \cdot d_N(v_p, x) = w_q \cdot d_N(v_q, x) \tag{2.3}$$

To explain the idea more clearly, let $\bar{D}_{ij}(v_k, t, N) = w_k \cdot d_N(v_k, t)$ be the weighted distance between v_k and a varying point t on edge e_{ij}. We plot this function as in Figure 2.2.1a. Observe that the slope of $\bar{D}_{ij}(v_k, t, N)$ is equal to $\pm w_k$. As shown in the figure, the function can have three shapes. A (B) corresponds to the case where a shortest path joining v_k and t, $\bar{P}_N(v_k, t)$, passes through v_i (v_j) for all $t \in e_{ij}$, while C corresponds to the case where $\bar{P}_N(v_k, t)$ passes through v_i for t in the subedge $[v_i, a]$ and through v_j for t in the subedge $[a, v_j]$. Here, a is the unique interior point of e_{ij} such that there are two shortest paths of equal length from a to v_k, one visiting v_i and the other visiting v_j. That is, a is the point where $d_N(v_k, t)$ is the same regardless of which vertex the shortest path passes through.

Now we can define the term *intersection point.* An intersection point is a point $x \in e_{ij}$ where two weighted distance functions $\bar{D}_{ij}(v_p, \cdot, N)$ and $\bar{D}_{ij}(v_q, \cdot, N)$ intersect, one with a positive, the other with a negative slope (see Figure 2.2.1b). The set P' is the set of such intersection points computed for all pairs v_p, v_q on each edge e_{ij} ($w_p, w_q > 0$ is required to maintain positive and negative slopes). Because we require opposite signs on slopes, v_p and v_q are two distinct vertices that satisfy (2.3). Once an intersection point is known to be a candidate location for an x_j (or a p-center is known to be restricted to the set $P' \cup V$), a candidate value for the p-radius $z_p^*(N)$ is the value of the functions at an intersection point, i.e. $\bar{D}_{ij}(v_p, x, N)$. In this way, we can compute all the candidate values as we compute all the intersection points.

The important implication of the above result is that the p-center and $z_p^*(N)$ can be the computed by performing a finite search on the set of candidate values. The algorithm, though not polynomial, relies on this fact. A broad

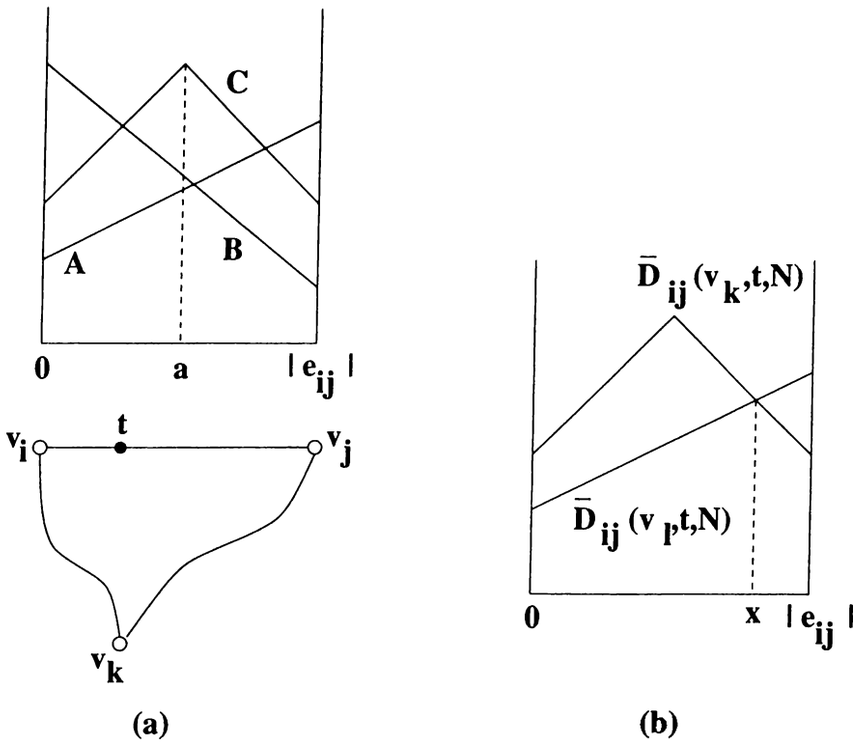

Figure 2.1: Weighted distance functions and intersection points

scheme of the algorithm is given below (we assume the distance matrix of N is constructed a priori, in $O(n^3)$ time, see Floyd's algorithm in [10]) :

p-CENT

0. Find all the intersection points on all edges and let $Z = \{z_1, \ldots, z_l\}$ be the ordered (ascendingly) set of distinct weighted distance function values associated with the intersection points. Set $z_{low} \leftarrow z_1, z_{high} \leftarrow z_l$.

1. Perform a binary search on the set $Z' = \{z_{low}, \ldots, z_{high}\}$. If $|Z'| = 1$, STOP, $z_p^*(N) = z_{low}$. Else, find the 'middle' value z_{mid} in Z'. Go to step 2.

2. Solve $C(z_{mid})$. If $n^*(z_{mid}) \leq p$, set $z_{high} \leftarrow z_{mid}$. Else set $z_{low} \leftarrow z_{mid+1}$. Go to step 1.

For the complexity of the algorithm, Step 0. involves computation of the intersection points which is $O(|E| \cdot n^2)$ and sorting of the z values which is

$O(|E|\cdot n^2\cdot\log(|E|\cdot n^2))$. Step 1. is $O(|E|\cdot n^2)$. In Step 2., the covering problem is solved for $O(\log(|E|\cdot n^2))$ z values where the computational effort for solving the covering problem is

$$O\left(\frac{|E|^p\cdot n^{2p}}{p!}\right)$$

Hence, the complexity of the p-CENT algorithm is

$$O\left(\log(|E|\cdot n^2)\cdot\frac{|E|^p\cdot n^{2p}}{p!}\right)$$

2.2.2 On a Tree Network

The p-CENT algorithm in the previous section is also applicable for tree networks. In this case, however, the set Z can be computed in $O(n^2)$ time. Note that, in the cyclic case, a pair of vertices $v_p, v_q \in V$ can form as many intersection points (and z values) as the number of distinct paths joining v_p and v_q (with the assumption that $w_p, w_q > 0$). In other words, weighted distance functions of v_p and v_q intersect exactly once on a particular path joining v_p and v_q. In the tree case, v_p and v_q can be connected by exactly one path, meaning that this pair can form only one intersection point on the entire network N. The weighted distance function value corresponding to that intersection point is the β_{pq} value defined by this pair. Once the intersection points and the corresponding z values are computed, the rest of p-CENT algorithm applies as in the cyclic case. This time the complexity of the algorithm is polynomial since the covering problem in Step 2 can be solved in $O(n^2)$ time (due to Tansel, Francis, Lowe and Chen 1982 [37]). However, Tansel, Francis and Lowe 1990 [36] give a reduction in this complexity, so that the covering problem on a tree can be solved in $O(n)$ time. The total complexity of the algorithm is then $O(n^2\cdot\log n)$ which comes from the sorting operation in Step 0.

Chapter 3

The Spanning Tree Approach

3.1 The Fundamental Theorem

In what follows, the spanning tree approach for solving the absolute p-center problem on cyclic networks will be discussed. The main idea is motivated by the fact that, for an arbitrary network N, there always exists a spanning tree of N, called an *optimal tree of* N, whose p-center is also a p-center of N with the same p-radius. Actually, this result is the fundamental theorem of this section and will be given at the end of the section. Before that, some related concepts have to be discussed.

Note that, any spanning tree $T = (V', E')$ of N is a connected subgraph of N that satisfies $V' = V$ and $E' \subseteq E$ with $|E'| = |V| - 1$. Further, it is assumed that the vertices of T have the same weights associated with the corresponding vertices of N. This structure of the spanning trees leads to the following observations :

Observation 3.1 $x, y \in T \Rightarrow x, y \in N$.

This follows from the definition of a spanning tree : $e_{ij} \in T \Rightarrow e_{ij} \in N$.

Observation 3.2 *Let $x, y \in T$. Then $d_T(x,y) \geq d_N(x,y)$. Further, for arbitrary v_i and X_T, $D_T(v_i, X_T) \geq D_N(v_i, X_T)$.*

Since N contains all the edges of T plus some additional edges, all the shortest paths on T are also on N. Further, the additional edges of N may create shorter paths on N between x and y, which implies $d_T(x,y) \geq d_N(x,y)$. For the latter part, note that X_T is also feasible on N (by Observation 3.1), which leads to

$$D_T(v_i, X_T) = \min_{j=1,\ldots,p} d_T(v_i, x_j) \geq \min_{j=1,\ldots,p} d_N(v_i, x_j) = D_N(v_i, X_T)$$

where the inequality follows from the first part of the observation.

Observation 3.3 *Let T be a spanning tree of N. Then, the p-radius of T and p-radius of N satisfy*

$$z_p^*(T) = f_T(X_T^*) \geq f_N(X_N^*) = z_p^*(N)$$

Since a p-center of T, X_T^*, is a feasible location set on N, we have

$$\begin{aligned} z_p^*(T) &= f_T(X_T^*) \\ &= \max_{v_i \in V} w_i \cdot D_T(v_i, X_T^*) \\ &\geq \max_{v_i \in V} w_i \cdot D_N(v_i, X_T^*) \\ &= f_N(X_T^*) \\ &\geq f_N(X_N^*) \\ &= z_p^*(N) \end{aligned}$$

where the first inequality follows from Observation 3.2 and the second follows from the optimality of X_N^* on N. In words, any spanning tree T of N has a p-radius (associated with a p-center) not less than the p-radius of N. Observe that an optimal tree $\bar{T}$ of N is a spanning tree of N that satisfies the inequality of Observation 3.3 as equality. The following discussion will show how such an optimal tree can be constructed once a p-center of N is known.

Let $X_N^* = \{x_1^*, \ldots, x_p^*\}$ be a p-center of N. Partition V into disjoint subsets V_j, for $j = 1, \ldots, p$ as follows :

$$V_1 = \{v_i \in V : D_N(v_i, X_N^*) = d_N(v_i, x_1^*)\}$$

$$V_2 = \{v_i \in V \backslash V_1 : D_N(v_i, X_N^*) = d_N(v_i, x_2^*)\}$$

$$V_3 = \{v_i \in V \backslash (V_1 \cup V_2) : D_N(v_i, X_N^*) = d_N(v_i, x_3^*)\}$$

In other words, V_j contains those vertices of V to which center x_j^* is nearest. Note that, if more than one center is nearest to some v_i, say $x_{k_1}^*, \ldots, x_{k_r}^*$ with $1 \leq k_1 < \cdots < k_r \leq p$, then v_i is put into the set V_{k_1} (i.e., ties are broken in favor of the smallest indexed center).

Let $T_j, j = 1, \ldots, p$ be a shortest path tree rooted at x_j^* and span vertices in V_j. T_j is defined to be the union of shortest paths $\bar{P}_N(v_i, x_j^*)$ between each $v_i \in V_j$ and x_j^*, hence its existence follows from the existence of $\bar{P}_N(v_i, x_j^*)$ for each $v_i \in V_j$. However, more than one shortest path may exist between x_j^* and a particular v_i and in such a case, ties between alternative shortest paths are broken arbitrarily. Such a rooted tree can be constructed in polynomial time using Dijkstra's well-known algorithm. Notice that, all the vertices in a particular V_j appear in T_j. Hence, V_j is a subset of the set of vertices of T_j and this result is given in Observation 3.4.

Observation 3.4 *T_j contains all the vertices of V_j but may contain some additional vertices $v \notin V_j$.*

However, in Proposition 3.2, it will be proved that V_j and the set of vertices of T_j are the same. Now, consider the following observation which is valid for each T_j :

Observation 3.5 *Let v_t be a tip vertex of T_j. Then $v_t \in V_j$.*

Observe that a vertex $v_i \in T_j$ can appear in T_j in only two ways : either $v_i \in V_j$ and, therefore, a shortest path $\bar{P}_N(v_i, x_j^*)$ is included in the union of shortest

paths that form T_j ; or $v_i \notin V_j$ but v_i appears on a shortest path $\bar{P}_N(\bar{v}, x_j^*)$ for some $\bar{v} \neq v_i$ and $\bar{v} \in V_j$ (A vertex $v_i \in V_j$ may satisfy both of these). Note that, the latter is not possible for a tip vertex v_t since v_t's being on some path $\bar{P}_N(\bar{v}, x_j^*)$ violates its being tip vertex on T_j. Hence, only the former is possible, which implies that if v_t is a tip vertex of T_j, it must be in V_j.

Why do we need these T_j's ? Recall that, our primary objective is to show the existence of an optimal tree of N. We will see that, there exists an optimal tree of N which contains the T_j's as subtrees. In other words, T_j's are combined in some way to form the optimal tree. The following proposition has an important role in the construction of an optimal tree from the T_j's.

Proposition 3.1 $T_j \cap T_k = \phi$ *for all* $1 \leq j < k \leq p$.

Proof : Suppose $T_j \cap T_k \neq \phi$ for some $j < k$. Then, $\exists v_r \in V$ such that $v_r \in T_j \cap T_k$. There are two cases, either $v_r \in V_j$ or not.

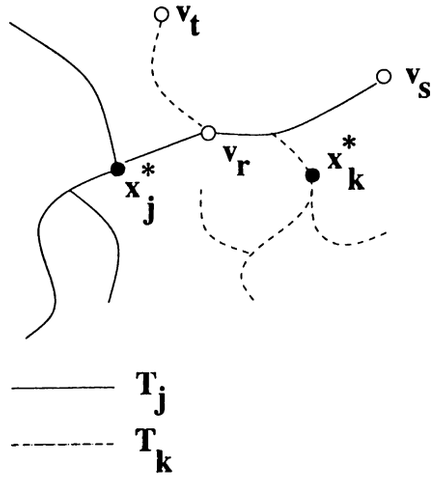

Figure 3.1: Intersecting rooted trees

Case 1. $(v_r \in V_j)$: $v_r \in V_j$ implies $v_r \notin V_k$. Since $v_r \in T_k$, v_r must lie on the path $P_{T_k}(v_t, x_k^*)$, joining some tip vertex $v_t \in V_k$ of T_k and x_k^* (v_r cannot be the tip vertex itself due to Observation 3.5). Observe that,

$$d_N(v_t, x_k^*) \;=\; d_{T_k}(v_t, x_k^*)$$

$$\begin{aligned} &= d_{T_k}(v_t, v_r) + d_{T_k}(v_r, x_k^*) \\ &= d_N(v_t, v_r) + d_N(v_r, x_k^*) \end{aligned} \tag{3.1}$$

since $P_{T_k}(v_t, x_k^*)$ is a shortest path between v_t and x_k^* on N. Now, from $v_r \in V_j$, we have

$$d_N(v_r, x_j^*) \leq d_N(v_r, x_k^*). \tag{3.2}$$

(3.1) and (3.2) together imply,

$$\begin{aligned} d_N(v_t, x_k^*) &= d_N(v_t, v_r) + d_N(v_r, x_k^*) \\ &\geq d_N(v_t, v_r) + d_N(v_r, x_j^*) \\ &\geq d_N(v_t, x_j^*) \end{aligned} \tag{3.3}$$

where the last inequality follows from the triangle inequality on networks. Since $v_t \in V_k$ and $k > j$, $d_N(v_t, x_k^*) < d_N(v_t, x_j^*)$ must hold. But this contradicts (3.3), hence $T_j \cap T_k = \phi$ for this case.

Case 2. ($v_r \notin V_j$) : This case will first be proved for $j = 1$, i.e. $T_j \cap T_k = \phi$ for all $k > j = 1$. Then, we will see that the proof can be repeated for $j = 2, 3, \ldots, p-1$ sequentially.

Since $v_r \notin V_1$, we have $d_N(v_r, x_1^*) > D_N(v_r, X_N^*)$ (by the construction of V_1). But since $v_r \in T_1$, there exists a tip vertex v_s of T_1 such that $s \neq r$ and $v_s \in V_1$ and

$$\begin{aligned} d_N(v_s, x_1^*) &= d_{T_k}(v_s, x_1^*) \\ &= d_{T_k}(v_s, v_r) + d_{T_k}(v_r, x_1^*) \\ &= d_N(v_s, v_r) + d_N(v_r, x_1^*) \end{aligned} \tag{3.4}$$

similar to Case 1. Let $l > 1$ be the index such that $v_r \in V_l$. We have,

$$d_N(v_r, x_1^*) > D_N(v_r, X_N^*) = d_N(v_r, x_l^*) \tag{3.5}$$

But then,

$$D_N(v_s, X_N^*) = d_N(v_s, x_1^*) \tag{3.6}$$

$$= d_N(v_s, v_r) + d_N(v_r, x_1^*) \tag{3.7}$$

$$> \; d_N(v_s, v_r) + d_N(v_r, x_l^*) \tag{3.8}$$
$$\geq \; d_N(v_s, x_l^*) \tag{3.9}$$
$$\geq \; D_N(v_s, X_N^*)$$

which is impossible (Here, (3.6) follows from $v_s \in V_1$, (3.7) from (3.4), (3.8) from (3.5), and (3.9) from the triangle inequality). Hence, $T_1 \cap T_k = \phi$, for $k > 1$.

Now, we will show that the above proof of Case 2 can be repeated for $j = 2, 3, \ldots, p-1$ sequentially. To illustrate how this happens, let $j = 2$. Recall that we have $v_r \in T_2 \cap T_k$ for some $k > 2$ (the main assumption of the entire proof). Further, we have $T_1 \cap T_2 = \phi$ from the proof of the case $j = 1$. Since $T_1 \cap T_2 = \phi$, we have $v_r \notin V_1$ (otherwise, $v_r \in V_1$ would imply $v_r \in T_1$. Note also that $v_r \in T_2$). This and the assumption of Case 2 imply $v_r \in V_l$ for some $l > 2$. Observe that, steps (3.5)-(3.9) are still valid when x_1^* is replaced with x_2^*, which lead to a contradiction. Hence, $T_2 \cap T_k = \phi$ for all $k > 2$. The process can be repeated for $j = 3, 4, \ldots, p-1$ (in that order) similarly. This completes the proof of Proposition 3.1.

□

The T_js' being disjoint gives the following proposition :

Proposition 3.2 *Let $V(T_j)$ be the vertex set of T_j. Then, $V(T_j) = V_j$.*

Proof : The proposition will be proved, again, for $j = 1$ first. Since T_1 spans all the vertices in V_1 (by construction of T_1), $V_1 \subseteq V(T_1)$. To show $V(T_1) \subseteq V_1$, let $v_r \in V(T_1)$. If v_r is a tip vertex of T_1, then $v_r \in V_1$ by Observation 3.5. If v_r is not a tip vertex, then there exists a tip vertex v_t of T_1 such that $v_r \in P_{T_1}(v_s, x_1^*)$. Assume $v_r \notin V_1$. Then

$$D_N(v_r, X_N^*) < d_N(v_r, x_1^*)$$

Let $l > 1$ be the smallest index for which $D_N(v_r, X_N^*) = d_N(v_r, x_l^*)$. Then, the same sequence of operations (3.6)-(3.9) can be repeated, which leads to a

contradiction. Hence, $v_r \in V_1$ even if v_r is not a tip vertex of T_1. It follows that $V(T_1) = V_1$.

The proposition is proved for $j > 1$ in a similar way to Proposition 3.1. For $j = 2$, $T_1 \cap T_2 = \phi$ (from the case $j = 1$) implies $v_r \notin V_1$. Let $l > 2$ be the smallest index for which $D_N(v_r, X_N^*) = d_N(v_r, x_l^*)$. Then (3.6)-(3.9) are valid and lead to a contradiction. Hence, $v_r \in V_2$ which implies $V(T_2) = V_2$. This can be repeated for $j = 3, 4, \ldots, p-1$ in that order to complete the proof.

□

Why is it important to have disjoint T_j's ? Observe that, T_j's partition N into p disjoint subtrees each containing only the vertices in the corresponding V_j. This implies that they can be connected to one another using $p-1$ additional edges (that are not in any of the T_j's) to obtain a spanning tree. In fact that's why the tie-breaking rule in the partitioning of V is essential for a correctly worked out proof. Without the rule, the T_j's need not be disjoint in which case the subgraph of N defined by vertex set V and the union of all edges in $T_1, \ldots, T_p$ may contain cycles which makes it highly difficult, if not impossible, to work out a satisfactory proof of the existence of an optimal tree. The following proposition states that the final tree constructed as described above is actually an optimal tree of N, that we are looking for.

Proposition 3.3 *Let $\bar{T}$ be a spanning tree of N formed by combining T_j's as described above. Then*

$$z_p^*(\bar{T}) = f_{\bar{T}}(X_{\bar{T}}^*) = f_N(X_N^*) = z_p^*(N) \tag{3.10}$$

Proof : We need to show only the middle equality since the others come from the definitions of $z_p^*(\cdot)$. Suppose we solve 1-center problem on each T_j and let $\bar{X} = \{\bar{x}_1, \ldots, \bar{x}_p\}$ be the set of the corresponding 1-centers. Note that, $\bar{X}$ is feasible on $\bar{T}$ and N. Hence, we have

$$f_{\bar{T}}(X_{\bar{T}}^*) \leq f_{\bar{T}}(\bar{X}) \tag{3.11}$$

from the optimality of $X_{\bar{T}}^*$ on $\bar{T}$. Furthermore, let $z_1^*(T_j)$ be the 1-radius

associated with T_j. Then, for each T_j, we have

$$\begin{aligned} z_1^*(T_j) &= \max_{v_i \in V_j} \{w_i \cdot d_{T_j}(v_i, \bar{x}_j)\} \\ &\leq \max_{v_i \in V_j} \{w_i \cdot d_{T_j}(v_i, x_j^*)\} \\ &= \max_{v_i \in V_j} \{w_i \cdot d_N(v_i, x_j^*)\} \\ &\leq z_p^*(N) \end{aligned}$$

which gives

$$f_{\bar{T}}(X_{\bar{T}}^*) \leq f_{\bar{T}}(\bar{X}) = \max_{j=1,\ldots,p} z_1^*(T_j) \leq z_p^*(N) = f_N(X_N^*)$$

We also have $f_{\bar{T}}(X_{\bar{T}}^*) \geq f_N(X_N^*)$ from Observation 3.3, which implies

$$f_{\bar{T}}(X_{\bar{T}}^*) = f_N(X_N^*).$$

This completes the proof of Proposition 3.3.

□

To summarize, the set of 1-centers, $\bar{X}$, is actually a p-center on both $\bar{T}$ and N. Therefore, the p-center of $\bar{T}$ is also optimal on N, regardless of the edges used to combine T_j's into $\bar{T}$ (since $\bar{T}$ contains T_j's as subtrees). Hence, we get $z_p^*(\bar{T}) = z_p^*(N)$.

The following theorem, which is actually the fundamental theorem of this section, is the summary of all the above discussion :

Theorem 3.1 *Let $N = (V, E)$ be a network as defined in* Chapter 1, *that satisfies the assumptions* A1-A6 *stated. Let $\mathcal{T}(N)$ be the collection of all spanning trees of N which have the same weights w_i as N. Then, there exists a spanning tree $\bar{T} \in \mathcal{T}$ such that*

$$z_p^*(\bar{T}) = z_p^*(N) \tag{3.12}$$

Proof : The tree $\bar{T}$ is the optimal tree that is constructed from the T_j's and (3.12) follows from Proposition 3.3.

□

Notice that, the construction procedure described in the above discussion does not help to find T_j's or construct $\bar{T}$ at all. This is because the construction is well-defined once we have a p-center $X_N^* = \{x_1^*, \ldots, x_p^*\}$ as a starting point. Normally, this is not the case because if we knew such an X^*, we would not need to construct all those T_j's in the first place.

The implications of Theorem 3.1 are quite important for solving the problem on a cyclic network. If we can, somehow, determine an optimal tree $\bar{T}$ for N, we solve the p-center problem on $\bar{T}$ and obtain the optimal solution for the cyclic network. Note that, the number of all spanning trees of N is not polynomial. If it were possible to reduce the search for optimal tree to a polynomial-size subset of $\mathcal{T}$, we would be able to solve the problem on N in polynomial time. Of course, this would mean $\mathcal{P} = \mathcal{NP}$. Hence, identifying some properties of the optimal tree or identifying a set of spanning trees that includes the optimal tree would still be a valuable effort for solving the problem. In the following section, two particular types of spanning trees will be introduced and their relation to the optimal tree will be discussed.

3.2 Rooted Shortest Path Trees

In this section, two types of spanning trees that are suspected to include the optimal tree are introduced. The trees will be referred to as *rooted shortest path trees* (or shortly RSPT) in general. As the name implies, RSPTs are constructed by picking some point(s) of N as root(s) and taking the collection of shortest paths between each vertex of N and the root(s).

First, the concept of an *antipodal* needs to be defined. The antipodal of vertex v_k on an edge e_{ij} is the unique point a where the distance function $d_N(v_k, t)$ (between v_k and a varying point t on e_{ij}) reaches its maximum point (See Figure 3.2a). Three different shapes of this distance function are labeled A,B and C as in the figure. Observe that, for a distance function of type A or B, the antipodal is at one of the endpoints of e_{ij}, whereas for type C, it is at an interior point of e_{ij}. For later use, the antipodals associated with

type A and B distance functions will be referred to as *type-1* antipodals, where the antipodals associated with type C distance functions will be referred to as *type-2* antipodals. Note that, each vertex has exactly one antipodal on each edge, meaning a total of $O(|E| \cdot n)$ antipodals on N.

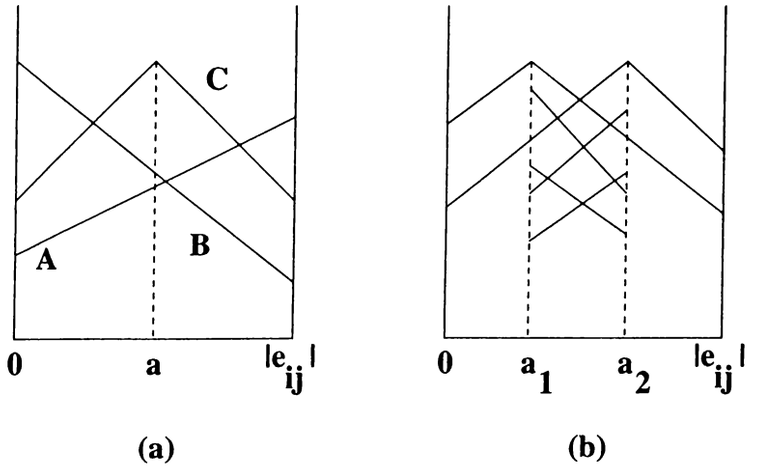

Figure 3.2: Distance functions and antipodals

Now we can describe the shortest path trees rooted at adjacent antipodal segments.

3.2.1 Trees Rooted at Adjacent Antipodal Segments

Consider any adjacent pair of antipodals, a_1, a_2, on edge e_{ij} and let the *adjacent antipodal segment* $[a_1, a_2]_{ij}$ be defined as the subedge of e_{ij} that lies between a_1 and a_2. Note that, all the weighted distance functions on the segment $[a_1, a_2]_{ij}$ have either shape A or shape B (see Figure 3.2b). This is because the existence of a type-C shape on $[a_1, a_2]_{ij}$ would imply a third antipodal strictly between a_1 and a_2 which violates the adjacency. As a consequence, the set of vertices is uniquely partitioned into two sets V_A,V_B as follows :

- $V_A = \{v_i \in V : d_N(v_i, a_1) < d_N(v_i, a_2)\}$
- $V_B = \{v_i \in V : d_N(v_i, a_1) > d_N(v_i, a_2)\}$

The equality case is impossible because of the strictly increasing or decreasing weighted distance functions on $[a_1, a_2]_{ij}$.

The shortest path tree, $T(a_1, a_2)$, rooted at the segment $[a_1, a_2]_{ij}$ is constructed as follows. Let $\bar{P}_N(v_i, a_1)$ be a shortest path on N between v_i and a_1 (if there are more than one, $\bar{P}_N(v_i, a_1)$ refers to the first one encountered). Then,

1. Construct $T_A(a_1) = \bigcup_{v_i \in V_A} \bar{P}_N(v_i, a_1)$
2. Construct $T_B(a_2) = \bigcup_{v_i \in V_B} \bar{P}_N(v_i, a_2)$
3. Define $T(a_1, a_2) = T_A(a_1) \cup T_B(a_2) \cup [a_1, a_2]_{ij}$.

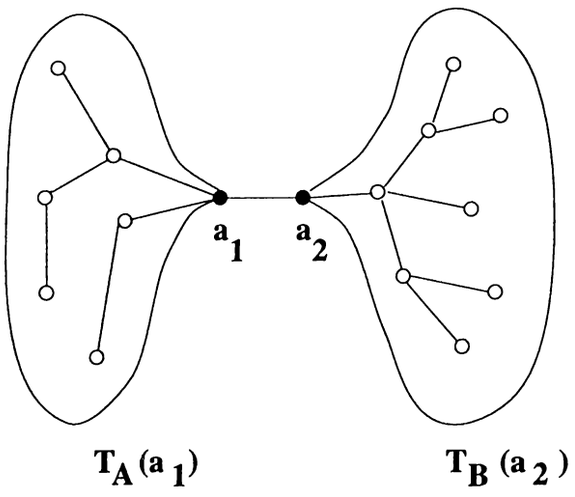

Figure 3.3: Shortest Path Tree Rooted at an Antipodal Segment

The trees constructed have the structure as shown in Figure 3.3. Actually, the idea why such trees might contain the optimal tree relies on some past empirical evidence. For many handworked small scale examples, it turned out that one of these trees always gave the optimal solution to the problem on a cyclic network. Hence, such trees are subject to search for the optimal tree.

Observe that, although the partition (V_A, V_B) is unique, alternative shortest paths joining vertices and antipodals may be encountered, in steps 1 and 2 of the above procedure. For the moment, we assume that the first shortest

path found is picked as $P(v_i, \cdot)$ and put into T_A or T_B. Note that, with this rule for breaking ties, one RSPT is constructed for each antipodal segment. Hence, a total of $O(|E| \cdot n)$ spanning trees of N is constructed (each of them is constructed by using Dijkstra's $O(n^2)$ shortest path algorithm and some of them may be identical). The total computational effort is, therefore, $O(|E| \cdot n^3)$ which can most likely be reduced to a lower order by avoiding repetitions in using Dijkstra's algorithm at neighboring segments, but that is not our point of focus in this thesis. The RSPTs rooted at adjacent antipodal segments will be referred to as *A-RSPT*.

The question is the following : Is the optimal tree included in the set of trees so constructed? To answer this question, at least empirically, an experimental study on random networks is designed and implemented. The p-center problem is solved on all the trees constructed and the best p-center is compared with the p-center of the network. This is the subject of the next chapter, however, we will say that the answer to the question is 'not always!'.

3.2.2 Trees Rooted at Intersection Points

One alternative set of spanning trees in which the optimal tree might be searched, is the set of shortest path trees rooted at intersection points of N. The consideration of this family of spanning trees does not rely on empirical evidence as in the case of antipodal segments, however, these trees have structural similarities to the previous ones so that it might be worth considering.

The spanning trees rooted at intersection points of N are constructed in a similar way to the antipodal case. This time, we do not partition V into V_A and V_B, but construct a single shortest path tree rooted at an intersection point of N (an RSPT rooted at an intersection point will be referred as *I-RSPT*). This construction is repeated for all intersection points of N which is $O(|E| \cdot n^2)$ ($O(n^2)$ on each edge). Again, each I-RSPT is constructed in $O(n^2)$ time (by Dijkstra's algorithm), which gives a total computational effort of $O(|E| \cdot n^4)$ (which may be reduced similarly). The alternative short paths are handled

similarly. Unfortunately, our empirical results show that the optimal tree need not be an I-RSPT of N.

One important remark is about the existence of vertices with zero weights. Suppose that some of the vertices of N have zero weights. Since a zero-weight vertex is not included in the computation of intersection points, N has fewer intersection points compared to the case that no vertex has zero weight. However, as long as the number of intersection points is $O(|E| \cdot n^2)$, the existence of zero-weight vertices reduces only the *number* of I-RSPTs constructed, not the order.

3.3 The Optimal Tree

The proof of Theorem 3.1 suggests that the optimal tree is composed of p shortest path trees (T_j's) rooted at optimal centers of the network N. Let us call the edges that appear on these trees critical edges. All the other edges that do not appear on T_j's but might be used to combine T_j's will be referred to as *non-critical edges*. Note that if there are alternative shortest paths in T_j's or more than one optimal solution to the problem, there may be other optimal trees as well. Whether an edge switches from being critical to non-critical (or vice versa) with respect to different optimal trees is an open question and not studied in this thesis. But if we can identify as many non-critical edges as possible, the network that remains after deleting such edges still contains each T_j as a subgraph and hence, a p-center of original N is still contained in the remaining N. In this case, we have the opportunity of working with a simpler network whose edges are all candidates for the edges of an optimal tree of N.

Is there a way to identify critical or non-critical edges? The answer to this question cannot be given easily for the absolute p-center problem. However, if we fix a value of z, we can partially identify non-critical edges for the associated covering problem, $C(z)$.

First, note that an edge e_{ij} on a tree T_k is the only path on T_k (actually a

shortest one on N, by Assumption A4 of Chapter 1) between v_i and v_j. Hence, these two vertices can best be covered by locating the 1-center of T_k (ignoring all the other vertices) at the point on e_{ij} where the weighted distance functions of the two vertices intersect. Note that, this is the point which is

$$w_j \cdot \frac{|e_{ij}|}{w_i + w_j}$$

far from v_i and

$$w_i \cdot \frac{|e_{ij}|}{w_i + w_j}$$

far from v_j (if this point is slightly perturbed, the weighted distance function value of one of the vertices definitely increases because of the positive and negative slopes). And the value of the weighted distance functions of both vertices at the intersection point will be

$$w_i w_j \frac{|e_{ij}|}{w_i + w_j} = \frac{d_N(v_i, v_j)}{\frac{1}{w_i} + \frac{1}{w_j}} = \beta_{ij}$$

Suppose we solve the covering problem $C(z)$ for fixed z. Then, a necessary condition for v_i and v_j to be on the same tree T_k (or in the same partition V_k) is

$$d_N(v_i, v_j) \leq \frac{z}{w_i} + \frac{z}{w_j} = z \cdot (\frac{1}{w_i} + \frac{1}{w_j})$$

and equivalently,

$$\frac{d_N(v_i, v_j)}{\frac{1}{w_i} + \frac{1}{w_j}} = \beta_{ij} \leq z \tag{3.13}$$

Consider Figure 3.4. The shaded regions correspond to the distances within which a center must be located. In (b) and (c), any single center located in the intersection of regions will cover both vertices. However, in (a), the two regions do not overlap since (3.13) is not satisfied. Hence, the two vertices cannot be covered with a single center, even on the shortest path joining them. This implies that v_i and v_j cannot be in the same partition, i.e. the edge e_{ij} cannot appear on any of the T_j's.

Obviously, $w_i, w_j > 0$ is required so that (3.13) can be well defined. Hence, the necessary condition is applicable for only the vertex pairs v_i, v_j

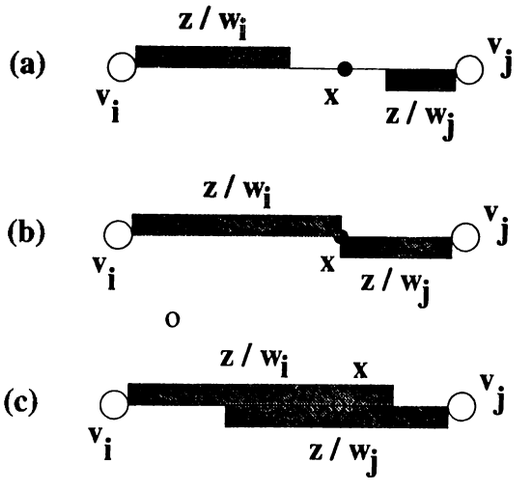

Figure 3.4: Covering two vertices with a single center

with $w_i, w_j > 0$ and $e_{ij} \in E$. Actually, if $w_i = 0$ for some i, then we do not need to consider covering v_i and any other vertex by the same center (i.e. put them in the same partition), because v_i does not even need to be covered (since it is not a real demand point).

This approach provides a method for eliminating some non-critical edges. Compute β_{ij} for all pairs v_i, v_j and sort descendingly (even include the repeating ones). Let $\beta_{max} \geq \cdots \geq \beta_2 \geq \beta_1$ be the sorted values. Find index r such that $\beta_i > z$, $\forall i \geq r$. Now consider $\beta_i = \beta_{st}$, for $i \geq r$. If $(v_s, v_t) \in E$, then by the argument above, it is a non-critical edge and satisfies

$$e_{st} = (v_s, v_t) \notin T_j, \forall j = 1, \ldots, p$$

Hence, e_{st} can be deleted from N without affecting $n^*(z)$, the optimal solution to $C(z)$ (since the structure of T_j's remains unchanged with the deletion of a non-critical edge). The process is repeated for each β_{ij} that satisfies $\beta_{ij} > z$.

The above procedure is for eliminating some of the non-critical edges of N (with respect to z). However, if $z > \beta_{max}$, there are no such edges. Another problem is when $(v_i, v_j) \notin E$ even if $\beta_{ij} > z$ exists. In this case, the problem $C(z)$ is irreducible and we cannot identify any non-critical edges. But usually (for $p > 1$) there exist $\beta_{ij} > z$. For relatively large p, it is almost guaranteed to find such β values. The procedure can be used to increase the efficiency of

the p-CENT algorithm by solving the covering problem on a simpler network.

Note that, the procedure assumes a fixed z. This means that, if the p-radius itself or an upper bound to p-radius is known, it is an easy task to identify some non-critical edges (if they exist). To obtain an upper bound on p-radius, we may construct all the A-RSPTs and I-RSPTs of the network N, solve the absolute p-center problem on each and pick the best p-radius as the upper bound.

3.3.1 Two Special Cases

For a general cyclic network, it may be quite difficult to find the optimal tree in an efficient way. The following special cases, which are more general than a tree network, are the cases where we can construct a polynomial number of spanning trees one of which, we know, is the optimal tree.

The Simple Cycle

The simple cycle is illustrated by the 8-vertex numerical example in Figure 3.5.a. It is a single n-vertex cycle, nothing else. As one may guess, this network is the most trivial after the tree network with respect to edge structure (it may be made even more trivial by assigning unit edge lengths and weights). Observe that, this n-vertex simple cycle has n spanning trees formed by deleting its edges one at a time. Hence, to find a p-center, we construct these n spanning trees by enumerating all edge deletions, solve the problem on each (in a total of $O(n \cdot n^2 \cdot \log n)$ time) and take the best p-radius (notice that each zero-weight vertex decreases by 1 the total number of trees that have to be considered).

The above discussion is still valid when the simple cycle is generalized further by adding edges that do not introduce a second cycle (see Figure 3.5.b). Still, there are n spanning trees formed by deleting its edges one at a time. Hence, in order to find a p-center of N, again we enumerate all edge deletions

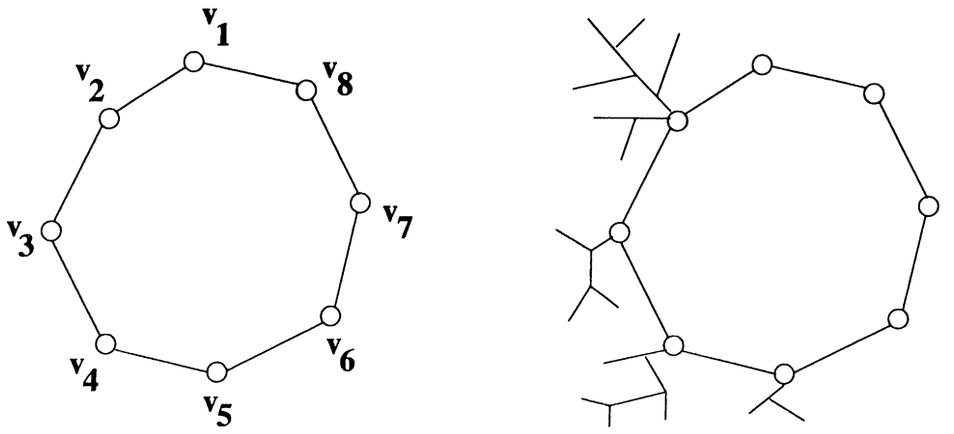

Figure 3.5: The simple cycle

to construct n spanning trees, solve the problem on each and take the best p-radius.

The Cactus-type Network

The cactus-type network, or simply cactus, is a one-step further generalization of the simple cycle. It contains W cycles which do not pairwise intersect at more than one vertex. More formally, cactus is the network where each edge cannot appear in more than one cycle (see Figure 3.6).

In the figure, big circles actually represent the cycles. Let the cycles be denoted by $C_1, C_2, \ldots, C_W$ and the number of edges they contain be denoted by $n_1, n_2, \ldots, n_W$, respectively. Obviously, there are $n_1 \cdot n_2 \cdots n_W$ spanning trees possible (all the combinations of deleting one edge from each cycle). Note that, there may be $O(n)$ cycles at most and each cycle may contain one additional edge (compared to a tree). Hence the total number of edges of the cactus is still $O(n)$. This implies that W and each n_i cannot be $O(n)$ at the same time (otherwise $|E|$ would be $O(n^2)$). Hence, if we denote the average number of edges per cycle by $\bar{n}$, we have $W \cdot \bar{n} \sim |E| \sim O(n)$. Note that the number of all spanning trees of the cactus is expressed by $S = n_1 \cdot n_2 \cdots n_W$ for which $\sum_i n_i$

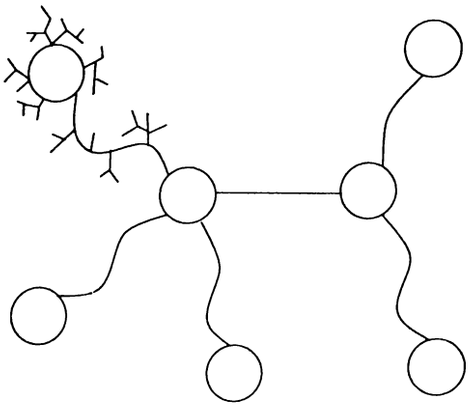

Figure 3.6: The cactus

is at most n (if there are no vertices outside the cycles and each vertex belong to exactly one cycle). From a well-known mathematical result, S gets its maximum value when $n_1 = n_2 = \cdots = n_W = \frac{n}{W}$. Then we have $S = (\frac{n}{W})^W$. Note that, the first derivative of the expression with respect to W is never zero, so that we can find a W^* for which S is maximized when n is fixed. Furthermore, it is not possible to find a positive integer k such that $S \leq C \cdot P(n^k)$ where C is a positive constant and $P(n^k)$ is a polynomial function in n of degree k. Hence, the number of all spanning trees of a cactus-network is not polynomial. For this reason, we need other techniques to identify the optimal tree, rather than polynomially solving all the trees.

It turns out that for the case $p = 2$, a polynomial number of spanning trees of a cactus include the optimal tree. In this case, the vertices are partitioned into two as V_1 and V_2 associated with trees T_1 and T_2. This bipartitioning implies that no two cycles can contain vertices from both V_1 and V_2 (otherwise T_1 and T_2 would have to use the same single path joining two cycles in order to be connected, which means $T_1 \cap T_2 \neq \phi$). Hence, all the vertices in a particular cycle (with at most one exceptional cycle) must belong to either V_1 or V_2. We will refer to such a cycle as a *homogeneous* cycle.

Consider Figure 3.7. Let the *inter-cycle path* between two cycles C_i and C_j be denoted by P_{ij}. Note that, this is the unique path on the cactus that connects C_i and C_j. Then, a *gate* $g_l(C_k)$ of a cycle C_k is defined as the unique

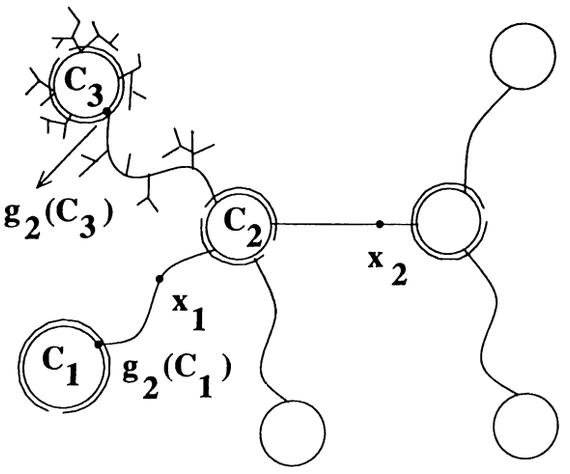

Figure 3.7: 2-center on cactus

vertex v_s that satisfies $v_s \in C_k \cap P_{kl}$ for some cycle C_l, with $k \neq l$. Let the *degree* of a cycle C_i be defined to be the number of gates of C_i and be denoted by δ_i (analogous to the case on a simple graph). Further, let the *inside* of a cycle be identified as the cycle itself plus all those edges that are branching out of the cycle at the vertices that are not gates of that cycle (as shown for the top-left cycle in the figure). Note that, the inside of a cycle cannot contain any edge of an inter-cycle path (except the endpoint of the inter-cycle path that intersects with the cycle at a gate) or the edges inside of other cycles. The remaining part of the cactus with respect to a cycle is the *outside* of that cycle. With these definitions, the homogeneity of a cycle has to be extended to include all the vertices inside of a cycle.

Suppose both x_1 and x_2 are outside of all of the cycles, i.e. they are both located on inter-cycle paths (which is the case in the figure). At this point, we are interested in identifying the edge(s) that should be removed from each cycle (the non-critical edges) so that we can construct the optimal tree. When a cycle is homogeneous, then there is only one non-critical edge on that cycle that has to be removed. For example, the cycle C_1 in the figure is a homogeneous cycle and all the vertices inside of C_1 belong to V_1 (since x_1 is strictly nearer than x_2). Therefore, there is only one non-critical edge on this cycle. Observe that, this edge is determined by the gate $g_2(C_1)$, namely the edge that contains the type-2 antipodal of $g_2(C_1)$. If the antipodal is on top of a vertex, then there

are two alternatives for the non-critical edge in either direction. We can select either of them as the non-critical edge on this cycle, without destroying the properties of T_1. This implies that, for a homogeneous cycle, the non-critical edge on that cycle can deterministically be identified.

What happens when a cycle is not homogeneous (recall that at most one cycle can have this property). A cycle C_j's being not homogeneous implies that some of the vertices inside of C_j are in V_1 and the remaining are in V_2 (e.g. the cycle C_2 in the figure). This cycle has actually two non-critical edges to be removed, but for the purposes of constructing the optimal tree, we will be interested in identifying and removing only one of them (if both are removed, the cactus becomes disconnected). Observe that, it is not sufficient to know only the inter-cycle path on which x_1 and x_2 are fixed, in order to identify the cycle that is not homogeneous. We have to know the exact locations of x_1 and x_2 to decide whether a cycle is homogeneous or not. However, if a homogeneous cycle exists, it must be on the way along some path joining x_1 and x_2. So, for x_1 and x_2 fixed on some pair of inter-cycle paths, we would like to enumerate all the cycles along such a path, which is $O(n)$. We still have to identify a non-critical edge on a cycle that is not homogeneous, but again this is not deterministically possible if the exact locations of x_1 and x_2 are not known. So, for a cycle considered as not homogeneous, we would like to enumerate all edge deletions on this cycle ($O(n)$). This implies that, once x_1 and x_2 are fixed on inter-cycle path(s) (there are $O(W^2)$ of such path combinations), we need to construct $O(n^2)$ spanning trees, which makes a total of $O(W^2 \cdot n^2)$, for the case x_1 and x_2 restricted to inter-cycle paths. Since $W \sim O(n)$, it becomes $O(n^4)$.

The other case is when x_1 and x_2 are restricted to inside of cycles. To analyze this case, let C_1 and C_2 be those two cycles, respectively, inside of which x_1 and x_2 are located. There are $O(W^2)$ combinations of C_1 and C_2. Observe that, the gates of C_1 (C_2) do not necessarily determine the non-critical edges on C_1 (C_2) anymore. Since we do not know the exact location of x_1 (x_2), we would like to enumerate all edge deletions for C_1 (C_2) (which is $O(n^2)$). Furthermore, there may be a cycle C_k (other than these two) which is not homogeneous,

and it is not possible to deterministically identify this cycle and one of its non-critical edges, as in the previous case. So again we enumerate all the cycles along some path joining C_1 and C_2 and on each cycle considered, enumerate all the edge deletions. Hence, we have to construct $O(W^2 \cdot n^4)$ spanning trees for this case (which dominates the order of previous case), which means that, for the case $p = 2$, we need to enumerate $O(n^6)$ trees (since $w \sim O(n)$) which will be guaranteed to contain the optimal tree.

One final remark will be for the $p = 2$ case on general networks. There is some empirical evidence which supports the idea that an optimal tree of N is an A-RSPT. Although not proved theoretically, the evidence is that an adjacent antipodal segment on some edge of N partitions V into two sets V_A and V_B exactly in the same way a 2-center of N does. This means that solving the 1-center problem separately on T_A and T_B of Figure 3.3 is equivalent to solving 2-center on N. The following conjecture states this idea :

Conjecture 3.1 *There exists a pair of adjacent antipodals a_1, a_2, located on some edge e_{ij} of N, such that one of the shortest path trees rooted at the segment $[a_1, a_2]_{ij}$, $T(a_1, a_2)$, satisfies $z_2^*(T(a_1, a_2)) = z_2^*(N)$.*

Chapter 4

The Experimental Study

4.1 The Design

The concept of an optimal tree brings a number of questions regarding how to find the p-center of a network N. One of the aspects of the optimal tree that is considered in this study is the matter of identifying a set of spanning trees in which the optimal tree is included.

In this chapter, an experimental study on two sets of spanning trees (which are suspected to contain the optimal tree) is presented and discussed. These are the sets of A-RSPTs and I-RSPTs that are introduced and described in Section 3.2.1 and Section 3.2.2, respectively. The construction of these trees are based on the procedures given in these sections.

The experimental study is carried on random cyclic networks. The following modules were coded (in C language) in order to be used in the study :

- NETGEN : Generates random cyclic networks.
- IPOINTS : Computes all the intersection points of the input network.
- ZFIND : Extracts all z values from intersection points.

- COVER : Solves $C(z)$ on an input network for an input z value.
- AROOT : Constructs A-RSPTs of an input network
- IROOT : Constructs I-RSPTs of an input network.
- AROOTE, IROOTE : Exhaustive versions of the modules AROOT and IROOT.
- PTPOL : Solves p-center problem on a range of trees constructed.

The random cyclic networks that are tested in the study are all generated by the NETGEN module according to the following factors and their associated levels :

n : Number of vertices, $n \in \{10, 20, 30, 40\}$.

d : Edge density, the ratio (in percent) $|E| / \binom{n}{2}$, $d \in \{25, 50, 75\}$.

w : Vertex weights, uniform from set $\{1, 2, 3\}$, or unweighted.

l : Edge lengths, uniform from set $\{1, 2, 3, 4, 5\}$.

p : number of centers, $p \in \{\lfloor n/4 \rfloor, n/2, \lceil 3n/4 \rceil\}$.

Observe that parameter n gives 4 levels; parameter d, 3 levels; parameter w, 2 levels (weighted and unweighted) and parameter p, 3 levels. Hence, a total of 72 factor settings are generated. For each case, 10 random instances are generated which makes a total of 720 instances. The NETGEN module generates an instance of a random network (independent from p) as follows : it takes n, d and w status (weighted or unweighted) as input and starts with an empty network (n vertices, but no edges). It continues to generate random edges until a total number of

$$\left\lceil \frac{d}{100} \cdot \binom{n}{2} \right\rceil \tag{4.1}$$

edges are generated. Note that, this number of edges guarantees a $d\%$-density network, but does not guarantee connectedness. Hence, if the network is not connected, NETGEN proceeds with generating additional random edges until the network becomes connected. In the second stage of the process, NETGEN deletes random edges from the network (without destroying the connectedness) until the total number of edges is reduced to $d\%$ level. Finally, it randomly generates vertex weights and edge lengths. It must be noted that, an unweighted instance is constructed by simply setting equal to 1 the weights of the corresponding weighted instance. This is for the purpose of testing whether or not, the unweightedness improve the results on the corresponding instances for which the optimal tree could not be found. Further, all the weighted instances were generated with positive weights in order not to simplify the structure of an instance with zero weights.

The IPOINTS and ZVALUE modules compute the intersection points and the associated z values to be used in modules IROOT, IROOTE and COVER, respectively. The COVER module solves the covering problem $C(z)$ by solving a sequence of set covering problems as described in Chapter 2. Then this module is used to solve the p-center problem on a particular instance.

The AROOT and the IROOT modules are the two basic ones of the experiments. AROOT and IROOT construct the A- and I-RSPTs, respectively. Recall that, we may encounter alternative shortest paths in constructing these trees, one of which must be chosen. AROOT (IROOT) constructs one tree (by picking one of the alternative paths) for one antipodal segment (intersection point) and is, therefore, polynomial. However, AROOTE (IROOTE) enumerates all the alternative shortest paths and constructs all the trees that are rooted at that segment (intersection point). Obviously, the number of trees generated by AROOTE or IROOTE is not polynomial. The need for exhaustively constructing the trees is based on the following. As mentioned earlier, some empirical evidence supported the idea that these rooted trees always included the optimal tree. During the experiments, it was observed that some instances violated this evidence. However, this might simply be because of not choosing the correct shortest path (in case of alternative shortest paths), not

because the trees do not include the optimal tree. To test this, the rooted trees were exhaustively constructed for those instances. However, it was observed that, even in this case, there were instances in which the optimal tree was not an A- or I-RSPT. This exhaustiveness also enables the experimenter to see how much improvement, in the gap between tree and network p-radii, is achieved by introducing this additional step. Note that, since enumerating all the shortest paths is not polynomial, this step is computationally hard to implement for large problem sizes.

Finally, the PTPOL module is designed to solve the p-center problem on the trees generated by the modules AROOT (AROOTE) and IROOT (IROOTE). The module uses the p-CENT algorithm given in Chapter 2 for solving the problem on a number of trees, for the p values given above.

4.2 The Results

The experiment is considered to have two stages. In the first stage, for each instance of a cyclic network (there are 240 of them, solved for 3 p values, which makes 720), only one rooted tree is constructed for each adjacent antipodal segment of the instance. Then, the problem is solved on both the network and the trees constructed and the best p-radius among the trees is compared with the p-radius of the network. Recall that the total number of A-RSPTs (I-RSPTs) is $O(|E| \cdot n^3)$ ($O(|E| \cdot n^4)$) and the p-center problem on each tree is solved in $O(n^2 \cdot \log n)$ time. Hence, the total complexity of the first stage is $O(|E| \cdot n^5 \cdot \log n)$ for A-RSPTs and $O(|E| \cdot n^6 \cdot \log n)$ for I-RSPTs. The second stage (the exhaustive stage) is applied to those instances for which the optimal tree could not be found in the set of trees tested, in the first stage (i.e., a gap occurred, in the first stage, between the p-radii of the network and all the trees constructed). In this stage, all the possible trees are constructed (by enumerating all the alternative shortest paths) and the rest is carried out the same as in the first stage.

The instances are grouped in the following four major groups so that the

results can be analyzed with respect to different criteria.

1. Weighted vs. unweighted (W and U).
2. Sparse vs. dense ($d = 25, 50, 75$).
3. The value of p relative to n ($p = \lfloor n/4 \rfloor, n/2, \lceil 3n/4 \rceil$).
4. The problem size ($n = 10, 20, 30, 40$).

The four summary tables display the results with respect to these four groups. All the groups except the first one, are further split into two subgroups as weighted (W) and unweighted (U). The tables contain two kinds of information under each group and subgroup, at each stage :

(a) the percentage of instances for which the optimal tree *is* in the set of trees tested.

(b) the maximum and average gap, for the instances in a particular (sub)group, between the p-radii of the network and that of the best A-RSPT (I-RSPT) in terms of the p-radius.

The summary tables 4.1, 4.2, 4.3 and 4.4 are actually extracted from the tables in Appendix A,B,C and D. The notation used in the latter set of tables has to be defined, before explaining the details of the summary tables.

The tables in the Appendix A,B,C and D list all the instances for which the optimal tree could not be found in the set of A- or I-RSPTs constructed and tested, either in the first or second stage (or both). In these tables, n, d and S denote the number of vertices, the edge density (in percent) and the random seed number, respectively, for a particular instance of network. Now, denote the class of A-RSPTs constructed in the first and second stages by $\mathcal{T}_A^{(1)}$ and $\mathcal{T}_A^{(2)}$, respectively, with $\mathcal{T}_A^{(1)} \subseteq \mathcal{T}_A^{(2)}$. Let T_A' and T_A'' be the best trees associated with each stage, for a particular instance of network N, that satisfy

$$z_p^*(T_A') = \min_{T \in \mathcal{T}_A^{(1)}} z_p^*(T)$$

$$z_p^*(T_A'') = \min_{T \in \mathcal{T}_A^{(2)}} z_p^*(T)$$

If an individual instance is denoted by I, then G1(I) and G2(I) (abbreviated as G1 and G2 in the tables) are defined as :

$$G1(I) = 100 \cdot \frac{z_p^*(T_A') - z_p^*(N)}{z_p^*(N)}$$

$$G2(I) = 100 \cdot \frac{z_p^*(T_A'') - z_p^*(N)}{z_p^*(N)}$$

Replace indices 'A' with 'I' for the set of I-RSPTs. Note that, a nonzero entry in a G2 column means that the optimal tree was not an RSPT for that instance. A nonzero entry in a G1 column but a zero entry in the associated G2 column means that the optimal tree is actually an RSPT for that instance, but since all the ties were broken incorrectly in Stage 1, it was not possible to construct and solve the optimal tree in the first stage.

Now we can move to the summary tables. Let the term *success* refer to finding the optimal tree in the set of trees tested, for a particular instance. Similarly, the term *failure* refers to failure in finding the optimal tree in the set of trees tested. The column headings used in the summary tables are, then, defined as follows :

G Group number

T Explanation of a particular group.

SG Subgroups of a particular group (**W**eighted and **U**nweighted).

T# Total number of instances in a particular (sub)group.

SR_i Success ratio in stage i, i.e. the percentage of the instances in a particular (sub)group for which the optimal tree was found in the set of trees tested.

CSR Cumulative success ratio after stage 2.

FR Failure ratio, the percentage of the instances for which the optimal tree could not be found even after the second stage.

MG_i Maximum gap in the corresponding (sub)group of instances in stage i. Let $\mathcal{I}$ denote a (sub)group of instances. Then, MG_1 (MG_2) of a particular (sub)group $\mathcal{I}$ is defined as :

$$MG_1 = \max_{I \in \mathcal{I}} G1(I)$$

AG_i Average gap in the corresponding (sub)group of instances, in stage i. Let $\mathcal{I}$ be defined similarly. Further, let $|\mathcal{I}|$ be the number of instances in the (sub)group $\mathcal{I}$. Then AG_1 (AG_2) of $\mathcal{I}$ is defined as :

$$AG_1 = \frac{\sum_{I \in \mathcal{I}} G1(I)}{|\mathcal{I}|}$$

Note that, in the summary tables, type (a) and type (b) results are displayed separately under the headings 'SUCCESS/FAILURE' and 'GAPS'.

In the rest of this chapter and in the Appendix, the 4-tuple $(n/d/WS/RTS)$ will be used to refer different instances of the random networks. Here, n and d are as defined before, WS is the *weight status* (either 'W' or 'U', referring to whether the network is weighted or unweighted) and RTS is the *rooted tree status* (either 'A' or 'I', referring to the types of trees, A- or I-RSPT, that are constructed from the network and tested).

The edge lengths and weights of random networks were initially designed to take values uniformly from sets $\{1,2,3,4,5\}$ and $\{1,2,3\}$, respectively. However, it might be the case that these values restrict the networks that are tested in this study, to a narrow subset of the entire population of networks (because of the narrow range of values for the two parameters). This may result in ignoring some instances that do not conform to the results regarding the instances actually tested. To avoid this, the edge lengths and weights were allowed to take values, again uniformly, from a wider set of values, namely the set $\{1,2,\ldots,20\}$. Again, 240 instances of random networks, with this property, were generated by NETGEN and tested for three p values, as in the previous case. However, for this set of instances, the exhaustive construction and testing (Stage 2) was NOT performed. The first stage on these instances, actually, resulted in an improvement on the percentages of finding the optimal trees, the

amounts of maximum and average gaps. There were no instances that gave gaps larger than 100% (the largest gap in the previous case) and the overall percentage of finding the optimal tree was better than that of the previous case. So, Stage 2 was not performed on this set of instances.

The following two sections discuss the results of the experiments on A- and I-RSPTs, respectively. The related statistics with respect to four major groups are displayed in Tables 4.1, 4.2, 4.3 and 4.4.

4.2.1 With Adjacent Antipodal Segments

The experiments on the A-RSPTs are performed for 240 random instances of cyclic networks with 3 different values of p for each, as described in the previous section. The major question is the following : does this set of spanning trees necessarily contain the optimal tree? The results of the experiment, which will be displayed in detail, show that the answer to this question is 'NO', in other words, the optimal tree *need not* be an A-RSPT. However, it turns out that such trees are good approximations, at least empirically, to the optimal tree, in terms of the p-radius of the cyclic network. Furthermore, the empirical evidence supports the idea that failure to find the optimal tree as an A-RSPT does not occur very frequently.

Tables 4.1 and 4.2 are the summary tables for experiments on A-RSPTs, that correspond to two groups of instances with respect to weight and edge length values. Let P_1 denote the group where edge lengths and weights come from set $\{1,2,3,4,5\}$ and $\{1,2,3\}$, respectively. Similarly, let P_2 denote the group where both edge lengths and weights come from set $\{1,2,3,\ldots,20\}$.

With P_1 (see Table 4.1), the success ratio in Stage 1 is 83.75% and the cumulative success ratio (CSR) is 96.11%, for $(\cdot/\cdot/\cdot/A)$, the overall set of instances. This means that, if all the ties were broken appropriately in stage 1, we would have only 3.89% of the instances with failure to find the optimal tree. Observe that all the values SR_1, SR_2, CSR and FR are *better* in the unweighted case compared to the weighted case, and this is true for almost

				SUCCESS/FAILURE				GAPS			
G	T	SG	T#	SR_1	SR_2	CSR	FR	MG_1	MG_2	AG_1	AG_2
	ALL		720	83.75	76.06	96.11	3.89	100.0	100.0	42.7	7.8
1	W		360	81.67	75.78	95.56	4.44	77.8	33.3	24.9	4.7
	U		360	85.83	76.53	96.67	3.33	100.0	100.0	64.1	11.7
2	25%	T	240	87.92	82.78	97.92	2.08	100.0	25.0	35.6	3.4
		W	120	87.50	80.00	97.50	2.50	77.8	11.1	28.7	1.8
		U	120	88.33	85.71	98.33	1.67	100.0	25.0	43.1	5.0
	50%	T	240	83.33	67.50	94.58	5.42	100.0	100.0	42.6	10.7
		W	120	80.83	69.57	94.17	5.83	50.0	25.0	24.4	5.9
		U	120	85.83	64.71	95.00	5.00	100.0	100.0	67.1	18.2
	75%	T	240	80.00	79.17	95.83	4.17	100.0	100.0	46.5	7.9
		W	120	76.67	78.57	95.00	5.00	50.0	33.3	25.8	5.3
		U	120	83.33	80.00	96.67	3.33	100.0	100.0	76.2	11.7
3	$\lfloor n/4 \rfloor$	T	240	70.00	65.27	89.58	10.42	100.0	100.0	36.1	10.4
		W	120	66.67	65.00	88.33	11.67	50.0	33.3	25.7	6.3
		U	120	73.33	65.63	90.83	9.17	100.0	100.0	49.1	15.5
	$n/2$	T	240	84.58	91.87	98.75	1.25	100.0	100.0	57.2	4.3
		W	120	84.17	89.47	98.33	1.67	66.7	33.3	27.1	3.1
		U	120	85.00	94.44	99.17	0.83	100.0	100.0	88.9	5.6
	$\lceil 3n/4 \rceil$	T	240	96.67	100.00	100.00	0.00	100.0	0.0	31.6	0.0
		W	120	94.17	100.00	100.00	0.00	77.8	0.0	21.8	0.0
		U	120	99.17	100.00	100.00	0.00	100.0	0.0	100.0	0.0
4	$n = 10$	T	180	95.56	100.00	100.00	0.00	100.0	0.0	41.6	0.0
		W	90	94.44	100.00	100.00	0.00	77.8	0.0	31.6	0.0
		U	90	96.67	100.00	100.00	0.00	100.0	0.0	58.3	0.0
	$n = 20$	T	180	86.67	83.33	97.78	2.22	100.0	33.3	29.5	4.1
		W	90	85.56	84.62	97.78	2.22	33.3	11.1	19.6	1.6
		U	90	87.78	81.82	97.78	2.22	100.0	33.3	44.2	7.1
	$n = 30$	T	180	81.11	76.47	95.56	4.44	100.0	50.0	43.9	6.3
		W	90	81.11	76.47	95.56	4.44	60.0	33.3	24.2	4.3
		U	90	81.11	76.47	95.56	4.44	100.0	50.0	64.2	8.3
	$n = 40$	T	180	71.67	68.63	91.11	8.89	100.0	100.0	47.1	11.6
		W	90	65.56	67.74	88.89	11.11	66.7	33.3	28.6	7.0
		U	90	77.78	70.00	93.33	6.67	100.0	100.0	75.8	18.8

Table 4.1: Summary table for $(\cdot/\cdot/\cdot/A)$ instances

				SUCC./FAIL.		GAPS	
G	T	SG	T#	SR_1	FR	MG_1	AG_1
	ALL		720	87.78	12.22	50.0	18.0
1	W		360	88.06	11.94	30.6	13.3
	U		360	87.50	12.50	50.0	22.3
2	25%	T	240	91.25	8.75	33.3	13.2
		W	120	90.83	9.17	27.0	12.5
		U	120	91.67	8.33	33.3	13.9
	50%	T	240	85.42	14.58	50.0	18.4
		W	120	86.67	13.33	30.6	15.5
		U	120	84.17	15.83	50.0	20.8
	75%	T	240	86.67	13.33	50.0	20.6
		W	120	86.67	13.33	26.6	11.7
		U	120	86.67	13.33	50.0	28.6
3	$\lfloor n/4 \rfloor$	T	240	70.00	30.00	50.0	16.9
		W	120	72.50	27.50	26.6	12.4
		U	120	67.50	32.50	50.0	20.7
	$n/2$	T	240	93.75	6.25	50.0	23.3
		W	120	92.50	7.50	30.6	17.1
		U	120	95.00	5.00	50.0	30.3
	$\lceil 3n/4 \rceil$	T	240	99.58	0.42	12.5	12.5
		W	120	99.17	0.83	12.5	12.5
		U	120	100.00	0.00	—	—
4	$n = 10$	T	180	96.11	3.89	13.0	7.7
		W	90	98.89	1.11	2.9	2.9
		U	90	93.33	6.67	13.0	8.6
	$n = 20$	T	180	90.56	9.44	33.3	13.4
		W	90	91.11	8.89	26.6	8.6
		U	90	90.00	10.00	33.3	17.8
	$n = 30$	T	180	86.67	13.33	50.0	20.1
		W	90	87.78	12.22	25.9	15.4
		U	90	85.55	14.44	50.0	23.8
	$n = 40$	T	180	77.78	22.22	50.0	20.4
		W	90	74.44	25.56	30.6	14.5
		U	90	81.11	18.89	50.0	28.0

Table 4.2: Summary results for $(\cdot/\cdot/\cdot/A)$ instances with w and l from set $\{1,\ldots,20\}$

all of the groups and subgroups. From this viewpoint, the A-RSPTs seem to perform better on unweighted instances.

In general, Stage 2 was able to eliminate most of the failure instances of Stage 1. In terms of the maximum gaps, no gap higher than 100% was observed. Contrary to the success ratios, the U instances had worse (higher) maximum and average gaps compared to the W instances. Stage 2, although does not change MG, decreases AG significantly for both W and UW groups.

The results with respect to groups 2,3 and 4 are summarized as follows :

2. A general pattern is that the Stage 1 and Stage 2 success ratios decrease as the density increases. For the U subgroups, the success ratio is again generally higher compared to W subgroups. The MG does not change with density (always 100%), but AG increases apparently as the density increases. Stage 2 decreases both the failure rates and average gaps of Stage 1 significantly but does not improve maximum gaps except for $d = 25\%$. The A-RSPTs apparently performed better on relatively sparse instances of networks.

3. The Stage 1 and 2 success ratios increase as p gets nearer to n or as p/n increases. Again, SR_i are higher for the U instances. The major problem, in terms of success ratios, is with $p = \lfloor n/4 \rfloor$. In this case, SR_1 is quite below the overall average SR_1. The least improvements in maximum and average gaps occurred again with this case, which implies that A-RSPTs show relatively poor performance for this case.

4. As n increases, all the success ratios decrease and amount of maximum and average gaps increase. Although the A-RSPTs again perform better for the U instances, the performance significantly decreases as the problem size (n) increases.

With P_2 (see Table 4.2), the success rate in Stage 1 is 87.78% which is better than the case with P_1. Since Stage 2 is not performed for this group of instances, FR in this case is not comparable with FR in the P_1 case. However, a general

pattern of the results indicate that when the edge lengths and weights take values from a wider set, both the success ratios, maximum and average gaps improve. Furthermore, we observe, contrary to the P_1 case, that the A-RSPTs perform better on W instances for most of the groups and subgroups. The patterns within each group, although similar to the P_1 case, are summarized as follows :

2. The success ratio is relatively high for sparse instances in this group. The maximum and average gaps in overall figures increase as density increases, though this is slightly violated for W and U subgroups. The A-RSPTs seem to perform better again on relatively sparse instances of networks.

3. The problem in the P_1 case is also observed in this group. For $p = \lfloor n/4 \rfloor$, the success ratio is quite below the overall figure and the maximum and average gaps are considerably large. The A-RSPTs performed poorly in this case. On the other hand, for $p = \lceil 3n/4 \rceil$, the A-RSPTs contain the optimal tree for almost all instances in the group. Again, the performance of the A-RSPTs increase as p gets nearer to n.

4. The pattern is similar to the P_1 case, as the problem size increases, the performance of the A-RSPTs (in terms of both the success ratios and gaps) decrease.

Observe that, in either of the cases P_1 or P_2, unweighted instances give larger maximum and average gaps than corresponding weighted instances. This is possibly because of the following : for the unweighted instances, a z value associated with an intersection point either has a fractional part of 0.5 or is integer. This is because the z value generated by a pair of vertices v_i, v_j on a particular path joining them is just half the distance between v_i and v_j. Because of this structure of z values, whenever a gap occurs between a network p-radius and a tree p-radius, it is a positive integer multiple of 0.5 and this may cause the gap to be high when the network p-radius is small in quantity (if the network p-radius is 0.5 and tree p-radius is the possible smallest value with a

gap, which is 1, then the gap is 100% in terms of network p-radius). In short, the z values for the unweighted instances are more discretized compared to those of the weighted instances. Observe from the tables in Appendix that a number of gap figures for unweighted instances have values 100%. However, for the weighted instances, the fractional parts of z values can theoretically be anything from the interval $[0, 1)$. So, the tree p-radii can get closer to network p-radii for the weighted instances.

Note that, the average gaps for the group P_2 are significantly smaller than those of the group P_1. This might be because a wider range of values for the weights and edge lengths produce a similar effect on the z values, as described above. That is, a wider range of values may generate a relatively more *continuous* set of z values. This argument can also be used to explain why the success ratios are relatively high in the P_1 case.

Another related observation is the following : in the P_1 case, the U instances apparently gave better success ratios. However, in the P_2 case, we observe that this distinction is not so strict. For a considerable number of W subgroups, the success ratio is higher than the U instances. This can be explained by a similar reasoning. Moving from P_1 to P_2, the z values corresponding to weighted instances become more *continuous* which may increase the possibility of finding the optimal tree. However, for the U instances, the structure of z values do not change, only a number of new values are added (in practice, it was observed that, the number of z values significantly increased from P_1 to P_2). Hence, it is reasonable for the W instances to have better success ratios compared to P_1 case and the success ratios of U instances.

Note that Stage 2 (with P_1) eliminates most of the failure instances of Stage 1 and improves maximum and especially average gaps significantly. However, since this stage involves the exhaustive construction of trees and is not polynomial, it is computationally very expensive to perform this stage for large problem sizes. To avoid such a computational burden, some rules for breaking ties among alternative shortest paths in Stage 1 would be useful in order to get better average or maximum gap figures. The concepts of *critical* and

non-critical edges (see Section 3.3) can be used to eliminate some of the edges (if they exist) once an upper bound $\bar{z}$ on the p-radius of the cyclic network is known. The non-critical edges are identified with respect to $\bar{z}$ and whenever such edges appear in a tie for the alternative shortest paths, they are ignored and the set of edges in the tie is reduced in size. The two problems with this method are the following : an upper bound may not be available, and even if $\bar{z}$ is known, there may be no non-critical edge. The first one is relatively easy to overcome, the best p-radius of a particular set of spanning trees may be used as an upper bound, but the value of $\bar{z}$ can be still so high that no non-critical edge can be identified with respect to this $\bar{z}$. To avoid this, some method to find a good $\bar{z}$ has to be known.

4.2.2 With Intersection Points

The same 1440 instances (720 for P_1 and 720 for P_2) are used to test the I-RSPTs. Recall that, the maximum number of I-RSPTs that can be constructed is $O(|E| \cdot n^2)$ where the same number of A-RSPTs is $O(|E| \cdot n)$. So the testing of I-RSPTs requires more computational effort.

The tables 4.3 and 4.4 are the two summary tables for the testing of I-RSPTs with P_1 and P_2, respectively. The answer to our major question is again 'NO', i.e. the optimal tree need not be an I-RSPT. However, the maximum gap occurred in the experiment is 100% which supports the idea that these trees are also good approximations of the optimal tree. In general, we observe that the success ratios are lower and maximum and average gaps are higher with the I-RSPTs compared to A-RSPTs. Hence, empirical evidence imply that the A-RSPTs are better approximations of the optimal tree than the I-RSPTs are (for both the P_1 and P_2 instances).

With P_1 (see Table 4.3), the overall success ratio is 81.94% in the first stage and this goes up to 93.89% after the second stage. That is, 6.11% of the instances ended up with failure to find the optimal tree. Note that, this figure is larger compared to that of the A-RSPTs (which was 3.89%). Also, the U

				SUCCESS/FAILURE				GAPS			
G	T	SG	T#	SR_1	SR_2	CSR	FR	MG_1	MG_2	AG_1	AG_2
	ALL		720	81.94	66.17	93.89	6.11	100.0	100.0	43.4	9.7
1	W		360	80.28	67.60	93.61	6.39	77.8	33.3	25.8	5.6
	U		360	83.61	64.42	94.17	5.83	100.0	100.0	64.6	14.6
2		T	240	86.25	66.69	95.42	4.58	100.0	25.0	34.6	5.8
	25%	W	120	85.83	64.71	95.00	5.00	77.8	20.0	27.3	4.2
		U	120	86.67	68.75	95.83	4.17	100.0	25.0	42.4	7.5
		T	240	81.67	61.38	92.92	7.08	100.0	100.0	44.6	11.1
	50%	W	120	80.00	66.67	93.33	6.67	66.7	25.0	26.6	5.5
		U	120	83.33	55.00	92.50	7.50	100.0	100.0	66.2	17.9
		T	240	77.92	69.81	93.33	6.67	100.0	100.0	48.5	10.9
	75%	W	120	75.00	70.00	92.50	7.50	50.0	33.3	24.7	6.5
		U	120	80.83	69.57	94.17	5.83	100.0	100.0	78.6	16.7
3		T	240	67.50	53.85	85.00	15.00	100.0	100.0	36.2	13.7
	$\lfloor n/4 \rfloor$	W	120	65.00	57.14	85.00	15.00	50.0	33.3	25.4	7.3
		U	120	70.00	50.00	85.00	15.00	100.0	100.0	48.9	21.2
		T	240	81.67	81.82	96.67	3.33	100.0	100.0	58.2	4.4
	$n/2$	W	120	81.67	77.27	95.83	4.17	66.7	33.3	27.8	4.2
		U	120	81.67	86.36	97.50	2.50	100.0	100.0	88.6	4.5
		T	240	96.67	100.00	100.00	0.00	100.0	0.0	31.6	0.0
	$\lceil 3n/4 \rceil$	W	120	94.17	100.00	100.00	0.00	77.8	0.0	21.8	0.0
		U	120	99.17	100.00	100.00	0.00	100.0	0.0	100.0	0.0
4		T	180	95.00	100.00	100.00	0.00	100.0	0.0	42.0	0.0
	$n = 10$	W	90	95.56	100.00	100.00	0.00	77.8	0.0	38.2	0.0
		U	90	94.44	100.00	100.00	0.00	100.0	0.0	45.0	0.0
		T	180	85.00	58.52	93.78	6.22	100.0	50.0	33.4	9.8
	$n = 20$	W	90	83.33	66.67	94.44	5.56	33.3	25.0	20.5	5.2
		U	90	86.67	50.00	93.33	6.67	100.0	50.0	49.6	15.6
		T	180	78.33	66.67	92.78	7.22	100.0	50.0	43.8	7.7
	$n = 30$	W	90	77.78	65.00	92.22	7.78	50.0	33.3	24.2	5.1
		U	90	78.89	68.42	93.33	6.67	100.0	50.0	64.5	10.5
		T	180	69.44	63.64	88.89	11.11	100.0	100.0	48.2	12.6
	$n = 40$	W	90	64.44	65.63	87.78	12.22	66.7	33.3	27.7	6.8
		U	90	74.44	60.87	90.00	10.00	100.0	100.0	75.8	20.7

Table 4.3: Summary table for $(\cdot/\cdot/\cdot/I)$ instances

				SUCC./FAIL.		GAPS	
G	T	SG	T#	SR_1	FR	MG_1	AG_1
	ALL		720	86.67	13.33	66.7	18.2
1	W		360	86.94	13.06	30.6	12.6
	U		360	86.39	13.61	66.7	23.3
2	25%	T	240	90.83	9.17	33.3	13.6
		W	120	90.00	10.00	27.0	11.7
		U	120	91.67	8.33	33.3	15.9
	50%	T	240	84.17	15.83	50.0	18.4
		W	120	85.83	14.17	30.6	16.0
		U	120	82.50	17.50	50.0	20.4
	75%	T	240	85.00	15.00	66.7	20.7
		W	120	85.00	15.00	26.6	10.0
		U	120	85.00	15.00	66.7	31.3
3	$\lfloor n/4 \rfloor$	T	240	68.33	31.67	66.7	17.1
		W	120	70.83	29.17	27.4	11.9
		U	120	65.83	34.17	66.7	21.4
	$n/2$	T	240	92.08	7.92	50.0	23.1
		W	120	90.83	9.17	22.8	14.9
		U	120	93.33	6.67	50.0	35.9
	$\lceil 3n/4 \rceil$	T	240	99.58	0.42	12.5	12.5
		W	120	99.17	0.83	12.5	12.5
		U	120	100.00	0.00	—	—
4	$n = 10$	T	180	96.11	3.89	13.0	8.6
		W	90	98.89	1.11	2.9	2.9
		U	90	93.33	6.67	13.0	9.5
	$n = 20$	T	180	88.89	11.11	33.3	14.9
		W	90	91.11	8.89	26.6	8.6
		U	90	86.67	13.33	33.3	19.1
	$n = 30$	T	180	85.56	14.44	33.3	18.4
		W	90	85.56	14.44	25.3	14.3
		U	90	85.56	14.44	33.3	22.4
	$n = 40$	T	180	76.11	23.89	66.7	21.1
		W	90	72.22	27.78	30.6	13.4
		U	90	80.00	20.00	66.7	31.9

Table 4.4: Summary results for $(\cdot/\cdot/\cdot/I)$ instances with w and l from set $\{1,\ldots,20\}$

instances perform better than W instances in terms of success ratios, but give larger maximum and average gaps than W instances do (as in the case with A-RSPTs). Note that, although the success ratios of stage 2 (SR_2) are not as high as the SR_1 values, the second stage eliminates a significant portion of the failure instances of stage 1.

The general pattern within each (sub)group is quite similar to the A-RSPT case, and will be summarized as follows :

2. Success ratios decrease as the density increases. Maximum gaps do not change too much with the density but average gaps increase as the density increases. U instances give better results in terms of the statistics. The I-RSPTs seem to perform better on relatively sparse and unweighted instances.

3. Success ratios significantly increase as p gets nearer to n. Maximum and average gaps do not change much with p. The performance of I-RSPTs is quite below the overall values for $p = \lfloor n/4 \rfloor$. The U instances give better results also in this group. The I-RSPTs perform better on values of p nearer to n.

4. The performance (in terms of SR_i, MG_i and AG_i) gets worse as the problem size increases. The I-RSPTs apparently give better results on U instances.

With P_2 (see Table 4.4), the success ratio is 86.67% which is slightly less than the 87.78% of the corresponding A-RSPTs. The general pattern of results shows an improvement in terms of the success ratios and gaps, compared to P_1 instances. We also observe that the distinction between W and U instances (in finding the optimal tree) is not so clear with this group of instances. We may explain this by the same argument in the previous section : the z values become more continuous when the edge lengths and weights come from a wider range of values.

The results within each group is summarized similarly as follows :

2. Performance decreases as the density increases. I-RSPTs perform better on relatively sparse instances.

3. Performance increases as p becomes nearer to n. A significant portion (31.67%) of the instances fail to give the optimal tree as an I-RSPT. The trees perform better on relatively larger values of p.

4. Performance apparently decreases as the problem size increases.

The difference between the W and U instances in terms of the success ratios and gaps may be explained by the continuity of the z values. In general, the I-RSPTs give similar results to the A-RSPTs, but the performance of I-RSPTs is slightly worse than A-RSPTs. This makes us prefer to approximate the optimal tree by the A-RSPTs because these spanning trees are also fewer in quantity. In addition to this, note that stage 2 (with P_1 instances) again reduces the number of failure instances significantly, hence breaking the ties appropriately in the construction phase is again an important matter of concern. The method with non-critical edges is also applicable with the I-RSPTs.

4.3 Summary

The following is a summary of the experiment results in general :

- The optimal tree *need not* be an A- or I-RSPT.
- A-RSPTs are better approximations of the optimal tree than I-RSPTs are.
- The trees perform better on relatively sparse instances of networks.
- The trees perform better on larger values of p with respect to n.
- The performance decreases as the problem size increases.

- Unweighted instances give better results when the weights and edge lengths come from a narrow range of values, but it is highly probable that W instances will perform better for wider range of values.

- Making a weighted instance unweighted does not always help to find the optimal tree. There were some unweighted instances with failure to find the optimal tree even though the corresponding weighted instance gave a success (see Tables A.3 and A.4 for the instances (40/75/W/A) and (40/75/U/A) for $p = n/2$).

Chapter 5

Conclusion

In this thesis, a new approach to solving the absolute p-center problem on cyclic networks is discussed. The problem is known to be $\mathcal{NP}$-Hard (Kariv and Hakimi 1979 [25]) on cyclic networks but polynomial-time solvable on tree networks (due to various polynomial algorithms). The new approach is based on identifying a spanning tree (an optimal tree) of a particular network under consideration, whose p-center and p-radius are also optimal to the cyclic network.

In Chapter 1, the absolute p-center problem is formulated. The related literature is discussed in Chapter 2 where, also, the solution procedures that are implemented in this study are given. The procedure for solving the problem on cyclic networks and the one for solving on tree networks are both based on solving a sequence of covering problems. The number of times the covering problem is solved is polynomial in either case, however, solving the covering problem itself on cyclic networks is $\mathcal{NP}$-Hard (by Kariv and Hakimi 1979 [25]) and polynomial on tree networks. Hence, the procedure used to solve the p-center problem on cyclic networks is performed in exponential time with time complexity $O\left(\log(|E| \cdot n^2) \cdot \frac{|E|^p \cdot n^{2p}}{p!}\right)$ and the one used for tree networks is performed in polynomial time ($O(n^2 \cdot \log n)$).

Chapter 3 is the core part of the thesis in that the fundamental theorem

regarding the existence of an optimal tree is proved and the optimal tree is analyzed for properties of identification. It turns out that, such an optimal tree is composed of p disjoint subtrees (T_j's) each of which is the subtree of the original network rooted at the optimal center x_j^* and that spans the vertices $v_i \in V_j$ covered by x_j^* (ties in 'covering' are broken by the smallest index of center).

When the network N has a cyclic and possibly a dense edge structure, there is an exponential number of distinct spanning trees of N. Hence, it is inefficient to construct all the spanning trees and solve the p-center problem on them. Instead, we may restrict the search for the optimal tree on a set of trees of polynomial number of elements and of some common property. Two types of spanning trees are introduced in Chapter 3 for this purpose : spanning trees rooted at (a) adjacent antipodal segments (A-RSPT) and (b) intersection points (I-RSPT) of N. The total number of A-RSPTs and I-RSPTs that can be constructed are $O(|E| \cdot n)$ and $O(|E| \cdot n^2)$. However, one major assumption here is that, alternative shortest paths may be encountered in the construction phase which may increase the number of trees exponentially, and to keep the number of trees polynomial, only one of such alternative paths is included in the construction.

The existence of alternative shortest paths impose another problem on the search for the optimal tree. We may pick a wrong path in all of the constructed trees, hence miss the optimal tree even if it is an A- or I-RSPT. One way to overcome this difficulty is to enumerate all the shortest paths and construct all the rooted trees. Since the number of these trees is exponential, this works only for small n. Actually, this is the only way to be sure to test all the rooted shortest path trees.

Chapter 3 also provides a necessary condition for an edge's being critical. Since, the vertices in the same partition must be covered by a single common center with the value $z_p^*(N)$, $v_i, v_j \in V_k$ for some partition V_k implies that

$$\beta_{ij} = \frac{d'_{ij}}{\frac{1}{w_i} + \frac{1}{w_j}} \leq z_p^*(N)$$

must be satisfied for the pair v_i, v_j (assume $w_i, w_j > 0$). Since there is no redundant edge on the network, each edge is also the shortest path between its endpoints, meaning that for edge-defining pairs $v_i, v_j \in V$ (i.e., $e_{ij} = (v_i, v_j) \in E$) with $w_i, w_j > 0$, the violation of the above inequality implies e_{ij}'s being non-critical (e_{ij} cannot appear on any of the T_j's). Of course, this analysis requires the knowledge of $z_p^*(N)$ a priori, however when there is an upper bound for $z_p^*(N)$, this may be used to eliminate some edges (without changing the p-center and the p-radius) from the network.

The last two sections of Chapter 3 discuss two special networks, that are more general than a tree, where the optimal tree can be found in a polynomial set of spanning trees. The first one is the simple cycle (and its generalized version) where the entire set of spanning trees is polynomial. The second one is the cactus network which is a collection of W simple cycles that do not pairwise intersect at more than one vertex. For $p = 2$, an optimal tree of any cactus is shown to be included in an $O(n^6)$ number of spanning trees of the cactus.

To see whether the optimal tree of a particular network is necessarily an A- or I-RSPT, an experimental study is designed and implemented, which is discussed in Chapter 4. A total number of 2880 random instances of the problem are tested for both types and it was observed, after the exhaustive stage of the experiment, that the optimal tree need not be an A- or I-RSPT. However, most of the time, such trees were at least empirically good approximations of the optimal tree. The gap, in the p-radius, between the best of the trees and the original network N was observed to be at most 2 times the p-radius of N. Actually, the best known heuristic for the problem provides a 2-approximation on general networks. It was also observed that the chance of finding the optimal tree with the RSPTs tested is higher for the unweighted instances when the edge lengths and weights come from a narrow range of values. Also the trees perform well on relatively sparse networks. But, such trees' failing to contain the optimal tree is more likely for small values of p relative to n. Although the RSPTs seemed to perform better in the unweighted case, they generated

larger gaps than the weighted case and the weighted instances gave larger success ratios when the edge lengths and weights were designed to come from a wider range of values. However, in either case, the maximum gap was 100%. Another major observation was that making a failure weighted instance unweighted did not help to find the optimal tree for some instances. And for some other instances, the unweighted network gave failure to find the optimal tree although the corresponding weighted instance did not give a failure.

The existence of an optimal tree of a particular network N brings some other questions that might sketch the direction of future research on the subject. The problem of identifying a polynomial number of spanning trees that include the optimal tree still remains unsolved. In fact, this problem must be at least as hard as the p-center problem since solving this problem would mean solving the p-center problem in polynomial time. Furthermore, more properties of the optimal tree may be identified so that the search may be restricted on a narrower set of spanning trees. Some restrictions on the data of the network might be introduced so that this narrow set of spanning trees is reduced to a polynomial set. In addition, to find good rules for breaking ties that arise in the construction of the RSPTs is an important matter of concern. All these aspects imply that this approach may have an important potential for solving the problem more efficiently, or devising more efficient solution procedures for the generalization of the absolute p-center problem, like the continuous or nonlinear versions.

Appendix A

Gap Summary for $(\cdot/\cdot/\cdot/A)$ Instances

			$\lfloor n/4 \rfloor$		$n/2$		$\lceil 3n/4 \rceil$	
n	d	S	$G1$	$G2$	$G1$	$G2$	$G1$	$G2$
10	25	10	25.0	0.0			77.8	0.0
	50	08	5.0	0.0				
	75	02			25.0	0.0		
		07	25.0	0.0				
20	25	02	25.0	0.0				
		08	11.1	0.0				
		10	4.2	0.0				
	50	03	25.0	9.4				
		05	25.0	0.0	12.5	0.0		
		08	33.3	11.1				
		10	12.5	0.0				
	75	02	20.0	0.0	33.3	0.0		
		03	20.0	0.0				
		09			12.5	0.0		
		10	20.0	0.0				
30	25	01	33.3	0.0				
		04			60.0	0.0		
		06	25.0	9.4				
		09	20.0	0.0				
	50	01	11.1	0.0				
		03			12.5	0.0		
		04	25.0	0.0				
		05	20.0	0.0				
		10	25.0	20.0				
	75	01			33.3	33.3		
		03					12.5	0.0
		04	25.0	0.0	11.1	0.0		
		05			11.1	0.0		
		07	25.0	11.1				
		08	25.0	0.0				
		10	25.0	0.0				

Table A.1: Results for $(10, 20, 30/\cdot/W/A)$ instances

			$\lfloor n/4 \rfloor$		$n/2$		$\lceil 3n/4 \rceil$	
n	d	S	$G1$	$G2$	$G1$	$G2$	$G1$	$G2$
10	25	10	50.0	0.0			100.0	0.0
	75	01	25.0	0.0				
20	25	03	25.0	0.0				
		09	20.0	20.0	50.0	0.0		
	50	05	25.0	0.0				
		08	25.0	0.0				
		09	25.0	25.0				
	75	04	33.3	0.0				
		05	33.3	33.3				
		08	50.0	0.0	100.0	0.0		
		10			100.0	0.0		
30	25	01	25.0	0.0				
		03			50.0	0.0		
		05	50.0	0.0				
		06	50.0	25.0				
		08			50.0	0.0		
	50	04	33.3	33.3				
		06	33.3	0.0	100.0	0.0		
		09	33.3	0.0				
		10	33.3	33.3	100.0	0.0		
	75	04	100.0	50.0	100.0	0.0		
		05			100.0	0.0		
		07	33.3	0.0				
		08			100.0	0.0		
		09	100.0	0.0				

Table A.2: Results for $(10, 20, 30/\cdot/U/A)$ instances

			$\lfloor n/4 \rfloor$		$n/2$		$\lceil 3n/4 \rceil$	
n	d	S	$G1$	$G2$	$G1$	$G2$	$G1$	$G2$
40	25	02	40.0	6.7				
		03			33.3	0.0		
		05	25.0	0.0				
		06	25.0	0.0				
		08			12.8	0.0		
		10	11.1	11.1				
	50	01	25.0	25.0				
		02	20.0	0.0	25.0	0.0		
		03					12.5	0.0
		04	33.3	20.0				
		05			25.0	0.0		
		06			66.7	25.0		
		07	38.9	0.0	25.0	0.0		
		08	50.0	25.0			12.5	0.0
		10	20.0	0.0				
	75	01	4.2	0.0				
		02	50.0	20.0			12.5	0.0
		03			25.0	0.0		
		04	50.0	33.3			12.5	0.0
		05			33.3	0.0	12.5	0.0
		06	50.0	0.0				
		08			33.3	0.0		
		09	50.0	25.0	25.0	0.0		
		10	25.0	25.0				

Table A.3: Results for $(40/\cdot/W/A)$ instances

			$\lfloor n/4 \rfloor$		$n/2$		$\lceil 3n/4 \rceil$	
n	d	S	$G1$	$G2$	$G1$	$G2$	$G1$	$G2$
40	25	01			50.0	0.0		
		02	25.0	0.0				
		05	25.0	25.0				
		08	33.3	0.0				
	50	01			100.0	0.0		
		04	100.0	100.0				
		05	100.0	50.0				
		07	33.3	0.0				
		08	100.0	50.0	100.0	0.0		
		10	100.0	0.0	100.0	0.0		
	75	01	50.0	0.0	100.0	0.0		
		04			100.0	0.0		
		05	50.0	0.0	100.0	100.0		
		06	50.0	50.0				
		07			100.0	0.0		
		09	100.0	0.0				

Table A.4: Results for $(40/\cdot/U/A)$ instances

Appendix B

Gap Summary for $(\cdot/\cdot/\cdot/I)$ Instances

			$\lfloor n/4 \rfloor$		$n/2$		$\lceil 3n/4 \rceil$	
n	d	S	$G1$	$G2$	$G1$	$G2$	$G1$	$G2$
10	25	10	25.0	0.0			77.8	0.0
	75	02			25.0	0.0		
		07	25.0	0.0				
20	25	02	25.0	0.0				
		07	20.0	20.0				
		08	11.1	0.0				
		10	4.2	0.0				
	50	03	25.0	0.0				
		05	25.0	0.0	12.5	0.0		
		06	20.0	11.1				
		08	33.3	11.1				
	75	02	20.0	0.0	33.3	0.0		
		03	20.0	11.1				
		05	33.3	25.0				
		09			12.5	0.0		
		10	12.5	0.0				
30	25	01	18.5	0.0				
		04			50.0	0.0		
		06	25.0	9.4				
		09	20.0	0.0				
		10	20.0	12.5				
	50	01	25.0	11.1	11.1	0.0		
		03			12.5	0.0		
		04	25.0	0.0				
		05	20.0	0.0	50.0	12.5		
		10	25.0	0.0				
	75	01			33.3	33.3		
		03					12.5	0.0
		04	25.0	0.0				
		05			11.1	11.1		
		07	25.0	0.0				
		08	25.0	0.0	25.0	11.1		
		10	25.0	0.0				

Table B.1: Results for $(10, 20, 30/\cdot/W/I)$ instances

			$\lfloor n/4 \rfloor$		$n/2$		$\lceil 3n/4 \rceil$	
n	d	S	$G1$	$G2$	$G1$	$G2$	$G1$	$G2$
10	25	10	50.0	0.0			100.0	0.0
	50	10	16.7	0.0				
	75	01	25.0	0.0				
		09	33.3	0.0				
20	25	03	25.0	0.0				
		06	25.0	25.0				
		09	20.0	20.0	50.0	0.0		
	50	01	33.3	33.3				
		09	25.0	25.0	50.0	0.0		
	75	04	33.3	0.0				
		05	33.3	33.3				
		08	100.0	50.0	100.0	0.0		
		10			100.0	0.0		
30	25	01	25.0	0.0				
		03			50.0	0.0		
		05	50.0	25.0				
		06	50.0	25.0				
		08	50.0	0.0				
	50	04	33.3	33.3				
		06	33.3	33.3	100.0	0.0		
		08	33.3	0.0				
		09	33.3	0.0				
		10	33.3	33.3	100.0	0.0		
	75	02			100.0	0.0		
		04	100.0	50.0	100.0	0.0		
		05			100.0	0.0		
		07	33.3	0.0				
		09	100.0	0.0	100.0	0.0		

Table B.2: Results for $(10, 20, 30/\cdot/U/I)$ instances

			$\lfloor n/4 \rfloor$		$n/2$		$\lceil 3n/4 \rceil$	
n	d	S	$G1$	$G2$	$G1$	$G2$	$G1$	$G2$
40	25	02	40.0	6.7				
		03			33.3	0.0		
		05	25.0	11.1				
		06	25.0	0.0	33.3	0.0		
		10	11.1	11.1				
	50	01	25.0	25.0				
		02	20.0	0.0	25.0	0.0		
		03					12.5	0.0
		04	33.3	11.1				
		05			25.0	0.0		
		06			66.7	25.0		
		07	38.9	0.0	25.0	0.0		
		08	50.0	25.0			12.5	0.0
		10	20.0	0.0				
	75	01	4.2	0.0				
		02	50.0	20.0	11.1	0.0	12.5	0.0
		03			25.0	0.0		
		04	50.0	33.3			12.5	0.0
		05			33.3	0.0	12.5	0.0
		08	20.0	0.0	33.3	0.0		
		09	50.0	25.0	25.0	0.0		
		10	25.0	25.0				

Table B.3: Results for $(40/\cdot/W/I)$ instances

			$\lfloor n/4 \rfloor$		$n/2$		$\lceil 3n/4 \rceil$	
n	d	S	$G1$	$G2$	$G1$	$G2$	$G1$	$G2$
40	25	01			50.0	0.0		
		02	25.0	0.0	50.0	0.0		
		05	25.0	25.0				
		08	33.3	0.0				
	50	01			100.0	0.0		
		02	33.3	0.0				
		04	100.0	100.0	100.0	0.0		
		05	100.0	50.0				
		08	100.0	50.0	100.0	0.0		
		10	100.0	0.0	100.0	0.0		
	75	01	50.0	0.0	100.0	0.0		
		04	50.0	50.0	100.0	0.0		
		05	50.0	0.0	100.0	0.0		
		06	100.0	100.0				
		07			100.0	100.0		
		09			100.0	0.0		

Table B.4: Results for $(40/\cdot/U/I)$ instances

Appendix C

Gap Summary for $(\cdot/\cdot/\cdot/A)$ Instances from group P_2

			$\lfloor n/4 \rfloor$	$n/2$	$\lceil 3n/4 \rceil$
n	d	S	$G1$	$G1$	$G1$
10	75	09	2.9		
20	25	03	3.9		
		07	11.3		
		10	4.3		
	50	03	5.4	3.1	
		09	5.0		
		10	9.1		
	75	04	26.6		
30	25	02	6.9		
		04	25.9		
		10	17.9		
	50	01	14.5		
		05	12.5		
		06	18.2		
		09	23.3		
	75	01	10.0		
		03	3.4	22.8	
		06	13.7		

Table C.1: Results for $(10, 20, 30/\cdot/W/A)$ instances

			$\lfloor n/4 \rfloor$	$n/2$
n	d	S	$G1$	$G1$
10	25	02	8.0	
	50	03	5.3	
		06	5.9	
		07		10.0
		10	13.0	
	75	09	9.1	
20	25	02	10.0	
		09	13.3	
	50	04	11.1	
		06	22.2	
		09	16.7	
		10	22.2	
	75	02	16.7	
		03	14.3	
		06	33.3	
30	25	05	10.0	
		07	18.2	
		10	15.4	
	50	01	33.3	
		05	16.7	
		08	14.3	25.0
		09	11.1	
	75	03	33.3	
		04	33.3	22.8
		06	16.7	50.0
		10	33.3	

Table C.2: Results for $(10, 20, 30/\cdot/U/A)$ instances

			$\lfloor n/4 \rfloor$	$n/2$	$\lceil 3n/4 \rceil$
n	d	S	$G1$	$G1$	$G1$
40	25	01	11.0		
		04	1.5	27.0	
		06	15.0		
		09	13.2		
	50	01		22.8	
		02	14.7		
		04	7.9	25.7	
		05	11.8		
		07	21.8	30.6	
		10	21.6		
	75	01	3.8		
		03	25.8	18.5	
		04		1.8	
		05	8.2		12.5
		07	13.0		
		08	8.4	1.4	
		10	15.1		

Table C.3: Results for $(40/\cdot/W/A)$ instances

			$\lfloor n/4 \rfloor$	$n/2$
n	d	S	$G1$	$G1$
40	25	01	9.1	
		05	9.1	33.3
		08	12.5	
	50	01	16.7	
		03	14.3	50.0
		06	33.3	
		07	25.0	
		09	50.0	
	75	02	33.3	
		03	50.0	50.0
		05	33.3	
		07	25.0	
		09	25.0	1.4
		10	33.3	

Table C.4: Results for $(40/\cdot/U/A)$ instances

Appendix D

Gap Summary for $(\cdot/\cdot/\cdot/I)$ Instances from group P_2

			$\lfloor n/4 \rfloor$	$n/2$
n	d	S	$G1$	$G1$
10	75	09	2.9	
20	25	03	3.9	
		07	11.3	
		10	4.3	
	50	03	5.4	3.1
		09	5.0	
		10	9.1	
	75	04	26.6	
30	25	03	25.3	
		05	13.3	
		06	11.5	
		09	7.1	
	50	01	14.5	
		05	12.5	
		06	24.1	
		09	23.3	
	75	01	10.0	
		03	1.7	22.8
		06	13.7	
		07	6.7	

Table D.1: Results for $(10, 20, 30/\cdot/W/I)$ instances

			$\lfloor n/4 \rfloor$	$n/2$
n	d	S	$G1$	$G1$
10	25	02	8.0	
	50	03	5.3	
		06	11.8	
		07		10.0
		10	13.0	
	75	09	9.1	
20	25	02	20.0	
		09	13.3	
	50	04	11.1	
		05	14.3	
		06	22.2	
		09	16.7	
		10	22.2	25.0
	75	02	16.7	
		03	14.3	
		06	33.3	
		09	20.0	
30	25	05	20.0	
		07	18.2	
		10	15.4	
	50	01	33.3	
		02	33.3	
		05	16.7	
		08	14.3	
		09	11.1	
	75	03	33.3	
		06	25.0	
		07	20.0	
		10	16.7	33.3

Table D.2: Results for $(10, 20, 30/\cdot/U/I)$ instances

			$\lfloor n/4 \rfloor$	$n/2$	$\lceil 3n/4 \rceil$
n	d	S	$G1$	$G1$	$G1$
40	25	01	11.0		
		04	1.5	27.0	
		06	10.4		
		09	13.2		
	50	01		22.8	
		02	19.0		
		04	7.9	25.7	
		05	11.8		
		07	27.4	30.6	
		09		8.7	
		10	21.6		
	75	01	3.8		
		03	12.3	18.5	
		04		1.8	
		05	8.2		12.5
		07	13.0	1.1	
		08	8.4	1.4	
		10	15.1		

Table D.3: Results for $(40/\cdot/W/I)$ instances

			$\lfloor n/4 \rfloor$	$n/2$
n	d	S	$G1$	$G1$
40	25	01	9.1	
		05	9.1	33.3
		08	12.5	
	50	01	16.7	
		03	14.3	50.0
		06	16.7	
		08	20.0	
		09	50.0	
	75	02	66.7	
		03	50.0	50.0
		05	33.3	
		07	25.0	
		08	33.3	50.0
		10	33.3	

Table D.4: Results for $(40/\cdot/U/I)$ instances

Bibliography

[1] CHANDRASEKARAN, R. AND DAUGHETY, A. "Location on Tree Networks : p-Centre and N-Dispersion Problems", *Math. Oper. Res.*, Vol. 6 (1981), pp. 50-57.

[2] CHANDRASEKARAN, R. AND TAMIR, A. "An $O((n \log p)^2)$ Algorithm for the Continuous p-Center Problem on a Tree", *SIAM J Algebraic Discrete Methods*, Vol. 1 (1980), pp. 370-375.

[3] CRISTOFIDES, N. AND VIOLA, P. "The Optimum Location of Multi-Centers on a Graph", *Oper. Res. Quart.*, Vol. 22 (1971), pp. 145-154.

[4] DEARING, P.M. "Minimax Location Problems with Nonlinear Costs" *J. Res. Nat. Bur. Standards*, Vol. 82 (1977), pp. 65-72.

[5] DEARING, P.M. AND FRANCIS, R.L. "A Minimax Location Problem on a Network", *Transportation Sci.*" Vol. 8 (1974), pp. 333-343.

[6] DEARING, P.M., FRANCIS, R.L. AND LOWE, T.J. "Convex Location Problems on Tree Networks", *Oper. Res.*, Vol. 24 (1976), pp. 628-642.

[7] DYER, M.E. AND FRIEZE, A.M. "A Simple Heuristic for the p-Centre Problem", *Oper. Res. Letters*, Vol. 3, No. 6 (1985), pp.285-288.

[8] ERKUT, E., FRANCIS, R.L., LOWE, T.J. AND TAMIR, A. "Equivalent Mathematical Programming Formulations of Monotonic Tree Network Location Problems", Oper. Res., Vol. 37 (1989), pp. 447-461.

[9] FRANCIS, R.L. "A Note on a Nonlinear Minimax Location Problem in a Tree Network", *J. Res. Nat. Bur. Standards*, Vol. 82 (1977), pp.73-80.

[10] FRANCIS, R.L., McGinnis, L.F.Jr., White, J.A., *Facility Layout and Location, An Analytical Approach*, 2nd ed., Prentice Hall, New Jersey, 1992.

[11] GARFINKEL, R., NEEBE, A. AND RAO, M. "The m-Center Problem : Minimax Facility Location", *Mgmt. Sci.*, Vol. 23 (1977), pp.1133-1142.

[12] HAKIMI, S. L. "Optimal Locations of Swithcing Centers and the Absolute Centers and Medians of a Graph". *Oper. Res.*, Vol. 12 (1964), pp. 450-459.

[13] HAKIMI, S.L. "Optimum Distribution of Switching Centers in a Communication Network and Some Related Graph Theoretic Problems", *Oper. Res.*, Vol. 13 (1965), pp. 462-475.

[14] HAKIMI, S.L., SCHMEICHEL, E. F. AND PIERCE J.G. "On p-Centers in Networks", *Transportation Sci.* Vol. 12 (1978), pp. 1-15.

[15] HANDLER, G.Y. "Minimax Location of a Facility in an Undirected Tree Graph", *Transportation Sci.* Vol. 7 (1973), pp. 287-293.

[16] HANDLER, G.Y. "Finding Two-Centers of a Tree; the Continuous Case", *Transportation Sci.*, Vol. 12 (1978), pp. 93-106.

[17] HANDLER, G.Y. "Complexity and Efficiency in Minimax Network Location", Chapter 10 in *Combinatorial Optimization*, N. Christofides, A. Mingozzi, P. Toth and C. Sandi, John Wiley and Sons, New York, 1979.

[18] HEDETNIEMI, S.M., COCKAYNE, E.J. AND HEDETNIEMI, S.T. "Linear Algorithms for Finding the Jordan Center and Path Center of a Tree", *Transportation Sci.* Vol. 15 (1981), pp. 98-114.

[19] HOCHBAUM, D.S AND SHMOYS, D.B. "A Best Possible Heuristic for the k-Center Problem", Math. Oper. Res., Vol. 10, No. 2 (1985), pp. 180-184.

[20] HOOKER, J. "Solving Nonlinear Single-Facility Network Location Problems", *Oper. Res.*, Vol. 34 (1986), pp. 732-743.

[21] HOOKER, J. "Solving Nonlinear Multipple-Facility Network Location Problems", *Networks*, Vol. 19 (1989), pp. 117-133.

[22] HSU, W.L. AND NEMHAUSER, G.L. "Easy and Hard Bottleneck Location Problems", *Discrete Appl. Math.*, Vol. 1 (1979), pp. 209-215.

[23] JAEGER, M. AND GOLDBERG, J. "A Polynomial Algorithm for the p-CENTER PROBLEM on CACTUS GRAPHS", Woking Paper, Department of Systems and Industrial Engineering, University of Arizona, Tucson, Arizona 85721, 1990.

[24] JORDAN, C. "Sur les assemblages des lignes", *J. Reine Angew. Math.* Vol. 70 (1869), pp. 185-190.

[25] KARIV, O., AND HAKIMI, S. L. "An Alogrithmic approach to network location problems. I : The p-centers", *SIAM J. Appl. Math.* Vol. 37 (1979), pp. 513-538.

[26] MARTINICH, J.S. "A Vertex-Closing Approach to the p-Center Problem", *Nav. Res. Log.*, Vol. 35 (1988), pp. 185-201.

[27] MEGIDDO, N., TAMIR, A., ZEMEL, E. AND CHANDRASEKARAN, R. "An $O(n \log^2 n)$ Algorithm for the kth Longest Path in a Tree with Applications to Location Problems", *SIAM J. Comput.*, Vol. 10 (1981), pp. 328-337.

[28] MINIEKA, E. "The m-Center Problem", *SIAM Rev.* Vol. 12 (1970), pp.138-139.

[29] MINIEKA, E. "Conditional Centers and Medians of a Graph", *Networks* Vol. 10 (1980), pp.265-272.

[30] MINIEKA, E. "A Polynomial Time Algorithm for Finding the Absolute Center of a Network", *Networks* Vol. 11 (1981), pp. 351-355.

[31] PLESNÍK, J. "A Heuristic for the p-Center Problem in Graphs", *Discrete Appl. Math.*, Vol. 17 (1987), pp. 263-268.

[32] ROSENTHAL, A., HERSEY, M., PINO, J., COULTER, M. "A Generalized Algorithm for Centrality Problems on Trees", *Proc. Allerton Conf. on Communication. Control and Computing* 1978, pp. 616-625.

[33] SHIER, D.R. AND DEARING, P.M. "Optimal Locations for a Class of Non-linear, Single-Facility Location Problems on a Network", *Oper. Res.*, Vol. 31 (1983), pp.292-303.

[34] TAMIR, A. "A Finite Algorithm for the Continuous p-Center Location Problem on a Graph", *Math. Prog.*, Vol. 31 (1985), pp. 298-306.

[35] TANSEL, B.C., FRANCIS, R.L. AND LOWE, T.J. "Location on Networks : A Survey, Part I : The p-Center and p-Median Problems", *Mgmt. Sci.* Vol. 29 No. 4 (1983), pp. 482-497.

[36] TANSEL, B.C., FRANCIS, R.L. AND LOWE, T.J. "Duality : Covering and Constraining p-Center Problems on Trees", Chapter 8 in **Discrete Location Theory**, eds. P.B. Mirchandani and R.L. Francis, Wiley Interscience, 1990.

[37] TANSEL, B.C., FRANCIS, R.L., LOWE, T.J. AND CHEN, M.L. "Duality and Distance Constraints for the Non-linear p-Center Problem and Covering Problem on a Tree Network", *Oper. Res.*, Vol. 30 (1982), pp. 725-743.

[38] TOREGAS, C., SWAIN, R., REVILLE, C. AND BERGMAN, L. "The Location of Emergency Service Facilities", *Oper. Res.*, Vol. 19 (1971), pp. 1363-1373.