MINIMAX NETWORK LOCATION: THEORY AND ALGORITHMS

Gabriel Y. Handler

Massachusetts Institute of Technology
Flight Transportation Laboratory

Report FTL-R74-4

August 1974

MIT

DEPARTMENT
OF
AERONAUTICS
&
ASTRONAUTICS

FLIGHT TRANSPORTATION
LABORATORY
Cambridge, Mass. 02139

Massachusetts Institute of Technology

Flight Transportation Laboratory

Report FTL-R74-4

MINIMAX NETWORK LOCATION: THEORY AND ALGORITHMS

Gabriel Y. Handler

August 1974

ABSTRACT

For a given network let P and N denote the set of all points and the set of all nodes respectively. Let G and T denote a cyclic network and a tree network respectively and let m denote the number of centers available. The categorization scheme $\begin{Bmatrix} P \\ N \end{Bmatrix} / \begin{Bmatrix} P \\ N \end{Bmatrix} / m / \begin{Bmatrix} G \\ T \end{Bmatrix}$, where the first and second cells refer to the possible locations of centers and demand generating points respectively, provides for compact identification of a variety of minimax network location problems. This dissertation presents algorithms which efficiently solve all problems in this class--for example, P/P/m/G-for virtually any size of network. Moreover, tree problems can usually be solved manually.

Methodologically, the tree-based results are graph-theoretic while the general case, formulated in a mathematical programming framework, leads to a highly efficient strategy for a class of massive generalized set covering problems.

ACKNOWLEDGEMENTS

My deepest appreciation to Professor Simpson for his constant support and guidance. My interest in networks stems largely from his own catching enthusiasm for the subject. Sincere thanks to the other members of the thesis committee including Professor Little, who was most helpful in guiding my doctoral program, Professor Magnanti, whose friendship and inspired teaching are valued experiences and Professor Odoni, who kept posing the 'right questions' in an uncannily fruitful manner.

Several results were prompted by suggestions from Professor Kleitman while a key idea can be traced back to Professor Shapiro's 'relaxed approach to life.' Thanks are due also to Lin Dearing, Richard Francis and Alan Goldman for helpful comments and to all my colleagues at FTL and the O.R. Center.

Debbie Brooks demonstrated remarkable professionalism in typing the thesis from a collection of hieroglyphics. Her competence was invaluable during the past weeks.

This has been a joint 'effort' with Amnon, Gideon and Gila, in ascending order of the duration of their endeavors. I hope their patience will be rewarded.

My greatest debt is to my dear parents, to whom I dedicate this work.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

1.1 Problem Statement and Background

Consider the network optimization model

$$\min_{x \in G} \max_{y \in N} d(x,y) \qquad (1.1)$$

where G denotes the set of all points on an undirected graph with node set N, arc set A and shortest path distances d(x,y). (1.1) has been termed the 'absolute center' problem in graph-theoretic terminology. As a model of a physical phenomenon its most obvious application is to the location of emergency centers on a network. In (1.1) let x represent the location of a facility, and y the location of a demand point. For example, we may wish to locate a first aid center somewhere on a street network in a manner that minimizes the greatest distance to be traveled to that center from any node in the network.

Problem (1.1) is but one example of a variety of minimax location models. The following modifications in model assumptions encompass much of this variety:

i) Facilities may be situated on all points of the graph (nodes and interior points of the arcs) as in (1.1) or on the nodes alone, in which case the problem becomes

$$\min_{x \in N} \max_{y \in N} d(x,y). \qquad (1.2)$$

ii) In (1.1) and (1.2) demand is generated only at the nodes. A natural extension provides for demand generation at any point on the network. (1.1) and (1.2) would now be reformulated respectively

$$\min_{x \varepsilon G} \max_{y \varepsilon G} d(x,y), \qquad (1.3)$$

$$\min_{x \varepsilon N} \max_{y \varepsilon G} d(x,y). \qquad (1.4)$$

In addition to the obvious differences in the physical assumptions, models (1.1) through (1.4) differ significantly in computational complexity. We shall see later that these differences are not always intuitively apparent.

iii) The network may be a tree or it may contain cycles. Though the tree case is not significant in the physical setting, the distinction does present a useful hierarchy in developing a theoretical foundation and computational procedures for the general (cyclic) case. By replacing G with T in (1.1) through (1.4) we can refer to the appropriate tree problems.

iv) In general, m centers may be desirable, leading to a 'multi-center' problem. In the previous models we implicitly assumed m = 1. The following model generalizes (1.1) to include m centers:

$$\min_{X^m \varepsilon G^m} \max_{y \varepsilon N} d(y,X^m) \qquad (1.5)$$

where $d(y,X^m) \equiv \min_{x \varepsilon X^m} d(y,x)$ and G^m denotes the set of all m points, X^m, on G. Problems (1.2) through (1.4) can be similarly generalized.

A related class of problems, which we term the 'inverse' multi-center problem, seeks to determine the minimal number of facilities, m, so that all nodes (points) are within a specified distance, λ, from some facility. For example, the inverse of (1.5) is

$$\min_{X^m \varepsilon G^m} m$$

$$\text{s.t.} \quad \max_{y \varepsilon N} d(y, X^m) \leq \lambda. \tag{1.6}$$

As an example of an application of this type of model, beyond the location of emergency facilities, consider the determination of the number and location of bus-stops in a school-bus system where no pupil is to reside further than a specified distance from the nearest bus-stop.

v) A more general formulation allows for facility and demand locations on subsets of N or G. Letting $N' \subseteq N$, $G' \subseteq G$ denote these subsets, problem (1.1) would become

$$\min_{x \varepsilon N'} \max_{y \varepsilon G'} d(x,y). \tag{1.7}$$

This formulation enables us to model various locational restrictions imposed by physical constraints.

vi) So far we have assumed an implicit weighting $w(y,x) = 1$ associated with the shortest path from the demand center at y to the facility at x. The most important extension of this type cited in the literature is to attach weights $w(y,x) = w(y)$.

In this case (1.1) becomes

$$\min_{x\in G}\max_{y\in N} w(y)d(x,y). \qquad (1.8)$$

In general, such a formulation appears unrealistic in the context of minimax problems, as can be seen by disaggregating node y into w(y) distinct nodes each with unit weight. However, the following military scenario, communicated to the author by Richard Francis, does suggest (1.8) as a suitable model. Let w(y) represent a 'rate of attrition' at a defensive outpost given an enemy attack. Then the critical 'holding time' for effective defense of the installation until the arrival of reinforcements is a function of this rate.

vii) Up to this point we have assumed an absence of arc orientation. It may be desirable to include directed graphs allowing, for example, for variations in travel times due to gradients or traffic patterns in the physical network.

viii) The basic topological assumption has been of a given network of arcs and nodes, most suitable to an urban planning configuration. Other distance measures commonly used in facility location models are primarily Euclidean and rectilinear distances in a plane.

ix) The models so far are based upon a 'one-shot' or static scenario. This describes realistically a situation where the facility is either stationary, as for a hospital, or where frequency of service is very low. In other cases, for example police patrols, a dynamic model would be preferred for a more realistic approximation of the physical process.

x) It may often be desirable to incorporate stochastic elements to allow, for example, for congestion, climatic variations, variable demand rates and so on.

Notice that the dynamic and stochastic extensions referred to in the previous two categories raise questions concerning the minimax criterion. Indeed, in many situations other optimization criteria are required. Perhaps the most extensively treated problem is the 'median' problem, where total weighted costs are to be minimized. (1.8) would now be reformulated as

$$\min_{x \in G} \sum_{y \in N} d(x,y)w(y). \qquad (1.9)$$

<u>Scope of the Thesis</u>

In this thesis we shall consider minimax network location models of all varieties with respect to categories (i) through (iv) above, and we shall indicate that many of our results are easily extended to cover categories (v) through (vii) as well. Furthermore, a 'mixed' formulation of center (1.1) and median (1.9) problems will be briefly treated as an initial effort towards future research. Categories (viii) through (x) are beyond the scope of this thesis. Unless otherwise stated, we shall henceforth adhere to the assumptions implicit in the models of categories (i) through (iv).

In order to facilitate identification of the varieties of location models in this dissertation we propose the following shorthand categorization scheme,

Facility Location Set		Demand Location Set		No. of Centers/ Max. Distance		Network Type
$\begin{Bmatrix}N\\P\end{Bmatrix}$	/	$\begin{Bmatrix}N\\P\end{Bmatrix}$	/	$\begin{Bmatrix}m\\\lambda^{-1}\end{Bmatrix}$	/	$\begin{Bmatrix}T\\G\end{Bmatrix}$

where N,P denote the node and point sets respectively, T,G denote tree and cyclic graphs respectively and m,λ refer to the number of available facilities and the maximal distance respectively. The inverse is used in "λ^{-1}" to distinguish from m where numbers replace the symbols. For example, (1.1) would be identified as P/N/1/G and (1.6) as P/N/λ^{-1}/G.

1.2 Previous Work

The literature on location theory is extensive. A recent selective bibliography by Francis and Goldstein [9] runs to some 220 titles while a literature review by Elshafei and Haley [6] lists over 390 references. Yet there are but a handful of publications of direct relevance to minimax network location problems. For example, a new book by Francis and White [9], devoted entirely to location problems, treats only Euclidean and rectilinear distance measures. On the other hand, brief surveys of minimax network location problems together with a variety of applications are reported in Elshafei and Haley [6], Frank and Frisch [11], Odoni [34] and ReVelle et al. [38].

Problems N/N/1/G and P/N/1/G, or the 'vertex center' and 'absolute center' problems, were introduced and solved by Hakimi [20]. Similar concepts were introduced prior to that by Ore [35] though in a more restricted setting and without algorithmic solutions. N/N/1/G is naturally solved by finding all shortest paths in the network. To solve P/N/1/G, Hakimi proposed an arc by arc search of all shortest paths from the points along each arc of the network. The resulting procedure is cumbersome for manual operations though it presents no major difficulties for a computer solution. We shall point out in the next section that single center problems are essentially tractable. Goldman [18] presented a reduction algorithm for P/N/1/G attempting to overcome some of the original 'brute force' in Hakimi's procedure. Application of this algorithm does not, however, guarantee an optimal solution and Hakimi's algorithm may yet have to be applied. As a special case of his algorithm, Goldman derived an efficient algorithm for P/N/1/T.

The m-center problem P/N/m/G was posed by Hakimi [21] and subsequently solution algorithms were proposed by Minieka [33] and by Christofides and Viola [2]. An algorithm for N/N/m/G was given by Torregas et al. [44]. All of these algorithms involve repeated solutions of generalized set covering problems. As we shall point out later, these problems have yet to be solved satisfactorily for large scale problems, with the implication that the m-center problem, too, must still be considered an open problem for large scale networks. This is particularly true with respect to P/P/m/G and N/P/m/G which, to our knowledge, have not received any attention at all so far.

Since the inauguration of this research the author has published, at times with Odoni, a number of preliminary results pertaining to single center problems, for both cyclic and acyclic networks [24,25,26], to be discussed later. Very recently this area has exhibited a dynamic growth in interest. Since the conclusion of the major part of this research effort a number of related publications have appeared. A report by Dearing and Francis [3] on single center problems contains some conceptual results akin to the 'minimum diameter tree' introduced in [24] and discussed in Chapter III. Furthermore, they propose another algorithm for single centers of a tree, complementing the algorithms by Goldman [18] and the author [25]. Yet another algorithm for the latter problem has been recently reported by Halfin [22], based upon Goldman's procedure.

Garfinkel et al. [12] have reported an improved algorithm for P/N/m/G based upon Minieka's scheme [33]. Interestingly, their procedure depends upon a 'column elimination' observation which we also employ in Chapter IV in our proposed solution for two of the multi-center problems. However, their procedure remains limited by the previously mentioned covering problem bounds, so that truly large scale problems cannot be entertained, particularly for small values of m. The recent comprehensive review by Elshafei and Haley [6] points out a number of additional, related reports [15,37,39,42]. Unfortunately, none of these are well documented and could not be obtained by the author in the short time available. Furthermore, some of these references are of dubious value as, for example, Rosenthal and Smith [39]. While Elshafei and Haley indicate their algorithm is efficient for P/N/m/G, the author has come across an algorithm by Rosenthal and Smith [40] for the same problem which turned out to be invalid [24]. It is possible the two algorithms are one and the same. It appears that this field is particularly susceptible to erroneous suggestions. In reporting an interesting application of $N/N/\lambda^{-1}/G$ to the location of school-bus-stops, Gleason [16] mistakenly attributes a unimodular structure to the matrix of the resultant covering problem. In the course of this research effort the author has frustratingly often formulated 'remarkable' theorems which turned out, indeed, remarkably invalid. Such errors are due chiefly to the mind-twisting 'min-max-min' criterion and the cyclic, 'anti-convex' nature of general networks.

Limitations

In conclusion, existing state of the art is seriously deficient in the following items:

* Single centers of cyclic networks can be found only in a most cumbersome manner.

* Multi-center problems P/N/m/G and N/N/m/G can be practically solved for small to medium networks. Truly large scale problems cannot be entertained.

* Multi-center problems P/P/m/G and N/P/m/G cannot be solved at all.

Finally, we note that in a recent paper, T. C. Hu [28] identified the major discrete optimization problem areas for which breakthroughs are needed. One of these areas is multi-facility network location of which one variety is the minimax type. Another variety is the m-median problem (with objective function as in (1.9)) which has been treated, for example, by Marsten [31].

1.3 Overview of the Thesis

1.3.1 Objectives

In this thesis we aim to overcome all of the deficiencies in the state of the art as listed in the previous section. Furthermore, we intend to develop a complete graph-theoretic foundation for minimax location on a tree network together with more efficient algorithms for single and double centers of a tree.

1.3.2 Methodology

It is important to differentiate between single and multi-center problems with respect to computational complexity. To formalize this dichotomy we might invoke concepts from complexity theory. Karp [29] distinguishes problems possessing solution algorithms terminating in a number of steps bounded by a polynomial in the length of the input from problems with no such algorithm. We shall refer to the former as algebraic and the latter as exponential problems.

Single center problems are algebraic, with upper bound polynomial of low order. Solution techniques are generally graph theoretic and the intent of this thesis is to develop a graph theoretic foundation for this class of problems. Multi-center problems, in contrast, are apparently exponential in complexity and mathematical programming techniques are needed for their solution. As indicated, the methods which have been proposed so far involve repeated solution of the set covering problem

$$\begin{aligned} &\min e^t x \\ &\text{s.t. } Ax \geq e \\ &\qquad x_j \varepsilon \{0,1\} \quad \forall j \end{aligned} \tag{1.10}$$

where $a_{ij} \varepsilon \{0,1\} \quad \forall i,j$ and $e^t = (1,1,\ldots,1)$. We shall develop a problem-oriented math programming technique to solve (1.10) efficiently.

1.3.3 Outline

Chapter II develops a complete graph-theoretic foundation and highly efficient algorithms for the full variety of single and double center problems on a tree. An initial effort is also made in formulating and solving network location models with 'mixed' objective functions, combining minimax (center) and minisum (median) criteria. The preliminary treatment of trees provides results and insights for the general network case. The relative difficulty in the latter results from the absence of a basic convexity property dominating the tree case.

Chapter III is addressed to the variety of single center problems for a cyclic network. Some graph-theoretic results are obtained, generalizing the notions of Chapter II and an algorithmic strategy is developed, greatly improving existing strategies. Above all, important insights are gained for the major developments of the following chapter.

Chapter IV presents a highly efficient algorithmic strategy for the full variety of multi-center problems. A problem-oriented math programming strategy is developed which enables us to solve truly large scale problems most efficiently.

A final chapter summarizes major results, indicates straightforward extensions and suggests important and promising directions for future research.

1.3.4 Basic Notation and Assumptions

Standard mathematical notation is employed in the areas of set theory, analysis, number theory and linear algebra. In the latter, matrices are always upper case while vectors and scalars are always lower case. Matrix operations are assumed to conform without explicit statement of dimensions. Because of non-standard terminology in the graph-theoretic literature we need to define our notation more carefully in this area.

Let G(N,A) denote an undirected graph with node set N, arc set A and infinite point set G. As we shall be dealing exclusively with undirected graphs we shall refer synonymously to arcs/edges/links; vertices/nodes; graphs/networks. The following list of basic notation will be supplemented in subsequent sections as required.

d_{ij}	$\equiv$	length of arc $(ij) \varepsilon A$.
$d(x,y)$	$\equiv$	minimum distance between $x \varepsilon G$ and $y \varepsilon G$.
$\|\|d(x,y)\|\|$	$\equiv$	minimum distance matrix $\forall xy \varepsilon N \times N$.
$I(x)$	$\equiv$	degree of x; number of arcs or partial arcs incident at $x \varepsilon G$. By convention $I(x) = 2$ if $x \notin N$.

MPT(x) $\equiv$ minimum path tree rooted at $x \varepsilon G$.

MST $\equiv$ minimum spanning tree of G.

Unless otherwise stated, we shall always assume the following network properties:

i) $d_{ij} = d_{ji} \geq 0 \quad \forall (ij) \varepsilon A$.

ii) $I(x) \neq 2 \Rightarrow x \varepsilon N$.

Standard algorithms will be assumed for the basic tree search problems, specifically, Dijkstra [4] for MPT(x), Kruskal [30] for MST and any of the algorithms in Dreyfus [5] for $\|d(x,y)\|$.

Finally, we shall often use the following math programming terminology. Consider the finite-dimensional optimization problem

$$z(C) = \inf_{\xi \varepsilon C} f(\xi) \qquad (1.11)$$

where $f: R^n \to R$, $\xi \varepsilon R^n$ and $C \subseteq R^n$. A 'relaxation' of (1.11) is any new problem with $C' \supset C$ replacing C in (1.11) while a 'restriction' of (1.11) is defined by substituting $C'' \subset C$ for C in (1.11). If $z(C'') = z(C)$ we shall refer to C" as a 'dominant set' for C in (1.11).

CHAPTER II

CENTERS OF A TREE

2.1 Introduction

In this chapter we treat a variety of facility location problems based upon tree networks. The results obtained, in addition to their elegance and utility for this class of problems, provide insight, supportive theory and a natural first step for the study of the more difficult general case treated later. Furthermore, a number of contributions are made to the more abstract state of the art of graph theory.

The fundamental results are developed in section 2.2, which contains a discussion of all four single center problems $\begin{Bmatrix}P\\N\end{Bmatrix}/\begin{Bmatrix}P\\N\end{Bmatrix}/1/T$ as defined in Chapter I. These results are extended in 2.3 to the variety of double center problems $\begin{Bmatrix}P\\N\end{Bmatrix}/\begin{Bmatrix}P\\N\end{Bmatrix}/2/T$. Section 2.4, an important digression from the central theme of the thesis, presents an initial synthesis of median and center problems. Some results are established for mixed median/center formulations, which we dub 'medi-centers,' for single facilities on a tree network. Again, the results of section 2.2 play a central role.

Definitions and Notation

Let T(N,A) denote the infinite collection of points on the tree graph with node set N and undirected arc set A, For brevity we shall refer to T in place of the set T(N,A).

We shall employ the following notation to characterize elements of T:

E	$\equiv$	end point set, set of nodes with degree one.
AC(T)	$\equiv$	'absolute center' of T, solution to P/N/1/T, denoted also as AC in this chapter.
VC(T)	$\equiv$	'vertex center' of T, solution to N/N/1/T, denoted also as VC in this chapter.
D(T)	$\equiv$	'diametral path' in T, a longest path in T, also referred to as a 'diameter' of T.
d(x,y)	$\equiv$	distance between two points $x,y\varepsilon T$. We note that the concept of shortest paths is irrelevant in the context of tree graphs as one and only one path connects any two points.
p(x,y)	$\equiv$	path connecting $x,y\varepsilon T$.
$\ell(x)$	$\equiv$	$\max\limits_{y\varepsilon T} d(x,y)$.
$p(\ell(x))$	$\equiv$	maximum distance path from x.

For the proofs we shall also require the following notation:

x(t)	$\equiv$	point $x\varepsilon T$ such that $d(AC,x) = t$.
$\ell(x(t))$	$\equiv$	maximum distance from x(t).
$p(\ell(x(t)))$	$\equiv$	maximum distance path from x(t).

These are clearly not necessarily unique, though we shall show that $\ell\big(x(t)\big)$ is in fact a unique function of t. Finally, we shall let 'initial directions' from x denote directions from a point $x \varepsilon T$ to adjacent nodes. If x is an interior point of an arc there are two initial directions. If $x \varepsilon N$ there are as many directions as the number of arcs incident to x.

2.2 Single Centers

In this section we develop a graph-theoretic foundation for single centers on a tree network and provide simple and efficient algorithms for locating AC(T) and VC(T). Notice, however, that these are also optimal solutions to P/P/1/T and N/P/1/T respectively since a furthest point from any $x \in T$ is always an end point $e \in E \subseteq N$. Indeed, we might equally solve the problems P/E/1/T and N/E/1/T respectively. For the remainder of this section we shall refer to the problems as AC and VC respectively.

Existing Algorithms

Goldman [18] first proposed an efficient algorithm for the AC. Here we shall consider a simpler and more efficient procedure and we shall point out that the AC and VC are located simultaneously. The results of this section were completed early in the course of this research effort and have been documented by the author in the literature [25]. Because of their importance for subsequent sections these results are reproduced here in full. More recently, Halfin [22] has proposed a different algorithm, based upon and improving Goldman's procedure. Dearing and Francis [3] present yet another recent approach in the context of a generalized distance measure. Finally, a very early work by Ore [35], brought to the author's attention by Goldman [19], includes some graph-theoretic ideas closely related to the following discussion.

Algorithm for AC and VC

Informally:

Step 1: Choose any point x on tree and find furthest point away--say e_1 .

Step 2: Find furthest point away from e_1--say e_2 .

Step 3: Absolute center is at mid-point of path from e_1 to e_2 . Vertex center is at closest node to AC.

Formally:

Step 1: From any $x \varepsilon T$ find e_1 such that $d(x,e_1) = \ell(x)$.

Step 2: Find $\ell(e_1)$.

Step 3: AC is at $\ell(e_1)/2$ from e_1 along $p\big(\ell(e_1)\big)$. $\ell(AC) = \ell(e_1)/2$. VC is at closest adjacent node to AC (perhaps itself). $\ell(VC) = \ell(e_1)/2 + d(VC,AC)$.

Computational Considerations

The major computational effort involves finding the longest distance from each of two points (x and e_1). This requires a special case of the algorithm for the least cost tree rooted at a node for a general graph. Here, once a node is labeled with some cost $\pi(n)$ it is closed so that no minimizations are required. $\ell(x)$ is the greatest node price.

Step 3 involves tracing back along the path $p\big(\ell(e_1)\big)$ from the furthest point--e_2. This is achieved by using the node costs $\pi(n)$ from the computation of $\ell(e_1)$. Beginning at e_2 adjacent nodes are chosen only where node prices are lower than the current node price--there exists one and only one such adjacent node at each stage--until for the first time $\pi(n) \leqq \ell(e_1)/2$. AC is then $\ell(e_1)/2 - \pi(n)$ from node n along the last arc traversed.

Goldman's algorithm [18] progresses from 'tips' (nodes $e \varepsilon E$), successively deleting 'quills' (links with an adjacent tip) from the modified network until AC is located. This procedure requires in general many computations of $\ell(x)$. The current algorithm involves just two such computations.

Proof: The algorithm may be stated as the following theorem:

Theorem 2.1

i) $\ell(AC) = \ell(e)/2$ for any $e \varepsilon T$ such that there exists an $x \varepsilon T$ such that $\ell(x) = d(x,e)$.

ii) $d(AC,e) = \ell(e)/2$ and AC is located on $p\big(\ell(e)\big)$.

iii) VC is located at closest adjacent node to AC.

iv) $\ell(VC) = \ell(AC) + d(AC,VC)$.

To prove the theorem we require the following lemmas, the first two of which apply to general networks.

Lemma 2.1; $\ell(AC)$ is the distance from AC to points on T in at least two initial directions from AC.

Proof: If $\ell(AC)$ is the distance from AC to points in T in only one initial direction from AC then $\ell(AC)$ may be reduced by moving in that direction a short distance. Thus AC is not the absolute center and a contradiction is apparent. ||

Lemma 2.2: $\ell(x) \leqq \ell(y) + d(x,y)$ for all $x,y \varepsilon T$.

Proof: For all $x,y \varepsilon T$.

by definition, $d(z,y) \leqq \ell(y)$ for all $z \varepsilon T$;

by inspection, $d(z,x) \leqq d(z,y) + d(y,x)$ for all $z \varepsilon T$;

hence, $d(z,x) \leqq \ell(y) + d(y,x)$ for all $z \varepsilon T$;

by definition, $\ell(x) = \max_{z \varepsilon T} d(z,x)$;

hence, $\ell(x) \leqq \ell(y) + d(y,x)$. ||

The main observation needed for the proof of the theorem is contained in the following lemma:

Lemma 2.3:

$$\ell\big(x(t)\big) = \ell(AC) + t,$$

$$\text{or} \quad \ell(x) = \ell(AC) + d(x,AC).$$

Proof: Using Lemma 2.1, we may write

$$\ell\big(x(t)\big) \geqq \ell(AC) + t$$

since in moving from AC to $x(t)$, the longest distance in at least one of the initial directions from AC is increased by t. From Lemma 2.2, we may write

$$\ell\big(x(t)\big) \leqq \ell(AC) + t;$$

hence

$$\ell\big(x(t)\big) = \ell(AC) + t. \ ||$$

Lemma 2.4: AC is unique.

Proof: Let AC_1 and AC_2 be two absolute centers. From Lemma 2.3,

$$\ell(AC_1) = \ell(AC_2) + d(AC_1,AC_2);$$

hence $d(AC_1,AC_2) = 0$ and AC_1 and AC_2 are identical points. ||

The following lemma is a major result.

Lemma 2.5: AC must lie on every maximal path $p(\ell(x))$ from all $x \varepsilon T$.

Proof: From the proof of Lemma 2.3, AC lies on a longest path from any $x \varepsilon T$. Denote this path $p^*(\ell(x))$. If there exists a maximal path $p(\ell(x))$ that does not include AC then at the point of departure of this longest path from $p^*(\ell(x))$ the function $\ell(x(t))$ is at a local minimum. By assumption this point is not AC. However, from the functional form given in Lemma 2.3, only AC has the property of a local minimum. Since AC is unique (Lemma 2.4) a contradiction is apparent. Hence AC must lie on every maximal path from all $x \varepsilon T$. ||

We are now in a position to prove the theorem.

AC must lie on every $p(\ell(x))$ for any $x \varepsilon T$ (Lemma 2.5). Let e_1 be the furthest point from x on one of the paths $p(\ell(x))$. From Lemma 2.3, $\ell(x(t))$ diminishes along $p(\ell(x(t)))$, reaching a value $\ell(AC)$ at AC $(t = 0)$ and then increases, reaching a value $\ell(e_1)$ at e_1.

To relate $\ell(AC)$ and $\ell(e_1)$, we apply Lemma 2.3 twice in

$$\begin{aligned}\ell(e_1) &= \ell(AC) + d(AC,e_1)\\ &= \ell(AC) + d(x,e_1) - d(x,AC)\\ &= \ell(AC) + \ell(x) - d(x,AC)\\ &= \ell(AC) + \ell(AC) + d(x,AC) - d(x,AC)\\ &= 2\ell(AC).\end{aligned}$$

But AC also lies on $p\big(\ell(e_1)\big)$. Thus we have shown:

i) $\ell(AC) = \ell(e_1)/2$,

ii) $d(AC,e_1) = \ell(e_1)/2$ and AC is located on $p\big(\ell(e_1)\big)$.

To prove (iii) and (iv) we note from Lemma 2.3 that VC must satisfy

$$\text{iii)}\ d(VC,AC) = \min_{n\varepsilon N} d(n,AC).$$

Hence VC is located at the node closest to AC--or at AC, if $AC\varepsilon N$--and from Lemma 2.3,

$$\text{iv)}\ \ell(VC) = \ell(AC) + d(VC,AC).\ \|$$

Additional Observations on the Absolute Center of a Tree

The algorithm presented appears most efficient for practical purposes. Several related ideas are interesting to consider for theoretical reasons and special cases.

a. First we note that the algorithm presented sets out to find the minimax location by finding first a maximax location. $\ell(e_1) = d(e_1,e_2) = 2\big(\ell(AC)\big)$ is the longest distance between any two points. Hence e_1 and e_2 are maximax locations. Note that, unlike the minimax, these are not unique. We showed that the minimax is at the center of the longest path. The validity of the algorithm could have been proved using this observation together with Lemma 2.3. These observation provide as well justification for the terminology D(T), diameter of T, corresponding to a maximal path $p(e_1,e_2)$ with mid-point AC(T).

b. Further insight can be gained from a physical analogue of the algorithm. Consider a 'string solution' to the problem. Holding up the string network at any point find the lowest point--e_1. Holding now at e_1 find the lowest point again--e_2. Then AC is obtained by joining e_1 to e_2 and locating the mid-point of $p(e_1,e_2)$.

c. Using Lemmas 2.3 and 2.5 we could think of another algorithm for locating AC--namely, as the intersection of as many longest paths as are needed to locate a sufficiently small interval containing AC. We observe that all longest paths from AC intersect only at AC (Lemma 2.1). Thus all longest paths intersect only at AC.

d. The essential property given by Lemma 2.3 is a convexity property of $\ell(x)$ over the network. This enables us to locate a global minimum by locating a local minimum. In fact, $\ell\big(x(t)\big)$ is a particular linear monotonic function enabling us to locate the minimum very efficiently. It is interesting, however, to note that an alternative scheme is to 'home in' on the absolute minimum by beginning at any point x T and moving in the direction of decreasing $\ell(x)$, necessarily along $p(\ell(x))$. The following lemma validates this procedure.

Lemma 2.6: From any point $x \varepsilon T$, $\ell(x)$ either

i) decreases in one and only one initial direction --on the single path from x to AC--and increases in all other initial directions, or

ii) increases in all initial directions if x = AC.

Proof: Let x be the point x(t). t decreases only in the direction of AC, unless t = 0, in which case it increases in all directions. This is true since for a tree graph one and only one path connects any two points. The lemma then follows from the functional form given in Lemma 2.3. ||

e. Finally, we note that for the case of a general graph the property of convexity no longer holds as it is no longer true that one and only one path connects any two points. Thus we have the problematic task of minimizing a nonconvex function.

2.3 Double Centers

In this section we extend the fundamental results of section 2.2 to provide simple and efficient algorithms for the full variety of double center problems on a tree network. We shall emphasize P/P/2/T and modify the results for the remaining three cases.

Definition

Let D_m denote twice the optimal value of P/P/m/T.

2.3.1 Algorithm for P/P/2/T

Informally:

Step 1: Find the absolute single center as in section 2.2.

Step 2: Bisect the tree at the absolute center forming two sub-trees.

Step 3: An optimal pair of locations is given by the absolute single centers of the two sub-trees.

Formally:

Step 1: Find AC(T) and a diametral path $p(e_1,e_2)$ (section 2.2).

Step 2: Partition T into two sub-trees T_1, T_2 such that

$$T_1 \cup T_2 = T; \; T_1 \cap T_2 = AC(T); \; e_1 \varepsilon T_1; \; e_2 \varepsilon T_2, \tag{2.1}$$

resolving a choice arbitrarily if $I\big(AC(T)\big) > 2$.

Step 3: Using the algorithm of section 2.2 twice, locate

$$X^2 = \{AC(T_1), AC(T_2)\}. \tag{2.2}$$

Then X^2 is an optimal solution to P/P/2/T with optimal value $D_2/2$ given by

$$\frac{D_2}{2} = \max_{i\varepsilon\{1,2\}} \ell_i\big(AC(T_i)\big) \tag{2.3}$$

where

$$\ell_i(x) \equiv \max_{y\varepsilon T_i} d(x,y) \quad x\varepsilon T_i \ , \ i\varepsilon\{1,2\}. \tag{2.4}$$

Computational Discussion

The algorithm is based on a transformation of a difficult two dimensional optimization problem into two very simple ones (Theorem 2.2). Computational effort is only marginally greater than for the single center problem. Having solved for AC(T), we have available diametral points e_1 and e_2 where $p(e_1,e_2)$ coincides with D(T) (section 2.2). We shall show in Lemma 2.9 that $\{e_i\}_{i=1,2}$ are also diametral points of $\{T_i\}_{i=1,2}$ respectively. Since $\ell_1(e_1)$ can be found by inspection from previous computation of $\ell(e_1)$, only $\ell_2(e_2)$ need be computed so that an additional effort of about 25% is all that is required to locate an optimal pair of centers.

Proof: The algorithm may be stated and proved as the following theorem:

Theorem 2.2

The 2-dimensional problem P/P/2/T may be transformed into three 1-dimensional problems as follows:

$$D_2 \equiv 2 \overbrace{\min_{x_1 x_2 \varepsilon T\times T} \max_{y\varepsilon T} \min_{i\varepsilon\{1,2\}} d(y,x_i)}^{P/P/2/T} \qquad (2.5)$$

$$= 2 \max_{i\varepsilon\{1,2\}} \overbrace{\min_{x\varepsilon T_i} \max_{y\varepsilon T_i} d(y,x)}^{P_i/P_i/1/T_i} \qquad (2.6)$$

where $\{T_i\}_{i=1,2}$ are defined in (2.1) and $\{P_i\}_{i=1,2}$ are corresponding symbols. The third problem, implicit in (2.6), is to locate AC(T).

Proof: Assume T is partitioned into T_1, T_2, not necessarily as in (2.11), with points y in T_1, T_2 assigned to x_1, x_2 respectively. Let e_1 and e_2 be the diametral end points obtained in locating AC(T). To prove the theorem we require the following lemmas:

Lemma 2.7:

$$D_2 \leq 2 \max_{i\varepsilon\{1,2\}} \ell_i\big(AC(T_i)\big) \leq D_1 \qquad \forall\, T_i \subseteq T,\ i = 1,2\ .$$

Proof: i) R.H.S.: Directly from the theory of section 2.2. A maximal path in $T_i \subseteq T$ cannot exceed a maximal path in T.

ii) L.H.S.: (2.5) can be rewritten as follows:

$$D_2 = 2 \min_{T_1 T_2 \varepsilon T\times T} \max_{i\varepsilon\{1,2\}} \min_{x\varepsilon T_i} \max_{y\varepsilon T_i} d(y,x) \qquad (2.7)$$

$$\leq 2 \max_{i\varepsilon\{1,2\}} \min_{x\varepsilon T_i} \max_{y\varepsilon T_i} d(y,x) \qquad \forall\, T_i \subseteq T,\ i = 1,2$$

$$\equiv 2 \max_{i\varepsilon\{1,2\}} \ell_i\big(AC(T_i)\big)\ . \ \|$$

<u>Lemma 2.8</u>:

$$D_2 < D_1 \Longrightarrow \left(e_1 \varepsilon T_i \Longleftrightarrow e_2 \notin T_i\right) \quad \forall T_i \subseteq T .$$

<u>Proof</u>: Otherwise, from (2.7) and section 2.2,

$$\begin{aligned} D_2 &= d(e_1,e_2) \\ &= D_1 . \quad \| \end{aligned}$$

<u>Lemma 2.9</u>: $\{e_i\}_{i=1,2}$ are (single) diametral end points of $\{T_i\}_{i=1,2}$ where $\{T_i\}_{i=1,2}$ are defined in (2.1).

<u>Proof</u>: From section 2.2

$$\begin{aligned} d\big(AC(T),e_i\big) &= \ell\big(AC(T)\big) \\ &\geq \ell_i\big(AC(T)\big) \\ &\geq d\big(AC(T),e_i\big) ; \end{aligned}$$

therefore,

$$\ell_i\big(AC(T)\big) = d\big(AC(T),e_i\big)$$

and e_i is a diametral point of T_i , $i = 1,2$. ||

We are now in a position to prove the theorem. Clearly, $D_2 \leq D_1$. Suppose $D_2 = D_1$, then the theorem holds from Lemma 2.7.

Assume $D_2 < D_1$ and construct $\{T_i\}_{i=1,2}$ as in (2.1). Let $\{f_i\}_{i=1,2}$ denote complementary diametral points to $\{e_i\}_{i=1,2}$ in $\{T_i\}_{i=1,2}$ so that $\{AC(T_i)\}_{i=1,2}$ are obtained at the mid-points of $\{p(e_i,f_i)\}_{i=1,2}$ (Lemma 2.9 and section 2.2, Figure 2.1). Assume, without loss in generality,

$$\Delta \equiv 2 \max_{i\varepsilon\{1,2\}} \ell_i\big(AC(T_i)\big) = 2\ell_1\big(AC(T_1)\big)$$

$$= d(e_1,f_1). \qquad (2.8)$$

To show $\Delta = D_2$ rather than $\Delta \geq D_2$ as in Lemma 2.7, suppose

$$D_2 < \Delta . \qquad (2.9)$$

(2.8), (2.9) and Lemma 2.8 imply

$$f_1,e_2\varepsilon T_2^* \ ; \ e_1\varepsilon T_1^*$$

where $\{T_i^*\}_{i=1,2}$ are optimal sub-trees in (2.7). From (2.7) and section 2.2

$$D_2 \geq \min_{x\varepsilon T_2^*} \max_{y\varepsilon T_2^*} d(y,x)$$

$$\geq d(f_1,e_2)$$

$$= d\big(e_2,AC(T)\big) + d\big(AC(T),f_1\big)$$

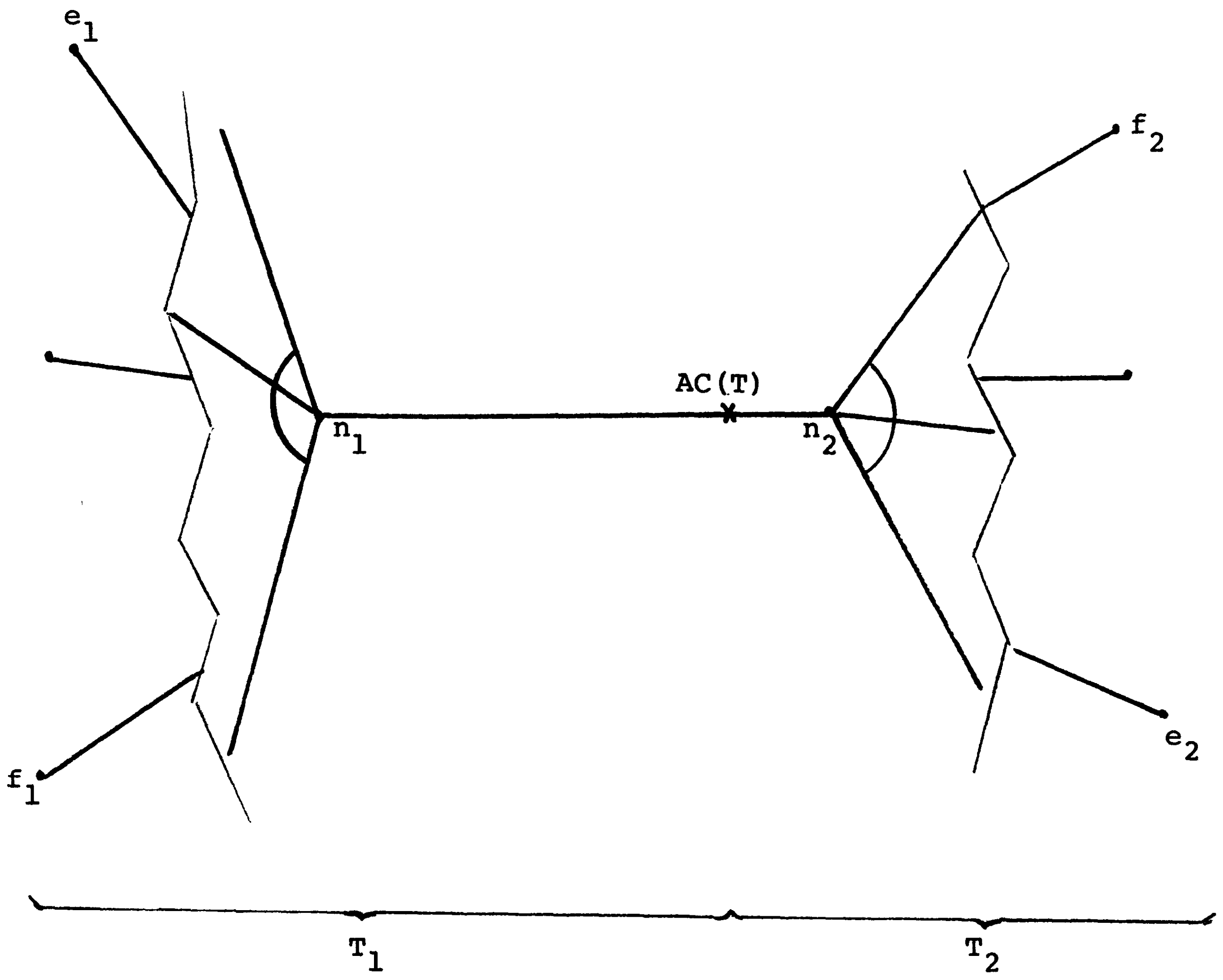


Figure 2.1: Partitioned Tree for P/P/2/T

$$= d\big(e_1, AC(T)\big) + d\big(AC(T), f_1\big)$$

$$\geq d(e_1, f_1)$$

$$= \Delta \text{ (from 2.8)}$$

contradicting (2.9). ||

Diameters and Double Centers of a Tree

Section 2.2 established the unity of the twin concepts of a diameter and single center of a tree. The following two theorems indicate an interesting relationship between diameters and double centers of a tree.

Theorem 2.3

Any diameter, D(T), and the intersection of all diameters is a dominant location set for optimal centers in P/P/2/T.

Proof: Directly from Theorem 2.2 and Lemma 2.5. ||

Theorem 2.4

$D_2 = D_1$ iff more than two 'diametral arcs' intersect at AC(T), where a 'diametral arc' is any arc $(ij) \varepsilon D(T)$.

Proof: i) Sufficiency

AC(T) is the unique intersection of all diameters of T and Theorem 2.3 implies no improvement is gained from a second center.

ii) Necessity

Consider the algorithm for P/P/2/T and assume, without loss in generality, $2\ell_1\big(AC(T_1)\big) = D_2 = D_1$. Then $AC(T_1)$

and $AC(T_2)$ are identical since both are unique (Lemma 2.4). Since $f_1 \neq e_2$ by construction, this implies $p(AC(T),e_1)$, $p(AC(T),e_2)$ and $p(AC(T),f_1)$ are three paths of length $\ell(AC(T))$ disjoint everywhere except at $AC(T)$. ||

2.3.2 Extensions

P/N/2/T

The algorithm and supporting theorem are very similar to those in section 2.3.1 for P/P/2/T, requiring minor modification.

Algorithm

As for P/P/2/T except where $AC(T)$ is an interior point on some arc $(n_1,n_2)\varepsilon A$ (see Figure 2.1) in which case Step 2 is modified to read

Step 2: Partition T into sub-trees T_1,T_2 by deleting the interior portion of arc (n_1,n_2).

The supporting theorem, identical to Theorem 2.2 with modifications corresponding to above changes, is proved as before. Lemmas 2.7 and 2.8 apply directly. To prove Lemma 2.9 note that

$$\ell_i(n_i) = d(n_i,e_i), \quad i = 1,2 .$$

The remainder of the proof of Theorem 2.2 is identical. Theorems 2.3 and 2.4 apply equally in this case.

N/N/2/T

When centers are restricted to the node set a complication arises which invalidates Theorem 2.2 and the associated algorithm. Specifically, (2.8) now becomes

$$\Delta \equiv 2 \max_{i\varepsilon\{1,2\}} \ell_i(VC(T_i))$$

$$\geq d(e_1,f_1)$$

from section 2.2, so that the final part of the proof of Theorem 2.2 no longer holds. Nevertheless, most of the fundamental properties, including Lemmas 2.7, 2.8 and 2.9, still apply and give rise to the following iterative algorithm.

<u>Algorithm</u>

Step 1: Find AC(T) using algorithm of section 2.2; proceed to Step 2.

Step 2: Partition T into two sub-trees T_1^o, T_2^o such that

$$e_1 \varepsilon T_1^o \ ; \ e_2 \varepsilon T_2^o$$

by

i) deleting from T the interior of arc (n_1,n_2), where AC(T) is an interior point on that arc, or

ii) if AC(T) is a node, by satisfying

$$T_1^o \cup T_2^o = T \ ; \ T_1^o \cap T_2^o = AC(T)$$

where $p(e_1,e_2)$ is a diametral path of T; proceed to Step 3.

Step 3: Find $VC(T_1^o)$, $VC(T_2^o)$ using algorithm of section 2.2. If $\ell_1^o(VC(T_1^o)) = \ell_2^o(VC(T_2^o))$ optimal solution is given by

$$X^2 = \{VC(T_1^o),\ VC(T_2^o)\}$$

where

$$\ell_i^k(x) \equiv \max_{y \varepsilon T_i^k} d(x,y) \quad x\varepsilon T_i^k \ ,\ i\varepsilon\{1,2\},\ k = 0,1,2,\ldots\ .$$

Otherwise, suppose T_1^o is a critical sub-tree, with diametral path $p(e_1,f_1^o)$; set k = 1 and proceed to Step 4.

Step 4: Repartition T into two sub-trees T_1^k, T_2^k such that

$$T_2^{k-1} \subset T_2^k \ ;\ T_1^k \subset T_1^{k-1}$$

$$f_1^{k-1} \varepsilon\ T_2^k \ ;\ f_1^{k-1} \notin T_1^k$$

by appending to T_2^{k-1} and excluding from T_1^{k-1} the node f_1^{k-1} and a minimal set of arcs to insure connectedness of the two sub-trees and deleting from T_1^{k-1} the interior portion of the arc joining the updated sub-trees; proceed to Step 5.

Step 5: Find $VC(T_1^k)$, $VC(T_2^k)$. If $\ell_1^k(VC(T_1^k)) \le \ell_2^k(VC(T_2^k))$ optimal value of N/N/2/T is given by

$$\min\{\ell_2^k(VC(T_2^k))\ ,\ell_1^{k-1}(VC(T_1^{k-1}))\}$$

and optimal solution is given by

$$X^2 = \{VC(T_1^j), VC(T_2^j)\}$$

where $j = k$ or $k - 1$ as determined in the minimization. Otherwise, let $p(e_1, f_1^k)$ denote a diametral path in T_1^k; set $k = k+1$ and go to Step 4.

Computational Remark

Computation of $VC(T_i^k)$ in Step 5, based upon the algorithm of section 2.2, is most easily performed by updating $VC(T_i^{k-1})$.

Proof of Algorithm: Briefly, corresponding forms of Lemmas 2.7, 2.8 and 2.9 apply here. We need to prove validity of the stopping condition in Step 5 for the two cases:

i) $D(T_2^k) = p(e_2, f_1^r)$ for some $r \in \{0,1,\ldots,k-1\}$,

ii) (i) does not hold and, necessarily,

$$D(T_2^k) = p(e_2, f_2^o).$$

In case (i) clearly no improvement is possible, but in case (ii) we need to demonstrate that transfering f_2^o to T_1^k cannot lead to any improvement.

Consider the points $e_1, e_2, f_1^o, f_2^o, n_1, n_2$ and $AC(T)$ as shown in Figure 2.1 after dropping superscripts and ignoring T_i labels in the figure.

Definition

Let pVq denote the relaxation of N/N/1/T,

$$pVq \equiv \min_{x\varepsilon N} \max\{d(x,p),d(x,q)\}.$$

We wish to demonstrate

$$e_1Vf_2 > e_2Vf_2 . \qquad (2.10)$$

By assumption (case (ii)) and section 2.2

$$d(n_2,f_2) > d(n_2,f_1). \qquad (2.11)$$

From section 2.2, any optimal location for e_1Vf_2 must lie on $p(e_1,n_2)$ so that, together with (2.11),

$$e_1Vf_2 \geq e_1Vf_1 \qquad (2.12)$$

and, by assumption (Step 3),

$$e_1Vf_1 > e_2Vf_2. \qquad (2.13)$$

(2.12) and (2.13) establish (2.10). ||

N/P/2/T

As for N/N/2/T except that interior of connecting link can no longer be neglected. This leads to some changes in the previous algorithm as specified below.

Step 2: Use (ii) always.

Step 4: Instead of deleting from T_1^{k-1} the interior portion of the connecting arc, let $T_1^k \cap T_2^k = x$, where x is connecting point.

Step 5: If $f_1^k = x$, require a one-dimensional search to avoid cycling.

Multi-Centers

The strategies for double centers do not, alas, extend to the case of three or more centers. However, combining such a generalized procedure with a simple perturbation scheme to obtain a local multi-center optimum does result in a most effective heuristic technique. Furthermore, such a procedure can be ideally combined with the optimal strategy of Chapter IV by furnishing quickly a good initial solution.

2.4 Medi-Centers

2.4.1 Preliminaries

In this brief digression from our main theme, we present an initial effort at combining the objective functions of the median and center problems in a single formulation, which we have dubbed the 'medi-center' problem. As indicated by Odoni [34], this is a crucial research area which has received practically no attention so far. We shall establish some results for the simplest cases, for single facilities on a tree network, hence the inclusion of this discussion in the present chapter.

We shall consider in turn the following two 'natural' medi-center formulations:

Constraint Approach

$$C(\lambda) = \min_{x\varepsilon T} \sum_{y\varepsilon N} d(x,y)w(y)$$

$$\text{s.t.} \quad d(x,y) \leq \lambda \quad \forall y\varepsilon N \tag{2.14}$$

Penalty Approach

$$P = \min_{x\varepsilon T} \sum_{y\varepsilon N} f\big(d(x,y)\big)w(y) \tag{2.15}$$

where $f(\cdot)$ is monotonic increasing and convex. (2.14) and (2.15) correspond to P/N/1/T and we shall see that the corresponding versions of N/N/1/T require no significant additional attention. In contrast, corresponding versions of N/P/1/T and P/P/1/T, which would require integration operators, are not treated here. We assume $w(y) \geq 0$ throughout this section.

A fundamental convexity property, analogous to the one established for $\ell(x(t))$ in Lemma 2.3, will prove useful.

<u>Definitions</u>

$$S(x) \equiv \sum_{y \varepsilon N} f\left(d(x,y)\right) w(y) \ , \quad x \varepsilon T \qquad (2.16)$$

$x^{ab}_{(t)} \equiv$ point on path $p(a,b)$, $a,b \varepsilon T$, distant t units from a.

<u>Lemma 2.10</u>: $S(x^{ab}_{(t)})$ is a convex function of $t \varepsilon [o,d(a,b)]$ $\forall ab \varepsilon T \times T$, if $f(\cdot)$ is monotonic increasing and convex.

<u>Proof</u>: $f\left(d\left(x^{ab}_{(t)},y\right)\right)$, $y \varepsilon N$ is a convex function of $t \varepsilon [o,d(a,b)]$ since $f(\cdot)$ is both convex and monotonic increasing. Then from basic convexity theory so is $\sum_{y \varepsilon N} f\left(d\left(x^{ab}_{(t)},y\right)\right) w(y)$. ||

<u>Corollary 2.1</u>: A local minimum of $S(x)$ is a global minimum satisfying (2.15), providing $f(\cdot)$ is convex and monotonic increasing.

<u>Proof</u>: Immediate from the lemma and the existence of a path joining any two points. ||

As a final preliminary, we shall find it convenient to reproduce here a very efficient and elegant algorithm for locating the median of a tree network or, equivalently, for solving (2.15) where $f(d) = d$. Let $W \equiv \sum_{y \varepsilon N} w(y)$.

<u>Algorithm for Median of a Tree (Goldman [17])</u>

Step 1: If N consists of a single node, stop; that node is an optimal solution.

Step 2: Search for a 'tip' $e \varepsilon E$ and associated 'quill' $(e\,i)$. If $w(e) \geq W/2$, stop; e is an optimal location.

Step 3: Modify T by deleting node e from N and link (ei) from A; also increment $w(i)$ by $w(e)$ and return to Step 1.

2.4.2 Constraint Approach

Definition

Let M^z denote a median (optimal solution to (2.15) with $f(d) = d$) which is closest to a given point $z \varepsilon T$. From the theory in [17], $M^z = z$ or $M^z \varepsilon N$.

A solution to the medi-center problem defined in (2.14) is immediately available upon location of AC and M^{AC}.

Theorem 2.5

An optimal solution to (2.14) is given by one or more of the following cases:

i) $\lambda < \ell(AC)$: no feasible solution,

ii) $\lambda = \ell(AC)$: unique solution at AC,

iii) $\lambda \geq \ell(AC) + d(M^{AC},AC)$: optimal solution at M^{AC},

iv) $\lambda \varepsilon \big(\ell(AC), \ell(AC) + d(M^{AC},AC)\big)$: unique optimal solution on the path $p(AC,M^{AC})$, $\big(\lambda - \ell(AC)\big)$ units from AC.

Proof: (i), (ii) and (iii) directly from Lemma 2.3. To show (iv), note that $f(d) = d$ satisfies the convexity conditions in Lemma 2.10. Lemmas 2.3 and 2.10 and the fact that M^z is a minimum establish the result. ||

Theorem 2.5 in effect defined an algorithm for solving (2.14), based upon solving first independently for AC and median. We can improve upon this by combining the two phases.

<u>Algorithm for (2.14)</u>

Step 1: Find AC using algorithm of section 2.2. If $\lambda < \ell(AC)$, stop; no feasible solution. If $\lambda = \ell(AC)$, stop; solution at AC. Otherwise, cut the tree at all points $(\lambda - \ell(AC))$ units from AC and amalgamate into the new end-points the sum of the node weights of those excluded nodes extending from the respective end-points.

Step 2: Find a median for the new tree using Goldman's algorithm (section 2.4.1), stop; this median is an optimal location.

<u>Additional Observations on Constrained Medi-Centers</u>

a. If a parametric solution is desired for (2.14) namely; $C(\lambda), \lambda \varepsilon [\ell(AC), \ell(AC) + d(AC, M^{AC})]$, it is preferable to use the strategy implied in Theorem 2.5. Such a solution is very easy to obtain as it is only necessary to consider the path $p(AC, M^{AC})$. Figure 2.2 illustrates the general form of such a solution.

b. An important property of pure median problems (including both general networks and multi-median problems) is the dominance of the node set as a location set for optimal solutions (e.g., see Hakimi [20]). This property is clearly lost in the constrained problem. Consequently, a different result will generally occur where (2.14) is replaced by the problem corre-

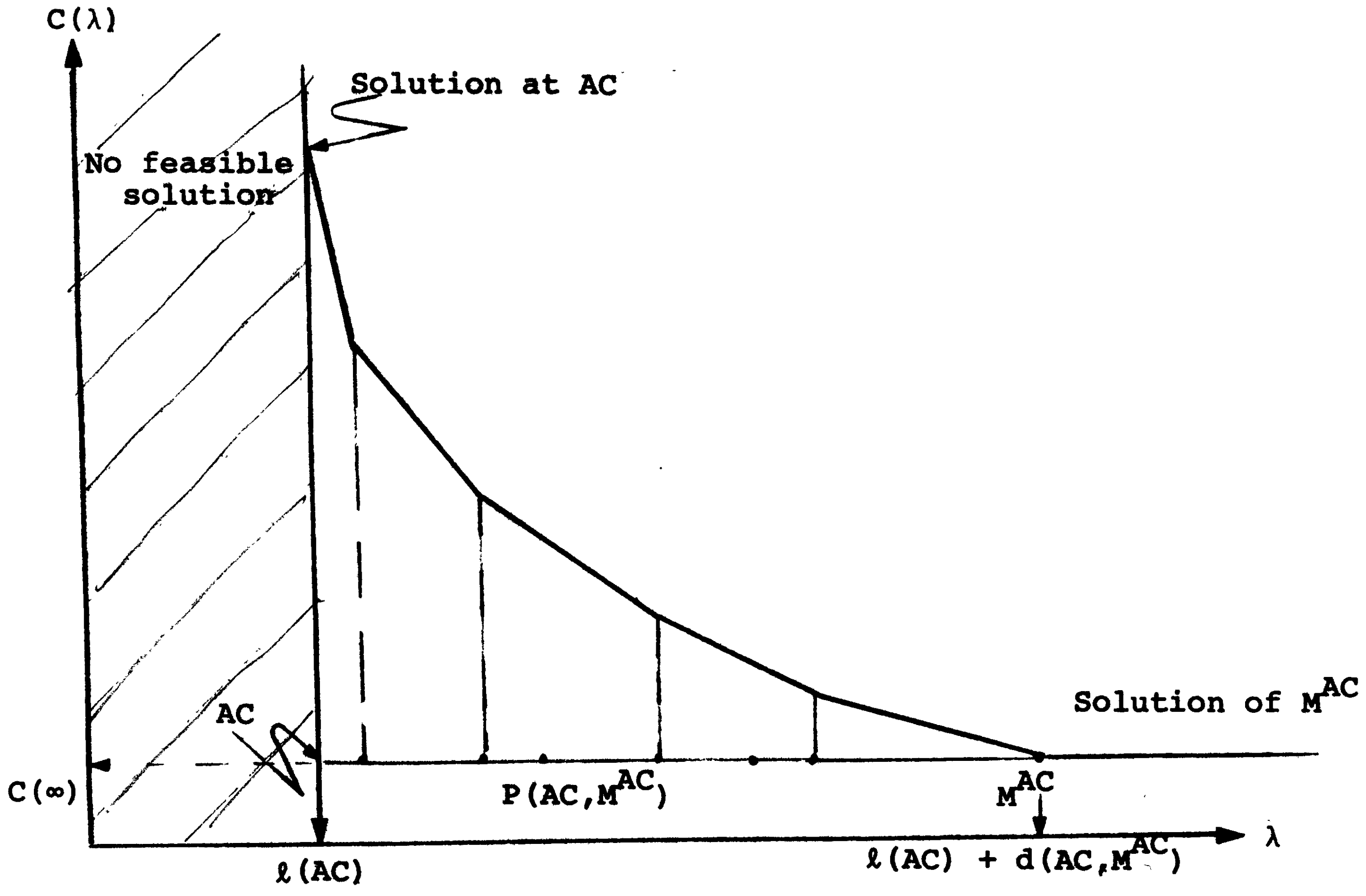


Figure 2.2: Parametric Solution to Constrained Medi-Center Problem

sponding to N/N/1/T. However, the modification is slight. Having solved as before the new solution is at the closest adjacent node in the direction of AC, providing λ is not violated.

c. Finally, notice the separate functions of node weights and arc lengths. While the former play no role in the center problem (in keeping with our original assumptions), the latter are irrelevant in the median algorithm.

2.4.3 Penalty Approach

Making use of Corollary 2.1, a solution to (2.15) can be obtained by computing values S(x) (2.16) while descending to a local and global optimal solution. A particularly simple algorithm results from the special case where $f(d) = e^d$ in (2.15).

Algorithm for (2.15) with Exponential Penalty

Step 1: Transform arc lengths d(i,j) as follows:

$$e^{d(i,j)} \rightarrow d^*(i,j) .$$

Step 2: Choose a node n and find S(n) as follows:

Search iteratively for a 'tip' $e \varepsilon E, e \neq n$ and associated 'quill' (ei). If n is the only node, $S(n) = w(n)$; otherwise, update w(i) according to

$$w(e)d^*(ei) + w(i) \rightarrow w(i),$$

delete e from N and (e,i) from A and continue the search.

Step 3: Restoring the original network, search for a direction of descent by locating a node k satisfying

$$k \varepsilon \Delta_n, \; S(k) < S(n)$$

where $\Delta_n \equiv$ set of nodes adjacent to n. (Note that S(k) is easily found from the information in Step 2.) Set $k \rightarrow n$ and repeat Step 3 until a local node minimum is established.

Step 4: Search all arcs (n,k) $k\varepsilon\Delta_n$ for an interior minimum as follows:

For any arc (n,k) find a point c on (n,k) such that

$$d^*(n,c) = \max\left\{\left(\frac{S_k}{S_n} d^*(n,k)\right)^{\frac{1}{2}}, 1\right\}$$

where $S_k = S(k)$ if (n,k) is cut from tree

$S_n = S(n)$ if (n,k) is cut from tree.

At most one interior minimum can exist, in which case the optimal location, c, is on the link (n,k), $d(n,c) = \frac{1}{2}\ell_n \frac{S_k}{S_n} + \frac{1}{2}d(n,k)$ units from n, with optimal value $P = S(c) = S_n d^*(n,c) + S_n \frac{d^*(n,k)}{d^*(n,c)}$. Otherwise, node n is the optimal location with optimal value $P = S(n)$.

Justification for Algorithm

i) Computation of function values S(x) in Step 2 follows immediately from the multiplicative form of (2.15) when

$f(d) = e^d$:

$$\min_{x \varepsilon T} \sum_{y \varepsilon N} \{w(y) \cdot \prod_{(a,b) \varepsilon p(y,x)} d^*(a,b)\}$$

where $p(y,x) = (y,i_1),(i_1,i_2),\ldots,(i_q,x),\ i_r \varepsilon N$.

ii) A minimum point on an arc (n,k) is found in Step 4 by minimizing the function

$$S(t) = S_n e^t + S_k e^{d(n,k) - t}$$

where t is a point on (n,k) t units from n.

Then

$$\frac{dS(t)}{dt} = S_n e^t - S_k e^{d(n,k) - t}$$

and at a minimum

$$e^t = \left(\frac{S_k}{S_n} e^{d(n,k)}\right)^{\frac{1}{2}} .$$

Since $t \geq 0$, a minimal point c satisfies

$$d^*(n,c) = \max\left\{\left(\frac{S_k}{S_n} d^*(n,k)\right)^{\frac{1}{2}}, 1\right\} .$$

iii) Finally, a local minimum is also a global minimum since e^d is convex and monotonic increasing and Corollary 2.1 applies.

The 'vertex medi-center,' corresponding to N/N/1/T, is located simultaneously. If the previous optimal location, c, is a node the solution is identical. Otherwise, if c is interior to an arc (n,k) the new optimal location is at n or k determined by

$$P = \min\{S(n), S(k)\}.$$

CHAPTER III

SINGLE CENTERS OF A GRAPH

3.1 Introduction

In this chapter we consider the class of problems $\begin{Bmatrix} P \\ N \end{Bmatrix} / \begin{Bmatrix} P \\ N \end{Bmatrix} /1/G$ where $G = G(N,A)$ is a cyclic network. P/N/1/G and N/N/1/G were originally formulated and solved by Hakimi [20] while P/P/1/G and N/P/1/G have not been treated in the literature, as far as the author is aware. We shall initially restrict the discussion to P/N/1/G reserving extensions to the remaining cases for a final section.

In section 2.2 we formulate a new tree search problem based upon the ideas of Chapter II and demonstrate its equivalence to P/N/1/G. Using problem restriction, this section also provides useful upper bounds for an algorithm for P/N/1/G which is presented in 2.3. Section 2.4 considers problem relaxation and provides useful lower bounds for the previous algorithm. An associated 'relaxation gap' leads naturally to the developments in Chapter IV. In a final section, extensions to the remaining three single center problems are described.

Definitions and Notation

Extending the terminology of Chapter II, section 2.1, assume any tree, T, is a spanning tree of G(N,A), namely;

$$T \equiv T(N,A^T),\ A^T \subseteq A,\ |A^T| = |N| - 1$$

and let Γ denote the set of all such trees of G.

In addition to AC(T), VC(T), D(T) and 'initial directions' from a point, define the terms:

AC	$\equiv$	'absolute center' of G, solution to P/N/1/G.
VC	$\equiv$	'vertex center' of G, solution to N/N/1/G.
$\ell(x)$	$\equiv$	$\max_{y \varepsilon N} d(x,y)$.
$\ell_T(x)$	$\equiv$	$\max_{y \varepsilon T} d(x,y)$.
D_1	$\equiv$	$2 \min_{x \varepsilon G} \ell(x)$, twice optimal value of P/N/1/G.
$D_1(T)$	$\equiv$	$2 \min_{x \varepsilon T} \ell(x)$, twice optimal value of P/N/1/T and length of D(T).

3.2 Minimum Diameter Trees

Definition

A 'minimum diameter tree' of G is a tree, MDT, satisfying

$$D_1(\mathrm{MDT}) = \min_{T\varepsilon\Gamma} D_1(T). \qquad (3.1)$$

The following theorem identifies an equivalence between P/N/1/G and the tree search problem (3.1).

Theorem 3.1

$D_1 = D_1(\mathrm{MDT})$ and AC(MDT) is also an AC.

Proof: P/N/1/T is a restriction of P/N/1/G since, by definition, $A^T \subseteq A$ and the node sets are identical. Hence

$$D_1(T) \geq D_1 \quad \forall\, T\varepsilon\Gamma \qquad (3.2)$$

establishing the 'weak' part of the theorem. To prove the equality, let $T^* \equiv \mathrm{MPT(AC)}$. By definition

$$\begin{aligned} D_1(T^*) &\leq 2\ell_{T*}(\mathrm{AC}) \\ &= D_1 \,. \end{aligned} \qquad (3.3)$$

(3.2) and (3.3) imply

$$D_1(T^*) = D_1 \qquad (3.4)$$

and (3.1), (3.2) and (3.4) establish the theorem and

$$\mathrm{MDT} = \mathrm{MPT(AC)}. \;\| \qquad (3.5)$$

Discussion

The theorem may be viewed, at least conceptually, as transforming P/N/1/G from a continuous optimization problem (on a graph) to a purely combinatorial, tree search problem. The diameter $D_1(T)$ of any tree $T \varepsilon \Gamma$ serves as an upper bound on D_1 and there exists at least one tree such that equality holds. Further, the class of trees to be searched may be restricted, again conceptually, to the set of trees $\{MPT(x): x \varepsilon G\} \subseteq \Gamma$ (of course, Γ is a finite set). We would like to find an efficient search procedure for MDT. To date, no such efficient procedure exists. A suggested approach by Rosenthal and Smith [40] , though couched in a different conceptual framework, would have implied a further restriction of the search to the set of, at most $|N|$, trees $\{MPT(i): i \varepsilon N\}$. Unfortunately, the suggested approach is invalid, as the author demonstrates in [24].

Whereas an efficient optimal approach based upon the preceding ideas appears unattainable, an approximate approach can be devised to obtain useful upper bounds. Such a procedure is discussed below. Furthermore, these bounds can be used in a hybrid algorithm as outlined in section 3.3.

Minimum Spanning Tree Approach - Upper Bounds

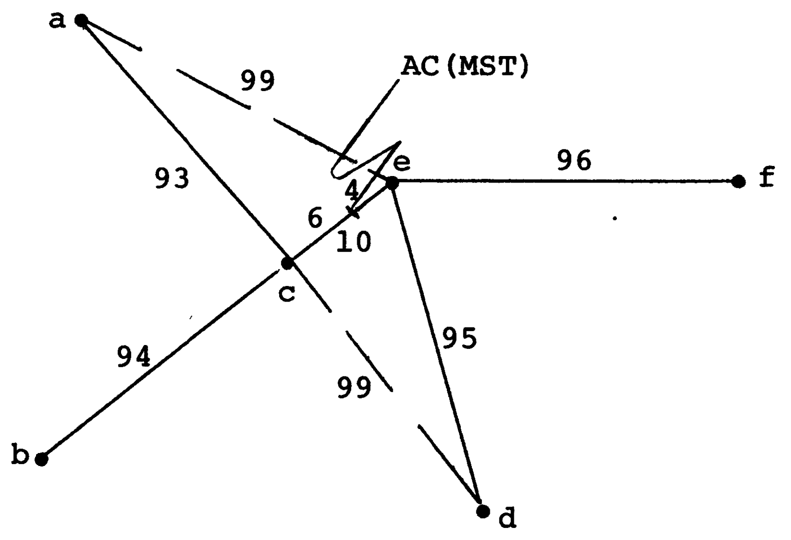
Consider first finding MST as a surrogate for MDT, with corresponding bound

$$D_1(MST) \geq D_1 \qquad (3.6)$$

necessarily true from (3.2). Note the ease of locating MST (e.g., Kruskal [30]) and D_1(MST) (Chapter II). Here is an example:

<u>Example 3.1</u>

G =

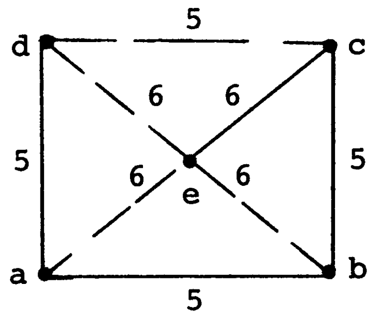


An MST is indicated by the continuous lines. The path b-c-e-f is a diametral path in MST, so that $D_1(\text{MST}) = 200$ and AC(MST) is at the mid-point of this path. Hence $D_1 \leq 200$ and AC(MST) is an initial approximation for AC. As we shall show later, AC(MST) is also optimal for G and $D_1 = 200$.

Though this strategy often locates an optimal solution, particularly for small examples, equality will not always hold in (3.6), for obvious reasons. The following example illustrates this.

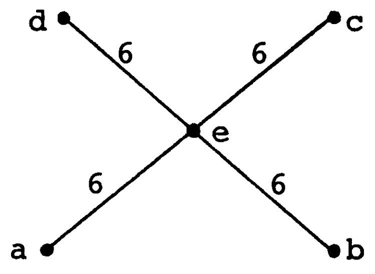
<u>Example 3.2</u>

G =



An MST is indicated by the continuous lines and $D_1(\text{MST}) = 21$. Now consider another sub-spanning tree

T =



Clearly, $D_1(T) = 12$ so that

$$D_1 \leq 12 < D_1(\text{MST}).$$

The last example motivates the following observation. Whereas MST is chosen with minimal total length as the sole criterion, MDT is also affected by the structure of the tree. Specifically, high degrees at nodes, $I(i)$, tend to reduce $D_1(T)$. This suggests that we might change Kruskal's algorithm to encourage higher degrees by modifying the entry criterion to

$$\min_{(ij)\varepsilon A} f\left(d_{ij}, I(i), I(j)\right)$$

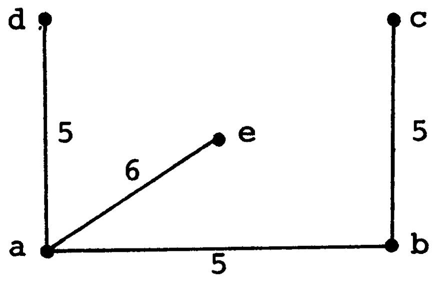
$$\text{s.t. no cycles being formed} \qquad (3.7)$$

where $I(i)$ and $I(j)$ are the new degrees resulting from adding (ij) to the current (partial) solution. For example, letting

$$f = d_{ij}/\left(I(i) + I(j)\right)$$

in (3.7) we find the following 'MST' for the previous example (here, it happens that 'MST' is still an MST),

'MST' =



so that D_1('MST') = 16 < D_1(MST), but still greater than D_1 .

This approach might be developed as a parametric procedure to obtain tighter upper bounds.

At this stage we note a conceptual similarity to Held and Karp's solution of the Traveling Salesman Problem [27]. Using a dual approach for the constrained MST problem, MSTs (more accurately, 'minimum spanning 1-trees') are successively generated by altering arc lengths in such a way that a minimal hamiltonian circuit is unaltered. In our case, modified MSTs are obtained by modifying the MST algorithm in order to solve a problem differing from MST only in objective function.

3.3 A 'Split and Bound' Algorithm for P/N/1/G

The difficulty in locating AC is due to the non-convex nature of the problem. In Chapter II, $\ell_T(x)$ was shown to be monotonically decreasing in the direction of AC(T) (Lemma 2.3). In the cyclic graph case, $\ell(x)$ obtains, in general, many local minima. Indeed, $\ell(x)$ can be multimodal even along one arc as illustrated in Figure 3.1.

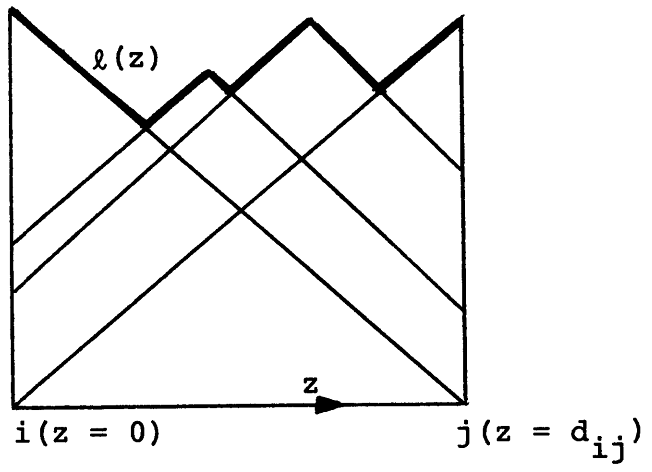


Figure 3.1: Form of $\ell(x)$

In the figure, $\ell(z)$ is the upper envelope of all d(n,z) for all $n \varepsilon N$, $z \varepsilon [0,d_{ij}]$ where z is a point on arc (ij) distant z units from i.

In the exhaustive search procedure suggested by Hakimi [20], the first step is the computation of $\|d(x,y)\|$. The second phase is a link by link search to identify local minima as 'candidate centers.' An overall minimum of these candidates is an AC. We shall formalize these concepts in our discussion of multi-center problems (Chapter IV). Here, suffice it to observe that the second phase of the search is an extremely cumbersome operation, particularly for large networks. The following ideas provide simple local and global lower bounds which are very useful in speeding up the search.

Theorem 3.2

$$\ell(z) \geq \left(\ell(i) + \ell(j) - d_{ij}\right)/2 \qquad \forall\, z\varepsilon[0,d_{ij}],(ij)\varepsilon A \tag{3.8}$$

where z is a point on arc (ij), z units from i.

Proof: Lemma 2.2 applies to cyclic graphs as well, so that

$$\ell(i) \leq \ell(z) + d(i,z) \tag{3.9}$$

and

$$\ell(j) \leq \ell(z) + d(j,z). \tag{3.10}$$

Summing (3.9) and (3.10) yields

$$\begin{aligned}\ell(i) + \ell(j) - 2\ell(z) &\leq d(i,z) + d(j,z)\\ &\leq d_{ij}\ . \ \|\end{aligned}$$

Geometrically, the lower bound for the arc is achieved if $\ell(z)$ is two-piece linear, sloping down (at 45°) from both ends.

Corollary 3.1:

$$D_1 \geq \min_{(ij)\varepsilon A} \{\ell(i) + \ell(j) - d_{ij}\} \tag{3.11}$$

Proof: Directly from (3.8). ||

Discussion

Theorem 3.2 may be employed to locate dominated links. Note that $\ell(VC)$, obtained by inspection from $||d(x,y)||$, serves as an initial upper bound for such pruning. Manual examples indicate that this simple procedure alone serves to reduce the number of arcs to be searched to a very small proportion of the original number. At this point, the remaining arcs may be searched by the exhaustive technique. However, a better approach is to use Theorem 3.2 iteratively. For instance, let (ij) be a candidate link and let z be the mid-point of this link. Then $\ell(z)$ is easily computed since

$$\ell(z) = \max_{y \varepsilon N}\left\{\min\{d(i,y),d(j,y)\}\right\} + \frac{d_{ij}}{2} \qquad (3.12)$$

and Theorem 3.2 can be reapplied to the newly created arcs (iz),(jz).

This procedure is combined with the upper bounds established in section 2.2 to yield an iterative algorithm for P/N/1/G. The following additional notation is adopted in the algorithm. $LB(i,j) \equiv \left(\ell(i) + \ell(j) - d_{ij}\right)/2$, UB and LB denote current upper and lower bounds respectively for D_1, S denotes the current set of candidate 'arcs' and $\delta(\geq 0)$ is some assigned optimality tolerance.

Algorithm

Step 0: Initialize

$A \rightarrow S$; $\frac{1}{2}D_1(MST) \rightarrow UB$; $VC \rightarrow z$.

Step 1: <u>Update Lower and Upper Bounds</u>

$\min\{UB, \ell(z)\} \to UB$

$\min_{(ij)\varepsilon S} LB(i,j) \to LB.$

Step 2: <u>Optimality</u>

If $LB \geq UB - \delta$, stop; $D_1 = UB$.

Step 3: <u>Eliminate Arcs</u>

If $LB(i,j) \geq UB - \delta$, $(ij) \to \bar{S}$, $\forall (ij)\varepsilon S$.

Step 4: <u>Split Arc</u>

Suppose $LB(i^*,j^*) = LB$ and z is mid-point of (i^*j^*), replace (i^*j^*) by (i^*z) and (j^*z) in S and go to Step 1.

<u>Remarks</u>: i) At some desired point the algorithm can continue with exhaustive search for remaining links. This would be advisable where few links remain and splitting continues to occur. Such a switch also guarantees finite convergence.

ii) For a true optimum set $\delta = 0$ in Steps 2 and 3.

iii) Initial LB(i,j) quantities require computation of $||d(x,y)||$. When an 'artificial node,' z, is added in Step 4, new quantities

$$d(z,y) = \min\{d(i,y), d(j,y)\} + \frac{d_{ij}}{2}$$

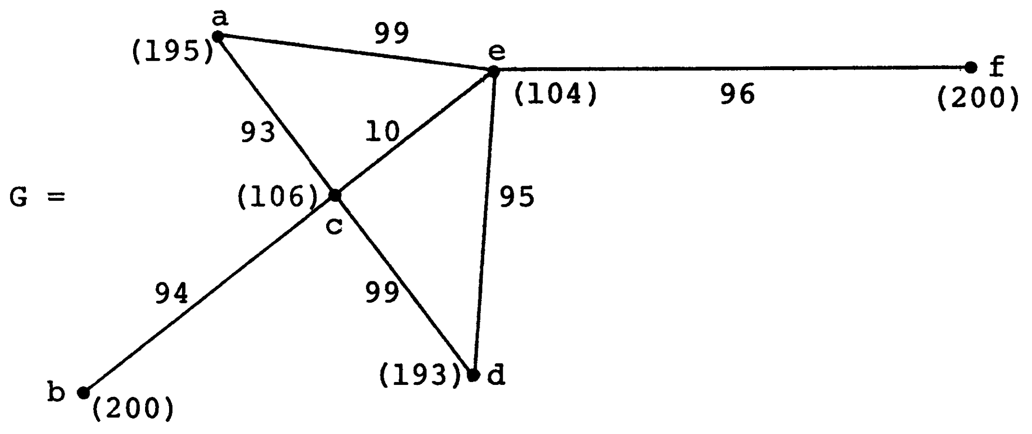
must be added to $||d(x,y)||$ for all current nodes, y, for computation of $\ell(z)$ (3.12).

iv) (UB - LB) is a non-increasing interval from one iteration to the next.

The following examples illustrate the algorithm.

Example 3.3

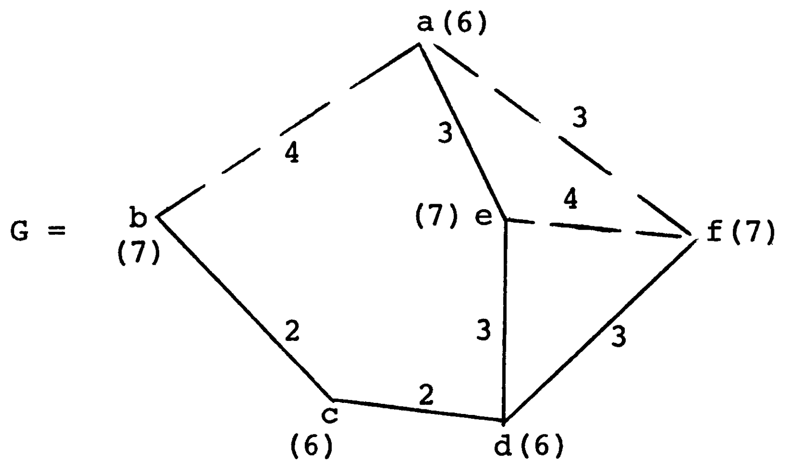
Consider again the graph in Example 3.1. The numbers in



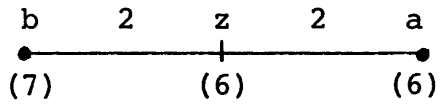
parentheses indicate $\ell(x)$. We found $D_1(MST) = 200$ and $\ell(VC) = \ell(e) = 104$, $UB = \min\{104, 100\} = 100$, $LB = 100$. Therefore, $D_1 = 200$ and AC(MST) is also AC, or, equivalently, MST is also MDT.

Example 3.4

Consider the example given by Hakimi [20].



Again, continuous lines make up an MST and numbers in parentheses represent $\ell(x)$. We find $D_1(\text{MST}) = 10$, $\ell(\text{VC}) = \ell(d) = 6$, $\text{UB} = \min\{6,5\} = 5$ and $\text{LB} = \text{LB}(a,b) = 4\frac{1}{2}$. Eliminating dominated arcs leaves (ab) the single remaining candidate which, on splitting, is replaced by



and we obtain UB = 5, LB = 5 so that $D_1 = 5$ and AC(MST) is also AC. Again, MST is MDT.

3.4 A Relaxation Approach for P/N/1/G

Theorem 3.3

$$\max_{xy \in N \times N} d(x,y) \leq D_1 \qquad (3.13)$$

or, equivalently,

$$\max_{x \in N} \ell(x) \leq 2 \min_{x \in G} \ell(x).$$

Proof: For any pair of nodes x,y

$$\begin{aligned} d(x,y) &\leq d(x,AC) + d(AC,y) \\ &\leq 2\ell(AC) \\ &= D_1 \,. \; \| \qquad (3.14) \end{aligned}$$

Discussion

The lower bound established by Theorem 3.3 is the result of a relaxation of P/N/1/G formed by eliminating from N all nodes except some pair x,y. In fact, $\frac{1}{2}d(x,y)$ is the optimal value of P/{x,y}/1/G. In contrast, the upper bound $D_1(T)$ (3.2) was the result of a problem restriction induced by eliminating a set of arcs. Notice that while $D_1 = D_1(MDT)$ from Theorem 3.1, here a 'relaxation gap' may well exist namely, a strict inequality in (3.13). In Chapter IV we develop a relaxation technique for multi-center problems which, when specialized to the single center problem, can be viewed as a technique for eliminating this gap. We shall return to this topic in section 4.5.

Notice that the bound in (3.13) can be incorporated in the algorithm of the previous section. While that algorithm and this bound require prior computation of $\|d(x,y)\|$ we can suggest a simple 'ascent' algorithm which will approximate this bound from below without requiring such preliminary computation. The procedure is especially useful in conjunction with the ideas of Chapter IV. Furthermore, the procedure is a direct generalization of the algorithm of section 2.2 for $D_1(T)$.

<u>Ascent Algorithm for Lower Bound</u>

Step 0: Choose any node $x \varepsilon N$; set $0 \rightarrow LB$.

Step 1: Locate a furthest node y such that $d(x,y) = \ell(x)$ by finding $MPT(x)$.

Step 2: If $\ell(x) = LB$, stop; LB is 'best' lower bound. Otherwise, set $\ell(x) \rightarrow LB$; $y \rightarrow x$ and go to Step 1.

<u>Remarks</u>: i) Successive values of LB are monotonically increasing since $\ell(x) = d(x,y) \leq \ell(y)$ and where equality holds the algorithm terminates.

ii) The final value of LB may be lower than $\max\limits_{x \varepsilon N} \ell(x)$, in which case a 'local optimum' has been obtained. This increases the size of the potential relaxation gap.

iii) In the case of a tree network, no relaxation gap exists and the above procedure locates a global optimum after exactly two iterations.

3.5 Extensions

N/N/1/G

According to existing state of the art VC is found far more easily than AC since the former is found from $||d(x,y)||$ by inspection. Consequently, the 'split and bound' algorithm of section 3.3 is irrelevant here. The upper and lower bounds of sections 3.2 and 3.4 are easily extended to this case by considering the restriction N/N/1/T and the relaxation N/$\{x,y\}$/1/G respectively. In Chapter IV we demonstrate how a generalized relaxation strategy can eliminate the need for the complete $||d(x,y)||$ matrix. Using a New York State network with 30 locations, an example is given in section 4.4.2. 7 MPTs are required to locate VC. Finally, we shall see that an interesting consequence of this approach is a marked change in the relative ease of solving P/N/1/G compared to N/N/1/G with VC at times more difficult to locate than AC.

P/P/1/G

The upper bound and tree search equivalence of section 3.2 apply here with the proviso that 'spanning trees' of G now span all points in G, requiring a minor extension in the MPT algorithm.

While Hakimi's exhaustive search [20] cannot be applied here because of the infinite set of demand locations, the 'split and branch' algorithm of section 3.3 is ideally suited to this problem. Redefining $\ell(x) \equiv \max_{y\varepsilon G} d(x,y)$, it is straightforward to obtain $\ell(x)$ for all $x\varepsilon N$ from $||d(x,y)||$. The algorithm is then applied as before. Notice, however, that finite convergence is not guaranteed in this case.

Most rewarding in this problem is the relaxation approach initiated in section 3.4. The ascent algorithm, with points replacing nodes to reflect the redefined $\ell(x)$, provides an initial lower bound, while the generalized relaxation strategy of Chapter IV provides an ideal algorithm for closing the relaxation gap. Finally, we note that here too finite convergence is not guaranteed, as is demonstrated in an example in section 4.4.1.

N/P/1/G

This case is similar to N/N/1/G, except that $\ell(x)$ is defined as in P/P/1/G above with corresponding changes in the MPT algorithm.

CHAPTER IV

MULTI-CENTERS OF A GRAPH

4.1 Introduction

In this chapter, we propose an efficient methodology for solving the complete variety of minimax multi-center network location problems for large scale networks. In the terminology of Chapter I we shall address the problems P/N/m/G, P/P/m/G, N/N/m/G, N/P/m/G for $m \geq 1$, as well as the associated 'inverse' problems where m is replaced by λ^{-1} .

For the special case $m = 1$, the proposed strategy provides efficient algorithms, particularly in conjunction with the ideas of Chapter III. We shall return to this issue in 4.5. For the remainder of this section we concentrate upon the more difficult case, $m > 1$.

Previous Work

Algorithms for P/N/m/G have been proposed by Christofides and Viola [2] and by Minieka [33]. For N/N/m/G the algorithm by Torregas et al. [44] can be used. Essentially similar algorithms solve the 'inverse' problems. All of these algorithms operate on the same principle, solving in succession a series of Set Covering Problems (CP). To the best of this author's knowledge no procedure has been suggested for P/P/m/G or N/P/m/G in the literature.

Our research has been prompted by two, related, deficiencies inherent in the state of the art:

a. The number of nodes, $|N|$, is critical. Because of the

'CP bottleneck', discussed at length in 4.2, problems with $|N| > 100$ are costly to solve and with $|N| > 1000$ probably entirely intractable.

b. P/P/m/G and N/P/m/G cannot be directly solved by any of the existing approaches. An approximate solution strategy is to discretize the set of 'demand points' by introducing artificial nodes at 'sufficiently close' intervals. By greatly increasing the size of $|N|$ this approach quickly leads to enormous computational requirements.

Proposed Algorithm - A Relaxation Approach

By developing a problem oriented relaxation technique which seems especially appropriate for this class of optimization problems we have been able to overcome the afforementioned deficiencies, specifically:

a. The size of problems of type P/N/m/G which can be reasonably solved may be increased by orders of magnitude.

b. Problems of type N/N/m/G can be solved very much more efficiently than hitherto, though not quite as dramatically as P/N/m/G.

c. Problems of type P/P/m/G and N/P/m/G can be solved directly and efficiently for large scale networks.

d. Virtually any size of problem of type P/N/m/T and P/P/m/T can be solved manually.

Outline

For expository purposes we shall initially restrict our discussion to P/N/m/G. This problem is seen as generic and extensions to the remaining cases are reserved for a later section. We have chosen as our point of departure Minieka's algorithm [33] although the general strategy can be equally applied to the other existing schemes.

Section 4.2 is devoted to the CP framework. After a brief description of Minieka's procedure [33] a greatly improved version of it is developed as a crucial preliminary to the proposed relaxation strategy. This section includes as well a brief summary of the state of the art of CPs in general and with respect to multi-center problems in particular. The relaxation technique, fully developed in 4.3, can be viewed as a problem-oriented solution technique for the large scale CPs encountered in multi-center problems. In 4.4 we extend the relaxation strategy to P/P/m/G, N/N/m/G, N/P/m/G and the 'inverse' problems. In a final section, 4.5, single center problems, treated initially in Chapters II and III, are reconsidered in the light of the proposed scheme.

4.2 Set Covering Strategy for P/N/m/G

Definition

$<x,c,y>$, denoting a 'local center' at a point $c \varepsilon G$ with respect to a pair of nodes xy, is said to exist iff

i) $d(x,c) = d(y,c)$

ii) $\left| \delta \varepsilon \Delta_c : \lim_{\varepsilon_\delta \to 0} \{d(z,\varepsilon_\delta) - d(z,c)\} < 0, \forall z \varepsilon \{x,y\} \right| = \emptyset$

where Δ_c is the set of nodes adjacent to c and ε_δ is a point distant ε_δ from c in the direction of the node $\delta \varepsilon \Delta_c$.

Remark: Note that $<x,c,x>$ $c \neq x$, never exists while $<x,x,x>$ always exists. We shall refer to the latter as the 'null center' at x.

The existence of $<x,c,y>$ identifies c as a local minimum of the function $\max\{d(x,z),d(y,z)\}, z \varepsilon G$. A more complete discussion appears in section 4.3.3 and in Appendix A.

Definition

$$C \equiv \{c \varepsilon G : <x,c,y> \text{ exists for some } xy \varepsilon N \times N\} \qquad (4.1)$$

Notice that C is the collection of intersection points, as depicted for an arc (ij) in Figure 3.1, over all $(ij) \varepsilon A$. While in P/N/1/G it was sufficient to consider such points on the upper envelope, $\ell(\cdot)$, alone, here it is necessary, and sufficient, to consider all intersection points, as we shall see next.

4.2.1 Minieka's Approach [33]

Minieka employs a different, but equivalent, definition for C (4.1). Our own, more explicit, definition is preferred for subsequent developments. The following theorem, which can be viewed as a generalization of Lemma 2.1, enables us to refer to 'local centers' as 'candidate centers.'

Theorem 4.1 (Minieka)

The finite set C defines a dominant set for the location of centers in P/N/m/G which can now be reformulated as

$$\min_{X^m \varepsilon C^m} \max_{y \varepsilon N} d(y, X^m) \tag{4.2}$$

where C^m denotes the set of all sets of m points, X^m, in C.

Proof: (Sketch): The reasoning is intuitively clear. A center at $x \not\in C$ can be perturbed in the direction of the furthest node being served by it, without loss of optimality. ||

A solution strategy for P/N/m/G can be devised by reformulating (4.2). Form the matrix $F = \| f_{ij} \|$, with $|N|$ rows and $|C|$ columns, where f_{ij} denotes twice the shortest distance from node i to the j^{th} candidate center. Then (4.2) can be reformulated as

$$\min_{J^m \varepsilon S^m} \max_{i \varepsilon N} \min_{j \varepsilon J^m} f_{ij} \tag{4.3}$$

where S^m is the set of all sets of m columns, J^m, of F.

Algorithm for P/N/m/G

The following algorithm solves (4.3), and hence P/N/m/G, in a finite number of operations.

Step 0: Choose an arbitrary initial solution, J^m .

Step 1: i) Let $d = \max_{i \in N} \min_{j \in J^m} f_{ij}$.

ii) Update the matrix $A = \|a_{ij}\|$ where

$$a_{ij} = \begin{cases} 0 & \text{if } f_{ij} \geq d \\ 1 & \text{otherwise} \end{cases} \qquad \begin{array}{l} i = 1,2,\ldots,|N| \\ j = 1,2,\ldots,|C| \end{array} \quad .$$

Step 2: Solve the CP

$$z = \min e^t x$$

$$\text{s.t.} \quad Ax \geq e \tag{4.4}$$

$$x_j \varepsilon \{0,1\} \quad \forall_j$$

where $e^t = (1,1,. . .,1)$.

Step 3: If $z > m$, stop; $D_m = d$ is the value of the optimal solution. Otherwise, an improved solution has been obtained. Update J^m and go to Step 1.

Remark: A feasible solution to (4.4) always exists because of the 'null set' of node centers, $N \subseteq C$.

4.2.2 Column Elimination

For large $|N|$ the number of candidate centers, $|C|$, becomes extremely large. The following theorem has the effect of greatly reducing the number of columns in successive iterations of (4.4).

Definition

For any candidate center $c \varepsilon C$, let d_c denote the 'diametral span'

$$d_c \equiv 2d(x,c) = 2d(y,c) \qquad (4.5)$$

where x,y are the generating nodes for c as defined in (4.1).

Theorem 4.2

Any candidate center $c \varepsilon C$ can be considered as 'covering' all of the nodes in the set

$$N_c \equiv \{n \varepsilon N: d(c,n) \leq \frac{d_c}{2}\} \qquad (4.6)$$

and only those nodes, without loss of optimality.

Proof: The justification for this theorem is implicit in the proof of Theorem 4.1. The generating pair of nodes x,y are, by definition, furthest away of the nodes served by c. ||

Corollary 4.1: In Minieka's algorithm, if d is the value of the incumbent solution for m centers, then all candidate centers (columns) in the set

$$K \equiv \{c \varepsilon C : d_c \geq d\} \qquad (4.7)$$

can be eliminated without loss of optimality.

Proof: For any $c \varepsilon K$ the generating nodes are no longer 'covered' by it (Step 1 of the algorithm). It follows from Theorem 4.2 that c is no longer a candidate center. ||

Corollary 4.2: In Minieka's algorithm, for any column j corresponding to a candidate center $c \varepsilon C$, the initialization procedure

$$a_{ij} = 0 \text{ if } f_{ij} > d_j ,$$

where $d_j \equiv d_c$, can be made without loss of optimality.

Proof: Directly from Theorem 4.2. ||

Revised Algorithm for P/N/m/G

Corollaries 4.1 and 4.2 lead directly to the following revised version of the algorithm of section 4.2.1 .

Step 0: i) Choose an arbitrary initial solution, J^m .

ii) Set up matrix $A = ||a_{ij}||$ where

$$a_{ij} = \begin{cases} 0 & \text{if } f_{ij} > d_j \\ 1 & \text{otherwise .} \end{cases} \qquad \begin{array}{l} i = 1,2,\ldots,|N| \\ j = 1,2,\ldots,|C| \end{array}$$

Step 1: i) Let $d = \max_{i \varepsilon N} \min_{j \varepsilon J^m} f_{ij}$.

ii) Eliminate from A all columns $\{j\} \ni d_j \geq d$.

Step 2: Solve the CP (4.4).

Step 3: If $z > m$, stop; $D_m = d$ is the value of the optimal solution. Otherwise, an improved solution has been obtained. Update J^m and go to Step 1.

Remark: Step 1, part (i) can be simplified to

$$d = \max_{j \in J^m} d_j$$

except for the first iteration. Though some minor redundancy may result, this procedure allows the matrix F to be discarded after initialization.

Discussion

In the revised algorithm columns of A are eliminated, whereas previously elements of A were zeroized, leaving A, in general, with its original dimensions. The effect of this reduction in size becomes increasingly significant as m increases. Notice, however, that for small m we still encounter a great number of columns which, coupled with large $|N|$, renders the problem intractable for large problems for reasons to be explained in section 4.2.3.

ILP Formulation for P/N/m/G

An interesting consequence of Theorem 4.2 and associated corollaries is the following formulation of P/N/m/G as a 'single shot' MILP

$$\begin{aligned} &\min d \\ &\text{s.t. } Ax \geq e \\ &\qquad e^t x \leq m \\ &\qquad d \geq d_j x_j \quad \forall j \\ &\qquad x_j \varepsilon \{0,1\} \quad \forall j \end{aligned} \tag{4.8}$$

4.2.3 Covering Problems and Their Solution

In all the existing methods for solving large scale multi-center problems, solving the CPs, as distinguished from matrix generation, is the major computational burden and bottleneck. This is obviously true for N/N/m/G (Torregas et al.[44]) and is clearly indicated for P/N/m/G by Christofides and Viola [2]. The observation is equally true for the algorithms of sections 4.2.1 and 4.2.2. This bottleneck, severely limiting the size of problems which can be optimally solved, is a result of the reliance of existing techniques upon standard algorithms for the solution of the CPs. By fully utilizing their special structure, it is possible to solve problems several orders of magnitude larger than hitherto, enabling truly large scale models to be considered. Prior to embarking upon a discussion of such a technique, a few words of explanation are justified to substantiate the observation that standard CP techniques render large scale problems intractable.

In the ensuing discussion we refer to the general CP obtained by replacing the unit cost vector in (4.4) with a general vector. We remark also that some of the discussion relates to the Partitioning Problem (PP), where equalities replace the inequalities in the CP, though it can be shown that PP is but a special case of CP (see Garfinkel and Nemhauser [13]).

CPs have attracted extensive and intensive attention due to their wide applicability, simple structure, tantalizing properties and frustrating inherent complexity. Some of the better known applications include airline crew scheduling

[23,32,41,43] and political districting [13]. Initial efforts at solving the CP concentrated on CP-oriented implicit enumeration techniques as in Pierce [36] These were found useful for small problems particularly due to the natural use of computer oriented binary logic and data representation. Such schemes were invariably exponentially dependent upon the number of columns. More recent efforts, as in Handler [23] and Thiriez [43] succeeded in converting the critical parameter to the number of rows by basing solution techniques on LP methodology. A well-known property of CPs is that basic feasible solutions of the imbedded LP provide a dominant set for an optimal solution of the CP. Moreover, computational experience indicates a vague property of 'near total unimodularity' (Handler [23] Thiriez [43] Torregas [44]), so that LP solutions are often integer and almost always 'close' to the CP solution. Given the generally relatively large number of columns in CPs, these techniques have rendered medium size problems solvable. However, for reasons of massive degeneracy, problems with very many rows are still beyond solution. Thus, it is commonly accepted as in Geoffrion [14] and Marsten [32] that the real bottleneck in solving the general CP is the imbedded LP itself.

Various avenues have been explored to 'beat the bounds.' Rubin [14] has developed an effective heuristic algorithm, with application to airline crew scheduling. Balas and Padberg [1], Fisher [7], Handler [23] and Trubin [45] have sought special properties of CP and PP structures in order to modify the Simplex method to avoid the degeneracy difficulty. Alas, a final breakthrough is yet to be achieved in these efforts. The recent work of Fisher et al. [8] in developing dual decomposition techniques for discrete optimization may also lead eventually to a method whereby the LP may be circumvented. It is entirely possible, however, that the general CP will remain inherently 'complex' and that problem-oriented techniques, as for the job shop scheduling problem in Fisher [7], are the most viable approach. The proposed strategy for multi-center location constitutes another example of the success of this approach.

4.3 Relaxation Strategy for P/N/m/G

4.3.1 Motivation

The 'relaxation strategy' proposed herein is based upon two fundamental observations:

a. For any number of centers, m, there exist a number of critical nodes, a 'relaxed' set $R \subseteq N$, which essentially determine the optimal location of centers. An optimal solution for P/R/m/G will automatically cover the remaining nodes, $\bar{R}$. Moreover, $|R|$ will be relatively unaffected by $|N|$ and instead will be fairly closely related to m.

In relation to other math programming problems, this observation, and implied strategy, is akin to a relaxation approach for minimax formulations where, again, the observation is that most of the constraints in the derived constrained problem will be non-binding at an optimal solution.

b. Unlike general math programming relaxation strategies, in this case both rows and <u>columns</u> are eliminated as a result of the manner in which columns (candidate centers) are generated by rows (nodes) as outlined in section 4.2. Thus a special feature here is that advantages generally resulting only from restriction obtain as well, rendering the relaxation strategy particularly attractive.

4.3.2 Relaxation Algorithm

Notation

For completeness, we include some previously defined quantities. Recall that G denotes the infinite collection of points on the graph G(N,A), with arc lengths d_{ij}, $(ij)\varepsilon A$.

$d(x,y)$	$\equiv$	shortest distance between two points x,y on G.
$\|\|d(x,y)\|\|$	$\equiv$	shortest distance matrix for all node pairs $xy\varepsilon N\times N$.
$d(X^m,y)$	$\equiv$	$\min_{x\varepsilon X^m} d(x,y)$, where X^m is a set of m points on G.
$\ell(x)$	$\equiv$	$\max_{y\varepsilon N} d(x,y)$.
$\ell(X^m)$	$\equiv$	$\max_{y\varepsilon N} d(X^m,y)$.
D_m	$\equiv$	$\min_{X^m\varepsilon G^m} 2\,\ell(X^m)$, where G^m is the set of all sets X^m on G, twice optimal value of P/N/m/G.
R	$\equiv$	'relaxed' set of nodes, $R \subseteq N$.
C	$\equiv$	set of 'candidate centers' as defined in (4.1).

d_c	$\equiv$	'diametral span' of candidate center c as defined in (4.5).
C_{xy}	$\equiv$	$\{c \varepsilon C : \langle x,c,y \rangle$ exists for a given pair $xy\}$.
${}^dC_{xy}$	$\equiv$	$\{c \varepsilon C_{xy} : d_c < d\}$.
${}^dC_R$	$\equiv$	$\bigcup_{xy \varepsilon R \times R} {}^dC_{xy}$.
$\{z, \hat{X}^z, \hat{d} \mid {}^dC_R\}$	$\equiv$	optimal solution quantities for the CP defined in (4.4) where
z	$\equiv$	value of objective function,
$\hat{X}^z$	$\equiv$	identity of the z centers in the solution,
$\hat{d}$	$\equiv$	$\max_{c \varepsilon \hat{X}^z} \{d_c\}$.
AC	$\equiv$	'absolute center,' optimal solution of P/N/1/G.
AC(T)	$\equiv$	optimal solution of P/N/1/T .
VC	$\equiv$	'vertex center,' optimal solution of N/N/1/G .
MPT(x)	$\equiv$	minimum path tree rooted at a point x.

We shall state the algorithm in the more general context of solving P/N/m/G for $m = k, k+1, \ldots, M$.

Step 0: <u>Initialization</u>

Set $m = k$ and select judiciously an initial set of m centers, X^m. Then $d = 2\ell(X^m)$ is an upper bound on D_m . Let the initial relaxed set of nodes, R, comprise a set of critical nodes for the m chosen centers and find all candidate centers ${}^dC_R$. Proceed to Step 1.

Step 1: <u>Relaxed CP</u>

Select from ${}^dC_R$ m centers, $\hat{X}^m$, covering all nodes in R. If this is impossible go to Step 4. Otherwise, proceed to Step 2.

Step 2: <u>Improvement/Node-to-Enter</u>

$\hat{X}^m$ is an improved solution for P/R/m/G with value $\hat{d}$. If all nodes in $\bar{R}$ are covered by $\hat{X}^m$ within a distance $\frac{\hat{d}}{2}$, then $\hat{X}^m$ is an improved solution for P/N/m/G; let $\hat{X}^m$ replace X^m as the incumbent solution; set $d = \hat{d}$; eliminate centers whose diametral spans exceed or equal d and go to Step 1. Otherwise, designate as node-to-enter a node, n, which is furthest away from $\hat{X}^m$. If $\hat{d} < 2\ell(\hat{X}^m) < d$ then $\hat{X}^m$ is still an improved solution; set $d = 2\ell(\hat{X}^m)$ and replace X^m by $\hat{X}^m$. In both cases proceed to Step 3.

Step 3: Column Generation

Add to ${}^{d}C_R$ candidate centers generated by node pairs nr for all $r\varepsilon R$, whose diametral spans are less than d. Add n to R and go to Step 1.

Step 4: Optimality

Incumbent solution, X^m, is optimal for P/N/m/G with value $D_m = d$. If $m = M$ stop. Otherwise, retain this solution as incumbent for $m + 1$; let $m = m + 1$ and go to Step 1.

A flow chart of the relaxation algorithm appears in Figure 4.1. Computational details of the subroutines are given below, followed by a proof of the algorithm and discussions of computational efficiency and experience. Examples illustrating the the algorithm appear in Appendix B.

4.3.3 Subroutines

The basic computational building blocks, depicted as subroutines in the flow chart of Figure 4.1, need to be clarified.

SR0-Initialization

Assuming first that we wish to locate $m = 1,2,\ldots,M$ multicenters we can initialize the algorithm for $m = 1$ in one of several ways. We present three alternatives:

a) Choose any node $x\varepsilon N$ and find a furthest node $y\varepsilon N$ from it by constructing MPT(x). Then an upper bound for D_1 is given by

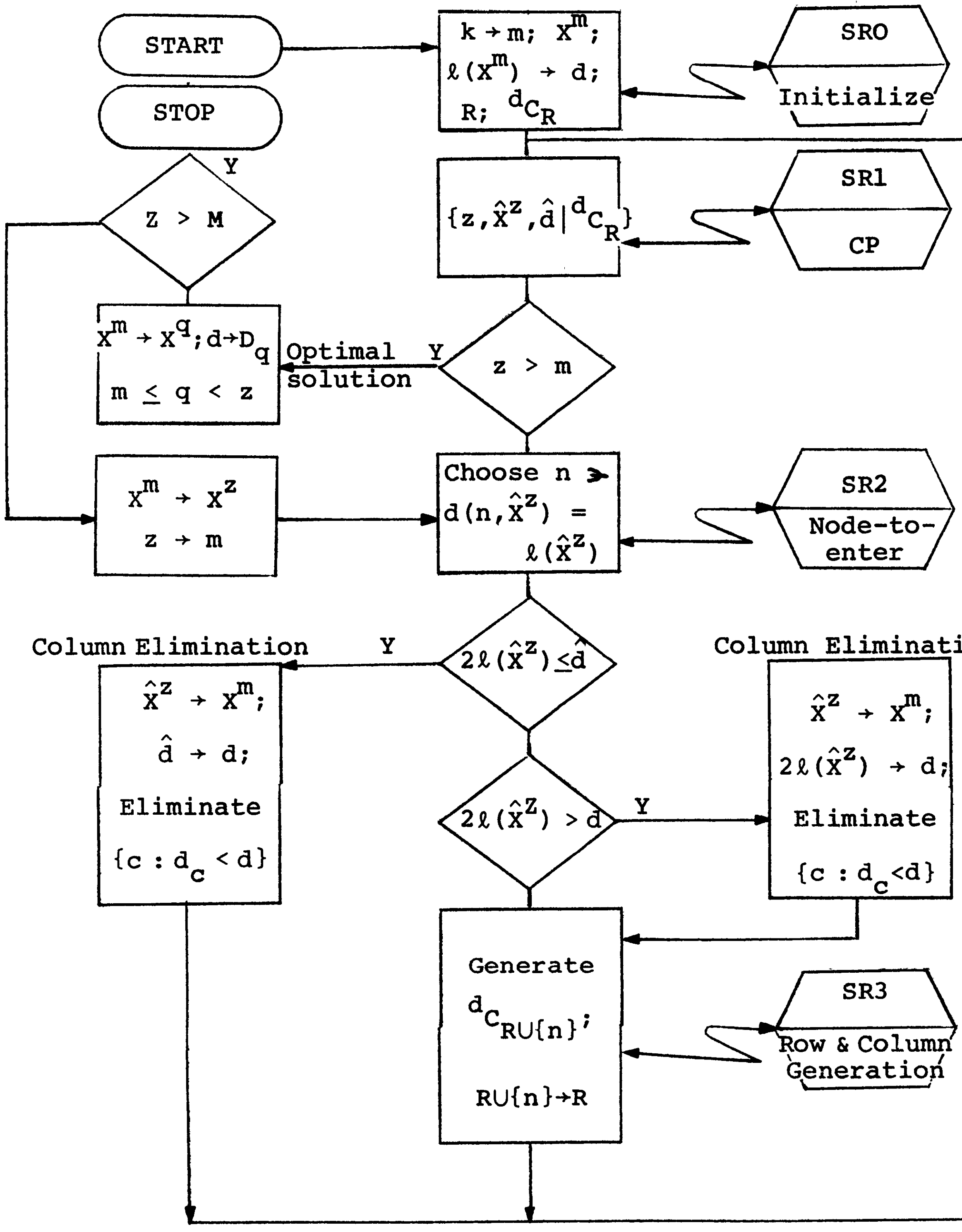


Figure 4.1: Flow Chart for Relaxation Algorithm

$$d = 2\ell(x) = 2d(x,y) \geq D_1 .$$

Let $R = \{x,y\}$ and $\hat{X}^1 = \{x\}$.

b) Choosing nodes x,y as in (a), a different upper bound for D_1 is given by the absolute center of the tree MPT(x) namely;

$$d = 2\ell\Big(AC\big(MPT(x)\big)\Big) \geq D_1$$

(See Chapter III). Let $R = \{e_1,e_2\}$ and $x^1 = \{AC\big(MPT(x)\big)\}$ where e_1,e_2 are diametral nodes found in constructing $AC\big(MPT(x)\big)$ using the algorithm of Chapter II.

Notice that given the information derived in (a) it is exceedingly simple to compute the quantities in (b). Thus e_1 corresponds to y and only one longest path need be found in MPT(x) in order to locate $AC\big(MPT(x)\big)$ as suggested in Chapter II.

c) The previous procedures are recommended for problems where M is small in relation to $|N|$. Otherwise, it becomes advantageous to compute initially the matrix $||d(x,y)||$ for subsequent computations in SR2 and SR3. In this case a solution to N/N/1/G, found by inspection, provides an upper bound on D_1, namely;

$$d = 2\ell(VC) \geq D_1 .$$

Futhermore, a furthest apart pair of nodes pq satisfying

$$d(p,q) = \max_{xy \varepsilon N \times N} d(x,y)$$

provides a useful lower bound on D_1, namely;

$$d(p,q) \leq D_1 .$$

Let $R = \{p,q\}$ and $\hat{X}^1 = \{VC\}$.

To complete the initialization procedure, ${}^dC_R$ has to be generated. A procedure for this is described in SR3 below.

Consider now the case where $m = k(>1),\ldots,M$ (including the case $k = M$). One possibility is to build up the solution by generating solutions for $m = 1,2,\ldots,k-1$ first. However, a more direct approach is preferable. The simplest procedure is to choose judiciously k points, X^k, as an incumbent solution for P/N/k/G with associated upper bound

$$d = \ell(X^k) \geq D_k .$$

We can conceive of various heuristic procedures for obtaining a good initial solution, for example, by successively adding centers and perturbing their locations to reduce the critical distance until a 'local optimum' is achieved for k centers. Let R be the set of critical nodes for the initial k centers and, as before, use SR3 to generate ${}^dC_R$.

SR1 - Set Covering Problem

We have already indicated in section 4.2.3 that the relaxation strategy in its entirety can be viewed as a problem oriented technique for solving the large scale CPs arising in a minimax facility location scenario. Operationally, we have transformed a succession of large scale CPs into a succession of miniscule CPs each of which requires a trivial computational effort for its solution.

The size of a generic CP is $|R| \times |{}^{d}C_{R}|$ compared with $|N| \times |C|$ and $|N| \times |{}^{d}C|$ in the algorithms of sections 4.2.1 and 4.2.2 respectively. Since $|R|$ is usually of the order of 2m (independently of $|N|$) it is evident that for small m the CP is indeed trivial. This is particularly true as a consequence of the sequential nature of the algorithm, enabling the CP (4.4) to be solved essentially as the feasibility problem

$$\begin{aligned} &Ax \geq e \\ &e^{t}x = m \\ &x_{j} \varepsilon \{0,1\} \quad \forall j . \end{aligned} \tag{4.9}$$

Indeed, all of the CPs encountered in the course of this research (for $m \leq 6$) were solved by manual inspection in less than one minute.

For large networks and large m it might become necessary to employ or modify some standard CP algorithm. However, it is difficult to envision a realistic situation where the CP part of the algorithm would require significant computational effort. We shall therefore not concern ourselves further with this particular issue.

SR2 - Node-to-Enter

Assuming prior computation of $||d(x,y)||$ it is straightforward to compute $\ell(\hat{X}^z)$. Alternatively, in the absence of the complete $||d(x,y)||$ matrix, it is necessary to compute MPT(x), $\forall x \varepsilon \hat{X}^z$. Note that since x is not necessarily a node a slight modification is required in the standard MPT algorithm.

SR3 - Column Generation

${}^dC_R$ requires updating to ${}^dC_{R\cup n}$ according to the expression

$$ {}^dC_{R\cup n} = {}^dC_R \bigcup_{r\varepsilon R} {}^dC_{nr} \cup \{n\} . \qquad (4.10) $$

A procedure for generating all candidate centers, ${}^dC_{xy}$, for a given pair xy is described below. This procedure is performed for all pairs nr, $r\varepsilon R$. Note also the inclusion of the null center C_{nn} in (4.10).

The necessary calculations can be performed directly from $||d(x,y)||$ or, in the absence of this matrix, after computing MPT(n) and utilizing the previously computed MPT(r), $\forall r\varepsilon R$. For each center, c, generated we need to record its location, diametral span, d_c and the subset of R,

$$ R_c \equiv \{r\varepsilon R : d(c,r) \leq \frac{d_c}{2}\} \qquad (4.11) $$

covered by it. Finally, we need to identify the subset of ${}^{d}C_R$,

$$Q \equiv \{c \in {}^{d}C_R : d(c,n) \leq \frac{d_c}{2}\} \qquad (4.12)$$

covering node n. Notice that $Q \cap \hat{X}^z = \emptyset$ according to the algorithm.

Generation of ${}^{d}C_{xy}$ $(x \neq y)$

For our discussion we shall find it convenient to extend the set of definitions of section 4.3.2. Consider a generating pair of nodes xy and a generic arc $(ij) \in A$.

Definitions

${}^{d}C^{ij}_{xy} \quad \equiv \quad \{c \in {}^{d}C_{xy} : c \text{ is on } (ij)\}.$

$S^{xy}_{(v)} \quad \equiv \quad \min\{d(v,x), d(v,y)\},\ v \in \{i,j\}.$

$<x,ij,y>$, denoting a 'flip-flop' condition on arc (ij) with respect to a pair of nodes xy, is said to exist iff:

$$\{S^{xy}_{(i)} - d(i,x)\}\{S^{xy}_{(j)} - d(j,x)\} + \{S^{xy}_{(i)} - d(i,y)\}\{S^{xy}_{(j)} - d(j,y)\} = 0.$$

The following lemma will enable us to construct an efficient algorithm for generating ${}^{d}C_{xy}$:

Lemma 4.1: $C^{ij}_{xy} \neq \emptyset$ only if $<x,ij,y>$ exists, in which case the following are mutually exclusive and exhaustive cases:

i) $<x,c,y>$ exists at an interior point of (ij), $|C^{ij}_{xy}| = 1$ and the diametral span and location of c are given by

$$d_c = d_{ij} + S^{xy}_{(i)} + S^{xy}_{(j)}$$

$$d_{ic} = \{d_{ij} + S^{xy}_{(j)} - S^{xy}_{(i)}\}/2 .$$

ii) $<x,c,y>$ exists at a node $v \varepsilon \{i,j\}$, $|C^{ij}_{xy}| = 1$

and

$$d_c = S^{xy}_{(v)} .$$

iii) $<x,c_i,y>$ and $<x,c_j,y>$ exist at nodes i,j respectively, $|C^{ij}_{xy}| = 2$ and

$$d_{c_v} = S^{xy}_{(v)}, \; v = i,j .$$

iv) $C^{ij}_{xy} = \emptyset$.

Futhermore, when all arcs $(ij) \varepsilon A$ are scanned, nothing is lost by assuming $|C^{ij}_{xy}| = \emptyset$ in case (iii).

A detailed but straight-forward proof of the lemma is given in Appendix A. The lemma leads directly to an efficient algorithm for generating $^dC_{xy} = \bigcup_{(ij) \varepsilon A} {}^dC^{ij}_{xy}$ as described in the flow chart in Figure 4.2.

Remark: In the algorithm described in Figure 4.2 redundancy can occur in two ways:

i) A node, v, can appear more than once as a candidate center. Such duplication can be avoided by 'closing'

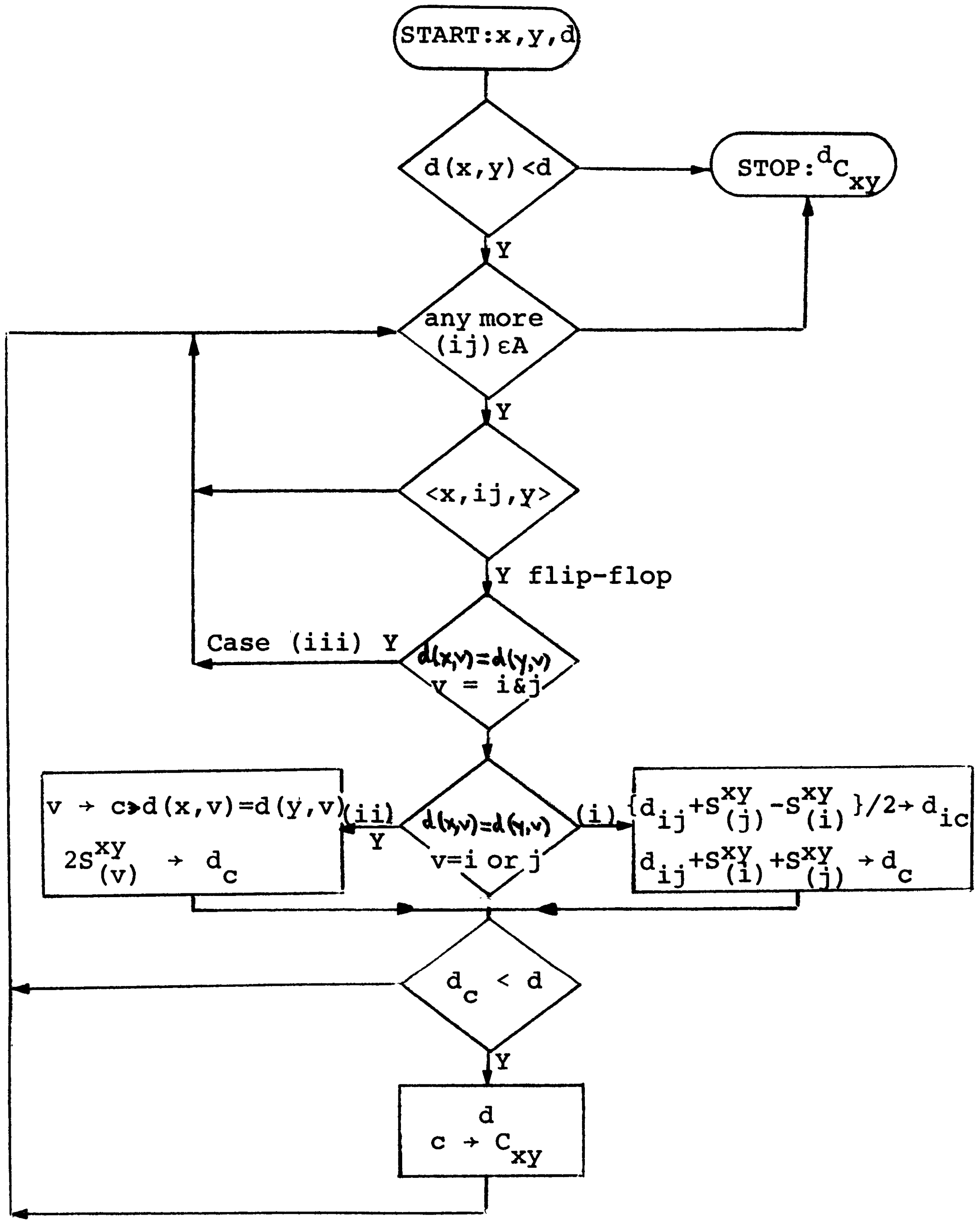


Figure 4.2: Flow Chart for Column Generation

node v after it is first designated as a center and subsequently skipping over all arcs incident to node v.

ii) A node, $\bar{v}$, designated as a candidate center may not qualify as a local center after all. To avoid this it is necessary to inspect the remaining incident arcs of v. Alternatively, the problem can be ignored since the additional computational burden due to such redundancy is generally insignificant.

4.3.4 Proof of Relaxation Algorithm

Theorem 4.3

The algorithm of section 4.3.2 achieves an optimal solution to P/N/m/G in a finite number of steps.

Proof: i) Optimality

A simple relaxation argument suffices. The algorithm finds an optimal solution, X^m, to P/R/m/G for some $R \subseteq N$ according to the results and algorithm of section 4.2.2. Futhermore,

$$d = \min_{X^m \varepsilon G^m} \max_{y \varepsilon R} d(X^m,y) = \max_{y \varepsilon N} d(X^m,y) \tag{4.13}$$

according to the algorithm. Now consider P/R/m/G reformulated as

$$\min_{X^m \varepsilon G^m} w$$

$$\text{s.t.} \quad w \geq d(X^m,y) \quad \forall\, y \varepsilon R \tag{4.14}$$

indicating that P/R/m/G is a relaxation of P/N/m/G. (4.13) and (4.14) establish that X^m is also optimal for P/N/m/G.

ii) Finite Convergence

An iteration of the algorithm consists of a CP and subsequent row/column generation, clearly a finite operation. At each iteration one and only one of the following occurs:

a) One row is added with possible column additions and eliminations.

b) At least one column is eliminated and none added.

Since $|N|$ is finite a necessary condition for an infinite number of iterations is that (b) is repeated infinitely for a given R. Since $|C_R|$ is finite (Lemma 4.1) this is impossible and finite convergence is guaranteed. ||

4.3.5 Computational Discussion

Algorithmic Framework

With respect to general math programming methodology, we have already indicated the special feature of the proposed relaxation algorithm incorporating as well restriction advantages. While the conceptual formulation (4.14) establishes P/R/m/G as a bona-fide relaxation problem, greater insight is derived by considering the MILP (4.8) from which it is evident how both rows and columns are eliminated in P/R/m/G.

In terms of multi-center strategies, notice that the proposed algorithm employs both algorithms of section 4.2. While

P/R/m/G is solved by the algorithm of section 4.2.2, P/N/m/G is implicitly solved also by Minieka's algorithm (section 4.2.1) since the condition of Theorem 4.2 does not necessarily apply to nodes in N-R.

The relaxation algorithm synthesizes the previous techniques to achieve massive savings in both CP and matrix generating phases.

Computational Efficiency

The critical factor is the cardinality of R at the optimal solution for a given m. In the worst case the algorithm will perform roughly as the algorithm of section 4.2.2. Consider P/N/1/G where G is a single loop and nodes are distributed at equal intervals. Then $|R| \approx |N|$ at the optimal solution (see example in section 4.4.1). But apart from such pathological cases yielding rather insignificant computational upper bounds, the algorithm can be expected to lead to immense savings. Preliminary computational experience suggests $|R| \approx 2m$ is a good approximation. This indicates an enormous relative advantage over the algorithm of section 4.2.2 where m is small. Since the latter algorithm is particularly efficient for large m it turns out that the proposed algorithm, by combining both features, is ideally suited for all values of m.

We now wish to focus attention on the number of columns $|{}^{d}C_{R}|$ which, apart from the direct dependence on $|R|$, is based upon the generic quantities $|{}^{d}C_{xy}|$.

Corollary 4.3:

$$^{d}C_{xy} \leq |A|$$

Proof: Directly from Lemma 4.1. ||

The following three observations indicate, however, why $|^{d}C_{xy}|$ is usually a very small number.

i) By definition, points $c \varepsilon G$ satisfying $<x,c,y>$ must satisfy requirements which severely limit their location. One manifestation of this is reflected in the following theorem:

Definition

Let $p(c)$ denote the diametral paths of length d_c joining x to y through c where $<x,c,y>$ exists.

Theorem 4.4

$<x,c,y> \Longrightarrow <x,z,y>$ does not exist for any $z \varepsilon p(c)$, $z \neq c$.

Proof: Let $z \varepsilon p(c)$ be a point closer to one end, say y, and assume $<x,z,y>$ exists. Then

$$\begin{aligned} d(c,x) &< d(c,z) + d(z,x) \\ &= d(c,y) - d(z,y) + d(z,x) \\ &= d(c,y) \end{aligned}$$

which is impossible. ||

ii) The effect of d in reducing $|^{d}C_{xy}|$ is all important. For small m the induced 'spread' of nodes in R sharpens this effect. Note that

$$d(x,y) \geq d \Rightarrow |{}^{d}C_{xy}| = 0$$

as utilized in the flow chart in Figure 4.2. For large m, small values of d again guarantee its effectiveness.

iii) Corollary 4.3 could be reformulated

$$|{}^{d}C_{xy}| \leq |A'| \tag{4.15}$$

where A' is the set of arcs derived after amalgamating adjacent arcs at nodes with degree 2. This observation is particularly important for P/P/m/G (section 4.4.1).

Finally, note the special case of trees where

$$|C_{xy}| = 1 .$$

Since now both CP and matrix generating efforts are minimal it appears that P/N/m/T can be reasonably solved manually for virtually any size of network as illustrated in example B.1 in Appendix B.

<u>Improvements</u>

Various possibilities exist for streamlining the basic algorithm. The following is a list of some of these:

* Observations (i) and (iii) above suggest possibilities for improving matrix generation efficiency.

* When $|R|$ exceeds 2m it may be advantageous to reduce $|R|$ by preserving only current critical nodes after an improve-

ment has been made. However, care must be taken to avoid cycling.

* Efficient heuristics can easily be developed for good initial solutions. A local optimum seeking perturbation technique appears attractive.

* For large m and very large $|N|$ it may be advantageous to combine a standard CP algorithm with a variety of problem-oriented reduction techniques.

4.3.6 Computational Experience

A major result of this research is that multi-center problems can generally be solved at minimal computational cost. Thus, for problems of type P/N/m/T no computer assistance is required, as illustrated in example B.1 of Appendix B. At least twenty other examples were generated with similar results. For the general case, P/N/m/G, for small values of m, it appears that computer assistance is required only for updating ${}^{d}C_{R}$, namely the matrix generating part, while solution of the resulting CPs is usually a trivial problem ideally suited for manual solution. This experience is reflected in Table 4.1 summarizing computational data for example B.2 of Appendix B.

Table 4.1: Computational Data for Example B.2

P/N/m/G, m = 1,2,3,4; $|N|$ = 53; $|A|$ = 81

			cpu - Secs.[1,2,3]
$\|d(x,y)\|$ matrix			0.24
m	$\|R\|$	CPs (rows × columns)[4,5]	
1	2	(2×2)	0.03
2	3	(3×6)	0.04
3	5	(4×10); (4×9); (4×2); (5×2); (5×4)	0.09
4	8	(6×16); (6×7); (7×18); (8×26); (8×16)	0.22

Legend: $\|d(x,y)\|$	≡	shortest distance matrix.
m	≡	number of centers.
$\|R\|$	≡	number of nodes in relaxed set at optimal solution.
CP	≡	set covering problem.
$\|N\|$	≡	number of nodes in network.
$\|A\|$	≡	number of arcs in network.

Notes: (1) Program written in Fortran IV, compiled under level G1, and run on an IBM 370/165.

(2) Times are incremental, but to solve for a given number of centers, M, would not necessitate solving for all previous $m < M$, as indicated in section 4.3.3.

(3) A large fixed cost was incurred in this example by first computing $\|d(x,y)\|$. This is very inefficient for small values of m. Thus, for $m = 1$, total time would be approximately 0.05 seconds instead of 0.27 as indicated.

(4) CPs were solved trivially by inspection. Corresponding cpu times would be insignificant.

(5) Number of columns in CPs does not include the identity matrix $I_{|R|}$ corresponding to the set of null centers at the nodes.

The advantages accruing from the relaxation scheme can be readily seen in this example. The number of nodes in R was generally twice the number of centers, m, in place of the full set of nodes, here 53, employed in existing algorithms. This represents a savings in orders of magnitude both with respect to the resultant CPs and matrix generation. Indeed it is doubtful whether Minieka's algorithm (section 4.2.1) can reasonably handle this problem, given the enormous number of generated columns. The computational data in Table 4.2 refers to the largest problem reported by Christofides and Viola [2] and serves to underscore the relative advantage of the relaxation scheme.

Table 4.2: Computational Data from [2]
for Graph with $|N| = 50$; $|A| = 80$

number of centers, m	1	2	3	4
cpu - seconds on CDC/6600	17.8	8.11	17.7	24.55

As problem size increases, the relative advantage of the relaxation approach increases. Indeed, it seems quite reasonable to solve problems with thousands or tens of thousands of nodes using this approach. For the existing techniques, networks with more than one hundred nodes would appear prohibitively large.

To further validate the efficiency of the algorithm, we constructed a 'worst case' example by generating a 'random network,' with uniform distributions for the number of arcs incident to any node, the identity of adjacent nodes and the lengths of arcs. Such a network loses physical distance properties which are particularly attractive for the algorithm. Furthermore, we generated an example for larger than can possibly be solved by existing techniques, namely; $|N| = 200$ and $|A| = 512$. Table 4.3 provides computational data for $m = 1$ and m = 3. The table is subject to the same legend and notes as Table 4.1. Note in particular the dominant cost of computing $\|d(x,y)\|$, which for m = 1 and m = 2 is really grossly redundant. Thus, realistic total times for m = 1 and m = 2 are on the order of 1.5 and 7 seconds respectively.

Table 4.3: Computational Data for Random Network with $|N| = 200$; $|A| = 512$

			cpu-Secs.
$\|d(x,y)\|$ matrix			19.8
m	$\|R\|$	CPs (rows × columns)	
1	4	(2×5); (3×41); (4×163)	0.4
2	8	(5×247); (6×476); (7×799) (7×586); (7×531); (7×328) (8×542)	3.3

4.4 Extensions

So far the discussion of the relaxation algorithm has been in the context of P/N/m/G. In this section we seek to demonstrate the applicability and efficiency of the algorithm for a wider range of related problems, thus completing our discussion of multi-center problems. In addition to the direct benefits of extended applicability, the ensuing discussion will add fresh insight into the nature of the relaxation scheme.

4.4.1 P/P/m/G

This problem is perhaps the most challenging of the variety of location problems we are considering.

Physically it represents what is often the most realistic situation while P/N/m/G is some discretized approximation of it. So far as is known to the author, no attempt has been made in the literature to solve this apparently difficult problem and certainly none of the techniques for P/N/m/G can be directly applied to it. The only existing feasible approach is indeed to discretize the set of demand generating points as finely as desired and solve as P/N/m/G. Clearly, such a scheme is doomed to failure as the approximation is tightened because of the critical effect of a large number of nodes in existing techniques.

Consider now the relaxation approach of section 4.3. All along we have been solving relaxed problems P/R/m/G, $R \subseteq N$, as surrogates for P/N/m/G. The only conceptual change here is that N is now the infinite set of 'artificial nodes,' G. The basic strategy remains unaltered and only minor modifications are required. To enable points $y \varepsilon G$ to be considered as 'nodes,' it is necessary to redefine

$$\ell(X^m) \equiv \max_{y \varepsilon G} d(X^m, y) \qquad (4.16)$$

necessitating a simple extension in the MPT computations in SR2 (section 4.3.3). To generate ${}^{d}C_{xy} = \bigcup_{(ij)\varepsilon A} {}^{d}C_{xy}^{ij}$ note that it is sufficient in SR3 (section 4.3.3) to consider arcs (ij) which join 'intersection nodes' (nodes with degree greater than two) so that artificial nodes need not be considered in this phase. (See also section 4.3.5 and (4.15).)

Although P/P/m/G utilizes to the fullest extent the capabilities of the relaxation method and provides the most striking context for its application, it nevertheless remains elusive in its convergence properties. We have shown that for P/N/m/G the algorithm converges finitely (Theorem 4.3). The same is not true for P/P/m/G and the following, possibly pathological, example illustrates this. Consider a network, as in Figure 4.3, composed of a single cycle and solve the problem P/P/1/G. Then application of the algorithm, beginning at some arbitrary point, a, will lead to an infinitely convergent series of intermediate solutions. The labels in Figure 4.3 represent successive additions to R of 'artificial nodes' as required in (4.16).

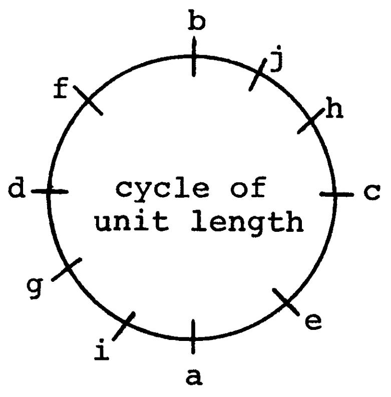


Figure 4.3: Unit Cycle for P/P/1/G

Successive iterations of the algorithm yield intermediate results as shown in Table 4.4.

Table 4.4: Intermediate Results for Unit Cycle Example - P/P/1/G

R	$\hat{X}^1$	$\hat{d}$	$2\ell(\hat{X}^1)$
a	a	0	1
a,b	d	1/2	1
a,b,c	c	1/2	1
a,b,c,d	f	3/4	1
a,b,c,d,e	e	3/4	1
a,b,c,d,e,f	h	3/4	1
a,b,c,d,e,f,g	g	3/4	1
a,b,c,d,e,f,g,h	j	7/8	1
etc...			

In general, $\hat{d} = \frac{2^s - 1}{2^s}$ where s = 0,1,... represents step number each of which entails 2^s iterations.

4.4.2 N/N/m/G

A recent treatment by Torregas et al. [44] is equivalent to Minieka's scheme for P/N/m/G (section 4.2.1) except that the dominant set of candidate centers C (4.1) is replaced by the set N. The mid-point property (Theorem 4.1) no longer holds though centers are now restricted to a finite set of points. Indeed, within the framework of Minieka's scheme, N/N/m/G is by far the easier of the two problems since in general, $|C| >> \binom{|N|}{2} + |N| >> |N|$.

The relaxation approach can be applied here with some modification. Note first that the column elimination scheme of section 4.2.2 is no longer applicable since that scheme was inextricably tied up with the mid-point property. Consequently, columns cannot be eliminated and the full complement of $|N|$ centers must be carried explicitly throughout. Otherwise, the relaxation scheme remains as before. Finite convergence is clearly guaranteed.

Though the improvement is not as dramatic as for P/N/m/G and P/P/m/G, substantial savings can be expected because of the importance of the number of rows in the solution of CPs (section 4.2.3). Furthermore, even in the classical case where m = 1 and no CP need be solved, substantial savings can be made. To illustrate this point consider the example given by Torregas et al. in [44]. Table 4.5 reproduces the $||d(x,y)||$ matrix for 30

Table 4.5: The Minimum-Distance Matrix for Locations in New York State (The distances are in miles)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	0	244	140	128	281	196	181	51	246	167	339	54	203	146	295	211	295	78	169	39	167	112	71	227	157	16	135	7	90	165
2	244	0	158	359	37	111	66	256	60	112	101	278	272	328	51	222	77	200	106	281	332	263	294	33	284	233	109	248	161	164
3	140	156	0	202	194	56	92	170	117	46	215	137	256	170	209	206	160	62	114	177	279	105	136	136	239	129	78	144	97	148
4	128	359	202	0	345	259	294	179	319	248	415	90	331	61	410	339	361	176	290	100	295	106	70	337	285	143	254	133	211	293
5	281	37	194	395	0	145	102	305	92	148	69	317	309	366	19	259	74	236	143	314	369	299	330	70	321	272	146	285	198	201
6	196	111	56	258	145	0	60	229	61	34	159	149	269	226	162	219	104	118	112	213	315	161	192	100	274	185	91	200	128	161
7	191	66	92	294	102	60	0	206	67	47	157	220	225	262	117	175	114	134	59	213	279	197	228	46	237	170	49	185	101	117
8	51	268	170	179	305	229	206	0	275	195	366	105	152	197	319	180	322	108	186	81	116	163	124	242	106	41	159	48	107	175
9	248	60	117	319	92	61	67	275	0	87	111	254	292	287	111	242	56	175	126	285	346	222	253	60	304	237	116	252	168	194
10	167	112	46	248	140	34	47	135	67	0	185	179	235	216	163	185	130	93	79	204	281	151	192	90	240	156	57	171	94	127
11	338	101	215	416	69	159	157	366	111	185	0	348	373	381	98	323	55	273	207	375	433	316	351	134	385	327	206	342	258	265
12	54	278	137	95	317	189	220	135	254	179	348	0	257	95	329	250	293	86	205	65	221	58	20	254	211	69	169	61	126	204
13	203	272	255	331	309	269	225	152	292	235	373	257	0	349	223	66	339	236	168	233	60	315	274	239	46	193	179	200	171	108
14	146	328	170	61	366	226	262	137	287	215	381	95	349	0	379	343	326	179	284	144	313	69	75	306	303	161	248	151	219	297
15	295	51	209	410	19	162	117	319	111	163	98	329	323	379	0	273	93	251	157	332	363	314	345	84	335	284	160	299	212	215
16	211	222	206	339	259	219	175	180	242	145	323	250	66	343	273	0	289	192	118	249	126	293	270	189	92	198	128	213	129	58
17	295	77	160	361	74	104	114	322	56	130	55	293	339	326	93	289	0	218	173	332	393	261	296	106	351	284	163	299	215	231
18	78	230	62	176	236	118	134	108	175	93	273	86	236	179	251	192	218	0	130	118	221	124	106	178	194	67	94	82	64	146
19	169	106	114	290	143	112	59	185	126	79	207	205	168	284	157	118	173	130	0	206	228	219	225	73	183	156	36	171	79	60
20	39	281	177	100	318	233	213	81	205	204	375	65	233	144	332	249	332	118	206	0	191	123	76	257	187	52	172	37	127	202
21	167	332	279	295	369	315	279	116	346	261	433	221	60	313	383	126	393	221	228	191	0	279	238	293	52	157	230	162	187	168
22	112	263	105	106	299	161	197	163	222	151	316	58	315	69	314	293	261	124	219	123	279	0	60	241	269	127	183	119	164	247
23	71	294	136	70	330	192	228	174	253	182	351	20	274	75	345	270	296	106	225	76	238	60	0	272	228	86	189	76	146	224
24	227	33	136	337	70	100	46	242	60	90	134	254	239	306	84	189	106	178	73	257	299	241	272	0	251	209	85	224	135	131
25	157	284	239	285	321	274	237	106	334	240	385	211	46	303	335	92	351	194	180	187	52	269	229	251	0	147	188	154	147	120
26	16	233	129	143	272	185	170	41	237	156	327	69	193	161	284	198	284	67	156	52	157	127	86	209	147	0	124	15	77	152
27	135	109	78	254	146	91	49	159	116	57	206	169	178	248	160	128	163	94	36	172	230	183	189	85	189	124	0	139	52	70
28	7	248	144	133	285	200	185	48	252	171	342	61	200	151	299	213	299	82	171	37	162	119	76	224	154	15	139	0	92	167
29	90	161	92	211	198	128	101	107	168	94	258	126	171	219	212	124	215	64	79	127	187	164	146	135	147	77	52	92	0	83
30	165	164	148	293	201	161	117	175	194	127	265	204	108	297	215	58	231	146	60	202	169	247	224	131	120	152	70	167	83	0

Table 4.6: Intermediate Solutions for N/N/1/G in New York State Example

R	$\hat{VC}$	$\hat{d}/2$	$\ell(\hat{VC}) = d(\hat{VC},n)$
1	1	0	338 = d(1,11)
1,11	7	181	294 = d(7,4)
1,11,4	3	215	279 = d(3,21)
1,11,4,21	27	254	254

locations in New York State. Consider now the problem of locating a solution VC to N/N/1/G. The classical approach of Hakimi [21] is to initially compute $||d(x,y)||$ and inspect the matrix for an optimal solution. Employing instead the relaxation technique and beginning arbitrarily with $R = \{1\}$ results in the series of solutions to N/R/1/G shown in Table 4.6, derived by inspection of the relevant columns and rows in Table 4.5. Thus, VC = 27 and $\ell(VC) = 254$ is a solution to N/N/1/G. Notice that to arrive at the solution required ocmputation of MPT(x), $\forall x \varepsilon \{1,11,7,4,3,21,27\}$ in place of the full $||d(x,y)||$ matrix or MPT(x), $\forall x \varepsilon \{1,2,...,30\}$. However, this advantage is somewhat mitigated by the relative efficiency of computing the complete matrix as in [5].

In conclusion, we have seen how N/N/m/G can be profitably solved with the relaxation algorithm. However, the advantages due to this approach are not quite as dramatic as for P/N/m/G and P/P/m/G and this serves to emphasize the unique features of the relaxation scheme when applied to these problems. With respect to N/N/m/G the relaxation technique can be viewed as a classical application of relaxation concepts to a minmax problem with corresponding elimination of constraints. But for P/N/m/G and P/P/m/G the technique is ideally suited because the linkage between columns and rows results in both row and column eliminations. Thus, problem relaxation alone carries the advantages of both relaxation and restriction.

Finally, note the reversal in relative computational difficulty for these problems. N/N/m/G now becomes the more difficult problem to solve, contrary to existing state of the art and, perhaps, to intuition. The situation is akin to the relative complexity of LP and ILP problems.

4.4.3 N/P/m/G

To the best of our knowledge, no treatment of this problem appears in the literature. Existing techniques can be applied directly to a discretized approximation, though this becomes impractical as the approximation is tightened. An exact formulation amenable to existing techniques is derived at the end of this section though the result is essentially conceptual.

Once again, the relaxation approach is ideally suited to this problem. The procedures outlined for N/N/m/G and P/P/m/G can be readily combined to form a solution strategy for N/P/m/G. An important distinction from P/P/m/G is contained in the following theorem.

Theorem 4.5

The relaxation algorithm appropriately modified to solve N/P/m/G converges to an optimal solution in a finite number of steps.

Proof: An iteration of the algorithm results in one of the following two cases:

i) A new artificial node is added to R.

ii) At least one non-zero element of the CP matrix is changed to zero.

To prove finite convergence it suffices to show that R has finite maximal cardinality. According to the algorithm, $R \subseteq W$ where

$$W \equiv \{w \varepsilon G : d(X^m, w) = \ell(X^m) \text{ for some } X^m \varepsilon N^m\}. \qquad (4.17)$$

Since $|N|$ is finite so is $|W|$ and convergence is guaranteed. ||

Corollary 4.4: N/P/m/G can be reformulated without loss of optimality as N/W/m/G where W is a finite set of nodes.

Proof: Directly from the previous proof, letting W be defined as in (4.17). ||

Remark: N/P/m/G can now be solved conceptually as N/N/m/G with existing tools. Clearly, however, such an approach is impractical while the relaxation scheme is an efficient method.

4.4.4 Solving the 'Inverse'

Consider first $P/N/\lambda^{-1}/G$ and let z_λ denote the minimal number of centers. The relationship between z_λ and the solution, D_m, to P/N/m/G is illustrated in Figure 4.4.

Solving $P/N/\lambda^{-1}/G$ requires minor and generally simplifying modifications to any procedure that solves P/N/m/G. Thus, the algorithms of section 4.2 can be used on a 'one shot' basis, solving just one CP. As usual, however, the relaxation technique is far more efficient. Setting $d = \lambda$ from the outset and ignoring intermediate updates of X^m where $z > m$, the algorithm of section 4.3.2 will yield $z_\lambda = z$ when $\ell(\hat{X}^z) = \lambda$ for the first time.

Extensions to the cases $P/P/\lambda^{-1}/G$, $N/N/\lambda^{-1}/G$ and $N/P/\lambda^{-1}/G$ are straightforward. Note that $N/N/\lambda^{-1}/G$ is the problem considered by Torregas et al. in [44].

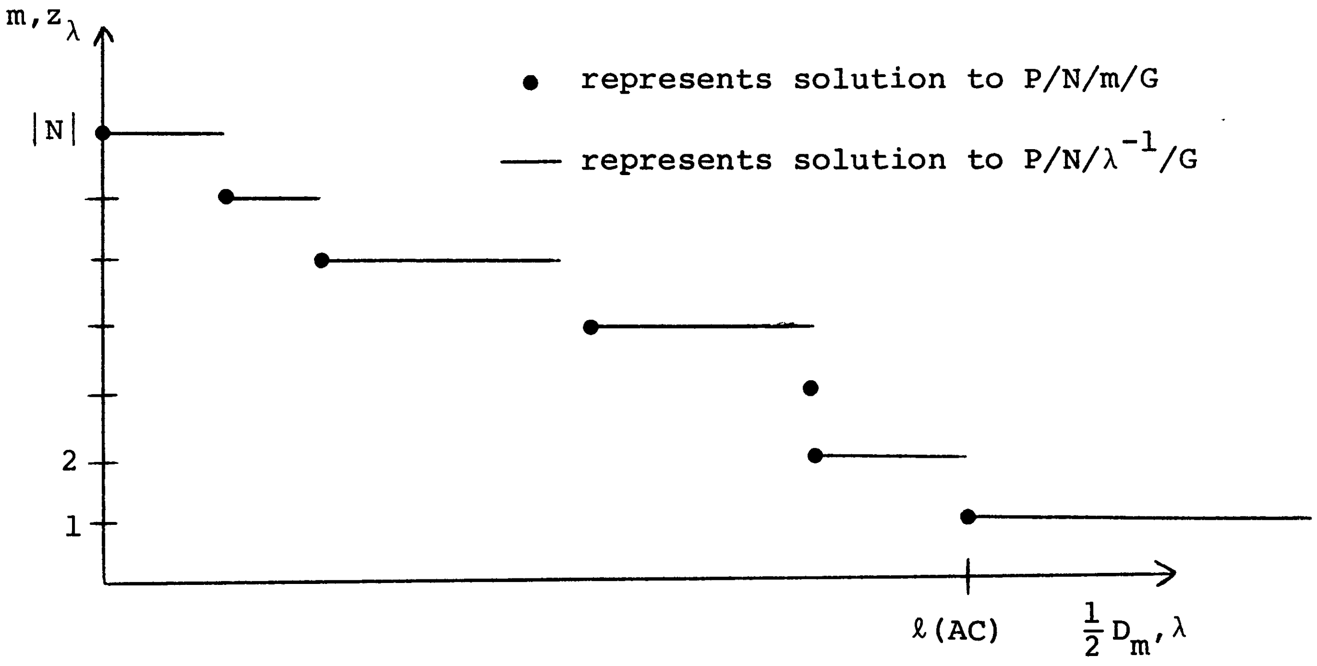


Figure 4.4: Relationship Between z_λ and D_m

4.5 Single Centers Revisited

By synthesizing the developments of the previous three chapters with respect to single center problems, a number of ideas come into clearer focus, leading to fresh insight and some further algorithmic suggestions. For expository purposes we confine the discussion to AC.

Resolution of Relaxation Gap

Chapter III introduced a relaxation approach for P/N/1/G and a relationship summarized as

$$\max_{x \in N} \ell(x) \leq \bar{D}_1 = D_1(\text{MDT}) \tag{4.18}$$

(Theorems 3.1 and 3.3). An ascent algorithm for the left-hand-side in (4.18) involved, possibly, a 'double relaxation gap' (section 3.4). The generalized relaxation algorithm of this chapter, for m = 1, effectively resolves this gap. Notice a conceptual similarity to the dual approach to some discrete optimization problems taken by Fisher et al. [8]. An ascent algorithm is used first to approximate the dual function, after which another strategy, possibly 'branch and bound,' is employed to resolve the resultant total gap from the primal function. Considering now the right-hand-side in (4.18), we now have an efficient relaxation strategy for locating an MDT. Again, analogies may be drawn to Held and Karp's strategy for locating a minimal hamiltonian circuit [27], which is also one of the examples analyzed in [8].

Composite Algorithm for P/N/1/G

Where $\|d(x,y)\|$ is readily available, employ the 'split and bound' algorithm of section 3.3, and if necessary, continue with the relaxation algorithm of this chapter. Notice that ${}^{d}C_R$ is particularly easily determined since many arcs have been eliminated in the first stage. If $\|d(x,y)\|$ is not available, use first the ascent algorithm of section 3.4 and continue with the relaxation algorithm of this chapter.

Tree Networks

Finally, we have seen in Chapter II that the ascent algorithm locates AC(T) in exactly two iterations so that no 'relaxation gap' need be resolved. Furthermore, it is interesting to observe that the relaxation algorithm of this chapter, when applied to a tree, reduces to a procedure essentially equivalent to the algorithm of section 2.2.

CHAPTER V

CONCLUSIONS

5.1 Introduction

In this chapter we briefly summarize the major results, indicate straightforward extensions and suggest some items for future research which are important and appear tractable in the light of this research.

5.2 Summary

Referring to the variety of minimax network location models defined in Chapter I, we have developed methodologies for solving truly large scale problems of all varieties included in categories (i) through (iv) in section 1.1. Chronologically, and in increasing order of complexity, theory and algorithms have been developed for

* $\begin{Bmatrix}P\\N\end{Bmatrix}/\begin{Bmatrix}P\\N\end{Bmatrix}/1/T$

* $\begin{Bmatrix}P\\N\end{Bmatrix}/\begin{Bmatrix}P\\N\end{Bmatrix}/2/T$

* $\begin{Bmatrix}P\\N\end{Bmatrix}/\begin{Bmatrix}P\\N\end{Bmatrix}/1/G$

* $\begin{Bmatrix}P\\N\end{Bmatrix}/\begin{Bmatrix}P\\N\end{Bmatrix}/\begin{Bmatrix}m\\\lambda\end{Bmatrix}\text{-}1/G$

as well as for a single center, tree based class of mixed minimax and minisum objective function problems. Networks with tens of thousands of nodes ought not to prove intractable even for the most general, multi-center, problems. Furthermore, virtually any tree based problem is now amenable to manual solution. Finally, an interesting result is the reversal in relative complexity of P/N/m/G and N/N/m/G, the latter now be-comming, usually, the more formidable problem.

In a wider context, a major result of this research has been the successful resolution of massive set covering problems arising in the minimax location scenario. By developing a problem-oriented relaxation technique, these were effectively transformed into similar, but miniscule covering problems. Though not pursued here, the implications for other set covering formulations, for minimax problems in general and for relaxation strategies are significant.

5.3 Extensions

We indicate the applicability of the results, with minor modifications, to the models described in categories (v) through (vii) in section 1.1. We shall confine the discussion initially to the relaxation strategy for multi-center problems given in Chapter IV. Consider the categories in turn.

Category (v) - Restricted Location Sets

Where demand locations are thus restricted, the results are directly applicable by virtue of the very nature of the relaxation scheme. Similarly, in the case of facilities on a restricted node set the algorithm remains identical. Indeed, in both cases the resultant problem is easier than before. However, some modification is required if facilities are restricted to a sub-set of the point set, since the 'mid-point property' (Theorem 4.1) no longer holds without qualification. To solve the problem define as 'nodes' all of the boundary points of the restricted set of points and adopt a hybrid algorithm combining the strategies for $P/\left\{\begin{smallmatrix}P\\N\end{smallmatrix}\right\}/m/G$ and $N/\left\{\begin{smallmatrix}P\\N\end{smallmatrix}\right\}/m/G$.

Category (vi) - Unequal Node Weights

The 'mid-point property' is easily generalized to account for node weights as are all of the remaining propositions, so that a generalized algorithm is simply derived.

Category (vii) - Arc Orientation

Again, all of the propositions are directly extended to this case. Note, however, that in this case it is necessary to specify also whether the 'server' or the 'customer' is the mobile unit.

Finally, note that even the basic assumptions of non-negative arc 'lengths' and the restriction $I(x) \neq 2 \quad x \epsilon N$ are not essential prerequisites for the algorithm.

The special features associated with tree networks, discussed in Chapter II, are affected by most of the afforementioned categories and suitable extensions need to be developed for every change in assumption.

5.4 Future Work

Further computational experience is required to accurately determine the cost of solving the variety of multi-center problems. In particular, it is interesting to ascertain the efficiency of the relaxation algorithm generalized to accomodate unequal node weights. Francis [10] observes a general tendency of weighted minimax location problems to be 'badly behaved' in comparison with equal weight problems. The relative efficiency of solving P/P/m/G, previously intractable, is also worth investigating. Furthermore, it should be possible to devise a modified algorithm for this problem which will guarantee finite convergence.

An area of critical importance is the combined minimax (center) and minisum (median) model. Though we have made some preliminary efforts in this direction, the issue remains unresolved for the general problem. It is felt that with efficient algorithms now available for the separate multi-facility problems, the time is ripe for a purposeful research effort in this direction.

In a wider context, we have already indicated the, at least conceptual, implications of the results of this research for related optimization areas. Specifically, set-covering problems, minimax formulations and relaxation strategies might usefully be reexamined in the light of our findings.

REFERENCES

1. Balas, E., and M. W. Padberg, "On the Set Covering Problem," Opns. Res. 20, 1152-1161 (1972).

2. Christofides, N., and P. Viola, "The Optimum Location of Multi-Centers on a Graph," Opnal. Res. Quart. 22, 145-154 (1971).

3. Dearing, P .M., and R. L. Francis, "A Minimax Location Problem on a Network," Technical Report No. 198, Department of Operations Research, College of Engineering, Cornell University, Ithaca, New York, 1973.

4. Dijkstra, E. W., "A Note on Two Problems in Connection with Graphs," Numer. Mathematik 1, 269-271 (1959).

5. Dreyfus, S. E., "An Appraisal of Some Shortest-Path Algorithms," Opns. Res. 17, 395-412 (1969).

6. Elshafei, A. N., and K. B. Haley, "Facilities Location: Some Formulations, Methods of Solution, Applications and Computational Experience," OR Report No. 91, North Carolina State University at Raleigh, 1974.

7. Fisher, M. L., "Optimal Solution of Resource Constrained Network Scheduling Problems," Technical Report No. 56, Operations Research Center, Massachusetts Institute of Technology, 1970.

8. Fisher, M. L., W. D. Northrup and J. F. Shapiro, "Using Duality to Solve Discrete Optimization Problems: Theory and Computational Experience," Working Paper No. 030-74, Operations Research Center, Massachusetts Institute of Technology, 1974.

9. Francis, R. L., and J. M. Goldstein, "Location Theory: A Selective Bibliography," Opns. Res. 22, 400-410 (1974).

10. Francis, R. L., and J. A. White, Facility Layout and Location: An Analytical Approach, Prenctice-Hall, Inc., Englewood Cliffs, N. J., 1974.

11. Frank, H., and I. T. Frisch, Communication, Transmission and Transportation Networks, Addison-Wesley, 1971.

12. Garfinkel, R. S., A. W. Neebe, and M. R. Rao, "The m-Center Problem: Bottleneck Facility Location," Working Paper Series No. 7414, Graduate School of Management, The University of Rochester, 1974.

13. Garfinkel, R. S., and G. L. Nemhauser, Integer Programing, John Wiley, 1972.

14. Geoffrion, A. M., "Preface and Guide," Management Sci. 20, 733-735 (1974).

15. Gillespie, C. M., Jr., "Locating Absolute 2-Centers of Undirected Graphs," unpublished M.S. thesis, Naval Postgraduate School, Monterey, California, 1968.

16. Gleason, J. M., "Bus-Stop Location on School Bus Routes," ORSA Bull. 22, Sup 1, B-76 (1974).

17. Goldman, A. J., "Optimal Center Location in Simple Networks," Transportation Sci. 5, 212-221 (1971).

18. Goldman, A. J., "Minimax Location of Facility on a Network," Transportation Sci. 6, 407-418 (1972).

19. Goldman, A. J., private communication, March 1973.

20. Hakimi, S. L., "Optimum Locations of Switching Centers and the Absolute Centers and Medians of a Graph," Opns. Res. 12, 450-459 (1964).

21. Hakimi, S. L., "Optimum Distribution of Switching Centers in a Communication Network and Some Related Graph Theoretic Problems," Opns. Res. 13, 462-475 (1965).

22. Halfin, S., "On Finding the Absolute and Vertex Centers of a Tree with Distances," Trans. Sci. 8, 75-77 (1974).

23. Handler, G. Y., "Airline Crew Scheduling and Zero-One Programming," unpublished M.Sc. thesis, Department of Industrial Engineering, Technion, Israel Institute of Technology, Haifa,1971.

24. Handler, G. Y., "Minimax Location in an Undirected Graph," Technical Memo. No. 72-7, Flight Transportation Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1972.

25. Handler, G. Y., "Minimax Location of a Facility in an Undirected Tree Graph," Trans. Sci. 7, 287-293 (1973).

26. Handler, G. Y., and A. R. Odoni, Minimax Location of a Facility in a Graph," ORSA Bull. 21, Sup. 1, B-42 (1973).

27. Held, M., and R. M. Karp, "The Travelling Salesman Problem and Minimum Spanning Trees," Opns. Res. 6, 1138-1162 (1970).

28. Hu, T. C., "Some Problems in Discrete Optimization," Mathematical Programming 1, 102-112 (1971).

29. Karp, R. M., "Reducibility Among Combinatorial Problems," Technical Report No. 3, Computer Science, The University of California, Berkeley, 1972.

30. Kruskal, J. B., "On the Shortest Spanning Subtree of a Graph," Proc. Amer. Math. Soc. 7, 48 (1956).

31 Marsten, R. E., "An Algorithm for Finding Almost All of the Medians of a Network," Discussion Paper No. 23, Center for Mathematical Studies in Economics and Management Science, Northwestern University, 1972.

32. Marsten, R. E., "An Algorithm for Large Scale Set Partitioning Problems," Management Sci. 20, 774-787 (1974).

33. Minieka, E., "The m-Center Problem," SIAM Review 12, 138-139 (1970).

34. Odoni, A. R., "Location of Facilities on a Network: A Survey of Results," Technical Report No. 3-74, Operations Research Center, Massachusetts Institute of Technology, Cambridge, MA, 1974.

35. Ore, O., "Theory of Graphs," American Mathematical Society Colloquium Publications 38, 1962.

36. Pierce, J. F., "Application of Combinatorial Programming to a Class of All-Zero-One Integer Programming Problems," Management Sci. 15, 191-209 (1968).

37. Reed, J. J., "Two Algorithms for Finding the Absolute m-Center of a Graph," unpublished M.S. thesis, Naval Postgraduate School, Monterey, California, 1971.

38. ReVelle, C. S., D. Marks and J. C. Liebman, "An Analysis of Private and Public Sector Location Models," Management Sci. 16, 692-707 (1970).

39. Rosenthal, M. R., and S. B. Smith, "Solution to the One-Center Problem, the m-Center Problem and the Location-Allocation Problem," paper presented at the ORSA meeting, New York, 1967.

40. Rosenthal, M. R., and S. B. Smith, "The m-Center Problem," Illinois Institute of Technology, reported in course 2695, Massachusetts Institute of Technology, Cambridge, MA, 1969.

41. Rubin, J., "A Technique for the Solution of Massive Set Covering Problems, with Application to Airline Crew Scheduling," Transportation Sci. 7, 34-48 (1973).

42. Singer, S., "Multi-Centers and Multi-Medians of a Graph with an Application to Optimal Warehouse Location," a report from Dunlop and Associates, Inc., Darien, Connecticut, U.S.A., 1968.

43. Thiriez, H., "Airline Crew Scheduling: A Group Theoretic Approach," Report No. R69-1, Flight Transportation Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1969.

44. Torregas, C., R. Swan, C. ReVelle, and L. Bergman, "The Location of Emergency Service Facilities," Opns. Res. 19, 1366-1373 (1971).

45. Trubin, V. A., "On a Method of Solution of Integer Linear Programming Problems of a Special Kind," Soviet Math Dokl. 10, 1544-1546 (1969).

APPENDIX A

PROOF OF LEMMA 4.1

We shall refer to the diagrams in Figure A.1. Consider first the general form of the function d(x,z) where z is a point on (ij), z units from i, $z \varepsilon [0,d_{ij}]$. The following properties of d(x,z) are readily established and illustrated in the generic example in (a):

* one or two piece linear
* gradient ± 1
* concave.

We consider the relationship between d(x,z) and d(y,z) and distinguish three mutually exclusive and exhaustive cases:

<u>i) $d(x,v) \neq d(y,v)$, v = i,j</u>

A necessary and sufficient condition for <x,z,y> in the range $z \varepsilon (0,d_{ij})$ is the existence of a 'flip flop,' <x,ij,y> , as illustrated in (b). Furthermore, such a local center, c, is unique. Finally, it is evident from the diagram that the diametral span $d_c = S^{xy}_{(i)} + S^{xy}_{(j)} + d_{ij}$ and that the location of c is given by $d_{ci} = (d_{ij} + S^{xy}_{(j)} - S^{xy}_{(i)})/2$.

<u>ii) d(x,v) = d(y,v) for v = i or v = j but not both</u>

Assume v = i without loss of generality and consider the generic case depicted in (c). Only the point c = i can possibly satisfy <x,c,y> though information from adjacent arcs is required to confirm this. Note that $d_c = 2S^{xy}_{(i)}$.

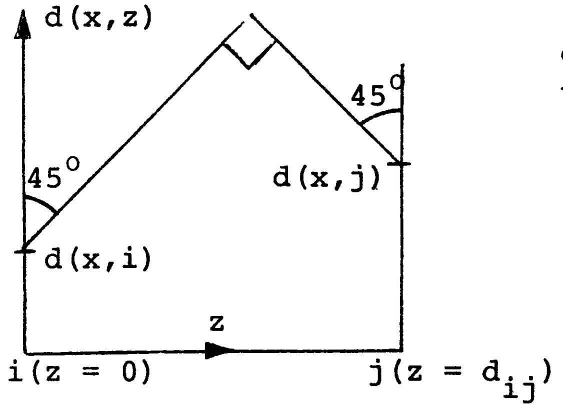
<u>iii) $d(x,v) = d(y,v)$, $v = i,j$</u>

Consider the generic case depicted in (d).

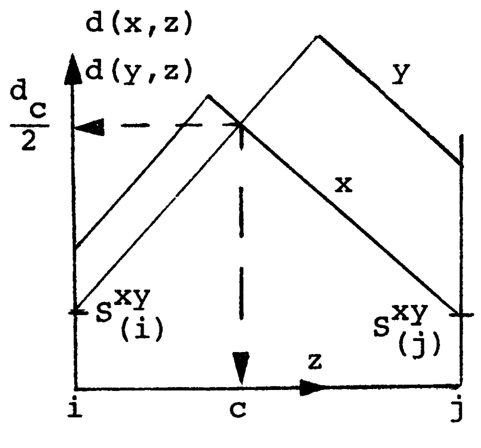
$d(x,z) = d(y,z) \quad \forall z \varepsilon [0,d_{ij}]$ and only the points i and j are potential local centers, with diametral spans $2S^{xy}_{(i)}$, $2S^{xy}_{(j)}$ respectively.

Note that $<x,ij,y>$ exists whenever $d(x,v) = d(y,v)$ for $v = i$ or $v = j$ or both, as well as in the situation illustrated in Figure A.2, thus establishing the four cases of Lemma 4.1.

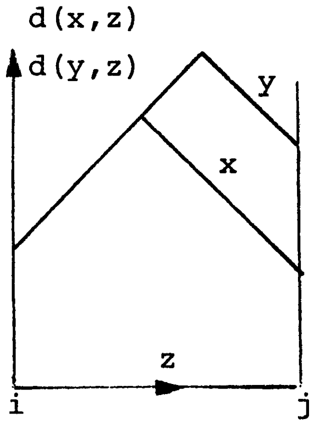
We now wish to show that nothing is lost by ignoring local centers in case (iii) providing all arcs $(ij)\varepsilon A$ are investigated. Assuming case (iii) obtains, consider the two possibilities illustrated in (d) and (e). Without loss of generality consider node i and let Δ_i denote the set of nodes adjacent to i. All arcs incident to i are in cases (ii) or (iii). In (d), $<x,i,y> \Rightarrow \exists \delta \varepsilon \Delta_i \ni (i\delta)$ is in case (ii). The contrary $\Rightarrow x = y (= i)$ which, by assumption, is impossible. In (e), $<x,i,y>$ does not exist. In conclusion, to generate C_{xy} it suffices to inspect all arcs $(ij)\varepsilon A$ in cases (i) and (ii) alone. ||



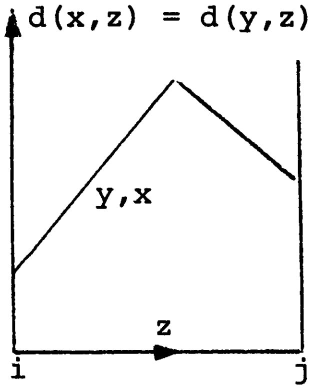
(a) - Form of d(x,z)



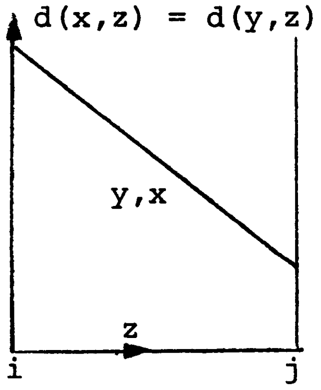
(b) - Case (i)



(c) - Case (ii)



(d) - Case (iii)



(e) - Case (iii)

Figure A.1: Local Center Conditions

APPENDIX B

RELAXATION ALGORITHM EXAMPLES

Two examples are solved with the relaxation algorithm of section 4.3.2. Note that the test $2\ell(X^z) \overset{?}{>} d$ has been omitted from this earlier version of the algorithm.

Example B.1: P/N/m/T, m = 1,2,...,7

The tree network with 60 nodes is shown in Figure B.1. The example illustrates how virtually any tree problem can be solved manually. We shall let x_{xy} denote the mid-point of the path linking x to y and, for compactness, shall also refer to x_{xy} as xy in labelling columns of CP matrices.

m = 1

Use the algorithm of Chapter II for P/N/1/T to obtain

$X^1 = x_{ab}$, $\underline{\underline{D_1 = 46}}$.

m = 2

Use the algorithm of Chapter II for P/N/2/T to obtain

$X^2 = \{x_{bc}, x_{ad}\}$, $\underline{\underline{D_2 = 34}}$.

m = 3

Initiate the relaxation algorithm, arbitrarily, with the following upper bound solution derived by breaking up the maximal span in X^2:

$$X^3 = \{x_{bc}, x_{ae}, x_{df}\} \text{ , } d = 30 \text{ , } R = \{a,b,c,d,e,f\}$$

which 'covers' all nodes in N - R . Update ${}^{d}C_R$ to obtain the

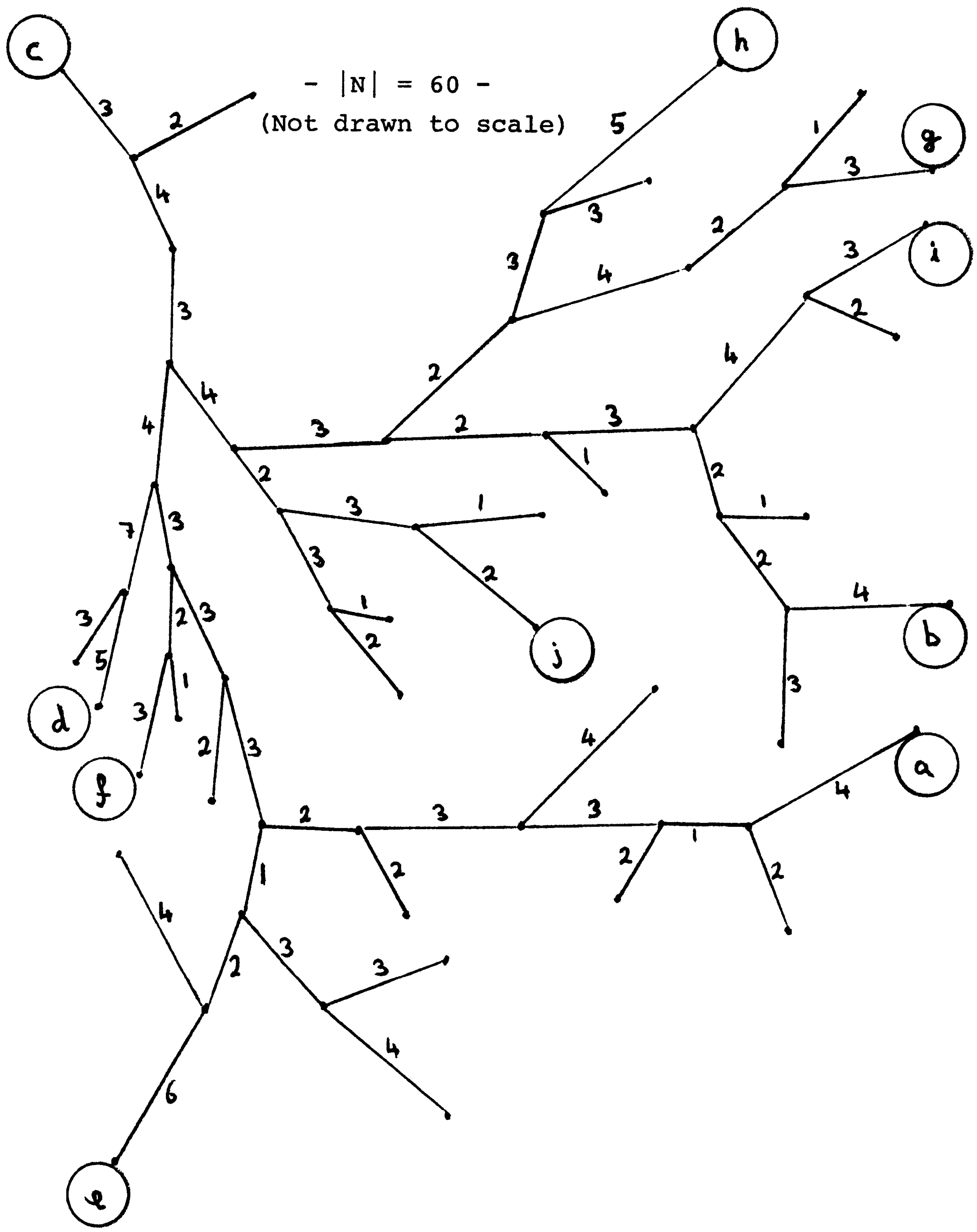


Figure B.1: Network for P/N/m/T Example

Numbers represent arc lengths.

Encircled node labels a,b,...,j represent successive additions to R.

following CP matrix:

$$\begin{array}{c|cccccc|cccccc|}
 & ae & af & cd & cf & df & ef & a & b & c & d & e & f \\
\hline
a & 1 & 1 & & & & & & & & & & \\
b & & & & & & & & & & & & \\
c & & & 1 & 1 & & & & & & & & \\
d & & & 1 & & 1 & & \multicolumn{6}{c|}{I_6} \\
e & 1 & 1 & & & & 1 & & & & & & \\
f & & 1 & 1 & 1 & 1 & 1 & & & & & & \\
\hline
 & & \bullet & \bullet & & & & & \bullet & & & & \\
d_c & [22 & 24 & 26 & 22 & 20 & 20 & 0 & \multicolumn{4}{c}{\longleftrightarrow} & 0]
\end{array}$$

Note that the objective function is $\sum x_j$. The row vector representing diametral spans, d_c, is included for subsequent column eliminations.

Obtain by inspection, $z = 3$, $\hat{X}^3 = \{x_{af}, x_{cd}, b\}$, $\hat{d} = 26$. $2\ell(\hat{X}^3) = 2d(g,\hat{X}^3) > 26$. Adding g to R and updating ${}^{d}C_R$ obtain the following enlarged CP matrix:

$$\begin{array}{c|cccccccc|ccccccc|}
 & ae & af & cd & cf & df & ef & gb & gc & a & b & c & d & e & f & g \\
\hline
a & 1 & 1 & & & & & & & & & & & & & \\
b & & & & & & & 1 & & & & & & & & \\
c & & & 1 & 1 & & & & 1 & \multicolumn{7}{c|}{I_7} \\
d & & & 1 & & 1 & & & & & & & & & & \\
e & 1 & 1 & & & & 1 & & & & & & & & & \\
f & & 1 & 1 & 1 & 1 & 1 & & & & & & & & & \\
g & & & & & & & 1 & 1 & & & & & & & \\
\hline
 & & \bullet & \bullet & & & & \bullet & & & & & & & & \\
d_c & [22 & 24 & 26 & 22 & 20 & 20 & 24 & 28 & 0 & \multicolumn{5}{c}{\longleftrightarrow} & 0]
\end{array}$$

Obtain by inspection, $z = 3$, $\hat{X}^3 = \{x_{af}, x_{cd}, x_{gb}\}$, $\hat{d} = 26$.

Since $2\ell(\hat{X}^3) = 26$ we have an improved global solution with $d = 26$. Attempting to improve upon this, eliminate columns with $d_c \geq 26$ thus obtaining the following CP matrix:

$$\begin{array}{c|cccccc c|}
 & ae & af & cf & df & ef & gb & a, b, c, d, e, f, g \\
\hline
a & 1 & 1 & & & & & \\
b & & & & & & 1 & \\
c & & & 1 & & & & \\
d & & & & 1 & & & I_7 \\
e & 1 & 1 & & & 1 & & \\
f & & 1 & 1 & 1 & 1 & & \\
g & & & & & & 1 & \\
\hline
 & & \cdot & \cdot & \cdot & & \cdot & \\
d_c & [22 & 24 & 22 & 20 & 20 & 24 & 0 \longleftrightarrow 0]
\end{array}$$

We find $z = 4$, $\hat{X}^4 = \{x_{af}, x_{cf}, x_{df}, x_{gb}\}$, $\hat{d} = 24$. Hence, the incumbent solution is optimal for $m = 3$, namely:

$X^3 = \{x_{af}, x_{cd}, x_{gb}\}$, $\underline{\underline{D_3 = 26}}$.

<u>m = 4</u>

Since $2\ell(\hat{X}^4) = 24$ we have an improved solution $X^4 = \hat{X}^4$ with $d = 24$. Eliminating columns with $d_c \geq 24$ the following CP matrix is obtained:

$$\begin{array}{c|cccc c|}
 & ae & cf & df & ef & a, b, c, d, e, f, g \\
\hline
a & 1 & & & & \\
b & & & & & \\
c & & 1 & & & \\
d & & & 1 & & I_7 \\
e & 1 & & & 1 & \\
f & & 1 & 1 & 1 & \\
g & & & & & \\
\hline
 & \cdot & \cdot & \cdot & & \cdot \qquad \cdot \\
d_c & [22 & 22 & 20 & 20 & 0 \longleftrightarrow 0]
\end{array}$$

We find $z = 5$, $\hat{X}^5 = \{x_{ae}, x_{cf}, x_{df}, b, g\}$, $\hat{d} = 22$. Hence, the incumbent solution is optimal for $m = 4$, namely:
$X^4 = \{x_{af}, x_{cf}, x_{df}, x_{gb}\}$, $\underline{\underline{D_4 = 24}}$.

$\underline{m = 5}$

$2\ell(\hat{X}^5) = 2d(h, \hat{X}^5) > 22$. Adding h to R and updating ${}^{d}C_R$ obtain the following enlarged CP matrix:

$$\begin{array}{c|cccccc|c|}
 & ae & cf & df & ef & hy & hb & a,\ b,\ c,\ d,\ e,\ f,\ g,\ h \\
a & 1 & & & & & & \\
b & & & & & & 1 & \\
c & & 1 & & & & & \\
d & & & 1 & & & & I_8 \\
e & 1 & & & 1 & & & \\
f & & 1 & 1 & 1 & & & \\
g & & & & & 1 & & \\
h & & & & & 1 & 1 & \\
 & \bullet & \bullet & \bullet & & \bullet & \bullet & \\
d_c & [22 & 22 & 20 & 20 & 17 & 23 & 0 \longleftrightarrow 0]
\end{array}$$

We find $z = 5$, $\hat{X}^5 = \{x_{ae}, x_{cf}, x_{df}, x_{hg}, x_{hb}\}$, $\hat{d} = 23$. Since $2\ell(\hat{X}^5) = 23$, $X^5 = \hat{X}^5$ is an improved global solution for $m = 5$ with $d = 23$. Eliminating columns with $d_c \geq 23$ and solving the revised CP we obtain $z = 5$, $\hat{d} = 22$, $2\ell(\hat{X}^5) = 2d(i, \hat{X}^5) > 22$. Adding node i to R and updating ${}^{d}C_R$ we obtain a new CP of size (9×7) (excluding null variables) yielding $z = 5$, $\hat{d} = 22$, $2\ell(\hat{X}^5) = 2d(j, \hat{X}^5) > 22$. Adding node j to R and updating ${}^{d}C_R$ we obtain an enlarged CP of size (10×10) yielding $z = 5$, $\hat{d} = 22$, $2\ell(\hat{X}^5) = 22$ so that $X^5 = \hat{X}^5$ is an improved solution with $d = 22$.

Eliminating columns with $d_c \geq 22$ and resolving the CP of size (10×7) yields $z = 6$, $\hat{d} = 21$ so that X^5 is an optimal solution with $\underline{\underline{D_5 = 22}}$.

<u>m = 6</u>

Since $2\ell(\hat{X}^6) = 21$, $X^6 = \hat{X}^6$ is an improved global solution for $m = 6$ with $d = 21$. Eliminating columns with $d_c \geq 21$ results in a CP of size (10×5) which yields $z = 7$, $\hat{d} = 20$ so that X^6 is optimal and $\underline{\underline{D_6 = 21}}$.

<u>m = 7</u>

$2\ell(\hat{X}^7) = 20$ so that $X^7 = \hat{X}^7$ is an improved global solution for $m = 7$ with $d = 20$. Eliminating columns with $d_c \geq 20$ we obtain a CP of size (10×2) yielding $z = 8$, $\hat{d} = 17$ so that X^7 is optimal and $\underline{\underline{D_7 = 20}}$.

<u>Comparative Note</u>: Minieka's algorithm (section 4.2.1), in contrast, involves generating and solving CPs with 60 rows and 1830 columns.

Example B.2: P/N/m/G, m = 1,2,3,4

The network, with 53 nodes and 81 arcs, is shown in Figure B.2. For the general graph problem the task of updating ${}^{d}C_{R}$ is tedious for a manual mode while the CPs are usually amenable to manual solution by inspection. In this example an interactive approach was adopted with successive updates of ${}^{d}C_{R}$ by computer and manual solution of CPs by inspection. Details are reproduced only for m = 1 and m = 2. Computational data for this problem appear in section 4.3.6.

m = 1

Computing first the matrix $||d(x,y)||$ we find first an upper bound solution $\ell(16) = 37$. Hence, $D_1 \leq 74$. A most distant pair of nodes is {40,53}. Initiate the algorithm by setting d = 74 and R = {40,53}. Compute ${}^{74}C_{40,53}$ to obtain the following CP matrix:

$$\begin{array}{cccccc}
 & x_{40,53}^{11,13} & x_{40,53}^{12,16} & 40 & 53 \\
40 & 1 & 1 & \multicolumn{2}{c}{\multirow{2}{*}{I_2}} \\
53 & 1 & 1 & & \\
 & \bullet & & & \\
d_c & [\ 71 & 73 & 0 & 0\ ] \\
d_{ic} & [\ 1/2 & 7/2 & 0 & 0\ ]
\end{array}$$

Where x_{xy}^{ij} is the candidate center on (ij) generated by the pair xy and d_{ic} is the distance of the center, c, from node i along (ij).

Choosing $\hat{X}^1 = x_{40,53}^{11,13}$, $\hat{d} = 71$ we find $2\ell(\hat{X}^1) = 71$ so that

- $|N| = 53$; $|A| = 81$ -

(Not drawn to scale)

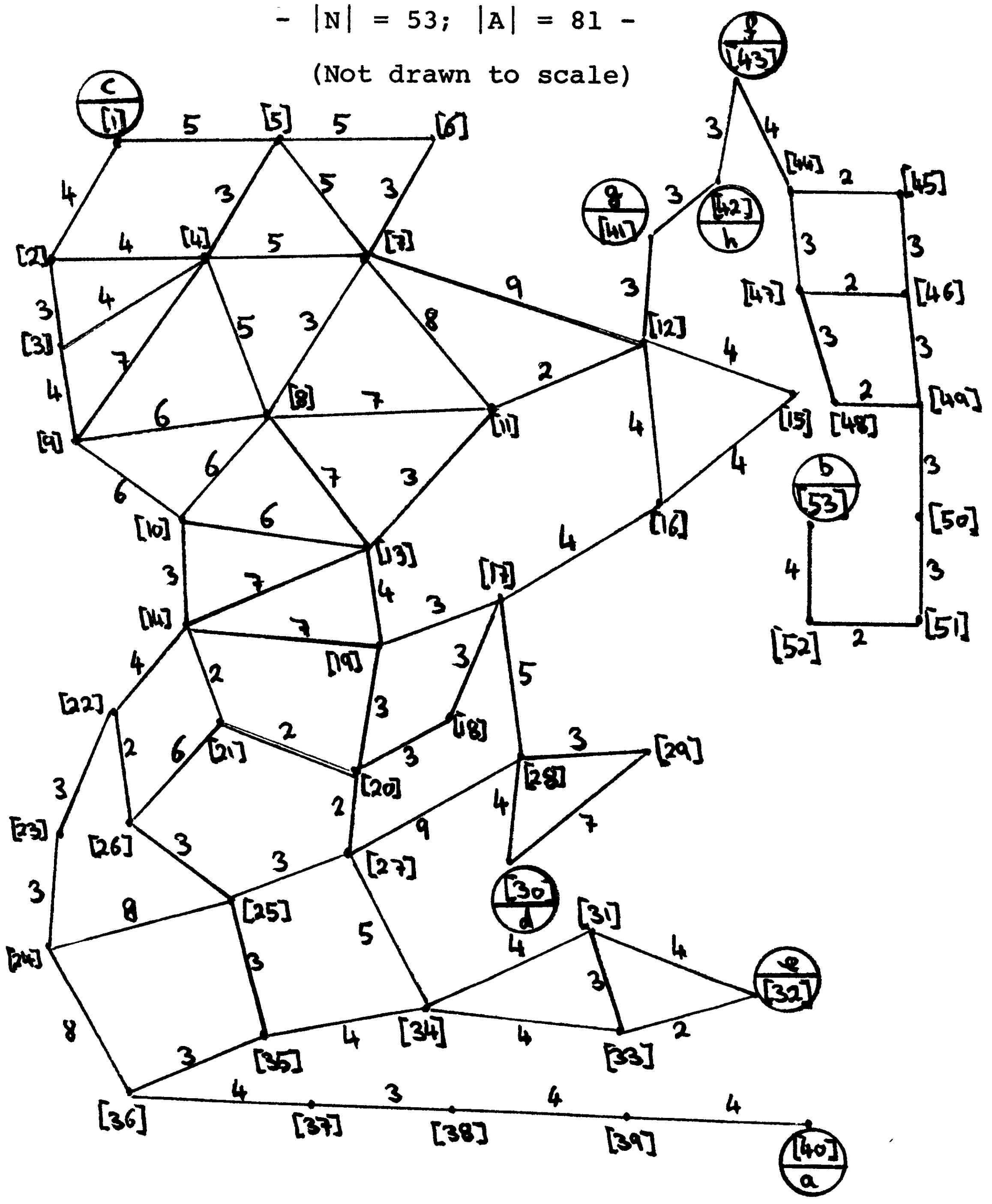


Figure B.2: Network for P/N/m/G Example

Numbers on links represent arc lengths.

Numbers in brackets identify nodes.

Encircled node labels a,b,...,h represent successive additions to R.

$X^1 = \hat{X}^1$ and $d = 71$. Clearly, no better solution exists for P/R/1/G so that the global solution is

$$X^1 = x^{11,13}_{40,53}\ , \quad \underline{\underline{D_1 = 71}}\ .$$

<u>Remark</u>: Note the ease with which the single center solution has been identified. Utilizing the technique suggested by Hakimi [20] would involve generating C_{xy} for all pairs $xy \varepsilon N \times N$.

<u>m = 2</u>

Eliminating columns with $d_c \geq 71$ reduces the previous CP matrix to the set of null centers with solution $z = 2$, $\hat{X}^2 = \{40,53\}$, $\hat{d} = 0$.

$\ell(\hat{X}^2) = d(1,\hat{X}^2) > 0$ and node 1 is appended to R. Updating d_{C_R} yields the new CP matrix:

$$\begin{array}{c|ccccccccc}
 & x^{22,26}_{40,1} & x^{20,27}_{40,1} & x^{21,26}_{40,1} & x^{42,43}_{53,1} & x^{22,23}_{40,1} & x^{27,28}_{40,1} & 40 & 53 & 1 \\
\hline
40 & 1 & 1 & 1 & 0 & 1 & 1 & & & \\
53 & 0 & 0 & 0 & 1 & 0 & 0 & & I_3 & \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & & & \\
 & & \bullet & & & & & & \bullet & \\
d_c & [50 & 50 & 52 & 52 & 54 & 65 & 0 & 0 & 0] \\
d_{ic} & [1 & 1 & 4 & 1 & 3 & 17/2 & 0 & 0 & 0]
\end{array}$$

Solving the CP yields $z = 2$, $\hat{X}^2 = \{x^{20,27}_{40,1}, 53\}$, $\hat{d} = 50$. Since $2\ell(\hat{X}^2) = 50$ we have an improved global solution $X^2 = \hat{X}^2$, $d = 50$. Eliminating all columns with $d_c \geq 50$ reduces the CP matrix to the null set. Clearly, no improvement can be made for P/R/2/G so that a global solution too is given by $X^2 = \{x_{40,1}^{20,27},53\}$ and $\underline{\underline{D_2 = 50}}$.

<u>m = 3</u>

Proceeding as before an initial solution for P/R/3/G with $d < 50$ is given by the null set and a most distant node, 30, is added to R. After five iterations an optimal solution for $m = 3$ is achieved with $R = \{40,53,1,30,32\}$ and $\underline{\underline{D_3 = 36}}$.

<u>m = 4</u>

Five iterations are needed to achieve optimality with $R = \{40,53,1,30,32,43,41,42\}$ and $\underline{\underline{D_4 = 25}}$.

BIOGRAPHICAL NOTE

Gabriel Y. Handler is on the research and teaching staff of the Flight Transportation Laboratory at M.I.T. His current research interests include combinatorial optimization, routing and scheduling in transportation networks and network location theory, and he teaches a course in airline operations analysis. Mr. Handler holds a B.Sc. in Economics and Computing from the London School of Economics and an M.Sc. in Operations Research from the Technion, Israel. His professional experience includes three years as an O.R. analyst in military systems analysis and consulting to an international airline in development of a computerized crew scheduling facility. He is a member of ORSA, ORSIS, TIMS and SIGMA XI.