A Discrete Equal-Capacity p-Median Problem.

by

Vikram Marathe

Thesis submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Master of Science

in

Industrial and Systems Engineering

APPROVED:

Dr. Hanif D. Sherali, Chairman

Dr. Konstantinos P. Triantis

Dr. Antonio A. Trani

May 15, 1992

Blacksburg, Virginia

A Discrete Equal-Capacity p-Median Problem.

by

Vikram Marathe

Dr. Hanif D. Sherali, Chairman

Industrial and Systems Engineering

(ABSTRACT)

This thesis deals with the analysis of a discrete equal-capacity p-median problem, where the costs are directly proportional to the shipping distance and the amount shipped. A mixed integer programming formulation of the unbalanced, but equal capacitated case is analyzed. First we develop a dynamic programming procedure for a p-median problem on a chain graph. In the second part we develop an algorithm to solve a p-median problem on a general network. First a heuristic algorithm is used to obtain an upper bound on the problem. Next, we obtain a lower bound on the problem by solving a Lagrangian relaxation of a reformulated problem via a conjugate subgradient optimization procedure. We obtain Benders' cuts from the above procedures and proceed to a modified Benders' approach to solve the continuous relaxation of the original problem. Finally a branch-and-bound algorithm that enumerates over the location decision variable space is used to obtain an integer optimal solution. Computational experience is provided to demonstrate the efficacy of the algorithm.

Acknowledgements

I am indebted to Dr. Hanif D. Sherali, the Chairman of my advisory committee, for his invaluable guidance and encouragement at all stages of my graduate study. He has been both mentor and friend, and I am grateful for all his help.

I am grateful to Dr. Konstantinos P. Triantis and Dr. Antonio A. Trani for their advice and very important comments. I feel very fortunate to have had an opportunity to work with them.

Finally, I would like to thank my family, without their support and encouragement, I would not have been able to accomplish what little I have.

Table of Contents

List of Tables

List of Figures

CHAPTER I

Introduction

1.1 Problem Statement

The p-median problem is a discrete optimization problem that belongs to the general class of problems traditionally referred to as location-allocation problems. These problems concern themselves with simultaneously locating a set of service facilities and satisfying the demands of a given set of customers, the objective usually being to minimize the combined cost of location and distribution. Examples of such service facilities include factories, warehouses, schools, machines, and departments within a production facility, and the model also arises in the design of transportation and information networks.

In general, the capacity of each service facility, or supply center, and the

location and demand of each customer are fixed and known. Costs to be considered include construction, production , and transportation costs. In the case of discrete problems, the facilities to be located can only be placed at a finite number of potential sites selected via some prior analysis. The most widely studied problems of this nature are: the p-median problem, the p-center problem, the uncapacitated facility location problem, and the quadratic assignment problem. In this research, we will consider the p-median problem.

The p-median problem is a special case of minisum location problems. The total transportation cost for this class of problems is obtained by summing the component transportation costs for each commodity and each demand point. Locations that minimize this total transportation cost, are called "minisum" locations. The p-median problem essentially deals with locating exactly p service facilities so as to minimize the total amortized location and transportation cost. In particular, the p-median problem considered here will assume that all the service facilities deal with a single commodity, and have equal size capacity restrictions.

1.2 Problem Formulation

The p-median problem tries to optimally locate exactly p facilities in a network, such that the demand at each of the nodes is satisfied. There are two versions possible a) Capacitated and b) Uncapacitated. In the capacitated version

each facility has a certain fixed supply to dispense. However, in the uncapacitated version, all the facilities have unlimited amount of supply available, although in reality, one typically encounters problems that do have capacity restrictions. The present thesis considers the equal-capacity case in which all the facilities have the same amount of supply, namely s units available. However, we can locate more than one facility at a site. This essentially means that we are trying to locate facilities such that at some of the nodes the available capacity is some integer multiple of s. The motivation in using such a model is that this model is fairly realistic, as mostly in practice it is easier to set up similar facilities with progressive size limits.

There are many variations of the p-median problem such as multiproduct, multiperiod (i.e. the demand varies with time) that one frequently encounters. Often, these variations lead to problems of structure similar to the basic models, and also admit similar solution procedures. Hence, the basic models are the ones that are most popularly used.

The p-median problem introduced above may be formulated as follows :

PMED: Minimize

$$\sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}x_{ij} \qquad ..1.1$$

Subject to :

$$\sum_{i=1}^{n} x_{ij} = d_j \qquad \forall\, j=1,...,n \qquad ..1.2$$

$$\sum_{j=1}^{n} x_{ij} \leq sy_i \quad \forall\, i=1,\ldots,n \qquad ..1.3$$

$$\sum_{i=1}^{n} y_i = p \qquad ..1.4$$

$$x_{ij} \leq u_j * y_i \quad \forall\, i,j=1,\ldots,n \qquad ..1.5$$

$$x \geq 0, \quad y \geq 0, \quad y \ \textit{integer} \qquad ..1.6$$

where n : number of demand points/nodes

s : supply available at each facility

d_j : demand at node j

u_j : $\min\{s,d_j\}$ gives an upper bound on x_{ij} if $y_i = 1$

p : number of supply facilities to be located

y_i : number of facilities located at node i

x_{ij} : amount shipped from facilities at site i to satisfy demand at location j

C_{ij} : cost of shipping one unit from facilities at site i to demand node j

The variables x_{ij} and y_i are the decision variables.

The Problem PMED stated above is a discrete location-allocation problem.

Given a fixed set of locations y_i, i = 1,...,n the above problem reduces to a simple transportation/allocation problem. Given a fixed set of allocations, i.e, the amount shipped to each of the nodes from the p facilities the problem reduces to p single facility location problems. Each of these can be solved separately, by locating the facility at the median location with respect to its allocation scheme as shown by Hakimi [1964,1965].

The p-median problem is known to be NP-hard; for a detailed discussion on this subject the reader is referred to Kariv and Hakimi [1979] and Garey and Johnson [1979]. However, as will be discussed in Chapter 2, the problem has been solved successfully using mathematical programming techniques, branch-and-bound procedures, and heuristics.

1.3 Objectives

The aim of this research is to develop two solution procedures for the p-median problem. The first of these is a dynamic programming algorithm applied to the p-median problem when the demand nodes are restricted to lie on a chain graph. The second method uses a modification of Benders' algorithm in a branch-and-bound framework for the p-median problem formulated on a general network.

This study is organized as follows. Chapter 2 presents a literature review on p-median problems, and on location-allocation problems. Chapter 3 details the

dynamic programming algorithm used to solve the p-median problem on a chain graph. Chapter 4 describes the Benders' method used in conjunction with a branch-and-bound algorithm. Computational experience has been presented in Chapter 5. Conclusions and suggestions for further research are presented in Chapter 6.

Chapter II

Literature Review

The p-median problem being a discrete location-allocation problem, we first discuss some discrete location-allocation problems, before proceeding to discuss direct treatments of the p-median problem itself.

2.1 Discrete Location-Allocation Problems

The discrete location-allocation problem involves a network in which the facilities may be located only at certain discrete locations in a network. The discrete capacitated location-allocation problem can be mathematically modelled as follows.

DLAP : Minimize

$$\sum_{i=1}^{m}\sum_{j=1}^{m} C_{ij}y_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{m} p_{ijk}x_{ijk} \qquad ..2.1$$

subject to :

$$\sum_{j=1}^{m} y_{ij} = 1 \quad \forall\, i=1,...,m \qquad ..2.2$$

$$\sum_{i=1}^{m} y_{ij} = 1 \quad \forall\, j=1,...,m \qquad ..2.3$$

$$\sum_{k=1}^{n} x_{ijk} = y_{ij}s_i \quad \forall\, i,j=1,...,m \qquad ..2.4$$

$$\sum_{i=1}^{m}\sum_{j=1}^{m} x_{ijk} = d_k \quad \forall\, k=1,...,n \qquad ..2.5$$

$$x_{ijk} \geq 0 \quad \forall\, i,j=1,...,m,\ \forall\, k=1,...,n \qquad ..2.6$$

$$y_{ij} \ \textit{binary} \quad \forall\, i,j=1,...,m \qquad ..2.7$$

where the decision variables are

$$y_{ij} = \begin{cases} 1 & \text{if supply center i is constructed on location site j} \\ 0 & \text{otherwise} \end{cases}$$

x_{ijk} = annual number of units produced by supply center i at site j for customer k

and where

C_{ij} = annualized cost of locating supply center i at

location j

p_{ijk} = total cost (production and transportation) of a unit produced by supply center i at site j for customer k

s_i = annual capacity of supply center i

d_k = annual demand of customer k

The new facilities are restricted to certain discrete location sites by constraints (2.2), (2.3) and (2.7). Constraint (2.4) ensures that whenever $y_{ij} = 0$, the corresponding x_{ijk}'s are zero, and when $y_{ij} = 1$, the sum of x_{ijk} equals the supply at facility i. Constraints (2.5) and (2.6) ensure that the customer demands are satisfied, and the amounts shipped are nonnegative.

This particular formulation is tackled in Sherali and Adams [1984]. They use a modified Benders' approach embedded within a branch-and-bound framework to solve the above problem. The problem is decomposed into a master problem and a subproblem. The continuous relaxation of the master problem is solved using Benders' decomposition in order to derive a lower bound on the problem, this is also used to derive a strongest surrogate constraint for the purpose of conducting logical tests. The solution of each subproblem gives an upper bound as well as a Benders' cut which cuts off the previous solution. On completion of solving the continuous relaxation, a branch-and-bound algorithm is used to obtain the integer optimal solution. The present thesis utilizes their technique in order to solve the p-median problem.

Christofides and Beasley [1983] use a different formulation of the capacitated location-allocation problem, which they solve using Lagrangian relaxation. Here, instead of using the location variables in a separate assignment like structure, or permitting the location of various capacitated facilities at any location as we do, their location variables simply indicate whether or not a facility is opened at a given location. These facilities can be located only at certain predefined potential warehouse locations. If a facility is opened at a given site, then the total shipment out of that location is bounded from above by a capacity, and is bounded from below by a minimum utilization factor. Moreover, they specify a range for the total number of facilities that may be located. Beasley [1988] modified this solution procedure to reduce the computational effort. First he utilizes Lagrangian relaxation to solve the continuous relaxation of the problem. The corresponding Lagrangian dual problem is solved using a subgradient optimization procedure. The problem is then solved using a binary depth-first tree search procedure, computing a lower bound at each tree node via the Lagrangian relaxation and the subgradient procedure. The problem considered in the present thesis differs in that if a site is used to locate a facility, we permit the design of the size of the facility that can be located thereon in multiples of a basic capacity size s, rather than continuously within a range. The present thesis also imposes the restriction that exactly p facilities must be located. The proposed solution procedure utilizes a conjugate subgradient procedure to solve the Lagrangian relaxation of the reformulated linear program to obtain a strongest surrogate

Benders' cut. The reformulation cuts off some of the non-integer linear programming relaxation region to obtain a tighter representation of the original mixed integer problem.

Some of the other papers in the area of capacitated location-allocation problems include the work of Jacobsen [1983] in which the author has extended some of the well known heuristic techniques used in the uncapacitated plant location problems to the capacitated facilities model, and the work of Van Roy [1981, 1986] who has applied the cross decomposition technique for mixed integer programming problems to facility location problems. In this approach he uses a Benders' decomposition method. This implementation requires only one full cross-decomposition iteration and does not require the solution of the Benders' master problem. It alternates between solving Benders' subproblems and Lagrangian subproblems. This technique has been successful in solving large-scale capacitated plant location problems.

Bartezzaghi et al [1981] have applied a tree search algorithm to solve capacitated location problems. They derive a lower bound by solving a transportation problem. Bitran et al [1981] have proposed an inverse optimization technique using both Lagrangian and group theoretic techniques. The reader is also referred to Magnanti and Wong [1990] for a further discussion on various decomposition methods applied to facility location problems.

The uncapacitated facility location problem has been studied very extensively.

For a detailed discussion and an up-to date survey the reader is referred to Cornuejols, Nemhauser, and Wolsey [1990].

2.2 Continuous p-Median Problems

The earlier formulations of the p-median problem, starting with the work of Kariv and Hakimi [1979], restricted the demand to occur only at the nodes of the network. However, in problems involving the location of emergency or public service or utility stations, one must allow for demands to occur continuously on links as well. The main difference here from the facility location problems involves the restriction that exactly p supply centers be located.

Handler and Mirchandani [1979] formulate such a problem and then approximate it by replacing the continuous demand on each link with a concentrated centroidal demand point. Alternatively, one can obtain a closer approximation by using several concentrated demand points to replace the continuous demands on each link. Minieka [1977] adopts a non-discretized approach to this problem. However, here, the demand on a link is said to be served by travelling to the furthest end of the link. This is called a general absolute median of the network. Slater [1981] considers a similar type of problem, where a tree network is specified in which the demand is characterized by a collection of subtrees. A facility is said to serve a demand subtree by travelling in the network to the closest point in that subtree. Chiu

[1987] considers a single facility, and the average distance function to be minimized is characterized as a function of the location of this facility.

Cavalier and Sherali [1986] consider minisum location-allocation problems on undirected networks in which demands can occur on links with uniform probability distributions. Two types of networks are considered, namely, a chain graph and a tree network. But these approaches consider only the case of the uncapacitated facilities.

Sherali and Nordai [1988a] analyze the capacitated p-median location-allocation problem on a chain graph, in which there may exist a continuum of demands on the links characterized by some demand distribution or density functions. Sherali and Nordai [1988b] extend the above problem to locating an absolute 2-median on an undirected tree network. This paper, however, assumes a balanced network, i.e., a network in which the total supply is equal to the total demand. Sherali and Rizzo [1991] consider the unbalanced case for p-medians on a chain graph. Two unbalanced cases of this problem are considered, namely, the over capacitated case when the total supply exceeds the total demand, and the deficitly capacitated case when the total supply is less than the total demand.

Cavalier and Sherali [1985] consider multiperiod minisum location problems on trees with continuous link demands, in which multiple facilities need to be located, but due to budgetary constraints or some other factor, at most one facility can be built per time period. Thus, the facilities must be located in a sequential fashion. They also allow demands to change dynamically over the time periods.

However, their paper considers only the uncapacitated case. Sherali [1991] provides the analysis for a capacitated case, in which one additional capacitated facility is located in each of the p specified periods. The paper addresses two types of objective criteria, the first is a myopic strategy in which the present period cost is minimized sequentially for each period, and the second is a discounted present worth strategy. In general, sequential and dynamic location-allocation problems involve time dependent components, information, and decisions. Typically, the demand varies in a given manner with respect to time.

The next section deals with p-median problems in which the demand occurs only at the nodes in the network. The present thesis addresses this discrete version of the p-median problem.

2.3 The Discrete p-Median Problem

The available approaches for solving the p-median problem can be classified into the following categories

1) Graph-theoretic
2) Heuristic
3) Mathematical Programming

Other than these three, one can also use total enumeration to enumerate all possible solutions to obtain an optimal solution. Although this approach is not very practical

for most real problems, it may be the best solution technique to use if the problem size is relatively small.

2.3.1 Graph-Theoretic Procedures

Graph-theoretic approaches take advantage of the special network structure to determine the p-medians. Goldman [1971] presents an algorithm for the 1-median problem. This algorithm, called the "Majority algorithm", finds the 1-median of any given tree network, and is also applicable when the tree network has probabilistic link lengths. Mirchandani and Oudjit [1980] explore solution procedures for solving the p-median problem for both deterministic and probabilistic tree networks. They present two algorithms, the "Improved link-deletion" algorithm for the deterministic case, and the "Selective enumeration" algorithm for the probabilistic case.

For tree networks the Link-deletion algorithm requires a total of $O(n^2)$ operations. In case the distance matrix for the tree network is not given, we can compute the distance matrix in $O(n^2)$ operations. Thus the overall complexity still remains $O(n^2)$.

2.3.2 Heuristic Approaches

Heuristic methods rely on intuitive trial and error methods, but these cannot

guarantee an optimal solution. However, such procedures are usually applied to any general network structure. Heuristic procedures are especially important when one can accept any "good quality" solution and optimality is not essential, or when the problem size is much too large to practically obtain an optimal solution.

Some of the important heuristic procedures include the "Node partitioning scheme" of Maranzana [1964], the "Greedy or Myopic strategy" proposed by Kuehn and Hamburger [1963], and the "Node substitution" procedure of Teitz and Bart [1968]. For a review of these methods, the reader is referred to Handler and Mirchandani [1979].

2.3.3 Mathematical Programming Techniques

Mathematical programming approaches are generally based on an integer programming formulation of the p-median problem. Since there are many integer programming routines available, and as there is a large base of theoretical research in Mathematical programming, this approach has attracted a great deal of scrutiny. The mathematical formulation for the p-median problem can be divided into two groups, the capacitated and the uncapacitated formulations.

The p-median formulation as given in Section 1.2 is for the capacitated version. Here we give one possible problem formulation for the uncapacitated case.

Minimize

$$\sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}x_{ij} \qquad ..2.8$$

Subject to

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad \forall\, i=1,...,m \qquad ..2.9$$

$$y_j \geq x_{ij} \qquad \forall\, j=1,...,n; i=1,...,m \qquad ..2.10$$

$$\sum_{j=1}^{n} y_j = p \qquad ..2.11$$

$$x, y \text{ binary} \qquad ..2.12$$

where

$$x_{ij} = \begin{cases} 1 & \text{if the demand at node i is serviced by facility at node j} \\ 0 & \text{otherwise} \end{cases}$$

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

and where there are n demand nodes and m potential sites. As can be seen, this problem statement does not allow for capacitated facilities. Further, note that each facility either serves the demand at a given node completely or does not serve that

particular node at all. In this formulation, it is possible that some facilities serve a lot more of the demand than others, leading to overburdening of certain facilities.

We now discuss some of the algorithms developed for the uncapacitated p-median problem. ReVelle and Swain [1970] and Scharge [1975] have used linear programming (LP) relaxations, in which they relax the integrality requirements in Equation (2.12) above. The resultant linear program may give integer solutions. It has been shown by Oudjit [1981] that the LP relaxation always gives an optimal solution for the class of simple networks such as a line network that consists of a single simple path, single cycle networks, or a forest having such components. The main problem, however, with these methods is that for general networks, the solution to the LP relaxation may turn out to be non-integer.

Cornuejols et al. [1977] present a two phase approach for generating and verifying near-optimal solutions for the uncapacitated p-median problem. In the first phase a greedy-interchange heuristic generates good upper bounds on the optimal solution value. In the second phase they obtain a Lagrangian relaxation by dualizing the assignment constraints (2.9) given above. Narula et al.[1977] have also successfully used the Lagrangian relaxation approach to obtain lower bounds for the p-median problem.

Galvão [1980] modified the dual based approach for the uncapacitated facility location problem proposed by Erlenkotter, to solve the p-median problem. He applied the heuristic ascent procedure to the dual problem obtained by relaxing the

assignment constraints (2.9) and the p-median constraint (2.11) above. The lower bounds generated by the dual problem were then used in a branch-and-bound solution procedure. Mavrides [1979] developed another dual approach in which he used Lagrangian relaxation by dualizing only the p-median constraint.

2.4 Variants of the p-Median Problem

The p-median problems discussed in the earlier two sections consider mostly uncapacitated facilities. Only a few studies of the continuous problem consider capacitated facilities. The present thesis deals with the discrete capacitated p-median variant. P-median problems in various other forms have also been studied. In some variants there are additional constraints, while some other variants consider the problem for spatially stochastic demands. Some of the important variants found in the literature are discussed below.

Capacitated facilities : In this type of formulation, including the one used for this thesis, each facility has a limitation on how much demand it can serve. Levy [1967] has shown that the node optimality property holds for this case as well. Thus, one need only consider the nodes in the network for possible facility locations. There are several procedures available for the capacitated facility location problem, some of the examples being Sherali and Adams [1984], Van Roy [1981,1986], Beasley [1988] .

Capacitated flows : In the previously considered problems, there is no limit on the amount that can be shipped between any two nodes in the network. In this particular variant, however, the amount of goods or services that may be shipped over any arc is restricted by the capacity on that arc. Hakimi and Maheshwari [1972] prove that the node optimality property holds for this case as well.

Distance constraints : These variants impose additional constraints on the distance between a facility and the demand nodes serviced by that facility. For example one might have a requirement that the closest facility to each demand point must be within a specified distance from the demand node. These problems are discussed by Moon and Chaudhry [1984], and Choi and Chaudhry [1991].

Other than the above variants, there are problems which use spatially continuous demand and/or stochastic demands which were mentioned earlier in Section 2.2.

This thesis deals with the capacitated p-median problem. The problem formulation assumes the capacities to be equal as motivated earlier. In Chapter 3, a dynamic programming procedure is used to solve such a p-median problem on a chain graph. Chapter 4, then presents a procedure for the p-median problem on a general network. First, a location-allocation interchange heuristic is used to calculate an upper bound on the objective value. Next, a reformulation linearization technique as given in Sherali and Adams [1988,1990] is used to tighten the linear-programming relaxation of the problem. However, as the size of this reformulated problem is

prohibitively large, Lagrangian relaxation is used to handle the resulting formulation. The resulting Lagrangian dual problem is then solved using a conjugate subgradient optimization procedure. This gives a strongest surrogate Benders' cut which is added to the Benders' master problem. From here on, Benders' procedure is used to solve the master problem with the integrality constraints relaxed. Finally, a branch-and-bound procedure is used to obtain an optimal integer solution.

Chapter III

Dynamic Programming Procedure

This chapter presents a dynamic programming based solution procedure to solve an equal-capacity p-median problem on a chain graph with demands located at the nodes. The proposed dynamic programming algorithm is developed for two cases, namely, the over-capacitated case in which the total supply exceeds the total demand, and the deficitly-capacitated case in which the total supply is less than the total demand. It has been shown by Sherali and Rizzo [1991], that for both the over-capacitated and the deficitly-capacitated cases, the p-median problem on a chain graph is NP-hard. As shown by Sherali and Nordai [1988 a] the balanced version of this problem is polynomially solvable.

3.1 Problem Statement

We begin by presenting the problem statement for the balanced case, along with certain optimality conditions. The next two sections then give the problem statement and the dynamic programming algorithm for the over-capacitated case and the deficitly-capacitated case.

Let G(N,A) be a chain graph, where N is the set of some n vertices or nodes, and A is the set of (n-1) undirected links or arcs l(i,j) connecting adjacent node pairs i and j, in N. Let this chain graph be represented as a closed, finite interval [0,c] on the real line. Hence, the shortest distance between any two points P and Q on G, and therefore on [0,c], is given by d(P,Q) = | P-Q |. Thus we can calculate the total distance matrix. In the formulation for the balanced case that follows it should be noted that the total demand equals the total available supply (p*s). This particular formulation can be very easily solved by applying the PFL allocation scheme and using median locations as shown by Sherali and Nordai [1988 a]

Accordingly, this problem can be stated as

CP : Minimize

$$\sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}x_{ij} \qquad ..3.1$$

subject to :

$$\sum_{i=1}^{n} x_{ij} = d_j \qquad \forall\, j=1,...,n \qquad ..3.2$$

$$\sum_{j=1}^{n} x_{ij} = sy_i \quad \forall\, i=1,\ldots,n \qquad ..3.3$$

$$\sum_{i=1}^{n} y_i = p \qquad ..3.4$$

$$x \geq 0, \quad y \geq 0, \quad y \text{ integer} \qquad ..3.5$$

where C_{ij} represents the cost of shipping a unit of the product from a facility located at node i to a demand point j. The decision variable x_{ij} represents the number of units shipped from facility i to node j. The decision variable y_i is 1 whenever a facility is located at node i, and is 0 otherwise. The quantity p is the number of facilities to be located. The parameter d_j denotes the demand in units at demand node j, and s denotes the supply available in number of units at each facility.

Note that we are considering the equal-capacity case in which all the facilities have equal s supply units. However, as suggested by Sherali and Rizzo [1991], we can use the procedures given below and extend them to problems with different supply values. This can be achieved by arranging the facilities in a sequence and solving this resultant problem using the dynamic programming procedure given in this chapter. However, one must apply this technique to all possible distinct permutations of supply values. Thus, if $s_1,\ldots,s_p$ denote the supply values and r is the number of distinct supply values, and if p_i is the number of facilities having the same i^{th} capacity value, then we need to solve $p!/p_1!*p_2!*\ldots*p_r!$ problems, to find an optimal solution.

The dynamic programming procedure given here uses the fact that, given an

allocation scheme, the problem is trivially solved by locating each facility at a median location with respect to its allocation scheme. The above formulation locates the facilities only at nodes of the chain graph. However, Levy [1967] has shown that there exists an optimal solution, for which all the p facilities could be located at the nodes in the network. Furthermore, from Sherali and Nordai [1988 a], for the balanced case, we can state the following theorem

Theorem 1. There exists an optimal solution to Problem CP, for which

(a) each facility i serves an interval $[\gamma,\beta]$, where γ and β are two nodes in G(N,A), and the location $y_i \in [\gamma,\beta]$, where the demand at nodes γ and β may be fully or partially supplied by the facility at i, and such that

(b)

$$\sum_{j=\gamma}^{\alpha-1} x_{ij} \leq D_i / 2$$

$$\sum_{j=\alpha+1}^{\beta} x_{ij} \leq D_i / 2$$

where α denotes the node at which y_i has been located, and D_i denotes the demand in the interval $[\gamma,\beta]$ served by the facility i.

Thus, part (a) above asserts that, there exists an optimal set of allocation functions, such that the demand regions served by facilities are all intervals with pairwise disjoint interiors, and appear from left to right in the same order as do the facilities themselves. Sherali and Nordai [1988 a] call this as the "Pack it From the

Left" or PFL allocation for each of the facility.

Part (b) of the theorem, simply states that each facility must be located at a median location with respect to its allocation scheme. Handler and Mirchandani [1979] show that the location $y_i \in [0,c]$ is an optimal location for facility i if and only if it is a median location with respect to its allocation function. The above theorem will be shown to be applicable for both the over-capacitated as well as the deficitly-capacitated case. For the solution procedure of this case the reader is referred to Sherali and Nordai [1988 a]. Further, note that the balanced case can be regarded as a special case of either the over-capacitated or the deficitly-capacitated cases discussed in the following sections, thus, we can also employ the solution procedures outlined for either of these cases.

3.2 Over-Capacitated Problem

The problem formulation for the over-capacitated case differs from the balanced case only in Equation (3.3), and can be written as follows.

OLAP: Minimize

$$\sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}x_{ij} \qquad ..3.6$$

subject to

$$\sum_{i=1}^{n} x_{ij} = d_j \qquad \forall\, j=1,...,n \qquad ..3.7$$

$$\sum_{j=1}^{n} x_{ij} \leq sy_i \quad \forall\ i=1,...,n \qquad ..3.8$$

$$\sum_{i=1}^{n} y_i = p \qquad ..3.9$$

$$x_{ij} \leq u_j * y_i \quad \forall\ i,j=1,...,n \qquad ..3.10$$

$$x \geq 0, \quad y \geq 0, \quad y \text{ integer} \qquad ..3.11$$

Thus, in this case, the demand at each node will be completely satisfied as given by Equation (3.7). However, certain supply facilities will have excess supply remaining unused.

3.2.1 Dynamic Programming Formulation

For problem OLAP given above, Sherali and Rizzo [1991] have shown that the PFL allocation scheme remains valid. Hence, an optimal allocation scheme is to dispense the supplies of the facilities from left to right, one at a time in the same order in which the facilities are permuted from left to right although not necessarily exhausting each facility. Let each unit of demand be numbered sequentially from left to right. Thus, if the total demand is N units, then {1,...,N} indexes each unit of demand sequentially from left to right.

Let us now define stages, states, and decisions as follows :

Stage(k) : There are p stages in this formulation. Each stage k, for $1 \leq k \leq p$, corresponds to a situation where, k facilities are remaining to be located to the right of the last facility located. Thus, at stage k, some (p-k) facilities are assumed to have been located.

State(s_k) : A state s_k at stage k denotes the indexed demand in {1,...,N} such that demand $\{s_k,...,N\}$ is to be supplied by the k remaining facilities and the demand $\{1,...,s_k-1\}$ has been supplied by the (p-k) facilities that have already been located to the left of these k facilities.

Decision(δ_k): A decision δ_k at stage k and for any state s_k denotes the indexed demand in {1,...,N} such that demand $\{s_k,...,\delta_k\}$ is served by the facility that is being currently located.

Let Δ denote the difference between the total supply and the total demand given by,

$$\Delta = p*s - N$$

We now present a state-decision space reduction technique that can be used for this problem.

3.2.2 State-Decision Space Reduction

For the over-capacitated case, we will only consider the cases where $s < N$

and $p < N$. Both of these conditions are in fact reasonable as

1) The condition $s \geq N$, is nothing but the uncapacitated case, which can be solved polynomially as shown by Kariv and Hakimi [1979], and Hassin and Tamir [1991].

2) For the case when $p \geq N$, we can locate one facility for every unit of demand at each of the nodes. Then each facility is assumed to supply one unit of demand at that node only.

The following steps detail the state-decision space reduction process

Step 1 : For stage 1, i.e., for $k = 1$, we know that

$$\delta_1 = N$$

Now, we need to find the range of values that s_1 can take. Let

$$s_1 \in \{L_1,...,M_1\}$$

denote the range of values for state s_1. Then we can show that

$$L_1 = \max \{p , N-s+1\}$$

Obviously, the leftmost demand from which this facility can start serving is $(N-s+1)$. However, if $(N-s+1) < p$ then this facility need supply at most the demand from $\{p,...,N\}$, since the $(p-1)$ demand units remaining to the left can be supplied by the earlier $(p-1)$ facilities. If the current facility were to serve the demands indexed less than $(p-1)$, then this case would be suboptimal, since there is no fixed charge associated with locating a facility, and one can always locate all the facilities with each serving at least one demand unit. Similarly,

$$M_1 = \min \{N-s+1+\Delta , N\}$$

That is, the rightmost demand from which point onward the current facility can serve demand is $(N-s+1+\Delta)$; which is same as $(p-1)*s+1$, and occurs if all the earlier facilities use all of their available supplies and the remaining demand is served by the current facility. However, clearly if $(N-s+1+\Delta) > N$ then the rightmost demand unit from which this facility can serve is N.

Step 2 : For stages k, such that $2 \leq k \leq p$, we have

$$\delta_k \in \{L_{k-1}-1,...,M_{k-1}-1\}$$

based on the fact that the current facility must serve demand upto $(s_{k-1}-1)$ for some state variable at stage (k-1), i.e. upto the demand immediately to the left of demand from which point onwards, the remaining k-1 facilities serve.

For the range of the state variables at stage k, we have,

$$s_k \in \{L_k,...,M_k\}$$

Here,

$$L_k = \max\{p-k+1,N-ks+1\}$$

gives the leftmost demand from which the current facility can start serving. This is $(N-ks+1)$ when the current facility and the facilities to the right of it serve their full supplies. However, if $(N-ks+1) < (p-k+1)$, then we need only consider demand from $(p-k+1)$ onwards, as the other p-k demand units will be served by the previous p-k facilities. Also,

$$M_k = \min\{N-ks+1+\Delta,M_{k-1}-1\}$$

since this facility can start serving from as far to the right as (N-ks+1+Δ), which is the same as (p-k)*s+1, and occurs if all the previous (p-k) facilities use up their supplies totally. However, if (p-k)*s+1 > M_{k-1}-1 then M_k must be equal to M_{k-1}-1, as M_{k-1} is the rightmost demand point from which point onward the $(k-1)^{st}$ facility must start serving. Also, since we are going to locate all the facilities, the state variable S_p must be equal to 1.

The range of values allowed for state and decision variables have been summarized in Table 3.1. The discussion on (a) locating a facility, given its state and

Table 3.1 The state-decision space for the over-capacitated problem

For k = 1,

$$\delta_1 = N$$

$$s_1 \in \{ L_1,...,M_1 \}$$

where $L_1 = \max \{ p , N-s+1 \}$

$M_1 = \min \{ N-s+1+\Delta , N \}$

or $M_1 = \min \{ (p-1)s+1 , N \}$

For $2 \leq k \leq p$,

$$\delta_k \in \{ L_{k-1}-1,...,M_{k-1}-1 \}$$

$$s_k \in \{ L_k,...,M_k \}$$

where $L_k = \max \{ p-k+1 , N-ks+1 \}$

$M_k = \min \{ N-ks+1+\Delta , M_{k-1}-1 \}$

or $M_k = \min \{ (p-k)s+1 , M_{k-1}-1 \}$

Note: For k = p,

$$s_p = 1$$

decision variables, (b) the cost computation given the location and allocation for each of the facilities, and finally, (c) the complexity analysis will be given in Section 3.4, following the discussion on the deficitly-capacitated case.

3.3 Deficitly-Capacitated Problem

The problem formulation for the deficitly-capacitated case is as given below

DLAP : Minimize

$$\sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}x_{ij} \qquad ..3.12$$

Subject to

$$\sum_{i=1}^{n} x_{ij} \leq d_j \qquad \forall\, j=1,...,n \qquad ..3.13$$

$$\sum_{j=1}^{n} x_{ij} = sy_i \qquad \forall\, i=1,...,n \qquad ..3.14$$

$$\sum_{i=1}^{n} y_i = p \qquad ..3.15$$

$$x_{ij} \leq u_j * y_i \qquad \forall\, i,j=1,...n \qquad ..3.16$$

$$x \geq 0, \quad y \geq 0, \quad y \text{ integer} \qquad ..3.17$$

In this case, the facilities being located are required to dispense with their total

supplies. However, certain unserved demand remains. Once again, for the proof of validity of the PFL allocation scheme as pertaining to the deficitly-capacitated problem, the reader is referred to Sherali and Rizzo [1991]. This means that in an optimal allocation scheme, each facility serves the demand over some contiguous subinterval of [0,c], with some unused demand being left over. Thus, for any given allocation scheme, i.e., given state and decision variables as described earlier, the corresponding facility must lie at a node, within the interval given by the nodes for its state and decision variables. Here, the stages, states, and the decision variables are defined as in Section 3.2.2. The following section discusses valid ranges for the state and decision variables to give the reduced state-decision space.

3.3.1 State-Decision Space Reduction

In this case, since the total demand is greater than total supply, we have

$$\Delta = N - p^*s > 0$$

and therefore $p < N$ and $s < N$. Also, note that since each facility uses up its full supply, given a possible decision δ_k, the accompanying state must be given by $s_k = \delta_k - s + 1$.

Step 1 : For $k = 1$, the last facility may serve demand upto p^*s if the facilities continue serving from 1 without any break. On the other hand it can serve up to N, if there is an unserved demand of Δ prior to this. Hence,

$$\delta_1 \in \{ p*s,...,N \}$$

and so,

$$s_1 \in \{ p*s-s+1,...,N-s+1 \}.$$

Step 2 : For stages k such that $2 \leq k \leq p$, the decision variables can range from

$$\delta_k \in \{ (p-k+1)*s,...,N-(k-1)*s \}$$

which is actually same as the range for s_{k-1} shifted down by 1, and the range for s_k is given by

$$s_k \in \{ (p-k)*s+1,...,N-k*s+1 \}$$

Also, for k = p, we get

$$\delta_p \in \{ s,...,N-(p-1)*s \}$$

which is the same as

$$\delta_p \in \{ s,...,\Delta+s \}$$

and

$$s_p \in \{ 1,...,N-P*s+1 \}$$

or

$$s_p \in \{1,...,\Delta+1 \}$$

These results have been summarized in Table 3.2.

Table 3.2 The state-decision space for the deficitly-capacitated problem

For k = 1,

$$\delta_1 \in \{ N-\Delta,...,N \}$$

$$\delta_1 \in \{ p*s,...,N \}$$

$$s_1 \in \{ p*s-s+1,...,N-s+1 \}$$

$$s_1 \in \{ (p-1)*s+1,...,N-s+1 \}$$

For $2 \leq k \leq p$,

$$\delta_k \in \{ (p-k+1)*s,...,N-(k-1)*s \}$$

$$s_k \in \{ (p-k)*s+1,...,N-k*s+1 \}$$

Note: For k = p,

$$\delta_p \in \{ s,...,\Delta+s \}$$

$$s_p \in \{ 1,...,\Delta+1 \}$$

3.4 Solution Technique

This section deals with the actual application of the dynamic programming algorithm. Sections 3.2.2 and 3.3.1 have already dealt with the determination of a reduced state-decision space for the over-capacitated and the deficitly-capacitated cases, respectively. Given particular state and decision variables values , the next task is to locate each facility at the median location with respect to this allocation scheme.

This is done by finding out the index of the demand in the interval $[s_k,...,\delta_k]$ served by the current facility being located, such that no more than half this total demand lies on either side of it. This is given by evaluating

$$m_k = \lfloor (s_k + \delta_k)/2 \rfloor$$

and locating the facility at the node where the demand m_k occurs.

The cost incurred to supply the demand in the allocation scheme, from the facility location as found above, is given as

$$C_k(s_k,\delta_k) = \sum_{i=s_k}^{\delta_k} | Dis(i) - Dis(m_k) |$$

where $C_k(s_k,\delta_k)$ denotes the cost at stage k as a function of the state and decision variables, and where Dis(i) is the distance of the node where indexed demand i occurs from the origin (or node 1).

Let $f_k^*(s_k)$ represent the optimal cost incurred by supplying the demand from state s_k onwards at stage k, given k available facilities to be located. Then the

recursive formula for this DP formulation can be written as

$$f_k^*(s_k) = \underset{\delta_k \in \{dec.\ space\}}{minimum} \{C_k(s_k,\delta_k) + f_{k-1}^*(\delta_k+1)\} \quad \forall s_k \in \{L_k,...,M_k\}, k = 1,...,p \qquad ..3.18$$

We will denote by $\delta_k^*(s_k)$ the optimal decision to (3.18), given state s_k at stage k. It should be noted that Bellman's principle of optimality holds, in that given any state s_k at stage k, the optimal policy of satisfying demand to the right of s_k depends only on s_k and the decisions adopted from hereon, and is independent of how one arrived at s_k.

3.4.1 Illustrative Example

Consider a n=5 node over-capacitated problem, with demands at each of the nodes given to be $d_1 = 20$, $d_2 = 15$, $d_3 = 7$, $d_4 = 13$, $d_5 = 25$; the distance of each of the nodes from node 1 is: $Dis_1 = 0$, $Dis_2 = 3$, $Dis_3 = 7$, $Dis_4 = 16$, $Dis_5 = 20$. There are p=3 facilities to be located, each with a supply of s=28 units.

Thus,

$$n = 5,\ p = 3,\ s = 28,\ N = 80,\ p^*s = 84, \text{ and } \Delta = 4$$

Also, on applying the state-decision space reduction technique given earlier for the over-capacitated case we get,

$$\delta_1 = \{80\} \quad \text{and} \quad s_1 \in \{53,...,57\}$$

$$\delta_2 \in \{52,...,56\} \quad \text{and} \quad s_2 \in \{25,...,29\}$$

$$\delta_3 \in \{24,...28\} \quad \text{and} \quad s_3 \in \{1\}.$$

However, it must be noted that not all the combinations of the state and decision variables are feasible. Whenever a particular combination is infeasible it has been marked by a dash in the following tables.

Now we can tabulate the calculations for the DP algorithm as follows:

Stage 1

s_1	δ_1[#]	$\delta_1^*(s_1)$	$f_1^*(s_1)$
	80		
53	12 (5)	80	12
54	8 (5)	80	8
55	4 (5)	80	4
56	0 (5)	80	0
57	0 (5)	80	0

[#]The first number gives the cost in dollars, given that the facility is located at the node given in parentheses.

Stage 2

s_2	δ_2					$\delta_2^*(s_2)$	$f_2^*(s_2)$
	52	53	54	55	56		
25	146 (3)	-	-	-	-	52	146
26	142 (3)	147 (3)	-	-	-	52	142
27	138 (3)	143 (3)	148 (3)	-	-	52	138
28	134 (3)	139 (3)	144 (3)	149 (3)	-	52	134
29	130 (3)	135 (3)	140 (3)	145 (3)	158 (4)	52	130

Stage 3

s_3	δ_3					$\delta_3^*(s_3)$	$f_3^*(s_3)$
	24	25	26	27	28		
1	158 (1)	157 (1)	156 (1)	155 (1)	154 (1)	28	154

Thus, the optimal locations for the three facilities are at nodes 1, 3, and 5. The first facility serves the demands 1 through 28, the second serves from 29 through 52, and finally the third facility serves the demand 53 through 80. The total cost is found to be 154.

3.4.2 Complexity Analysis

The complexity of the proposed algorithm can be shown to be $O[p(\Delta+1)^2]$. To prove this we must show that at each stage at most $(\Delta+1)$ states and decisions are

possible. We will now consider both over-capacitated and deficitly-capacitated cases to show that this is indeed true in both cases.

a) Over-capacitated case:

Here, as discussed earlier, for k=1, we know that $\delta_1 = N$, and $s_1 \in \{L_1,...,M_1\}$

where $$L_1 = \max\{p, N-s+1\}$$

and $$M_1 = \min\{N-s+1+\Delta, N\}$$

Hence,

$$(M_1 - L_1) = \min\{N-s+1+\Delta, N\} - \max\{p, N-s+1\} \leq (N-s+1+\Delta) - (N-s+1) = \Delta$$

In a similar fashion, we can show for stages 2 through p, that

$$(M_k - L_k) = \min\{N-ks+1+\Delta, M_{k-1}-1\} - \max\{p-k+1, N-ks+1\}$$

$$\leq (N-ks+1+\Delta) - (N-k+1) = \Delta$$

Thus, the number of possible values that state and decision variables at any stage can take is at the most equal to $(\Delta+1)$. Since there are a total of p stages and a maximum of $(\Delta+1)$ states and decisions each; the order of computations for the overall algorithm is clearly $[p(\Delta+1)^2]$.

b) Deficitly-capacitated case:

For the deficitly-capacitated case for $1 \leq k \leq p$, we can write

$$\delta_k \in \{ (p-k+1)*s,...,N-(k-1)*s \}$$

Since $\Delta = N - p*s$, or $N = p*s + \Delta$, we get,

$$\delta_k \in \{ (p-k+1)*s,...,(p-k+1)*s+\Delta \}$$

Thus a total of $(\Delta + 1)$ decisions are possible at each stage.

Similarly, the state variables can take values in the range

$$s_k \in \{ (p-k)*s+1,...,N-k*s+1 \}$$

which can be alternatively written as

$$s_k \in \{ (p-k)*s+1,...,(p-k)*s+1+\Delta \}$$

Hence, once again, there are at the most $(\Delta + 1)$ possible state variable values at each stage. Thus the order for this algorithm is again $[p(\Delta + 1)^2]$.

It should be noted that we are considering a capacitated p-median problem. However, if s > N, we can treat this problem as uncapacitated as stated earlier. In this case we can use the algorithm given by Hassin and Tamir [1991] which has a complexity bound of O(pn).

CHAPTER IV

A Decomposition Procedure

In this chapter we discuss the solution procedure used to solve the p-median problem on a general network. The problem formulation PMED used is as given in Chapter I. The overall solution procedure can be briefly explained as given below. Each of the components of this procedure are later discussed in detail in the following sections. First, we apply a location-allocation interchange heuristic to obtain a feasible solution to the problem. The objective is to get as good an upper bound on the problem as possible. This also gives feasible solutions to the original problem which are used to obtain Benders' cuts. Next, we use a conjugate subgradient optimization algorithm on a Lagrangian relaxation of a reformulated version of the original problem. This procedure gives us two Benders' cuts as explained later. Then we apply Benders' procedure to solve the continuous relaxation of the original

problem. Here, we also add the cuts generated by the above two procedures. Finally, when the Benders' procedure completes solving the continuous relaxation, we proceed to a branch-and-bound algorithm that uses the Benders' cuts obtained earlier, to obtain an integer optimal solution to problem PMED.

4.1 Location-Allocation Interchange Heuristic

This procedure makes use of the fact that given a y feasible to the Problem PMED, Problem PMED reduces to a transportation problem, and conversely given a set of feasible allocations x, the problem of finding the p locations can simply be solved as p independent 1-median problems. Let us denote the set of feasible y's by Y and that of x's by X. We begin by locating one facility each at the nodes arranged in a decreasing order of demands. If there are more facilities remaining to be located, then this process is continued until all of the facilities have been located. Therefore, we now have a set $y \in Y$. Then, we solve the following transportation problem

$$\textit{Minimize} \qquad \sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}x_{ij}$$

$$\textit{Subject to} \qquad \sum_{i=1}^{n} x_{ij} = d_j \qquad \forall\, j=1,..,n \qquad ..4.1$$

$$-\sum_{j=1}^{n} x_{ij} \geq -s\bar{y}_i \qquad \forall\, i=1,\ldots n \qquad ..4.2$$

$$x \geq 0 \qquad ..4.3$$

This gives us some feasible allocation values x ϵ X. Given this set of allocations we try to find out an optimum location for each facility with respect to its allocation. We now solve these p 1-median problems by total enumeration. However, note that if the underlying network is known then we could use Goldman's algorithm (see Mirchandani and Oudjit [1980]).

For each facility, we first find the cost of supplying the demand in its allocation scheme from each of the nodes in the network. This is simply done by multiplying the allocation at each node by the distance of that node from the node at which the facility is being located. Thus, this gives rise to an nxn matrix of weighted costs. We then simply sum each column in this matrix to get the total cost of supplying the demand in the allocation scheme from each node. The facility is then located at the node which has the minimum cost. Let us consider an example, where n = 3, p = 2, the distance matrix is as given below

n=	1	2	3
1	0	2	5
2	2	0	7
3	5	7	0

The Distance Matrix

Let the first facility supply the demands 10, 9, and 7, units at each of the three nodes respectively. Then we can calculate the cost of supplying this demand, when the facility has been located at each of the three nodes. The cost of supplying a unit of demand is taken equal to the distance over which that unit is being shipped. These calculations have been tabulated in the next table.

Facility located at node		
1	2	3
0	20	50
18	0	63
35	49	0
53	69	113

The Cost Matrix

The last row in the above table gives the sum of the rows above it and

represents the total cost for the given allocation scheme, whenever the facility is located at each of the nodes. Hence, in this example, the given facility should be located at the first node. The procedure is then repeated to find the location of the second facility. In this fashion we locate all the facilities and go back to solving the above transportation problem. This process is continued until there is no improvement between two consecutive iterations. The duals that are obtained by solving the transportation problem are used to obtain Benders' cuts. This procedure will be explained in Section 4.2.5. Note that since we solve the above transportation problem for some y ϵ Y, we get an upper bound on the solution to Problem PMED.

4.2 Continuous Relaxation Solution

Before proceeding to Benders' decomposition procedure, we solve Lagrangian relaxations of problems CPMED and CRLT (given in the next section), where a prefix C is used to denote the corresponding continuous relaxation of the problem. This relaxation yields a lower bound that is equal to the linear programming (LP) relaxation value. The reformulation employed here is the one suggested by Sherali and Adams [1988,1990], and it serves to give a tighter LP relaxation based bound. In the next section, we discuss the reformulation of Problem PMED, appropriately modified to account for general integer restrictions rather than the 0-1 restrictions as assumed to be the case in Sherali and Adams [1988,1990].

4.2.1 Reformulation Technique

The continuous relaxation of problem PMED is as given below.

CPMED: Minimize

$$\sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}x_{ij} \qquad ..4.4$$

Subject to :

$$\sum_{i=1}^{n} x_{ij} = d_j \quad \forall\, j=1,...,n \qquad ..4.5$$

$$\sum_{j=1}^{n} x_{ij} \le sy_i \quad \forall\, i=1,...,n \qquad ..4.6$$

$$\sum_{i=1}^{n} y_i = p \qquad ..4.7$$

$$x_{ij} \le u_j * y_i \quad \forall\, i,j=1,...,n \qquad ..4.8$$

$$x \ge 0, \quad y \ge 0 \qquad ..4.9$$

In order to tighten the LP relaxation of Problem PMED, we multiply each of its constraints by y_i where i = 1,...,n, and we also multiply Equation (4.7) by x_{ij} $\forall$ (i,j). Then, we relinearize the problem by making the following substitutions :

$$w_{ijk} = x_{ij}y_k$$

$$Y_{ik} = y_iy_k$$

We add four more constraints, namely constraints (4.19) through (4.22), in order to further tighten this formulation. The variable z is used in order to partially reflect the relationship between Y and y. Also, in order to reflect, that $y_i y_k = y_k y_i$ we add the constraints (4.19). After applying this Reformulation Linearization Technique (RLT), we can write the continuous relaxation of the reformulated problem as follows.

CRLT:

$$\textit{Minimize} \quad \sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}x_{ij}$$

$$\textit{Subject to} \quad \sum_{i=1}^{n} x_{ij} = d_j \quad \forall\, j \qquad ..4.10 \ (\alpha)$$

$$-\sum_{j=1}^{n} x_{ij} \geq -s\,y_i \quad \forall\, i \qquad ..4.11$$

$$-x_{ij} \geq -u_j\,y_i \quad \forall\, (i,j) \qquad ..4.12$$

$$\sum_{i=1}^{n} y_i = p \qquad ..4.13$$

$$\sum_{i=1}^{n} w_{ijk} = d_j\,y_k \quad \forall\, (j,k) \qquad ..4.14 \ (\beta)$$

$$-\sum_{j=1}^{n} w_{ijk} \geq -s\,Y_{ik} \quad \forall\, (i,k) \qquad ..4.15$$

$$-w_{ijk} \geq -u_j\,Y_{ik} \quad \forall\, (i,j,k) \qquad ..4.16$$

$$\sum_{i=1}^{n} Y_{ik} = p\, y_k \quad \forall\, k \qquad ..4.17$$

$$\sum_{k} w_{ijk} = p\, x_{ij} \quad \forall\, (i,j) \qquad ..4.18\ (\eta)$$

$$Y_{ik} - Y_{ki} = 0 \quad \forall\, (i<k) \qquad ..4.19\ (\gamma)$$

$$y_i = \sum_{t=0}^{p} t\, z_{ti} \quad \forall\, i \qquad ..4.20\ (\delta)$$

$$Y_{ii} = \sum_{t=0}^{p} t^2 z_{ti} \quad \forall\, i \qquad ..4.21\ (\Delta)$$

$$\sum_{t=0}^{p} z_{ti} = 1 \quad \forall\, i \qquad ..4.22$$

$$x \geq 0,\ w \geq 0,\ y \geq 0,\ Y \geq 0,\ z \geq 0 \qquad ..4.23$$

This reformulated problem is very large; we have $O(n^2)$ constraints and $O(n^3)$ variables. Thus, to solve the linear programming relaxation is itself a very tedious task. Hence, we employ a Lagrangian relaxation approach to solve this linear program. In order to check whether or not the above reformulation really tightens the lower bound, we ran a number of test problems for different values of n and p. The problems were solved with GAMS using MINOS 5.1. The following Table 4.1 summarizes optimal values $v(\cdot)$, obtained for problems CPMED and CRLT. As can

be discerned from this table, CRLT affords a slightly tighter relaxation than does CPMED.

Table 4.1 Optimal values from CPMED and CRLT: A comparison

n	p	v(CPMED)	v(CRLT)
5	4	8.56	10.47
7	3	105.10	113.01
7	5	48.20	51.27
8	2	290.82	290.99
8	4	82.90	85.18
9	5	57.86	60.28
10	2	230	230

4.2.2 Lagrangian Relaxation of Problem CPMED

The Lagrangian relaxation procedure is useful for exploiting special structures inherent in a subset of the constraints, and in deriving good quality solutions. In this

approach, some of the constraints are relaxed and are accommodated within the objective function using some multipliers. In order to get an advanced start to solve Problem RLT, we first solve a Lagrangian relaxation of Problem PMED, obtained by dualizing Equation (4.5). The Lagrangian dual problem can then be written as follows.

LD1 :

$$\textit{Maximize } \{\theta(\alpha): \ (\alpha) \textit{ unrestricted}\} \qquad ..4.24$$

where

$$\theta(\alpha) = \textit{Minimum}\{\sum_i \sum_j x_{ij}(C_{ij} - \alpha_j) + \sum_j \alpha_j d_j\} \qquad ..4.25$$

$$\textit{subject to} \quad (4.6), (4.7), (4.8), (4.9)$$

where $\alpha_j \ \forall j$, denotes the duals associated with Equation (4.5). This problem is solved using a conjugate subgradient optimization technique which searches over the dual space. This technique will be discussed in Section (4.2.4). At every iteration of this algorithm we solve for $\theta(\alpha)$, given a fixed vector of α. This involves solving the following two subproblems. In the x-subproblem, since the variable y multiplies all its constraints' right-hand-sides, we drop this variable and solve the resulting problem, since the variable multiplying the right-hand-side only has the effect of proportionately scaling the resulting objective value. In the x-subproblem description, given below, we show this discarded right-hand-side variable in parentheses. We now write the x-subproblem as

x-subproblem :

$$\sum_i X_i^* = \sum_i \{ Minimum \sum_j (C_{ij} - \alpha_j) x_{ij} \} \quad ..4.26$$

$$subject\ to \quad \sum_j x_{ij} \leq s \quad (y_i) \quad \forall\ i \quad ..4.27$$

$$0 \leq x_{ij} \leq u_j \quad (y_i) \quad \forall (i,j) \quad ..4.28$$

The optimal x_{ij}^*'s obtained by solving the above problem are then multiplied by the corresponding y_i's. This then gives us an objective function in terms of y's, and we can now write the subproblem in y as follows.

y-subproblem :

$$Z_{CPMED} = minimize \sum_i X_i^* y_i + \sum_j \alpha_j d_j \quad ..4.29$$

$$subject\ to \quad \sum_i y_i = p \quad ..4.30$$

$$y \geq 0$$

During every iteration of the conjugate subgradient optimization procedure we solve $\theta(\cdot)$ by solving the above two subproblems. The value of $\theta(\cdot)$ is then given by Z_{CPMED}, which is a lower bound on the original problem. It may be noted that the problem in x's can be separated into n bounded variable knapsack problems, and the problem in y's is also a knapsack problem. The duals α, obtained at the end of this

procedure are used to initialize the duals α associated with Equation (4.10) in CRLT, and the conjugate subgradient optimization procedure for solving CRLT is then implemented using this advanced start solution.

4.2.3 Lagrangian Relaxation of Problem CRLT

Let the Lagrangian multipliers associated with different constraint blocks be as follows : α_j $\forall$j with (4.10), β_{jk} $\forall$jk with (4.14), η_{ij} $\forall$ij with (4.18), γ_{ik} $\forall$ i<k with (4.19), δ_i $\forall$i with (4.20), and finally Δ_i $\forall$i with (4.21). Dualizing these constraints we get the following Lagrangian dual problem.

LD2 :

$$\textit{Maximize } \{\theta(\alpha,\beta,\gamma,\delta,\Delta): \ (\alpha,\beta,\gamma,\delta,\Delta) \textit{ unrestricted}\} \qquad ..4.31$$

where

$$\begin{aligned}\theta(\alpha,\beta,\gamma,\delta,\Delta) = \textit{Minimum}\{&\sum_i\sum_j x_{ij}(C_{ij}-\alpha_j+p\eta_{ij}) + \sum_i\sum_j\sum_k w_{ijk}(-\beta_{jk}-\eta_{ij}) \\ &+ \sum_i\sum_t z_{ti}(t^2\Delta_i + t\delta_i) \ + \ \sum_i y_i(\sum_j d_j\beta_{ji} - \delta_i) \\ &+ \sum_k[-\Delta_k Y_{kk} + \sum_{i<k} Y_{ik}(-\gamma_{ik}) + \sum_{i>k} Y_{ik}(\gamma_{ki})] + \sum_j \alpha_j d_j\}\end{aligned} \qquad ..4.32$$

subject to (4.11), (4.12), (4.13), (4.15), (4.16), (4.17), (4.22), (4.23)

When using the conjugate subgradient optimization procedure which is explained in the next section, we fix the duals for Equations (4.10), (4.14), (4.19),

(4.20), (4.21) and then solve for $\theta(\cdot)$ in the above Problem LD. Thus, in the following discussion we will assume that these duals are known. $\theta(\cdot)$ can be solved by separating it into five subproblems. In the subproblems that have a single variable multiplying all its constraints' right-hand-sides, we drop this variable, and solve the resulting problem. This is possible, since as stated earlier, the variable multiplying the right-hand-sides only has the effect of proportionately scaling the resulting objective value. Hence, once we obtain optimal solutions to the resulting problems, we can multiply the optimal solution that we obtain for each problem by the corresponding right-hand-side variable. In the subproblem descriptions, given below, we show the discarded right-hand-side variable in parentheses. The subproblems then can be written as follows.

x-subproblem :

$$\sum_i X_i^* = \sum_i \{ \textit{Minimum} \sum_j (C_{ij} - \alpha_j + p\eta_{ij}) x_{ij} \} \qquad ..4.33$$

$$\textit{subject to} \quad \sum_j x_{ij} \le s \quad (y_i) \quad \forall i \qquad ..4.34$$

$$0 \le x_{ij} \le u_j \quad (y_i) \quad \forall (i,j) \qquad ..4.35$$

When solving the problem in x we drop the y's on the right-hand-side. Then we have n continuous bounded variable knapsack problems in x, which can be solved very easily. X_i^* denotes the objective value obtained as a result of solving the i^{th} knapsack problem. Then, when we obtain the optimal values for x's as x_{ij}^*, we

multiply each x_{ij}^* by its corresponding y_i to get a function in terms of y's. Thus, in other words, each X_i^* denotes the coefficient of y_i.

The subproblem in terms of variable w can be written as follows.

w-subproblem :

$$\sum_i \sum_k W_{ik}^* = \sum_i \sum_k \{Minimum \sum_j (-\beta_{jk} - \eta_{ij}) w_{ijk}\} \quad ..4.36$$

$$subject\ to \quad -\sum_{j=1}^{n} w_{ijk} \geq -s \quad (Y_{ik}) \quad \forall\ (i,k) \quad ..4.37$$

$$0 \leq w_{ijk} \leq u_j \quad (Y_{ik}) \quad \forall\ (i,k) \quad ..4.38$$

For the subproblem in w we drop the variables Y's on the right-hand-side and solve the resulting n^2 separable continuous bounded variable knapsack problems. W_{ik}^* above denotes the objective value obtained for each of the n^2 knapsack problems. The optimal w_{ijk}^* values are then multiplied by corresponding Y_{ik} to obtain a function in terms of Y's.

Next, we write the subproblem in terms of Y. Note that the objective function here includes the objective in terms of Y's from Equation (4.32), as well as the function in terms of Y's obtained by solving the w-subproblem.

Y-subproblem :

$$\sum_k YY_k^* = \sum_k \{Minimum[-\Delta_k Y_{kk} + \sum_{i<k} Y_{ik}(-\gamma_{ik}) + \sum_{i>k} Y_{ik}(\gamma_{ki}) + \sum_i W_{ik}^* Y_{ik}]\} \quad ..4.39$$

$$\textit{subject to} \quad \sum_{i=1}^{n} Y_{ik} = p \quad (y_k) \qquad \forall\, k \qquad ..4.40$$

$$Y \geq 0$$

Once again we solve this problem by dropping the y's on the right-hand-side of the constraints. This then gives us n separable knapsack problems. For every k, we find the index i for which the minimum coefficient of Y_{ik} in Equation (4.26) occurs. We then simply set that particular Y_{ik} equal to p. The optimal value for each knapsack problem is being denoted by YY_k^*. The optimal Y_{ik}^* thus obtained are then multiplied by the corresponding y_k variables to get a function in terms of y. The coefficient of each y_k is then given by YY_k^* in (4.32).

The problem in terms of y now includes the coefficients of y from Equation (4.32) and coefficients obtained as a result of solving subproblems in x and Y. The y-subproblem can then be written as

y-subproblem :

$$OPT = \textit{minimize} \sum_i [\sum_j d_j \beta_{ji} - \delta_i + X_i^* + YY_i^*] y_i \qquad ..4.41$$

$$\textit{subject to} \quad \sum_i y_i = p \qquad ..4.42$$

$$y \geq 0$$

This gives us one knapsack problem in y. We find the minimum coefficient of

y and set that particular y equal to p. Note that we thus obtain integer values for y. OPT in equation (4.41) denotes the optimal value obtained.

The problem in terms of z does not have any variables on the right hand sides and hence can be solved directly. The optimal obtained from z-subproblem is then added to the final objective value. The z-subproblem can be written as follows.

z-subproblem :

$$Z^* = \sum_i \{Minimum \sum_t z_{ti}(t^2\Delta_i + t\delta_i)\} \quad ..4.43$$

$$subject\ to \quad \sum_{t=0}^{t} z_{ti} = 1 \quad \forall\ i \quad ..4.44$$

$$z \geq 0$$

The z-subproblem can also be separated into n continuous knapsack problems. For every i, we find the index t for which the minimum coefficient of z occurs in Equation (4.43), and set this particular z_{ti} equal to 1. Let Z^* denote the summation of objective values for all the n knapsack problems.

We can now write the value of $\theta(\alpha,\beta,\gamma,\delta,\Delta)$ as equal to

$$Z_{CRLT} = \theta(\alpha,\beta,\gamma,\delta,\Delta) = OPT + Z^* + \sum_j \alpha_j d_j \quad ..4.45$$

Equation (4.45) above denotes the objective function value Z_{CRLT} of the Lagrangian relaxation subproblem of the reformulated p-median problem, given a set of Lagrangian multipliers for the dualized constraints. However, we are interested in

finding the solution or a lower bound to the continuous relaxation of Problem PMED. Hence, we must employ some search routine to find the maximum value of $\theta(\cdot)$ over the unrestricted Lagrangian multipliers. In the next section we discuss a conjugate subgradient optimization algorithm.

4.2.4 Conjugate Subgradient Optimization

In order to solve Problem LD1 and LD2 as formulated earlier we must use some kind of a search procedure to search for an optimal value, over the space of the Lagrangian multipliers. Let us denote these Lagrangian multipliers by $\boldsymbol{\omega}$, then for Problem CPMED we have $\boldsymbol{\omega} = (\alpha)$, and for Problem CRLT $\boldsymbol{\omega} = (\alpha,\beta,\gamma,\delta,\Delta)$. Thus we will be searching over the space of $\boldsymbol{\omega}$. There are several techniques available for this kind of a search procedure. However, not all of them are very good in their convergence behavior. For example, Sherali and Myers [1988] have shown that subgradient optimization procedures may fail to converge practically. This happens mainly due to the fact that as the iterates progress, the angle between the subgradient based direction and the direction towards optimality, although acute, tends to approach 90°. As a result the step size used along the direction of search must be reduced considerably before a descent in the objective function value is realized. Hence, it becomes desirable to adopt some suitable deflection or rotation scheme in order to accelerate the convergence behavior. This is what has motivated

us to consider a suitable conjugate subgradient optimization algorithm. The following algorithm is based on the average direction conjugate subgradient (ADS) algorithm developed by Sherali and Ulular [1989].

The algorithm given below will be the same for solving problems CPMED and CRLT, except as stated earlier, the dual space $\boldsymbol{\omega}$ for the two problems is different. Our problem can be written as given below :

$$\text{Maximize } \{ \theta(\boldsymbol{\omega}) : \boldsymbol{\omega} \text{ unrestricted } \}$$

Let ξ denote the subgradient of the objective function above. Then we can write this subgradient for Problem CPMED as

$$\xi = \left| \; d_j - \sum_i x_{ij}^* y_i^* \quad \forall j \; \right|$$

and the subgradient for Problem CRLT is given by

$$\xi = \left| \begin{array}{ll} d_j - \sum_i x_{ij}^* y_i^* & \forall j \\ (d_j - \sum_i w_{ijk}^* Y_{ik}^*) y_k^* & \forall (j,k) \\ p x_{ij}^* y_i^* - \sum_k w_{ijk}^* Y_{ik}^* y_k^* & \forall (i,j) \\ Y_{ki}^* y_i^* - Y_{ik}^* y_k^* & \forall i<k \\ \sum_t t z_{ti}^* - y_i^* & \forall i \\ \sum_t t^2 z_{ti}^* - Y_{ii}^* y_i^* & \forall i \end{array} \right|$$

We must start the algorithm with some value of $\boldsymbol{\omega}$ denoted as $\boldsymbol{\omega}_1$. For Problem CPMED we will use the duals obtained from the last transportation problem solved in the location-allocation interchange heuristic algorithm. If σ_j, $\forall j$, denotes the duals

with respect to the demand constraints (4.1) then we set

$$\alpha_j = \sigma_j \quad \forall j$$

in the staring solution $\boldsymbol{\omega}_1$.

For problem CRLT, we start the algorithm by setting the duals α_j associated with Equation (4.10) equal to the duals α_j obtained at termination, after solving Problem CPMED. We then set the rest of the variables in $\boldsymbol{\omega}_1$ equal to zero. Let the current Lagrangian dual subproblem be denoted by LD, i.e., LD denotes LD1 or LD2, depending on whether we are solving CPMED or CRLT.

We also use the upper bound on PMED obtained by the heuristic procedure, and denote it by UB, to compute step lengths below. The algorithm is partitioned into T number of blocks of iterations. Let N = the total number of iterations allowed, N_t = cumulative number of iterations allowed to the end of block t, N_f = maximum number of consecutive failures allowed (counted by k_f below) after which we reset the solution to the last incumbent and change the step length, a = step length parameter, and β_t = step length parameter multiplier in block t, for t = 1,...,T.

Initialization : We start the algorithm with $\boldsymbol{\omega}^* = \boldsymbol{\omega}_1$ and $\theta(\boldsymbol{\omega}^*) = -\infty$. Set t = 1, start the iteration counter k = 1, and take a = 1. Proceed to Step 1.

Step 1 : Solve LD($\boldsymbol{\omega}_k$), and hence compute $\theta(\boldsymbol{\omega}_k)$ and a subgradient ξ_k of θ at $\boldsymbol{\omega}_k$. If $\lceil\theta(\boldsymbol{\omega}_k)\rceil \geq (1-\epsilon)$UB, then STOP; the current solution is ϵ-optimal. Otherwise, proceed to Step 2.

Step 2 : If $\theta(\omega_k) > \theta(\omega^*)$, then set $\omega^* = \omega_k$, $\xi^* = \xi_k$ and $\theta(\omega^*) = \theta(\omega_k)$ and $k_f = 0$. Otherwise, let $k_f \leftarrow k_f + 1$. If $k_f = N_f$, then reset, meaning, set $\omega_k = \omega^*$, $\xi_k = \xi^*$, $a \leftarrow a/2$, and $k_f = 0$.

Step 3 : If $\|\xi_k\| = 0$, then clearly all the equality constraints that were dualized are binding, hence the solution is feasible with respect to all the original problem constraints and so must be an optimal solution. Also, since the y's obtained from the y-subproblem are all integer, we have solved the original problem.

Step 4 : For $k = 1$, or if Step 2 resulted in resetting then the direction of search is taken as

$$d_k = \xi_k$$

Otherwise, we use

$$d_k = \xi_k + \|\xi_k\|*(\omega_k - \omega_{k-1})/\|\omega_k - \omega_{k-1}\|$$

The step size used is calculated as follows

$$\lambda_k = a*\beta_t*[UB - \theta(\omega_k)]/\|d_k\|^2$$

The new iterate is then given as

$$\omega_{k+1} = \omega_k + \lambda_k*d_k$$

Set $k \leftarrow k + 1$. If $k > N$, then STOP. If $k = N_t + 1$, then set $t \leftarrow t + 1$, (enter the next block), and set $a = 1.0$. Return to Step 1.

The following parameter values are recommended for use :

$N=200$, $N_f=10$, $T=3$, $N_1=75$, $N_2=150$, $N_3=200$, $\beta_1=0.75$, $\beta_2=0.75$, and $\beta_3=0.25$

4.2.5 Benders' Decomposition Procedure

In this section, a decomposition solution procedure based on Benders' partitioning scheme is developed for problem PMED. The motivation behind applying this procedure, is that this procedure exploits the problem structure, namely that for a fixed $y \in Y$, we get a transportation problem in x.

The procedure iterates between the relaxed master problem and the transportation subproblem. On the completion of each transportation problem, either the original problem is solved or a new cutting plane is generated which is then added to the relaxed master problem. In this fashion we generate only a few of the possible cuts to solve the original problem. Furthermore, to reduce the effort involved in solving each of the relaxed master problems, we drop the integer restrictions on y's. Thus, we will have solved the continuous relaxation of Problem PMED at the end of this procedure. Finally, a branch-and-bound procedure is employed to obtain an integer optimal solution. As discussed in Chapter 2, Sherali and Adams [1984] have applied this procedure successfully to solve the discrete location-allocation problem.

In this section we employ an adaptation of Benders' partitioning algorithm to solve Problem PMED. The PMED formulation as given in Chapter 1, can be rewritten after projecting onto x space as follows.

P1:

$$\underset{\substack{y \in Y \\ y \text{ integer}}}{\text{Minimize}} \quad \{ \text{Minimum} \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} x_{ij} \qquad ..4.46$$

$$\text{Subject to} \quad \sum_{i=1}^{n} x_{ij} = d_j \quad \forall\, j=1,..,n \qquad ..4.47$$

$$-\sum_{j=1}^{n} x_{ij} \geq -s\, y_i \quad \forall\, i=1,...n \qquad ..4.48$$

$$-x_{ij} \geq -u_j y_i \quad \forall\, i,j=1,..,n \qquad ..4.49$$

$$x \geq 0$$

$$\text{where} \quad Y = \{y: \sum_{i=1}^{n} y_i = p, y \geq 0\} \qquad ..4.50$$

Let r_j, $\forall j$, represent the dual variables associated with constraints (4.47), q_j $\forall i$, be the dual variables associated with constraints (4.48), and v_{ij} $\forall(i,j)$, be the duals associated with the constraints (4.49). Using these dual variables, we can write the equivalent dual problem associated with the Problem P1, as

P2:

$$\underset{\substack{y \in Y \\ y \text{ integer}}}{\text{Minimize}} \quad \{ \text{Maximum} \sum_{j=1}^{n} d_j r_j - \sum_{i=1}^{n} s q_i y_i - \sum_{i=1}^{n} \sum_{j=1}^{n} u_j v_{ij} y_i \qquad ..4.51$$

$$\text{subject to} \quad r_j - q_i - v_{ij} \leq C_{ij} \quad \forall\, i,j=1,...,n \qquad ..4.52$$

$$q, v \geq 0,\ r \text{ unrestricted}$$

P1 is feasible and bounded for all y ϵ Y, y integer; therefore its associated dual problem P2 must also be feasible and bounded for all y ϵ Y, y integer. Let P denote the polytope given by the constraints of the inner problem in P2, i.e., let

$$P = \{(r,q,v): r_j - q_i - v_{ij} \leq C_{ij}\ \forall\ (i,j),\ q,v \geq 0\}$$

and let (r^k,q^k,v^k), k = 1,...,E denote the extreme points of P. Since, we know that for any y ϵ Y, there exists an optimal extreme point solution to the inner problem in P2, we can write Problem P2 equivalently as follows.

P3:

$$\underset{\substack{y\in Y \\ y \text{ integer}}}{\text{Minimize}} \left\{ \underset{k=1,\ldots,E}{\text{Maximum}} \sum_{j=1}^{n} d_j r_j^k - \sum_{i=1}^{n} s q_i^k y_i - \sum_{i=1}^{n}\sum_{j=1}^{n} u_j v_{ij}^k y_i \right. \qquad ..4.53$$

Finally, Problem P3 may be rewritten as the following Master Problem (MP)

MP:

$$\underset{\substack{y\in Y \\ y \text{ integer}}}{\text{Minimize}} \quad z \qquad ..4.54$$

$$\text{subject to} \quad z \geq \sum_{j=1}^{n} d_j r_j^k - \sum_{i=1}^{n} (s q_i^k + \sum_{j=1}^{n} u_j v_{ij}^k) y_i \qquad \forall\ k=1,\ldots,E \qquad ..4.55$$

Note that the coefficients of y in the above equation must be less than or equal to zero. However, to be able to derive lower bounds within the branch-and-bound procedure explained in the next section, we would like to have nonnegative coefficients. Hence, we will add the zero term $\Phi_M(\Sigma y_i - p)$, where Φ_M is given by

$$\Phi_M = \max_l \{sq_l^k + \sum_j u_j v_{ij}^k\}$$

to the right-hand-side in (4.55). This gives the cut

$$z \geq \sum_{i=1}^{n} \alpha_i^k y_i + \beta^k \qquad ..4.57$$

where

$$\alpha_i^k = \Phi_M - (sq_i^k + \sum_{j=1}^{n} u_j v_{ij}^k) \ \forall i \quad and \quad \beta^k = \sum_j d_j r_j^k - p\Phi_M \ \forall k \qquad ..4.58$$

It is obvious that the number of constraints in Problem MP is equal to the number of extreme points of the set P. Problem MP thus can be in general prohibitively large, and hence does not directly lend itself to be solved in reasonable time. Therefore, a relaxation strategy which successively generates violated constraints is used. The Relaxed Master Problem can be written as

RMP:

$$\begin{array}{ll} \textit{Minimize} & z \\ y \in Y & \\ y \ \textit{integer} & \end{array} \qquad ..4.59$$

$$\textit{subject to} \quad z \geq \sum_{i=1}^{n} \alpha_i^k y_i + \beta^k \quad \forall\ k = 1,...,K << E \qquad ..4.60$$

Let $(\bar{z},\bar{y})$ denote an optimal solution to a given Relaxed Master Problem RMP. Then we need to check whether or not this solution is feasible to MP, and hence solves it, by examining if

$$\bar{z} \geq \sum_{i=1}^{n} \alpha_i^k \bar{y}_i + \beta^k \qquad \forall\, k=1,\ldots,E \qquad ..4.61$$

In order to verify this we need to determine if $\bar{z}$ is greater than or equal to

$$\underset{k=1,\ldots,E}{Maximum} \quad \sum_{i=1}^{n} \alpha_i^k \bar{y}_i + \beta^k \qquad ..4.62$$

Note that the transformations that we made for α and β did not change the objective function as we added and subtracted the same quantity from the right-hand-side. Hence, from (4.58), the value of the above Maximum may be determined by solving the following linear program

SP($\bar{y}$):

$$Maximum \quad \sum_{j=1}^{n} d_j r_j - \sum_{i=1}^{n} s q_i \bar{y}_i - \sum_{i=1}^{n}\sum_{j=1}^{n} u_j v_{ij} \bar{y}_i \qquad ..4.63$$

$$subject\ to \quad r_j - q_i - v_{ij} \leq C_{ij} \qquad \forall\, i,j=1,\ldots,n \qquad ..4.64$$

$$q, v \geq 0,\ r\ unrestricted$$

Problem SP($\cdot$) is known as Benders' subproblem. However, for a given y, the dual to this problem has a structure that makes it more attractive to solve. This dual DSP($\cdot$) can be written as follows.

DSP($\bar{y}$):

$$v(\bar{y}) = \textit{Minimum} \quad \sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}x_{ij} \qquad ..4.65$$

$$\textit{Subject to} \quad \sum_{i=1}^{n} x_{ij} = d_j \quad \forall\, j=1,..,n \qquad ..4.66$$

$$-\sum_{j=1}^{n} x_{ij} \geq -s\bar{y}_i \quad \forall\, i=1,...n \qquad ..4.67$$

$$-x_{ij} \geq -u_j\bar{y}_i \quad \forall\, i,j=1,..,n \qquad ..4.68$$

$$x \geq 0$$

Note that the above dual subproblem is a transportation problem in x. We solve DSP($\cdot$), to obtain an optimal set of dual multipliers (r^{K+1},q^{K+1},v^{K+1}), associated with constraints (4.66), (4.67), (4.68), respectively. Once these optimal values are obtained one needs to check whether

$$\bar{z} \geq \sum_{j=1}^{n} d_j r_j^{K+1} - \sum_{i=1}^{n}\sum_{j=1}^{n} (sq_i^{K+1} + u_j v_{ij}^{K+1})\bar{y}_i \equiv v(\bar{y}) \qquad ..4.69$$

holds. If this condition holds, then ($\bar{z}$,$\bar{y}$) solves MP and hence we have solved the Problem P1. Otherwise, we add the most violated constraint to Problem RMP, which is given by

$$z \geq \sum_{j=1}^{n} d_j r_j^{K+1} - \sum_{i=1}^{n}\sum_{j=1}^{n} (sq_i^{K+1} + u_j v_{ij}^{K+1}) y_i \qquad ..4.70$$

after transforming this as in (4.58), we obtain the new Benders' cut

$$z \geq \sum_{i=1}^{n} \alpha_i^{K+1} y_i + \beta^{K+1} \qquad ..4.71$$

The new cut clearly deletes the current solution $(\bar{z},\bar{y})$. Incrementing K by 1, we resolve the new RMP. This process continues to iterate between the Relaxed Master Problem and the Dual Subproblem until an optimal solution to RMP that is feasible to MP is obtained.

It is worth noting that as additional constraints are added to the Relaxed Master Problem it yields a monotone nondecreasing sequence of objective values until the optimal solution is obtained. Therefore, RMP gives a monotone sequence of lower bounds for the optimal solution to problem PMED. Similarly, since DSP(·) solves the Problem PMED for a particular set of y values, it gives upper bounds for the optimal solution to Problem PMED. However, these upper bounds are not necessarily decreasing, hence, one must keep track of the least upper bound. Thus, whenever the lower bound generated via Problem RMP equals the least upper bound obtained from Problems DSP(·), the procedure terminates. Finite convergence is guaranteed by the fact that only a finite number of extreme point solutions to P exist, and hence there are a finite number of constraint rows possible for MP. Furthermore, as stated earlier, each new cut deletes the most current solution, and

hence no constraint is being regenerated.

Sherali and Adams [1984] relaxed the binary requirements in their problem in a first phase of their solution procedure before reinforcing these restrictions, and found that the solution procedure becomes much more easy to solve. We will also employ the same approach by relaxing the integrality on y variables. This amounts to solving the continuous relaxation of the original mixed integer PMED formulation via Benders' Decomposition method. In solving this problem, a valid set of Benders' cuts are generated since the subproblems SP(y) yield dual basic solutions even for nonintegral y.

Let CRMP and CPMED denote the continuous relaxations (i.e. integrality on y's relaxed) of Problems RMP and PMED, respectively. The procedure employed to solve Problem CPMED is as outlined below.

First, the location-allocation interchange heuristic algorithm given earlier is employed to obtain a "good" feasible solution to problem PMED. This algorithm gives us improving solution values and since this algorithm only considers integer solutions, at every iteration, we set the current values of (z,y) equal to the incumbent values namely (z^*,y^*). We now obtain a cut for CRMP by solving the subproblem $SP(y^*)$, which is done as stated earlier by solving its dual $DSP(y^*)$. The optimal values (r^1,q^1,v^1) obtained from solving this are used to obtain the first Benders' cut. Let LB denote the lower bound on Problem PMED or MP, and let UB_c and UB respectively denote the upper bounds on CPMED and PMED. Thus, currently LB = 0, and UB_c

$= UB = z^*$.

Next, we solve the continuous Lagrangian relaxation of Problem PMED using a conjugate subgradient optimization procedure. This gives us a lower bound Z_{CPMED} on Problem PMED, and so we set $LB = Z_{CPMED}$. The upper bound UB obtained above is used to find the step size λ_k for this algorithm. If the lower bound LB, obtained from Equation (4.29), at any iteration is greater than or equal to $(1-\epsilon)UB$, then we stop, and declare the incumbent solution optimal. Also, if $\|\xi_k\| = 0$, then we stop, as this means that the current solution is feasible to all the original constraints and hence must give an optimal to Problem PMED, as we have integer y's. Otherwise, at the end of N iterations, we use the incumbent dual solution α^* along with the optimal dual solution associated with the constraints (4.27) in the subproblem for $\alpha = \alpha^*$ in order to obtain a strongest surrogate constraint, given by

$$z \geq \sum_i X_i^* y_i + \sum_j \alpha_j^* d_j$$

Once again using a similar transformation in the spirit of (4.57), in order to make the coefficients of y nonnegative, we can rewrite the above equation as

$$z \geq \sum_{i=1}^{n} \alpha_i y_i + \beta \qquad ..4.72$$

In addition, for the solution $\hat{y}$ obtained from the final y-subproblem, we solve Problem SP($\hat{y}$), to derive another Benders' cut to add to Problem CRMP, after the usual transformations.

Next, we solve the Lagrangian relaxation of Problem CRLT, in order to tighten the lower bound. We initialize the procedure by setting the duals associated with Equation (4.10) equal to the duals associated with Equation (4.5) obtained above. During every iteration of this procedure we get integer y solutions. Thus, after every 20 iterations or so we solve the corresponding dual subproblem DSP(y). If the objective value $v(y)$ is lower than the current incumbent objective value then we update the incumbent solution. If the lower bound Z_{CRLT} (see Equation 4.45), obtained at any iteration exceeds or equals $(1-\epsilon)$UB, or if $\|\xi_k\| = 0$, then the procedure stops with the incumbent declared as an ϵ optimal solution. Otherwise, we generate a strongest surrogate Benders' cut obtained from the final y-subproblem, given by

$$z \geq \sum_i [\sum_j d_j \beta_{ji}^* - \delta_i^* + X_i^* + YY_i^*] y_i + Z^* + \sum_j \alpha_j^* d_j$$

and derive a Benders' cut by solving Problem SP($\hat{y}$), where $\hat{y}$ is the solution of the final y-subproblem in this procedure. Both these cuts are then added to Problem CRMP, after the usual transformations.

We now employ Benders' procedure with the current CRMP having the above obtained five cuts, to solve the Problem CPMED. Figure (4.1) depicts the important steps in Benders' algorithm. First, we solve the current Problem CRMP. Let $(\bar{z},\bar{y})$ be an optimal solution. Since, $\bar{z}$ gives a lower bound on Problem PMED which is nondecreasing from one iteration to another, we set LB = $\bar{z}$. Next Problem DSP($\bar{y}$)

is solved to obtain an optimal objective value $v(\bar{y})$ and a set of optimal dual variables. Next, we check whether $v(\bar{y}) < UB$ and whether $\bar{y}$ is integer, and if that is the case, then we have a new incumbent solution to PMED. Hence, the incumbent solution y^* and its value UB are updated. Further, if $v(\bar{y}) < UB_c$, then we set $UB_c = v(\bar{y})$. Finally, if the gap between the upper and lower bounds UB_c and LB, respectively, obtained for Problem CPMED is less than 5%, i.e., if $(UB_c - LB)/UB_c \leq 0.05$, we terminate Benders' procedure as applied to CPMED. Since, we are not interested in an optimal solution to CPMED, this 5% is used to avoid the slow tail-end convergence of the procedure. However, if the decision to terminate the procedure is not reached, then a new Benders' cut is generated and added to Problem CRMP and the process is repeated. If at the end of this procedure the gap between the lower and upper bounds on Problem PMED (i.e., for the Problem MIP) is less than $100\epsilon\%$ (a suitable tolerance), i.e if $(UB - LB)/UB \leq \epsilon$, then the current incumbent solution is declared as a near optimal solution.

Note that if y solves Problem CRMP, then y is not necessarily integral. However, to solve problem DSP(•), which is a transportation problem our code requires integer right-hand-sides. Hence, we employ **scaling**, i.e., we multiply the demand d_j at each node as well as all the y values by 1000 and then use the floor function to get integer y's. The objective value obtained upon solving DSP(•), i.e. v(•) is then divided by 1000 to estimate the value that would have been obtained without scaling.

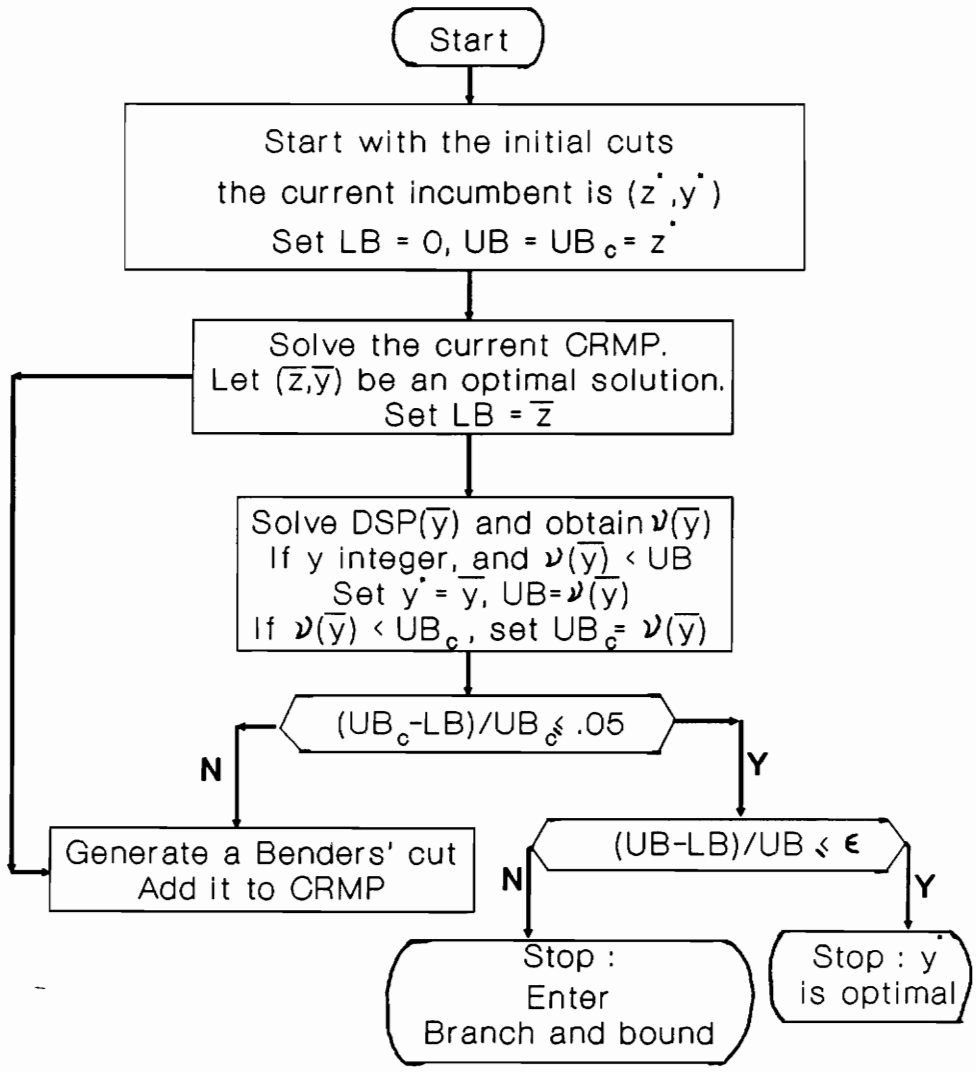

Figure 4.1 Benders' Decomposition Algorithm: Solution of the Continuous Relaxation

4.3 Branch-and-Bound Procedure

Once we have successfully solved Problem CRMP, we proceed to the implicit enumeration scheme outlined in this section. First, we use the duals obtained from the final Problem CRMP that is solved during the Benders' procedure, to obtain a strongest surrogate Benders' cut. This is done by simply multiplying each of the Benders' cuts by its corresponding dual variable, and adding all these resulting cuts together. We add this cut to Problem CRMP. We employ this instance of Problem CRMP to start the branch-and-bound procedure. The enumeration is conducted using Dakin's algorithm for partitioning the problem in the space of the y variables. The Problem CRMP can be written as follows.

CRMP:

$$\textit{Minimize} \quad z \qquad \text{..4.73}$$

$$\textit{subject to} \quad z \geq \sum_{i=1}^{n} \alpha_i^k y_i + \beta^k \qquad \forall\ k = 1,\ldots,K << E \qquad \text{..4.74}$$

$$\sum_{i=1}^{n} y_i = p \qquad \text{..4.75}$$

$$y \geq 0$$

The following sections explain the various components of this algorithm.

4.3.1 Book Keeping

A depth first (LIFO) strategy is used in the branch-and-bound algorithm. A partial solution list PS keeps track of the enumeration tree using the framework due to Geoffrion [1967]. Each entry in PS has two attributes. These are recorded in two arrays, say PSI, and PSV. The first array keeps track of the indices of the branching variables associated with the links on the path connecting the current node to the root node of the branch-and-bound tree. Whenever, we have an upper bound imposed as a branch restriction on any of these links, we carry the corresponding negative of the index of the variable on that link in PSI, and for imposed lower bounds, we carry the positive index value in PSI. The array PSV records the actual corresponding lower or upper bound value on each link. Note, that for any variable, we will always branch with an imposed lower bounding restriction first. Thus, whenever we fathom a particular node, we backtrack to the last positive PSI entry, which represents a lower bounding restriction, we fathom (delete) all the entries to the right of this entry, and we also change the sign on this PSI entry. The corresponding PSV entry is then reduced by one. This then represents a new upper bounding restriction. For example, if the current restrictions are as given below in the stated order:

$$y_1 \geq 2, \quad y_3 \leq 4, \quad y_1 \leq 3$$

then we can write the current partial solution arrays as follows:

$$PSI = \{ 1, -3, -1 \}$$

$$PSV = \{ 2, 4, 3 \}.$$

if we were to fathom the current node, the new PSI and PSV arrays would be given by the following, corresponding to the branch restriction $y_1 \leq 1$.

$$PSI = \{ -1 \}$$

$$PSV = \{ 1 \}$$

We also keep two arrays, namely lb and ub, which respectively carry the current lower and upper bounds on the y variables.

4.3.2 Logical Tests

In order to range restrict the y-variables, we perform objective function based reduced cost logical tests. At any node of the enumeration tree, each variable has explicit lower and upper bounds stored in arrays lb and ub as explained earlier. Let B^k be computed as

$$B^k = \beta^k + \sum_i \alpha_i^k (lb_i) \qquad \forall k$$

denote the minimum right hand side value for each cut at the current node. Since, we are interested in objective function values better than the current incumbent value UB, we can restrict $z \leq UB$, for every cut. (Further, in order to avoid excessive computations involved in going through alternate optimal solutions or close to global solutions, we will use $UB \Leftarrow (1-\epsilon) \cdot UB$ from hereon. Thus, when we get the final

solution we can only claim it to be ϵ-optimal.). This then yields the following

$$\sum_i \alpha_i^k (y_i - lb_i) \leq UB - B^k \qquad \forall k$$

We can now use the objective function reduced cost logical test, in order to get upper bounds on each y.

$$ub_i \leftarrow \min[ub_i,\ lb_i + \min\{\ \lfloor (UB - B^k)/\alpha_i^k \rfloor \quad \forall k\}]$$

Thus, this permits a range restriction on the y variables.

4.3.3 Lower Bound Computations

We used a total of four strategies to compute a lower bound on the objective value of Problem PMED. Given any partial solution, we first compute the lower bound L1 as follows

$$\gamma^l = \beta^l + \sum_i \alpha_i^l * lb_i \qquad \forall\ l=1,..,K$$

then

$$L1 = \gamma^g = \max\{\gamma^l : l=1,...,K\}$$

where g denotes the index of a cut that gives the above maximum. If L1 is greater than or equal to the upper bound UB, we fathom the current partial solution. If this lower bound fails to fathom, we try to find a sharper lower bound.

The cut indexed by g is likely to be the one that determines z in a feasible

completion to the Problem CRMP. Hence, we derive a lower bound L2 by solving the following bounded variable knapsack, given by the above indexed cut.

$$L2 = \beta^g + \textit{minimum} \sum_i \alpha_i^g y_i$$

$$\textit{subject to} \sum_i y_i = p$$

$$lb_i \le y_i \le ub_i$$

We then substitute the y solution obtained for the above problem through all the available cuts of CRMP. We denote the index of the cut that gives the maximum right hand side value as m. If the above lower bound L2, also fails to fathom the current partial solution, then we employ the two cuts g and m obtained above in order to get a lower bound L3. In order to obtain L3 we relax all the constraints in Problem CRMP other than constraints g and m. We then employ Lagrangian relaxation to dualize these two constraints the resulting problem can then be written as follows.

$$L3 = \textit{maximum}\{\theta(\lambda) : \lambda \ge 0\}$$

$$\textit{where} \quad \theta(\lambda) = \textit{minimum}\{(\sum_{i=1}^{n} \alpha_i^g y_i + \beta^g)\lambda + (\sum_{i=1}^{n} \alpha_i^m y_i + \beta^m)(1-\lambda)\}$$

$$\textit{subject to} \quad \sum_{i=1}^{n} y_i = p$$

$$lb_i \leq y_i \leq ub_i$$

The function $\theta(\lambda)$, is known to be a concave function. Thus, we can calculate the maximum value of this function by obtaining the point of intersection of the two tangents that have the shallowest positive and negative valued slopes respectively.

If the lower bound L3 obtained above is also lower than the current incumbent solution, then, we use a conjugate subgradient optimization procedure to update the solution to the previous Problem CRLT. Since, we have imposed lower and upper bounding restrictions on the y variables, we hope to get a lower bound which may lead to fathoming the current partial solution. We perform a fixed number of iterations; we found that 50 iterations were adequate. Also, note that we include the lower and upper bounding restrictions on y variables when solving the y subproblem at every iteration. This yields a lower bound L4 = Z_{CRLT}. We check to see whether this is greater than or equal to UB, and if that is the case, then we fathom the current PS. Otherwise, we branch on a variable to obtain the next node. The next section describes the strategy used to select the new branching variable.

4.3.4 Branching Variable Selection

We used the following strategy to find the next branching variable. To obtain the lower bound L2, we had to solve a bounded variable knapsack problem. This

knapsack can be solved by first fixing $y_i = lb_i \ \forall i$, after which we increment the variables in increasing order of their coefficients, provided that their upper bounds permit this increase and if the sum of the y's is not equal to p. We will use the first variable that was incremented in this procedure as our new branching variable. This is then given by

$$t = \underset{i}{\text{argmin}} \{\alpha_i^g : ub_i > lb_i\}$$

We then branch on the lower bounding restriction of this variable y_t by incrementing the current lower bound by one. Thus, the new restriction is given by

$$y_t \geq (lb_t + 1)$$

The following section gives us the actual steps of the algorithm.

4.3.5 Branch-and-Bound Algorithm

First we obtain the strongest surrogate constraint by multiplying each of the Benders' constraints in the current CRMP by their corresponding duals obtained by solving this CRMP during Benders' procedure as outlined in Section (4.2.5), and summing. This is then added to the current set of constraints, and K is then incremented by one. Next, we perform logical tests in order to obtain tighter upper bounds on the y variables. If the sum of the upper bounds is less than p, then the current incumbent is optimal; hence, we stop. Otherwise, we branch on the variable

that has the greatest upper bound, if m denotes this variable then we add the lower bounding restriction

$$y_m \geq 1$$

Proceed to Step 1.

STEP 1: Compute lower and upper bounds on the y variables, based on the current node list PS. Thus, update the lb_i and ub_i arrays. Let *lbsum* and *ubsum* denote the sum of the current lower and upper bounds, respectively. Proceed to Step 2.

STEP 2: If L1 ≥ UB, then fathom the current node and go to Step 8. Otherwise, if *lbsum* = p, then since L1 < UB, set $y_i = lb_i$ ∀i, perform Step 7, and then fathom the current node by proceeding to Step 8. If we failed to fathom during this step then perform logical tests on all the constraints in order to tighten the upper bounds on the y variables. Update the array ub_i and *ubsum*.

STEP 3: If ubsum < p, then fathom by transferring to Step 8. If *ubsum* = p, then set $y_i = ub_i$ ∀i, substitute this solution y in all of the Problem CRMP constraints, and find the cut with a maximum right hand side value. Let us denote this by z_{max}. If z_{max} < UB, then perform Step 7, and proceed to Step 8 to fathom the current solution. Else, if *ubsum* > p, proceed to Step 4.

STEP 4: Solve the bounded variable knapsack problem obtained for the constraint g (refer Section 4.3.3). Substitute this y solution in all of the constraints of Problem CRMP as before and obtain z_{max}, denote the index of the constraint that yields this maximum by m. If $z_{max} < UB$, then perform Step 7. In any case, proceed to Step 5.

STEP 5: If the lower bound $L2 \geq UB$, then go to Step 8 to fathom, otherwise find lower bound L3. If $L3 \geq UB$, then proceed to Step 8 to fathom. If neither L2 nor L3 lead to fathoming, proceed to Step 6.

STEP 6: Compute the lower bound L4, using the proposed Lagrangian relaxation of CRLT. If $\|\xi_k\| = 0$ (then the corresponding y solution solves the current node problem) perform Step 7, and then go to Step 8 to fathom. If $L4 \geq UB$, then fathom, this node at Step 8. Otherwise, branch on the variable y_t, according to $y_t \geq (lb_t + 1)$ (refer Section 4.3.4).

STEP 7: Solve DSP(y). If $v(y) < UB$, then update the incumbent by setting $y^* = y$ and $UB = v(y)$. Use the duals to obtain a new Benders' cut, and add this cut to the current CRMP. Increment K by 1, and return to the next statement in the step from which this step was called. Note, that in case the number of cuts generated exceeds the storage allocated for these cuts, we replace the last cut with this new cut.

STEP 8: Fathom the current solution. Backtrack from the current node, by finding the latest nonnegative PSI entry, and discarding the entries to the right of this entry, if any. Negate this PSI entry and subtract one from its corresponding PSV entry. If there are no more nonnegative entries in the current PSI list, then stop, declaring the current incumbent solution to be ϵ-optimal. Otherwise, return to Step 1.

CHAPTER V

Computational Experience

In this chapter we present our computational experience with the proposed dynamic programming algorithm for chain graphs and the decomposition and branch-and-bound procedure for general networks.

5.1 Problem Generation

This section explains the scheme used in order to generate the test problems. Here, the values for customer demands, the supply available at each facility and the distance matrix are generated, given the values of n and p. The problems are generated using a uniformly distributed pseudo random generator. Integer customer demands are generated over the interval [0,25]. The supply value is obtained by

dividing the total demand by p and then adding one to this value. The distance matrix is an nxn matrix, this matrix is symmetrical about its diagonal, hence we generate only the upper triangular values, and the lower triangular values are set equal to the corresponding values from the upper triangular matrix. The diagonal obviously contains zero values. All the distance values are generated between the interval [0,10].

5.2 Computational Results

We first give the results obtained for the dynamic programming algorithm. The proposed algorithm was coded in FORTRAN and was implemented on an IBM 3090-300E computer. The solution times obtained for solving problems of various sizes have been summarized in Table 5.1. It can be seen that this algorithm can very easily solve fairly large problems.

Table 5.1 Computational experience with the dynamic programming procedure

n No. of nodes	p No. of fac.	Obj ($)	Time CPU seconds
10	2	381	0.000
20	10	771	0.009
30	30	516	0.019
50	30	1111	0.259
60	50	1223	0.099
80	10	6600	0.009
80	50	1662	2.139
80	80	1245	0.439
100	50	3254	0.189
100	80	1643	5.109

For the decomposition procedure, during the branch-and-bound procedure we tried getting lower bounds using the Lagrangian relaxation of the RLT but it was found that this lower bound was not leading to any fathoming. The following two tables present computational experience for some of the problems solved using these lower bounds and the same problems solved without using these lower bounds.

Table 5.2 Results using Lagrangian relaxation lower bounds

n	p	No. of nodes		Solution obtained at node	Total no. of cuts	Total time	Obj. value
		visited	fathomed				
7	5	186	94	24	140	12.599	76
9	5	34	18	6	24	14.789	67
12	10	754	375	-	-	≥ 300	-
15	3	32	17	27	33	33.689	319
20	10	169	82	-	-	≥ 300	-

Table 5.3 Results without using Lagrangian relaxation lower bounds

n	p	No. of nodes		Solution obtained at node	Total no. of cuts	Total time	Obj. value
		visited	fathomed				
7	5	186	94	24	140	1.339	76
9	5	34	18	6	24	1.749	67
12	10	6,936	3,469	1,110	400	67.129	89
15	3	32	17	27	33	7.089	319
20	10	17,989	8,992	-	-	≥ 300	-

From Tables 5.2 and 5.3, we see that even when we obtain the Lagrangian relaxation based lower bounds during the branch-and-bound procedure, we seem to enumerate the same tree as we visit the same number of nodes, fathom the same number of nodes, and obtain the solution at the same node. Thus, clearly, the extra effort involved in calculating this lower bound is not paying off. Hence, we decided to drop the calculation of these Lagrangian relaxation based lower bounds from the branch-and-bound algorithm. However, we continue to use the Lagrangian relaxation as applied to CRLT at node zero as this gives us a strongest surrogate Benders' cut as well as a new incumbent solution in many cases. The following tables give the results for various test problems. The column notations used are as explained below. n = no. of nodes in the network, p = no. of facilities to be located, ϵ = optimality

tolerance, UB_H = upper bound obtained from the location-allocation interchange heuristic, LB_{RLT} = lower bound obtained from the Lagrangian relaxation of the continuous RLT problem, T_B = time in CPU seconds at the end of the Benders' decomposition procedure, LB_B = lower bound obtained from the Benders' procedure by solving CPMED, UB_B = upper bound obtained at the end of Benders' procedure as a result of any incumbent solution updates, N_V = number of nodes visited during the branch-and-bound procedure, N_F = number of nodes fathomed, N_S = node number at which the final optimal solution was obtained, T_S = time at which this final solution was obtained, B_N = total number of Benders' cuts generated, T = execution time in CPU seconds for the overall algorithm, OBJ = optimal objective value.

Note that if the cumulative time exceeded 300 CPU seconds, the execution of the algorithm was halted prematurely. The execution time for the input of the data is not included in the reported total execution time. In Tables 5.4 and 5.5 the objective values with a * indicate that the algorithm had to be halted prematurely as the total execution time exceeded 300 CPU seconds. The objective value given in these rows indicates the incumbent value at the time of halting.

We find that for many of the problems given in Table 5.4, the number of nodes enumerated is quite large. Since we are solving a general mixed integer programming formulation, the combinatorial content of this problem is fairly large. Thus, if the variable y was restricted to be binary, we could anticipate having to

enumerate fewer nodes. In order to verify whether this really leads to enumerating fewer nodes, we solved the Problem PMED with binary restrictions on the y variables. The computational results obtained are as given in Table 5.5. The problems were generated using the same random number generator and the seed value. Thus, any two problems in Tables 5.4 and 5.5, having the same values of n and p have the same data, and hence can be compared to each other to see the relative change in computational effort. We find that the enumeration tree does become considerably smaller for the binary case.

Table 5.4 Computational results for the branch and bound based decomposition procedure

n	p	100ϵ(%)	UB_H	LB_{RLT}	T_B	LB_B	UB_B	N_V	N_F	N_S	T_S	B_N	T	OBJ
7	5	1	82	49	0.90	49	82	186	94	24	0.94	140	1.34	76
9	5	1	80	54	1.72	58	80	34	18	6	1.73	24	1.75	67
9	6	1	59	33	1.64	42	59	122	62	0	0.02	21	1.72	59
10	6	1	70	51	2.27	62	70	42	22	0	0.02	25	2.30	70
10	7	1	51	28	2.14	35	51	48	25	12	2.16	26	2.19	40
10	7	5	51	28	2.15	35	51	48	25	12	2.17	26	2.20	40
10	9	1	88	23	2.16	24	88	1,576	789	179	2.59	104	6.24	57
12	7	1	87	63	3.54	68	87	286	144	29	3.61	40	3.99	85
12	10	1	117	21	3.67	34	117	6,936	3,469	1,110	8.69	400	67.13	89
12	10	5	117	21	3.69	34	117	6,644	3,323	962	7.25	316	51.58	89
15	10	5	61	40	6.10	46	61	2,544	1,273	499	7.09	42	11.19	50
15	13	5	129	17	6.91	29	129	21,456	10,722	-	-	400	≥ 300	70*
15	13	10	129	17	6.89	29	129	21,559	10,775	-	-	400	≥ 300	65*
20	5	5	353	276	14.40	276	353	48	25	6	14.45	35	14.56	293
20	8	5	245	192	15.12	196	245	3,404	1,703	136	15.68	61	27.43	221
20	10	5	180	131	14.30	132	180	17,972	8,984	-	-	400	≥ 300	146*
20	10	10	180	131	14.34	132	180	16,084	8,043	11,684	189.87	400	268.08	148
22	8	5	207	173	18.97	177	207	8,404	4,203	5,702	54.55	149	75.44	196
22	8	10	207	173	18.98	177	207	3,550	1,776	0	0.07	52	31.45	207
25	8	5	378	279	25.96	280	378	12,799	6,397	-	-	400	≥ 300	304*
25	8	10	378	279	25.89	280	378	2,140	1,071	2,135	63.85	400	63.86	304

Table 5.5 Computational results for the Problem PMED with y binary

n	p	100ϵ(%)	UB_H	LB_{RLT}	T_B	LB_B	UB_B	N_V	N_F	N_S	T_S	B_N	T	OBJ
12	10	1	117	41	4.22	94	103	30	16	0	4.22	13	4.22	103
15	10	1	61	40	7.33	47	50	46	24	0	7.33	15	7.35	50
20	10	1	180	131	25.67	133	158	3,788	1,895	1,587	28.06	55	31.77	140
20	10	5	180	131	25.54	133	158	1,746	874	1,309	27.11	39	27.73	142
22	8	1	207	173	25.69	177	196	6,128	3,065	0	25.69	80	38.89	196
22	8	5	207	173	25.78	177	196	2,124	1,063	0	25.78	31	28.14	196
22	8	10	207	173	25.62	177	196	0	0	0	25.62	27	25.62	196
25	8	5	378	279	30.45	280	368	13,950	6,976	9,770	155.80	400	212.32	299
25	8	10	378	279	30.76	280	368	1,126	564	1,119	33.17	68	33.17	309
25	10	5	255	215	26.43	253	294	10,044	5,023	3,759	75.98	384	225.56	221
25	10	10	255	215	26.43	215	243	44	23	1	26.46	15	26.56	226
30	10	5	323	253	79.80	253	294	17,039	8,516	17	79.88	400	≥ 300	269*
30	10	10	323	253	80.07	253	294	68	35	59	80.20	42	80.20	266
30	12	10	303	205	242.32	209	262	17,893	8,943	-	-	≥ 77	≥ 300	236*

In order to verify the robustness of the proposed algorithm, we changed the data generation scheme and solved a few test problems for the general integer case. The following three variations in the data generation scheme were tried out:

a) Supply values generated were increased by a random number between [50,100].

b) The demand at each node was allowed to vary between [0,50].

c) The distances between the nodes were allowed to vary between [0,25].

The results obtained for the test problems have been summarized in the following tables.

Table 5.7 Test problems with demand values between [0,50]

n	p	N_v	N_f	N_s	B_N	T	OBJ
10	9	1,626	814	934	385	13.87	95
12	10	4,488	2,245	691	159	21.84	182
20	8	4,004	2,003	1,421	400	70.51	416
22	8	3,580	1,791	832	59	32.98	388

Table 5.8 Test problems with distance values between [0,25]

n	p	N_v	N_f	N_s	B_N	T	OBJ
10	9	1,110	556	254	74	4.16	124
12	10	6,234	3,118	1,437	252	43.99	185
20	8	10,590	5,296	7,495	400	191.10	372
22	8	11,462	5,732	3,184	127	99.56	342

Table 5.6 Test problems with higher supply values

n	p	N_v	N_f	N_s	B_N	T	OBJ
10	9	0	0	0	0	0.13	3
12	10	0	0	0	0	0.19	5
20	8	5,396	2,699	2,699	393	95.48	151
22	8	16,158	8,075	4,864	400	$\geq$ 300	197*

CHAPTER VI

Conclusions and Further Research Suggestions

In this thesis, we have studied the equal-capacity p-median problem on a general network. The literature on this subject mostly deals with either capacitated p-median problems on special networks such as chain and tree graphs, or otherwise, deals with the uncapacitated p-median problem. We have examined in this thesis a capacitated discrete p-median problem on a general network, with all supply facilities having equal capacities, and have developed a viable solution algorithm for this problem.

In the proposed algorithm it was found that even though the reformulation technique leads to tighter lower bounds on the problem, our conjugate subgradient optimization technique did not converge to this tighter bound in most cases. Similarly, during the branch-and-bound procedure, the lower bounds obtained using

the Lagrangian relaxation on the restricted space (restricted by the lower and upper bounds on the y variables) of the RLT did not improve the fathoming efficiency in the test problems solved.

We were able to solve problems of size n = 25, p = 8, within five CPU minutes on an IBM 3090 computer. However, as p was increased the effort involved increased dramatically and we were unable to solve problems with a size of n = 15, p = 13, within the limit of five CPU minutes. Some problems e.g. n= 20, p=10 remained unsolved with a tolerance of 5% on optimality within the given time limit, but the same problems were solved when an optimality tolerance of 10% was used.

We find that the number of nodes enumerated in most cases is fairly large. This number is especially large whenever the gap between the lower and upper bounds obtained at the end of the Benders' procedure has been large.

The dynamic programming procedure seems to give very reasonable times for even large problems having n = 100, and p = 80. Thus, it can be used as an alternative algorithm whenever the underlying network is known to be a chain graph. Also, it is possible to develop some heuristic techniques in which some specific types of graphs can be converted to a chain graph, in which case the proposed dynamic programming algorithm can be used to efficiently solve such problems. The specific conversion techniques can be explored as further research.

We found that if we imposed binary restrictions on the y variables, the branch-and-bound tree size reduced significantly. Thus, we were able to solve problems of

size n = 30, p=10. Also, a significant drop in the size of the enumerated tree was observed in all the problems as compared to the general integer case.

The capacitated facility location problem which is fairly closely related to the capacitated p-median problem has been studied extensively in the literature. We present here the computational experience reported in some of these papers. Beasley [1988] who has solved a capacitated warehouse location problem reports problems with sizes upto n = 100 and p = 8 within 300 seconds on a Cray-ls with vector processing and maximum optimization. It must be noted that in their problem formulation once the supply parameters for each facility are fixed, there is no control in locating facilities with higher supply values, whereas in this thesis, whenever having higher supply at a particular node gives a lower cost, the algorithm automatically locates more than one facility at such a node. Sherali and Adams [1984] have been able to solve problems with m = 10 and n = 9, (where m denotes the total number of facilities to be located on as many nodes, and n denotes the number of customers) within 300 seconds on an IBM 3081 series D24 computer.

In order to check the sensitivity of the proposed algorithm with respect to the data values, we have solved certain test problems with modified data. The results for these were reported in Tables 5.6, 5.7, and 5.8. Clearly they do not indicate any particular increase or decrease in computational effort when compared with similarly sized problems of Table 5.4. Thus, we can conclude that the computational times are mainly dependent upon the problem sizes rather than the problem data.

The above research considers fixed demands over time, this thesis may be extended to handle multiperiod problems. Some of the problems with additional constraints such as distance constraints may also be solved using the approach used in this thesis. Since, the times required for the dynamic programming algorithm are significantly smaller when compared to the decomposition procedure of Chapter 4, researching a procedure that heuristically converts a general network to a chain graph would be a direct extension of the present thesis.

References

Bartezzaghi, E., A. Colorni, P. C. Palermo [1981] : "A Tree Search Algorithm for Plant Location Problems", **European Journal of Operational Research**, 7: 371-379.

Beasley, J.E. [1988] : "An Algorithm for Solving Large Capacitated Warehouse Location Problems", **European Journal of Operational Research**, 33: 314-325.

Bitran, G. R., V. Chandru, D. E. Sempolinski, J. F. Shapiro [1981] : "Inverse Optimization: An Application to the Capacitated Plant Location Problem", **Management Science**, 27: 1120-1141.

Cavalier, T. M., H. D. Sherali [1986] : "Network Location Problems with Continuous Link Demands: p-Medians on a Chain and 2-Medians on a Tree", **European of Journal of Operational Research**, 23: 246-255.

Cavalier, T. M., H. D. Sherali [1985] : "Sequential Location-Allocation Problems on Chains and Trees with Probabilistic Link Demands", **Mathematical Programming**, 32: 249-277.

Chiu, S. S. [1987] : "The Minisum Location on an Undirected Network with Continuous Link Demands", **Computers and Operations Research**, 14: 369-383.

Choi, I. C., S. S. Chaudhry [1991] : " A Lagrangian Relaxation and Subgradient Approach for Solving the p-Median Problem with Maximum Distance

Constraints: A Computational Experience", Under Review at Papers in Regional Science: **The Journal of the Regional Science Association International.**

Christofides, N., J. E. Beasley [1983] : "Extensions to a Lagrangian Relaxation Approach for the Capacitated Warehouse Location Problem", **European Journal of Operational Research**, 12: 19-28.

Cornuejols, G., M. L. Fisher, G. L. Nemhauser [1977] : "Locations of Bank Accounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms", **Management Science**, 23: 789-810.

Cornuejols, G., G. L. Nemhauser, L. A. Wolsey [1990] : "The Uncapacitated Facility Location Problem", **Discrete Location Theory** (ed. by, P. B. Mirchandani, R. L. Francis), Wiley Interscience Series in Discrete Mathematics and Optimization 119-171.

Galvão, R. D. [1980] : "A Dual-Bounded Algorithm for the p-Median Problem", **Oper. Res.**, 25: 1112-1121.

Geoffrion, A. M. [1967] : "Integer Programming by Implicit Enumeration and Balas' Method", **SIAM Review**, 7: 178-190.

Goldman, A. J. [1971] : "Optimal Center Location in Simple Networks", **Transportation Science**, 5: 212-221.

Garey, M. R., D. S. Johnson [1979] : **Computers and Intractability : A Guide to the Theory of NP-Completeness**, Freeman, San Francisco.

Hakimi, S. L. [1964] : "Optimum Locations of Switching Centers and the Absolute Centers and Medians of a Graph", **Oper. Res.**, 12: 450-459.

Hakimi, S. L. [1965] : "Optimum Distribution of Switching Centers in a Communication Network and some Related Graph Theoretic Problems", **Oper. Res.**, 13: 462-475.

Hakimi, S. L., S. N. Maheshwari [1972] : "Optimum Locations of Centers in Networks", **Oper. Res.**, 20: 967-973.

Handler, G. Y., P. B. Mirchandani [1979] : **Location on Networks**, The MIT press, Cambridge, Massachusetts.

Hassin, R., A. Tamir [1991] : "Improved Complexity Bounds For Location Problems on the Real Line", **Oper. Res. Letters**, 10: 395-402.

Jacobsen, S. K., [1983] : "Heuristics for the Capacitated Plant Location Model", **European Journal of Operational Research**, 12: 253-261.

Kariv, O., S. L. Hakimi [1979] : " An Algorithmic Approach to Network Location Problems", **SIAM Journal of Applied Mathematics**, 37: 513-560.

Kuehn, A. A., M. J. Hamburger [1963] : "A Heuristic Program for Locating Warehouses", **Management Science**, 9: 643-666.

Levy, J. [1967] : "An Extended Theorem for Location on a Network", **Operational Research Quarterly** 18: 433-442.

Magnanti T. L., R. T. Wong [1990] : "Decomposition Methods for Facility Location Problems", **Discrete Location Theory** (ed. by P. B. Mirchandani, R. L. Francis), Wiley Interscience Series in Discrete Mathematics and Optimization", 209-262.

Maranzana, F. E. [1964] : "On the Location of Supply Points to Minimize Transport Costs", **Operational Research Quarterly**, 15: 261-270.

Mavrides, L. [1979] : "An Indirect Method for the Generalized k-Median Problem Applied to Lock-Box Location", **Management Science**, 25: 990-996.

Minieka, E. [1977] : "The Centers and Medians of a Graph", **Oper. Res.**, 25: 641-650.

Mirchandani, P.B. [1990] : "The p-Median Problem and Generalizations", **Discrete Location Theory** (ed. by P. B. Mirchandani, R. L. Francis), Wiley Interscience Series in Discrete Mathematics and Optimization", 55-118.

Mirchandani, P. B., A. Oudjit [1980] : "Localizing 2 Medians on Probabilistic and Deterministic Tree Networks", **Networks**, 10: 329-350.

Moon, D. I., S. S. Chaudhry [1984] : "An Analysis of Network Location Problems with Distance Constraints", **Management Science**, 30: 290-307.

Narula, S. C., U. I. Ogbu, H. M. Samuelson [1977] : "An Algorithm for the p-Median Problem", **Oper. Res.** 25: 709-712.

Oudjit, A. [1981] : "Median Locations on Deterministic and Probabilistic Multidimensional Networks", Ph.D. Dissertation, Rensselaer Polytechnic Institute, Troy, New York.

ReVelle, C., R. W. Swain [1970] : "Central Facilities Location", **Geographical Analysis**, 2: 30-42.

Scharge, L. [1975] : "Implicit Representation of Variable Upper Bounds in Linear Programming", **Mathematical Programming Study** 4: 118-132.

Slater, P. [1981] : "On Locating a Facility to Service Areas within a Network", **Oper. Res.**, 29: 523-531.

Sherali, H. D. [1991] : "Capacitated, Balanced, Sequential Location-Allocation Problems on Chains and Trees", **Mathematical Programming**, 49: 381-396.

Sherali, H. D., W. P. Adams [1984] : "A Decomposition Algorithm for a Discrete Location-Allocation Problem", **Oper. Res.**, 32: 878-900.

Sherali, H. D., W. P. Adams [1988] : "A Hierarchy of Relaxations Between the Continuous and Convex Hull Representations for Zero-One Programming Problems", Working Paper, Department of Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA.

Sherali, H. D., W. P. Adams [1990] : " A Hierarchy of Relaxations and Convex Hull Characterizations for Mixed Integer Zero-One Programming Problems", Working Paper, Department of Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA.

Sherali, H. D., D. C. Myers [1988] : " Dual Formulations and Subgradient Optimization Strategies for Linear Programming Relaxations of Mixed-Integer Programs", **Discrete Applied Mathematics**, 20: 51-68.

Sherali, H. D., F. L. Nordai [1988 a] : "NP-Hard, Capacitated, Balanced p-Median Problems on a Chain Graph with a Continuum of Link Demands", **Mathematics of Operations Research**, 13: 32-49.

Sherali, H. D., F. L. Nordai [1988 b] : "A Capacitated, Balanced, 2-Median Problem on a Tree Network with a Continuum of Link Demands", **Transportation Science**, 22: 70-73. (Technical Note)

Sherali, H.D., T. P. Rizzo [1991] : "Unbalanced, Capacitated p-Median Problems on a Chain Graph with a Continuum of Link Demands", **Networks**, 21: 133-163.

Sherali, H. D., O. Ulular [1989] : " A Primal-Dual Conjugate Subgradient Algorithm for Specifically Structured Linear and Convex Programming Problems", **Applied Mathematics and Optimization**, 20: 193-221.

Teitz, M. B., P. Bart [1968] : "Heuristic Methods for Estimating the Generalized Vertex Median of a Weighted Graph", **Operations Research**, 16: 955-961.

Van Roy, T. J. [1981] : "Cross decomposition for Mixed Integer Programming with Application to Facility Location", **Operational Research** (J. P. Brans, ed.), 579-587.

Van Roy, T. J. [1986] : "A Cross decomposition Algorithm for Capacitated Facility Location", **Operations Research**, 34: 145-163.

Vita

Vikram J. Marathe was born on April 3, 1967 in Bombay, India. He received a B.Tech. in Mechanical Engineering from Indian Institute of Technology (IIT), Bombay. He received his M.S. degree in Industrial and Systems Engineering in May 1992 from Virginia Polytechnic Institute and State University (VPI&SU).