A Reformulation-Linearization Based Implicit Enumeration Algorithm for the Rectilinear Distance Location-Allocation Problem.

by

Sridhar Ramachandran

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:

[illegible]

Dr. Hanif D. Sherali, Chairman

[illegible]

Dr. Jeffrey D. Tew

[illegible]

Dr. Anthony A. Trani

May 24, 1991

Blacksburg, Virginia

LD
5655
V855
1991
R363
C.2

A Reformulation-Linearization Based Implicit Enumeration Algorithm for the Rectilinear Distance Location-Allocation Problem.

by

Sridhar Ramachandran

Dr. Hanif D. Sherali, Chairman

Industrial and Systems Engineering

(ABSTRACT)

This thesis is concerned with the analysis of a Rectilinear Distance Location Allocation Problem, where the costs are directly proportional to rectilinear distances and the amount shipped. The problem is formulated as a Mixed Integer Bilinear Programming Problem and as a Discrete Location Allocation Problem. Using linear programming relaxations constructed via the Reformulation-Linearization Technique (RLT), the latter formulation is shown to provide stronger lower bounds, and is therefore adopted for implementation. In addition, cutting planes are developed to further strengthen the linear programming relaxation. The special structure of the resulting linear program is exploited in order to get a quick lower bound via a suitable Lagrangian dual formulation. This lower bounding scheme is embedded within a finitely convergent Branch and Bound algorithm that enumerates over the location decision variable space. An illustrative example and computational experience are provided to demonstrate the efficacy of the proposed algorithm.

Acknowledgements

I would like to thank my advisor, Dr. Hanif D. Sherali, for his input and encouragement during each stage of this work. Without his guidance and support, this thesis would not have been possible. I would also like to thank Dr. Jeffrey D. Tew and Dr. Anthony A. Trani for serving on my thesis committee.

I am deeply indebted to my parents for their encouragement and financial support during my stay at Virginia Tech..

Table of Contents

List of Illustrations

List of Tables

Chapter I

INTRODUCTION

1.1 Problem Statement

In many practical situations there exist problems concerned with how to serve a set of destinations that have fixed and known locations in an optimal fashion. This statement of a general problem is due to Cooper (1964) : " Given the location of each destination its associated demand and a set of shipping costs for the region of interest, determine the number, location and capacities of sources so as to minimize the total location and shipping costs ". In most cases one does not have to deal with the problem in all its generality but with a certain class of restricted problems. In this research we will consider a specific location-allocation problem that belongs to a class of restricted problems described as follows.

Given the fixed location of n existing facilities or customers on a continuous plane and their associated demands, we wish to determine the location of m new facilities or sources with known capacities and satisfy the demand requirements of the customers at a minimum total cost. The decisions are where to locate the *m* sources and how much shipment to send from each source to each customer, so that the total cost which is assumed to be directly proportional to the amount shipped and the rectilinear distance over which the shipment occurs is minimized. For convenience, we will also assume without loss of generality that total supply is equal to the total demand. For unbalanced situations, a dummy demand point $(n+1)$, with $c_{i,(n+1)} = 0 \forall i$, can be introduced, if necessary, or the methodology developed can be easily modified to treat inequality supply constraints. This problem can be mathematically stated as follows

$$\begin{array}{lll}
\text{RDLAP}: & \text{minimize} & \displaystyle\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}w_{ij}\left[|x_i - a_j| + |y_i - b_j|\right] \\
& \text{subject to} & \displaystyle\sum_{j=1}^{n} w_{ij} = s_i \qquad i = 1, \dots, m \\
& & \displaystyle\sum_{i=1}^{m} w_{ij} = d_j \qquad j = 1, \dots, n \\
& & w_{ij} \geq 0 \qquad i = 1, \dots, m \quad j = 1, \dots, n
\end{array}$$

where (a_j, b_j): location of demand point *j*,

s_i: supply of source point *i*,

d_j: demand of destination point *j*,

c_{ij}: cost of shipping one unit from source point *i* to demand point *j*,

m: number of supply points,

n: number of demand points,

and where the decision variables are:

(x_i, y_i): location of source *i*,

w_{ij}: amount shipped from source *i* to destination *j*

For convenience, let us also denote

$W = \{w \equiv w_{ij} :$ w satisfies the (transportation) constraints of RDLAP$\}$

The above problem is a **Rectilinear Distance Location Allocation Problem** (RDLAP). For a fixed set of locations $(x_i y_i)$ $i = 1, \dots, m$ RDLAP reduces to a simple transportation/allocation problem and for a fixed $w \in W$, ie. when the allocations are known, RDLAP reduces to m pure location (single facility) problems, each of which is separably solvable in the x_i and the y_i variables. Furthermore, the median (half-sum) location gives the optimal solution. However as shown by Sherali and Nordai (1988) the combined location-allocation problem is NP-hard even if all the demand points are located on a straight line.

It can be shown (see Francis and White (1974)) that the optimal value of x_i and y_i for $i = 1, \dots, m$ must satisfy the following conditions:

$$x_i = a_j \text{ for some } j \in \{1, \dots, n\}$$

$$y_i = b_j \text{ for some } j \in \{1, \dots, n\}$$

Note that the above conditions mean that the rectilinear location problem always has an optimal solution with the sources located at grid points of vertical and horizontal lines drawn through the existing facility locations. Also as shown by Wendell and Hurter (1973) it is only necessary to consider "intersection points" in the convex hull of existing facility locations as candidates for optimal source locations.

1.2 Problem Formulations

1.2.1 Mixed-Integer Bilinear Programming Formulation :

Exploiting the above optimality conditions we can equivalently write

$$
\begin{aligned}
x_i &= \sum_{t=1}^{n_1} a_t \lambda_{it} & i = 1, \dots, m \\
y_i &= \sum_{t=1}^{n_2} b_t \gamma_{it} & i = 1, \dots, m
\end{aligned}
$$

where

$$
\begin{aligned}
\sum_{t=1}^{n_1} \lambda_{it} &= 1 & i = 1, \dots, m \\
\sum_{t=1}^{n_2} \gamma_{it} &= 1 & i = 1, \dots, m \\
\lambda, \gamma &\text{ binary}
\end{aligned}
$$

where $n_1 =$ number of distinct a_j values.

$n_2 =$ number of distinct b_j values.

Problem RDLAP can then be restated as follows :

$$\text{MIBLP}: \quad \text{minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{t=1}^{n_1} C^a{}_{ijt} w_{ij} \lambda_{it} \; + \; \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{t=1}^{n_2} C^b{}_{ijt} w_{ij} \gamma_{it}$$

$$\begin{aligned}
\text{subject to} \quad & \sum_{t=1}^{n_1} \lambda_{it} = 1 && i = 1, \dots, m \\
& \sum_{t=1}^{n_2} \gamma_{it} = 1 && i = 1, \dots, m \\
& \sum_{j=1}^{n} w_{ij} = s_i && i = 1, \dots, m \\
& \sum_{i=1}^{m} w_{ij} = d_j && j = 1, \dots, n \\
& w_{ij} \geq 0 && i = 1, \dots, m \quad j = 1, \dots, n \\
& \lambda_{it} \;\; \text{binary} && i = 1, \dots, m \quad t = 1, \dots, n_1 \\
& \gamma_{it} \;\; \text{binary} && i = 1, \dots, m \quad t = 1, \dots, n_2
\end{aligned}$$

where
$$\begin{aligned}
C^a{}_{ijt} &= c_{ij} \, |a_t - a_j| && \forall (i, j, t) \\
C^b{}_{ijt} &= c_{ij} \, |b_t - b_j| && \forall (i, j, t)
\end{aligned}$$

Problem MIBLP is a special case of the mixed-integer bilinear programming problem, where the continuous relaxation of the underlying zero-one integer program over the λ,γ variables (the "complicating variables") for a fixed $w \in W$ yields a binary optimum. As noted by Sherali and Adams (1988) problem MIBLP can be equivalently solved as a bilinear programming problem. Note that the λ,γ are not jointly constrained, thereby not enforcing the fact that optimal source locations belong to the convex hull of existing facility locations. Although this optimality condition is implied in the integer sense it does not necessarily hold once the λ,γ variables are allowed to be continuous over the interval (0,1). As will be shown later this leads to a considerable weakening in the linear programming reformulation-relaxation of problem MIBLP. The following model, due to the nature of its formulation obviates this difficulty.

1.2.2 RDLAP as a Discrete Location Allocation Problem :

Enumerating the "intersection points", $k = 1, \ldots, K$, belonging to the convex hull of existing facility locations and defining

$$z_{ik} = \begin{cases} 1 \text{ if source i is located at point k} \\ 0 \text{ otherwise} \end{cases}$$

we get the following model:

$$\begin{aligned} \text{DLAP}: \quad & \text{minimize} \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K} \overline{C}_{ijk} w_{ij} z_{ik} \\ & \text{subject to} \sum_{k=1}^{K} z_{ik} = 1 \qquad i = 1, \ldots, m \\ & \sum_{j=1}^{n} w_{ij} = s_i \qquad i = 1, \ldots, m \\ & \sum_{i=1}^{m} w_{ij} = d_j \qquad j = 1, \ldots, m \\ & w_{ij} \geq 0 \qquad i = 1, \ldots, m \quad j = 1, \ldots, n \\ & z_{ik} \text{ binary} \qquad i = 1, \ldots, m \quad k = 1, \ldots, K \end{aligned}$$

where $\overline{C}_{ijk} = c_{ij} \, d_{jk} \quad \forall (i, j, k)$

and

d_{jk} : is the rectilinear distance between existing facility j and point k

The aim of this research is to develop a solution procedure for problem RDLAP. The procedure is a reformulation-linearization based branch and bound algorithm which exploits the above optimality condition in the partitioning scheme and in deriving tight lower bounds via the solutions to linear programming relaxations. Upper bounds are obtained via heuristic procedures.

This study is organized as follows. In Chapter 2 we present a literature review on bilinear programming problems, and on location allocation problems. Chapter 3 describes the development of linearization based lower bounds and the cutting plane generation scheme used. Additionally, the special structure of the lower bounding linear program is exploited to derive Lagrangian dual based bounds. Conjugate subgradients are used in order to obtain dual ascents. This technique is shown to provide significant savings in the amount of effort required to generate the lower bound and is hence adopted for implementation. Chapter 4 describes the overall algorithm imbedded in a branch and bound framework and provides an illustrative example. Computational experience is provided in Chapter 5. Conclusions and suggestions for further research are presented in Chapter 6.

Chapter II

LITERATURE REVIEW

The problem discussed in this study is essentially a bilinear programming problem. Hence we have grouped our discussion of current literature into two topics namely, bilinear programming problems and location-allocation problems.

2.1 Bilinear Programming Problems

Consider the following bilinear programming problem as stated in Vaish and Shetty (1976).

$$\begin{aligned} \text{BLP :}\ \ \text{minimize}\ \ \phi(x,y) &= c^t x + d^t y + x^t C y \\ \text{subject to}\ \ x \in X_o &= \{x \in R^m \mid Ex \le e,\ x \ge 0\} \\ y \in Y_o &= \{y \in R^n \mid Gx \le g,\ y \ge 0\} \end{aligned}$$

where X_O and Y_O are bounded polytopes.

Note that the constraint sets defining X_O and Y_O are disjoint and hence problem BLP is called a separably constrained bilinear programming problem. Suppose X_O and Y_O are nonempty and compact. Then a global minimum is attained at $(\bar{x}, \bar{y})$, where $\bar{x}$ is a vertex of X_O and $\bar{y}$ is a vertex of Y_O, since $\phi(x, y)$ is linear in x for a fixed y and vice versa. A point $(\hat{x}, \hat{y})$ is a local star minimum if and only if $\hat{y}$ solves the problem

$$\min\ \phi(\hat{x}, y)\text{: subject to } y \in Y_O$$

and $\hat{x}$ solves the problem

$$\min\ \phi(x, \hat{y})\text{: subject to } x \in X_O.$$

Problem BLP is a nonconvex programming problem and hence a local minimum need not be a global minimum. It is this aspect of the problem that causes the essential difficulty in solution procedures for this problem.

Vaish and Shetty (1976) propose a polytope annexation algorithm for BLP. A sequence of expanding polytopes are generated such that the global minimum within each polytope is known. The algorithm terminates when one of the polytopes in the sequence contains the feasible region X_O. At any stage i their procedure requires the generation of all the facets of a bounded polytope $P_{i+1} = \text{conv}\,[P_i \bigcup \{v\}]$ where $v \in R^m$, $v \notin P_i$. The authors recognize the fact that memory requirements might prohibit the implementation of their algorithm for realistic problems. Another approach for solving problem BLP is by using the extreme point optimality property, and overcoming local minima by using a cutting plane based algorithm. Also one would like to specify the cuts only in terms of the x or y variables in order to preserve the special structure of the sets Y_O or X_O, respectively, so that when one of the variables is fixed, the resultant subproblem in the space of the other variable is easily solvable. Konno (1976) proposed a cutting plane algorithm for problem BLP. From a local star minimum $(\hat{x}, \hat{y})$ he defines a direction g^t

and obtains the value of a parameter $\overline{\alpha}$ such that $g^t x \geq \overline{\alpha}$ cuts off $\hat{x}$ but does not cut off solutions $(\tilde{x}, \tilde{y})$ for which $\phi(\tilde{x}, \tilde{y}) \leq \overline{v} - \varepsilon$, where $\overline{v}$ is the incumbent solution value and ε is some prespecified positive quantity. Thus his method leads to an *ε-optimal solution.* Gallo and *Ülkücü* (1977) give an algorithm for bilinear programming which is along the lines of Tuy's algorithm for concave minimization problems (see Pardalos and Rosen (1987)). In the cutting plane version of their algorithm, cuts are applied in the X_O set. Using duality theory they consider the following equivalent problem.

$$\begin{aligned} \text{minimize} \quad & \psi(x,u) = (c^t x + \text{maximum } g^t u) \\ \text{subject to} \quad & x \in X_o \\ & u \in U(x) = \{G^t u \leq d + Cx, \ u \leq 0\} \end{aligned}$$

From a point $x^k \in \text{Vert}(X_o)$, they obtain a point u^k by solving

$$\text{maximize } g^t u, \ \text{ subject to } u \in U(x^k).$$

A cut is generated from point x^k if there exists an $x^{k,j} \in N(x^k)$ (where $N(x^k)$ is the set of neighboring vertices of x^k in X_k), such that $\psi(x^{k,j}, u^{k,j}) < \psi(x^k, u^k)$. Otherwise, one has to move to a point $x^{k,j} \in N(x^k)$ which produces a higher objective function value in order to generate a cut. The original constraint set defining X_O is augmented with the cuts generated in a manner such that at any stage k,

$$X_{k+1} = X_k \cap \{ \text{ region feasible to currently generated cuts} \}.$$

Convergence of this method is not guaranteed. Vaish and Shetty (1977) propose an infinitely convergent algorithm for problem BLP, which yields a global optimal solution. Using the concept of generalized polars they generate cuts in the x space from a pseudo global minimum $(\overline{x}, \overline{y})$, a point which satisfies $\phi(\overline{x}, \overline{y}) \leq \phi(x^i, y^i)$ for $i = 1, \dots, m$ where x^i, $i = 1, \dots, m$ are the adjacent extreme points of $\overline{x}$ (assuming nondegeneracy), and y^i is the corresponding extreme point of Y_O obtained by solving the following linear program

$$\text{minimize } \phi(x^i, y), \quad \text{subject to } y \in Y_o.$$

They then specify a valid cut of the form

$$\sum_{j \in J} \frac{p_j}{\bar{\lambda}_j} \geq 1 \;\; \forall\, p \in P$$

in the spirit of Tuy (1964), based on a defined generalized polar set where

$$p^t \equiv (x^t, u^t), \;\; P = \{(x^t, u^t)\colon \; Ex + u = e, \; x \geq 0, \; u \geq 0\},$$

and where J is the index set for the nonbasic variables at $\bar{x}$. In order to compute the $\bar{\lambda}_j$ for $j = 1, \ldots m$ they solve m parametric linear programs over Y_O , using a Bolzano bisection search procedure. A new pseudo global minimum is found over the unexplored feasible region

$$X_O \cap \{ \text{ region feasible to the cuts generated}\},$$

and the process is repeated until the entire feasible region has been explored. The authors prove the dominance of their cuts over those generated by Konno (1976). Sherali and Shetty (1980) propose a finitely convergent algorithm for BLP using polar cuts and disjunctive face cuts. They first find an extreme face of X_O relative to Q (the set of points feasible to previously generated cuts). If the dimension of the extreme face is greater than zero then a disjunctive cut which deletes the entire extreme face is developed. If the extreme face happens to be an extreme point of X_O, a polar cut using negative edge extensions is developed. The disjunctive face cut makes use of the fact that the global minimum of BLP occurs at an extreme point of X_O. At some point x^o of the extreme face which is not an extreme point of X_O, some x or $u \in P$ is currently positive but must be zero at an extreme point of X_O which belongs to Q. The cut developed is a mathematical statement of this disjunction. Suppose an extreme point $\tilde{x}$ of X_O which belongs to Q is at hand. They then solve the problem

$$\text{minimize } \phi(\tilde{x}, y) \text{ subject to } y \in Y_O$$

to get a point $(\tilde{x}, \tilde{y})$ which is a vertex of $X_O \times Y_O$. Searching over $N(\tilde{x}) \bigcap Q$ they arrive at a *weak pseudo global minimum* (WPGM), $(\bar{x}, \bar{y})$ for which

$$\phi(\bar{x}, \bar{y}) \leq \phi(x^i, y) \;\; \forall x^i \in N(\tilde{x}) \bigcap Q \text{ for any } y \in Y_O$$

Based on this solution, they generate a polar cut which is deeper than the cuts developed in Vaish and Shetty (1977). They also suggest a very efficient method of solving the parametric linear programs associated with the above cut generation process. This method is an adaptation of Newton's procedure and is shown to be more effective than the Bolzano bisection search technique. The finiteness of the algorithm is ensured because at any stage one of the two types of cuts is generated and the number of extreme faces of X_O are finite.

Sherali and Alameddine (1990) propose a novel **Reformulation-Linearization Technique** (henceforth referred to as the **RLT** method) for bilinear programming problems of the form

$$\begin{aligned} \text{BLP(JC)}: \;\; & \text{minimize } \;\; \phi(x, y) = c^t x + d^t y + x^t C y \\ & \text{subject to } \;\; (x, y) \in Z \bigcap \Omega \end{aligned}$$

where

$$Z \equiv \left\{ (x, y) : \begin{array}{l} A_1 x + D_1 y \leq b_1 \\ A_2 x + D_2 y = b_2 \end{array} \right\}$$

is a polyhedron in R^{m+n} and

$$\Omega \equiv \{ (x, y) : 0 \leq l \leq x \leq u < \infty \ , \ 0 \leq L \leq y \leq U < \infty \}$$

is a hyper rectangle in R^{n+m}.

Note that the x and the y variables are related in the constraints defining the set Z. Hence BLP(JC) is known as a jointly constrained bilinear programming problem. In such problems the extreme point optimality property is lost; however, it is known that an optimal solution occurs at a boundary point of $Z \cap \Omega$. The RLT method has been successfully applied to separably constrained bilinear programming problems as well. Since our problem is separably constrained we direct the interested reader to Pardalos and Rosen (1987) and Alameddine(1990) for further information on other available approaches to solving problem BLP(JC).

Our approach to solving the rectilinear distance location-allocation problem considered in this study is centered around the RLT method. This method consists of two phases, namely, the Reformulation phase and the Linearization phase. The Reformulation phase calls for an enumeration of variable factors (of the form [variable - its lower bound] and [upper bound - the corresponding variable]) to multiply constraints as well as using constraints to multiply constraints in order to generate new nonlinear constraints. The Linearization phase uses an appropriate variable substitution strategy in order to linearize these newly generated constraints. This process transforms the representation of the nonconvex BLP from the original defining space into a higher dimensional space associated with a lower bounding linear program (henceforth referred to as LBLP) that approximates the closure convex hull of feasible solutions to the nonconvex BLP, when its objective function is accommodated into the constraints. Note that the tightness of

LBLP is inherently dependent on the tightness of the bounds on the variables. Also note that given a set of bounds for the variables, a corresponding LBLP can be constructed. The LBLP is then imbedded in a branch and bound algorithm, where the partitioning is performed by altering the bounds on the variables, and the corresponding LBLP is solved at each of the subnodes to yield lower bounds. Since the original constraints and bounds (which have possibly been further restricted) of problem BLP have been included, a feasible solution yielding an upper bound on the original objective function value is at hand whenever any LBLP is solved. A heuristic is employed to further improve this solution. The branch and bound process is continued until it converges to an optimal solution. Also it has been shown by Sherali and Alameddine (1990) that branching on only one set of variables (x or y) will not affect the convergence of the algorithm.

2.1.1 Mixed-Integer Bilinear Programming Problems :

Consider the following mixed-integer bilinear programming problem as stated in Sherali and Adams (1988)

$$\begin{aligned} \text{MIBLP :}\quad & \text{minimize} \quad c^t x + d^t y + x^t C y \\ & \text{subject to} \quad x \in X \\ & \qquad\qquad\quad y \in Y \\ & \qquad\qquad\quad y \text{ binary} \end{aligned}$$

where X and Y are nonempty, bounded polyhedral sets.

For a fixed $y \in Y$, Problem MIBLP reduces to a linear program in x, and for a fixed $x \in X$, it reduces to a zero-one linear integer program in y. In our case the x variables correspond to the allocation variables, and the y variables correspond to the location variables. Furthermore, as stated earlier, the continuous relaxation of the zero-one linear integer program in y for a fixed $x \in X$, yields a binary y as optimum.

Sherali and Adams (1988) have designed a composite Lagrangian relaxation-implicit enumeration-cutting plane algorithm for problem MIBLP. The mixed-integer bilinear program is converted into an appropriate mixed integer program (MIP) by using the reformulation technique along with an appropriate variable substitution strategy, and subjected to an automatic reformulation phase, in order to possibly fix some of the y variables at 0 or 1. A strongest surrogate Benders′ constraint is then generated. In order to get an optimal set of dual multipliers required for this constraint generation, they do not solve the continuous relaxation of problem MIP, a large scale linear program even for a reasonably sized MIBLP. Instead, they use a subgradient optimization scheme on a Lagrangian dual formulation of the problem, in order to obtain a good quality dual feasible solution. This is further improved upon by using a dual ascent technique to provide a near optimal dual solution to get a near-strongest surrogate Benders′ constraint. Other features of their algorithm include logical tests over Y to improve bounds on the variables, and the generation of Benders′ and disjunctive cuts. The interested reader is directed to their paper for further details and implementation comments.

2.2 Location-Allocation Problems

A mathematical statement of a location-allocation problem is as follows

$$\begin{aligned} \text{LAP :}\quad & \text{minimize} \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij} w_{ij}\ d(L_i, P_j) \\ & \text{subject to}\quad w \in W \\ & \qquad\qquad L_i \in L \quad i = 1, \dots, m \end{aligned}$$

where $d(L_i, P_j)$: represents the distance between points L_i and P_j,

P_j: is the location of demand point j,

c_{ij}: cost of shipping one unit from source point i to demand point j,

m: number of supply points,

n: number of demand points,

W: is the set of transportation constraints,

L: is the solution space for the location of sources;

and where the decision variables are:

m: number of sources to be located.,

$L_i = (x_i, y_i)$: location of source i,

w_{ij}: amount shipped from source i to destination j

The distance measures that arise frequently are rectilinear and Euclidean distances. Other distance measures include the squared Euclidean distance and general ℓ_p distances (see Francis et al. (1983)). Also in many practical situations, m is fixed and known. Based on the characterization of L, the solution space for the location of new facilities, Location-Allocation problems can be classified into three different categories: Continuous Location-Allocation Problems, Discrete Location-Allocation Problems, and Location-Allocation Problems on Networks.

2.2.1 Continuous Location-Allocation Problems :

2.2.1.1 Continuous Location-Allocation Problems Using the Rectilinear Metric

Here the new facility locations are allowed to be points on a continuous plane and the existing facility (customer) locations are fixed discrete points on the plane. Cooper(1961) was the first to formulate such location-allocation problems, and suggested some heuristic solution procedures (see Cooper(1964)). Among these is the alternating location-allocation heuristic which is widely used to provide starting solutions. In uncapacitated location-allocation problems, the sources are assumed to have infinite capacity. Hence, each customer is served by the closest facility. In this context the word "allocation" refers to the assignment of customers to sources. Using Wendell and Hurter's result, Love and Morris (1975) developed a two stage algorithm to solve the uncapacitated version of the rectilinear distance location-allocation problem exactly. The first stage uses a set reduction algorithm which reduces the set of all possible optimal source locations. This has been shown to be equivalent to finding the p-median of a weighted connected graph. The second stage makes use of a technique to solve p-median problems in order to determine the optimal locations and allocations. Kuenne and Soland (1972) developed a branch and bound algorithm for uncapacitated rectilinear distance location-allocation problems and presented computational results for near optimal solutions. A partial assignment of customers to sources is maintained and branching is done by arbitrarily assigning free customers to sources. Lower bounds are computed based on these partial solutions using the triangle inequality. Love and Juel (1982) considered the uncapacitated location-allocation problem with ℓ_p distances, and suggested five heuristic solution procedures. These heuristics involve the determination of a local optimum and the perturbation of the allocations with the hope of finding improving solutions. Computational experience has been provided for $p = 1$, i.e., for the rectilinear norm. Sherali and Shetty (1977) developed a convergent cutting plane algorithm for the capacitated problem. They consider a bilinear programming formulation of the above problem and develop a polar cutting plane algorithm. The cuts are specified in terms of the location variables only in order to preserve the special structure of W. At any stage of their algorithm, they determine an extreme point feasible to previously generated cuts by solving a phase I type (see Bazaara et al. (1990)) 0-1 integer program implicitly. If no feasible solution is found the algorithm terminates with the best found solution being optimal. Otherwise a search is made among the neighboring extreme points (excluding those which are not feasible to the cuts) for improving solutions. A new cut is generated as in Vaish and Shetty (1977). Shetty and Sherali (1980) consider a more general case in which there exist interactions between new facilities and multiple products are shipped among facilities. Their procedure has the following advantages over the procedure in Sherali and Shetty (1977) (a) deeper cuts are developed using negative edge extensions, (b) a more efficient method is developed for the solution of parametric transportation problems encountered during the cut generation, and (c) to determine extreme points of L feasible to the cuts, a simplified version of the extreme point ranking procedure in Murty (1968) is used. A computationally efficient way of implementing this modified procedure is also incorporated. Maruchek and Aly (1981) considered the case where potential customer locations may be more accurately represented as random points uniformly distributed over rectangular regions, and suggested a branch and bound procedure to obtain an exact solution. Selim (1979) considered the case where demands are allowed to be probabilistic as well, and developed solution procedures for problems employing both rectilinear and Euclidean distance metrics.

2.2.1.2 Continuous Location-Allocation Problems Using Other Distance Metrics

Cooper (1972) gave an exact algorithm to solve the capacitated Euclidean distance location-allocation problem. This involves the enumeration of all the basic feasible solutions of W and solving the associated location problems. He also suggested two heuristics which are centered around the alternating location allocation procedure. Murtagh and Niwattisyawong (1982) treated the uncapacitated Euclidean location-allocation problem as a nonlinear programming problem, and suggested a heuristic procedure using the commercial software MINOS (see Murtagh and Saunders (1987)). To get a starting solution they use an initial estimate of the source locations and obtain the corresponding optimal allocations. They then allow the location and allocation variables to vary simultaneously. Sherali and Tuncbilek (1990) have considered a capacitated squared Euclidean distance location-allocation problem. Using calculus the entire problem is transformed in terms of the allocation variables and is shown to be equivalent to a convex maximization problem. The RLT method is applied to this transformed problem to obtain an Upper Bounding Linear Program (UBLP). This UBLP is imbedded in a branch and bound framework which implicitly enumerates the extreme points of W. Branching decisions involve fixing a particular flow variable at a positive level or zero level. Logical tests are then conducted over the constraints defining W (exploiting the special structure) in order to further tighten the bounds on the free variables. If necessary, the corresponding UBLP's at the subnodes are solved (using MINOS as a subroutine as in Tuncbilek (1990)). Upper bounds based on the objective function are also derived. However, the RLT based upper bounds are found to be dominant. The general idea of our approach is the same, but due to the nature of our problem the solution methodology is different. More specifically, we do not project onto the allocation variable space, our RLT and cut generation schemes for tightening the underlying linear programming relaxations are different, and we perform our partitioning in the location variable space.

2.2.2 Discrete Location-Allocation Problems :

In continuous location-allocation problems, any point on the plane is a candidate new facility location. The cost parameters are associated with facilities to be located rather than the sites at which they are located. Also, the cost functions may not include fixed costs. In situations where the above limitations are unacceptable, discrete location-allocation problems arise. The solution set L is finite, discrete, and contains a fixed number of potential new facility locations. Sherali and Adams (1984) have considered a specific capacitated, balanced, discrete location-allocation problem, where the set L consists of m preselected sites, and the objective is to locate m new facilities in a one to one fashion at these sites, so that the total combined cost of location and allocation is minimized. There is a fixed cost c_{ij} associated with locating facility i at site j for each $i = 1, \dots, m$ and $j = 1, \dots, m$. They use Benders' decomposition strategy on a reformulated version of the problem. This involves projecting the problem onto the space of the assignment (facility to site) variables, the "complicating variables". The partitioning scheme is then incorporated in an implicit enumeration framework. Computational experience is provided for problems of realistic sizes. Our formulation of the Rectilinear Distance Location Allocation Problem is based on their suggestion of using the "intersection points" to state this problem as a discrete location allocation problem. However, our solution methodology is based on more recently available techniques to solve Bilinear Programming Problems. More specifically our lower bounding and cut generation schemes are different. For further reading on discrete location problems, the reader is directed to Francis et al. (1983), Aikens (1985), and Hansen et al. (1987).

2.2.3 Location-Allocation Problems on Networks :

In network location problems the potential source locations are restricted to be on a network. Locating an emergency service facility along a road network so as to minimize the maximum response time, locating a plant along a transportation network so as to minimize the total cost of transportation, are some examples of real life problems. The two most common problems encountered are "Minimize the total weighted distance to each customer from its nearest source" (the minisum network location problem), and "Minimize the maximum weighted distance between a customer and its nearest source" (the minimax network location problem). Since we are not directly concerned with this class of problems we refer the reader to Tansel et al. (1983), and Handler and Mirchandani (1979) for a survey and further details on this class of problems.

Chapter III

LOWER-UPPER BOUNDING SCHEMES

This section presents two lower bounding schemes, obtained by applying the RLT method to the two alternative formulations of problem RDLAP. The development of upper bounds and cutting planes are also described. Additionally, Lagrangian dual based bounds are developed, exploiting the special structure of the problem and dual ascents are obtained using conjugate subgradient based directions.

3.1 Lower bounds via Formulation MIBLP

For the purpose of the RLT scheme consider the following variable-bound factors

$$
\begin{array}{rlll}
w_{ij} \geq 0 & i = 1, \ldots, m & j = 1, \ldots, n & (1a) \\
(u_{ij} - w_{ij}) \geq 0 & i = 1, \ldots, m & j = 1, \ldots, n & (1b) \\
\lambda_{it} \geq 0 & i = 1, \ldots, m & t = 1, \ldots, n_1 & (1c) \\
\gamma_{it} \geq 0 & i = 1, \ldots, m & t = 1, \ldots, n_2 & (1d) \\
(1 - \lambda_{it}) \geq 0 & i = 1, \ldots, m & t = 1, \ldots, n_1 & (1e) \\
(1 - \gamma_{it}) \geq 0 & i = 1, \ldots, m & t = 1, \ldots, n_1 & (1f)
\end{array}
$$

where $u_{ij} = \min \{s_i, d_j\}$ is the upper bound on variable w_{ij} implied by $w \in W$, and consider the following constraint factors.

$$\sum_{t=1}^{n_1} \lambda_{it} = 1 \quad i = 1, \ldots, m \qquad (2a)$$

$$\sum_{t=1}^{n_2} \gamma_{it} = 1 \quad i = 1, \ldots, m \qquad (2b)$$

$$\sum_{j=1}^{n} w_{ij} = s_i \quad i = 1, \ldots, m \qquad (2c)$$

Let us add to problem MIBLP, the following sets of implied constraints :

1. *mn* constraints obtained by multiplying the i^{th} equality in (2a) with w_{ij}, $j = 1, \ldots, n$, for each $i = 1, \ldots, m$,

2. *mn* constraints obtained by multiplying the i^{th} equality in (2b) with w_{ij}, $j = 1, \ldots, n$, for each $i = 1, \ldots, m$,

3. mn_1 constraints obtained by multiplying the i^{th} equality in (2c) with λ_{it}, $t = 1, \dots, n_1$, for each $i = 1, \dots, m$,

4. mn_2 constraints obtained by multiplying the i^{th} equality in (2c) with γ_{it}, $t = 1, \dots, n_2$, for each $i = 1, \dots, m$,

5. mnn_1 constraints (nonnegativity restrictions) obtained by multiplying the i^{th} set of j inequalities in (1a) with λ_{it}, $t = 1, \dots, n_1$, for each $i = 1, \dots, m$,

6. mnn_2 constraints (nonnegativity restrictions) obtained by multiplying the i^{th} set of j inequalities in (1a) with γ_{it}, $t = 1, \dots, n_2$, for each $i = 1, \dots, m$,

7. mnn_1 constraints obtained by multiplying the i^{th} set of j inequalities in (1b) with λ_{it}, $t = 1, \dots, n_1$, for each $i = 1, \dots, m$,

8. mnn_2 constraints obtained by multiplying the i^{th} set of j inequalities in (1b) with γ_{it}, $t = 1, \dots, n_2$, for each $i = 1, \dots, m$.

This produces the following equivalent representation :

$$\text{RLT I}: \quad \text{minimize} \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{t=1}^{n_1} C^{a}{}_{ijt} w_{ij}\lambda_{it} + \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{t=1}^{n_2} C^{b}{}_{ijt} w_{ij}\gamma_{it}$$

$$\text{subject to} \quad \sum_{t=1}^{n_1} \lambda_{it} w_{ij} - w_{ij} = 0 \qquad i = 1, \dots, m \;\; j = 1, \dots, n \tag{3a}$$

$$\sum_{t=1}^{n_2} \gamma_{it} w_{ij} - w_{ij} = 0 \qquad i = 1, \dots, m \;\; j = 1, \dots, n \tag{3b}$$

$$\sum_{j=1}^{n} w_{ij}\lambda_{it} - s_i\lambda_{it} = 0 \qquad i = 1, \dots, m \;\; t = 1, \dots, n_1 \tag{3c}$$

$$\sum_{j=1}^{n} w_{ij}\gamma_{it} - s_i\gamma_{it} = 0 \qquad i = 1, \dots, m \;\; t = 1, \dots, n_2 \tag{3d}$$

$$w_{ij}\lambda_{it} \geq 0 \qquad \forall (i, j, t) \tag{3e}$$

$$w_{ij}\gamma_{it} \geq 0 \qquad \forall (i, j, t) \tag{3f}$$

$$- w_{ij}\lambda_{it} + u_{ij}\lambda_{it} \geq 0 \qquad \forall (i, j, t) \tag{3g}$$

$$- w_{ij}\lambda_{it} + u_{ij}\lambda_{it} \geq 0 \qquad \forall (i, j, t) \tag{3h}$$

$$\&$$

$$\{ \text{ original constraints of problem MIBLP } \} \tag{3i}$$

Note the (2a) and (2b) have not been multiplied by (1b), since the resulting constraints would be implied by (3a), (3b) and (3i). For the same reason (2c) has not been multiplied by (1e) or (1f). Constraints generated by multiplying (1a) with (1e) and (1f) are clearly implied by (3c) and (3d), respectively, and are hence also not added to RLT I.

Now, consider the constraint of the type

$$w_{ij}\lambda_{it} - w_{ij} - u_{ij}\lambda_{it} \geq - u_{ij} \tag{2e}$$

obtained by multiplying (1b) and (1e). From (3a) and (3g) we have

$$w_{ij}\lambda_{it} - w_{ij} = -\sum_{k \neq t} w_{ij}\lambda_{ik} \geq -\sum_{k \neq t} u_{ij}\lambda_{ik}$$

But from (2a),

$$-\sum_{k \neq t} u_{ij}\lambda_{ik} = -u_{ij}(1 - \lambda_{it}).$$

Hence, (2e) is implied, and so, the constraints obtained by multiplying (1b) and (1e), or similarly, by multiplying (1b) and (1f), need not be generated. Also note that (3c), & (3d), respectively, imply

$$\begin{aligned} -w_{ij}\lambda_{it} + s_i\lambda_{it} &\geq 0 \\ -w_{ij}\gamma_{it} + s_i\gamma_{it} &\geq 0. \end{aligned}$$

Hence (3g) and (3h) need to be included only for pairs (i,j) such that $u_{ij} = d_j$.

Next, in order to eliminate the nonlinearities in problem RLT I, we employ the following variable substitution :

$$w_{ij}\lambda_{it} = X_{ijt} \tag{4a}$$
$$w_{ij}\gamma_{it} = Y_{ijt} \tag{4b}$$

and subsequently relax the binary restrictions on the λ, γ variables to write it as the following lower bounding linear programming problem.

$$\text{LP}_{\text{MIBLP}}:\ \text{minimize} \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{t=1}^{n_1} C^a{}_{ijt}X_{ijt} + \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{t=1}^{n_2} C^b{}_{ijt}Y_{ijt}$$

$$\text{subject to} \sum_{t=1}^{n_1} X_{ijt} - w_{ij} = 0 \quad i = 1, \dots, m \ j = 1, \dots, n$$

$$\sum_{t=1}^{n_2} Y_{ijt} - w_{ij} = 0 \quad i = 1, \dots, m \ j = 1, \dots, n$$

$$\sum_{j=1}^{n} X_{ijt} - s_i\lambda_{it} = 0 \quad i = 1, \dots, m \ t = 1, \dots, n_1$$

$$\sum_{j=1}^{n} Y_{ijt} - s_i\gamma_{it} = 0 \quad i = 1, \dots, m \ t = 1, \dots, n_2$$

$$- X_{ijt} + u_{ij}\lambda_{it} \geq 0 \ \ \forall(i, j) \ni u_{ij} = d_j$$

$$- Y_{ijt} + u_{ij}\gamma_{it} \geq 0 \ \ \forall(i, j) \ni u_{ij} = d_j$$

$$\sum_{t=1}^{n_1} \lambda_{it} = 1 \qquad i = 1, \dots, m$$

$$\sum_{t=1}^{n_2} \gamma_{it} = 1 \qquad i = 1, \dots, m$$

$$\sum_{j=1}^{n} w_{ij} = s_i \qquad i = 1, \dots, m$$

$$\sum_{i=1}^{m} w_{ij} = d_j \qquad j = 1, \dots, n$$

$$w_{ij} \geq 0 \qquad i = 1, \dots, m \ \ j = 1, \dots, n$$

$$X_{ijt} \geq 0 \qquad \forall\ (i, j, t)$$

$$Y_{ijt} \geq 0 \qquad \forall\ (i, j, t)$$

Problem LP_{MIBLP} has approximately $2mn^2$ additional linearization variables and approximately $2mn^2 + 4mn$ additional constraints, and is a large scale linear program for even small location allocation problems. For example, if $m = 4$, $n = 12$, then problem LP_{MIBLP} can have upto 1296 variables and 1386 constraints. This linear programming problem is a relaxation of problem MIBLP in that for each feasible solution (λ, γ, w)

to MIBLP there corresponds a feasible solution $(\lambda, \gamma, w, X, Y)$ to LP_{MIBLP} with the same objective value, with X and Y defined as in (4a) and (4b). However, since (4a) and (4b) do not necessarily hold for any feasible solution $(\lambda, \gamma, w, X, Y)$ to LP_{MIBLP}, the converse is not necessarily true. However, for any feasible solution to LP_{MIBLP} having binary values of λ and γ, (4) holds and so, LP_{MIBLP} is equivalent to MIBLP if λ and γ are restricted to be binary valued.

3.2 Lower bounds via Formulation DLAP

As in the previous case, we can apply the RLT method to problem DLAP and equivalently write it as

$$
\begin{aligned}
\text{RLT II}: \quad & \text{minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K} \overline{C}_{ijk} w_{ij} z_{ik} \\
& \text{subject to} \quad \sum_{k=1}^{K} z_{ik} w_{ij} - w_{ij} = 0 \qquad i = 1, \dots, m \;\; j = 1, \dots, n \\
& \qquad \sum_{j=1}^{n} w_{ij} z_{ik} - s_i z_{ik} = 0 \qquad i = 1, \dots, m \;\; k = 1, \dots, K \\
& \qquad w_{ij} z_{ik} \geq 0 \qquad \forall (i, j, k) \\
& \qquad - w_{ij} z_{ik} + u_{ij} z_{ik} \geq 0 \qquad \forall (i, j, k) \\
& \qquad \qquad \& \\
& \qquad \{ \text{ original constraints of problem DLAP } \}.
\end{aligned}
$$

Note that the constraints of the type

$$w_{ij} z_{ik} - w_{ij} - u_{ij} z_{ik} \geq - u_{ij} \;\; \forall (i, j, k)$$

obtained by multiplying

$$(u_{ij} - w_{ij}) \geq 0 \text{ and } (1 - z_{ik}) \geq 0$$

are not generated since they are implied, as in the previous formulation. Also constraints of the type

$$- w_{ij}z_{ik} + u_{ij}z_{ik} \geq 0$$

are included only for pairs (i, j) such that $u_{ij} = d_j \; \forall k$. We eliminate the nonlinearities in RLT II by using the following substitution

$$w_{ij}z_{ik} = X_{ijk},$$

and then relax the binary restrictions on the z variables to obtain the following lower bounding linear program.

$$\text{LP}_{\text{DLAP}}: \text{ minimize } \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K}\overline{C}_{ijk}X_{ijk}$$

$$\text{subject to } \sum_{k=1}^{K}X_{ijk} - w_{ij} = 0 \quad i = 1, \dots ,m \; j = 1, \dots ,n \tag{5a}$$

$$\sum_{j=1}^{n}X_{ijk} - s_i z_{ik} = 0 \quad i = 1, \dots ,m \; k = 1, \dots ,K \tag{5b}$$

$$-X_{ijk} + u_{ij}z_{ik} \geq 0 \;\; \forall(i, j) \ni u_{ij} = d_j, \;\; \forall k \tag{5c}$$

$$\sum_{k=1}^{K}z_{ik} = 1 \qquad i = 1, \dots ,m \tag{5d}$$

$$\sum_{j=1}^{n}w_{ij} = s_i \qquad i = 1, \dots ,m \tag{5e}$$

$$\sum_{i=1}^{m}w_{ij} = d_j \qquad j = 1, \dots ,n \tag{5f}$$

$$w_{ij} \geq 0 \qquad i = 1, \dots ,m \;\; j = 1, \dots ,n$$

$$z_{ik} \geq 0 \qquad i = 1, \dots ,m \;\; k = 1, \dots ,K$$

$$X_{ijk} \geq 0 \;\;\; \forall \; (i, j, k)$$

Again, if z is restricted to be binary valued in LP_{DLAP}, the resulting problem is equivalent to DLAP.

3.3 Upper bounds via a Heuristic Procedure

We use the well known "alternating location-allocation" heuristic in conjunction with local search, in order to derive upper bounds on problem RDLAP. The optimal allocations corresponding to a given set of source locations are obtained by solving a transportation problem. For this purpose we use an improved specialized version of NETFLO, the network flow code developed by Kennington and Helgason (1980). The problem of determining optimal source locations given the allocations, as stated before, is easily solvable.

3.3.1 Heuristic Upper Bounding (HUB) Procedure :

Let L be the polyhedron defining the solution space of the location decision variables. The heuristic scheme used is as follows :

Step 1: Given a starting location vector $\tilde{L}$, enter the **alternating location-allocation** scheme to obtain a **fixed point solution** $(\bar{L}, \bar{w})$, where, $\bar{L}$ is a vertex of L and $\bar{w}$ is a vertex of W. Let the corresponding objective function value be $\bar{v}$. Proceed to step 2.

Step 2: For each location $\hat{L}$ in $N(\bar{L})$, the set of neighboring vertices of $\bar{L}$ in L, solve the corresponding transportation problem. Let the objective function value be $v(\hat{L})$. If $v(\hat{L}) < \bar{v}$ return to step 1 with $\tilde{L} = \bar{L}$. If no such $\hat{L} \in N(\bar{L})$ exists, then stop; $(\bar{L}, \bar{w})$ is a **weak pseudo global minimum (WPGM)**, and is the prescribed heuristic solution.

3.4 Comparison of Lower Bounds via Alternative Formulations

For various sizes of randomly generated problems RDLAP, the problems LP_{MIBLP} and LP_{DLAP} were solved using MINOS 5.1. Upper bounds were obtained by rounding their respective linear programming solutions, to obtain a binary solution, in the following manner. Given an optimum solution $\hat{z}$ to LP_{DLAP}, for each $i = 1, \dots, m$ we find

$$k^* = \underset{k=1,\dots,\boxed{K}}{\operatorname{argmax}} \, [\hat{z}_{ik}],$$

intersecty pt. (pp6)

and set $\tilde{z}_{ik^*} = 1$, $\tilde{z}_{ik} = 0$, $\forall\, k \neq k^*$. The solution $\tilde{z}$ is then used as a starting location vector for the HUB scheme, in order to produce the desired upper bounding solution.

The results in Table 1 indicate a significant difference between the values of LP_{MIBLP} and LP_{DLAP}. This is due to the difference in the nature of the alternative formulations. Also, LP_{DLAP} seems to provide better starting solutions for the HUB scheme, thereby usually resulting in better upper bounds. In the worst case, the value of LP_{DLAP} was 15% away from optimality, while that of LP_{MIBLP} was 89% away from optimality. Note that LP_{MIBLP} has at most $2mn^2 + 3mn$ variables, and $2mn^2 + 4mn + 3m + n$ constraints and LP_{DLAP} has $mnk + mn + mK$ variables, and $mnK + mn + mK + 2m + n$ constraints. For $K \leq 2n$, LP_{DLAP} would be smaller in size. The maximum possible value of K, however, is $(n-2)^2$, and in our test problems, K was typically of $O(mn)$. This explains the consistently higher times taken to solve LP_{DLAP}. However, as will be seen later, a conjugate subgradient algorithm which exploits the special structure of LP_{DLAP} to derive Lagrangian dual based lower bounds considerably reduces the amount of effort taken to obtain a lower bound, while maintaining the quality of bounds. This motivates the choice of LP_{DLAP} as the lower bounding linear program adopted by our algorithm.

3.5 Development of Cutting Planes

Recall that DLAP is a Bilinear Programming Problem. Hence polar cutting planes can be generated as in Shetty and Sherali (1980). Problem DLAP may be stated as follows

Table 1. Comparison of Lower Bounds via Alternative Formulations

m n	LB_1	UB_1	t_1	LB_2	UB_2	t_2
2 4	168.00	168.00	0.28	168.00	168.00	0.27
2 4	80.00	80.00	0.28	80.00	80.00	0.27
2 4	101.00	101.00	0.32	101.00	101.00	0.26
3 5	65.00	65.00	0.75	65.00	65.00	0.50
3 5	146.17	300.00	0.78	258.00	258.00	0.82
3 5	108.21	113.00	0.62	109.66	113.00	0.36
3 8	324.00	324.00	1.88	324.00	324.00	5.29
3 8	265.48	331.00	0.78	330.89	334.00	5.57
3 8	180.25	208.00	2.90	201.65	208.00	4.34
4 10	148.15	201.00	14.72	197.49	201.00	59.95
4 10	138.29	363.00	10.04	243.45	283.00	37.78
4 10	141.25	305.00	9.63	304.00	309.00	44.45
4 12	297.11	548.00	24.40	467.25	480.00	218.04
4 12	371.37	494.00	17.72	458.31	505.00	72.12
4 12	259.38	423.00	26.29	421.33	423.00	215.84
5 12	210.35	294.00	36.14	274.39	294.00	244.96
5 12	201.09	325.00	30.09	305.00	305.00	155.63

LB_1 is the value of LP_{MIBLP}
LB_2 is the value of LP_{DLAP}
UB_1 is the upper bound obtained via HUB on the solution to LP_{MIBLP}
UB_2 is the upper bound obtained via HUB on the solution to LP_{DLAP}
t_1 is the time (in cpu secs.) to solve LP_{MIBLP} using MINOS 5.1
t_2 is the time (in cpu secs.) to solve LP_{DLAP} using MINOS 5.1

$$\text{minimize}\{\ \phi(z\,,w):\ z \in Z\,,\ w \in W\ \},$$

where

$$Z \equiv \left\{\ z : \sum_{k=1}^{K} z_{ik} = 1\,, \quad \forall i, \quad z_{ik} \geq 0\ \right\},$$

and where $\phi(z\,,w)$ is the objective function of DLAP. By the separability of Z, its vertices of can be characterized as follows

$$\text{VERT}(Z) \equiv \underset{i}{\times}\ [\ \text{VERT}\ Z(i)]$$

where

$$Z(i) \equiv \left\{\ (z_{ik}\ \ \forall\ k): \ \sum_{k=1}^{K} z_{ik} = 1\,, \quad z_{ik} \geq 0\ \right\} \quad \forall\ i.$$

Now, consider $Z(i)$ for any $i \in \{\,1, \dots ,m\,\}$. We have $Z(i) \subset R^K$. Let $(\tilde{z},\ \tilde{w})$ be a vertex of $Z \times W$ corresponding to this solution, for each $i = 1, \dots ,m$, some z_{ip} is basic and the remaining $z_{ik},\ k \neq p$ are nonbasic, with $\tilde{z}_{ip} = 1$, and $\tilde{z}_{ik} = 0, \forall k \neq p$. The vertices adjacent to $\tilde{z}$ correspond to exchanging each $z_{ik},\ k \neq p,$ for $z_{ip},\ \forall\ i = 1, \dots ,m$. By making $m(K-1)$ such switches from a fixed point solution as in heuristic HUB we obtain a WPGM $(\bar{z},\ \bar{w})$. Let $\bar{v}$ be the objective function value of the incumbent solution. We now define the generalized reverse polar set

$$P = \{\,z : \underset{w \in W}{\text{minimum}}\ \phi(z,\ w) \geq \bar{v}\,\}.$$

Let

$$J \equiv \{ (i,\ k):\ z_{ik} \text{ is nonbasic in } \bar{z} \},$$

and let ξ_{ik} be the edge direction corresponding to z_{ik}, $\forall\ (i,\ k) \in J$. As in Shetty and Sherali (1980) we define

$$\Psi(\lambda_{ik}) = \underset{w \in W}{\text{minimum}} [\ \phi(\bar{z} + \lambda_{ik}\xi_{ik},\ w)\ ]$$
$$\overline{\Psi}(\lambda_{ik}) = \underset{w \in W}{\text{minimum}} [\ \phi(\bar{z} - \lambda_{ik}\xi_{ik},\ w)\ ].$$

Note that ξ_{ik} is a column vector of size mK, with only two nonzero elements, namely a $+1$ in the row corresponding to nonbasic variable z_{ik}, and a -1 in the row corresponding to basic variable z_{ip}. Determine

$$\lambda^*_{ik} = \sup\{\lambda_{ik} : \Psi(\lambda_{ik}) \geq \bar{v}\ \} \quad \forall\ (i,\ k) \in J.$$

If $\lambda^*_{ik} < \infty$, we put $(i,\ k)$ in some partitioned subset J_1 of J. On the other hand, if $\lambda^*_{ik} = \infty$, we put $(i,\ k)$ in the set $J_2 = J - J_1$, and compute

$$\lambda^{**}_{ik} = \sup\{\lambda_{ik} : \overline{\Psi}(\lambda_{ik}) \geq \bar{v}\ \} \quad \forall\ (i,\ k) \in J_2.$$

A valid cutting plane is then given by

$$\sum_{(i,\ k) \in J_1} \frac{z_{ik}}{\lambda^*_{ik}} - \sum_{(i,\ k) \in J_2} \frac{z_{ik}}{\lambda^{**}_{ik}} \geq 1$$

3.5.1 Cut Generation :

Given $\bar{z}$, we repeat the following procedure $m(K-1)$ times, once for each nonbasic variable z_{ik}, $i = 1, \dots ,m$, $k = 1, \dots ,K$. Consider some nonbasic variable z_{ik}. Increasing z_{ik} simply decreases z_{ip} by the same amount. This corresponds to moving along ξ_{ik} . Hence we have

$$\Psi(\lambda_{ik}) = \underset{w \in W}{\text{minimum}} \left[\sum_{i' \neq i} \sum_{j=1}^{n} \sum_{k} \bar{C}_{i'jk} w_{i'j} \bar{z}_{i'k} + \sum_{j=1}^{n} \bar{C}_{ijp} w_{ij} + \sum_{j=1}^{n} \bar{C}_{ijk} w_{ij} \lambda_{ik} - \sum_{j=1}^{n} \bar{C}_{ijp} w_{ij} \lambda_{ik} \right]$$

The above transportation problem is solved with λ_{ik} fixed at M, a very large number. If $\Psi(M) < \infty$ we put $(i,\ k) \in J_1$, and determine λ^{*}_{ik} by solving a parametric transportation problem, as in Shetty and Sherali (1980), using Newtons' search method. Otherwise, if $\Psi(M) = \infty$, we put $(i,\ k) \in J_2$, and determine λ^{**}_{ik}, where,

$$\bar{\Psi}(\lambda_{ik}) = \underset{w \in W}{\text{maximum}} \left[\sum_{i' \neq i} \sum_{j=1}^{n} \sum_{k} \bar{C}_{i'jk} w_{i'j} \bar{z}_{i'k} + \sum_{j=1}^{n} \bar{C}_{ijp} w_{ij} - \sum_{j=1}^{n} \bar{C}_{ijk} w_{ij} \lambda_{ik} + \sum_{j=1}^{n} \bar{C}_{ijp} w_{ij} \lambda_{ik} \right]$$

If $\bar{\Psi}(M) \geq \bar{v}$ we put $\lambda^{**}_{ik} = \infty$. Otherwise we search for λ^{**}_{ik} using Newtons' method. If at any stage in the cut generation process, either J_1 is empty, or λ^{**}_{ik} is zero then we stop with the conclusion that the incumbent solution is optimal (see Shetty and Sherali(1980) for a proof).

3.6 Deriving Lagrangian Dual Based Bounds

Problem LP_{DLAP} has a nice structure for deriving Lagrangian dual based bounds. Dualizing (5a), we obtain the following Lagrangian dual of LP_{DLAP} :

$$\underset{\delta\ unres.}{maximize}\ \ \theta(\delta)$$

where $\theta(\delta)$ is given by solving

$$\text{LDSP}(\delta):\ \theta(\delta) = \quad \text{minimum} \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K}[\overline{C}_{ijk} - \delta_{ij}]X_{ijk} + \sum_{i=1}^{m}\sum_{j=1}^{n}\delta_{ij}w_{ij}$$
$$\text{subject to} \quad [\text{All constraints in } LP_{DLAP} \text{ except } (5a)],$$

and where $\delta \equiv \delta_{ij}\ \forall(i,\ j)$ is the set of dual variables associated with the constraints in (5a).

Problem LDSP(δ) is separable into one transportation problem, and mK bounded variable knapsack problems. The w variables can be independently solved for via the transportation problem over $w \in W$, and the X variables can be solved for in terms of the z variables, which in turn are solved for via trivial knapsack problems. Specifically, given a $\hat{\delta} \equiv \hat{\delta}_{ij}\ \forall\ (i,\ j)$, the associated Lagrangian dual subproblem can be solved as follows.

Step 1 : Solve a transportation problem to obtain

$$\theta_1(\hat{\delta}) \equiv \underset{w \in W}{\text{minimum}} \sum_{i=1}^{m}\sum_{j=1}^{n} \hat{\delta}_{ij}w_{ij}$$

Let $w^* \in W$ be the corresponding optimum obtained.

Step 2 : For each (i, k), solve Problem P_{ik} given by

$$\begin{aligned} P_{ik}:\ \text{minimize} \ & \sum_{j=1}^{n} X_{ijk}[\,\overline{C}_{ijk} - \hat{\delta}_{ij}\,] \\ \text{subject to} \ & \sum_{j=1}^{n} X_{ijk} = s_i \\ & 0 \le X_{ijk} \le u_{ij} \ \forall\ j = 1, \dots, n. \end{aligned} \tag{6}$$

Problem P_{ik} is a bounded variable knapsack problem which can be solved trivially. Let $(\overline{X}_{ijk}\ \forall j)$ be the corresponding optimal solution obtained of objective value $v(ik)$. Determine

$$k^* = \underset{k=1,\dots,K}{\text{argmin}}\,[v(ik)].$$

The optimal value z^* of z, to problem LDSP($\hat{\delta}$) is then given by

$$z^* \equiv \{ z^*_{ik^*} = 1\ \forall i,\ z^*_{ik} = 0\ \forall k \neq k^*,\ \forall i \}.$$

The corresponding optimal value X^*, of X, is now given by

$$X^*_{ijk} = \overline{X}_{ijk}\, z^*_{ik}\ \forall j,\ \forall (i, k).$$

Denote

$$\theta_2(\hat{\delta}) = \sum_{i=1}^{m} v(ik^*).$$

Then, the lower bound obtained via LDSP($\hat{\delta}$) is given by

$$\theta(\hat{\delta}) = \theta_1(\hat{\delta}) + \theta_2(\hat{\delta}).$$

The following algorithm is motivated by the fact that solving LD_{DLAP} using a straight-forward simplex based approach can become prohibitively time consuming, especially in a branch and bound context. Additionally the ease of solving associated subproblems encourages one to pursue this approach. As demonstrated by Sherali and Myers (1988), subgradient optimization procedures may fail to converge practically, resulting in a significant weakening of the computed lower bound. To eliminate this problem we employ a conjugate subgradient based strategy.

3.6.1 A Conjugate Subgradient Algorithm for the Lagrangian Dual Problem

Step 0 : Start with $\delta_1 = 0$, and let $\bar{v}$ be the objective function value of the incumbent solution to DLAP. Set the iteration counter $\kappa = 1$, the number of consecutive failures counter $\kappa_f = 0$, and the step factor $\alpha = 1.0$. Also, select values for the parameters $N =$ maximum number of iterations allowed, $T =$ number of blocks into which the N iterations are partitioned, $N_t =$ number of iterations in block t, $N_f =$ maximum number of consecutive failures allowed before resetting, and $\beta_t =$ step length parameter in block t. Take $\delta^* = \phi$, and $\theta(\delta^*) = -\infty$. Set $t = 1$, and proceed to Step 1.

Step 1 : Solve LDSP(δ_κ), to obtain a lower bound $LB_\kappa = \theta(\delta_\kappa)$, and the corresponding solution (w^κ, X^κ, z^κ). If $|LB_\kappa| \geq (1 - \varepsilon)\bar{v}$, STOP; the incumbent solution is *ε-optimal.* Otherwise, proceed to Step 2.

Step 2 : If $\theta(\delta_\kappa) > \theta(\delta^*)$, the current iteration has resulted in a success. Set $\delta^* = \delta_\kappa$, $z^* = z^\kappa$, $X^* = X^\kappa$, $w^* = w^\kappa$ and put $\kappa_f = 0$. Otherwise, the current iteration has resulted in a failure. Hence, increment the failure counter κ_f by 1. If $\kappa_f = N_f$, then set $\delta_\kappa = \delta^*$, $z^\kappa = z^*$, $X^\kappa = X^*$, $w^\kappa = w^*$, and set $\alpha = \alpha/2$, and $\kappa_f = 0$. Proceed to Step 3.

Step 3 : Compute the subgradient ξ_κ of the objective function of LDSP(δ_κ) in the following manner :

$$\xi_\kappa = \left[w_{11}^\kappa - \sum_{k=1}^{K} X_{11k}^\kappa , \quad \dots \quad , w_{mn}^\kappa - \sum_{k=1}^{K} X_{mnk}^\kappa \right]^T$$

If $\|\xi_\kappa\| = 0$, the subproblem solution is feasible to the equality constraints (5a), and so $(z^\kappa, w^\kappa, X^\kappa)$ is an optimal solution to $\mathrm{LP}_{\mathrm{DLAP}}$. Since z^κ is binary, this solution also solves Problem DLAP. Otherwise, proceed to Step 4.

Step 4 : If $\kappa = 1$, or if δ_κ had been reset in Step 2, compute the direction of motion :

$$d_\kappa = \frac{\xi_\kappa}{\|\xi_\kappa\|}$$

Otherwise, take

$$d_\kappa = \xi_\kappa + \phi_\kappa d_{\kappa-1} \tag{7}$$

where

$$d_{\kappa-1} = (\delta_\kappa - \delta_{\kappa-1})$$

is the direction of motion at the previous iteration and,

$$\phi_\kappa = \frac{\|\xi_\kappa\|}{\|d_{\kappa-1}\|}$$

Step 5 : Calculate the step size λ_κ at the current iteration as follows

$$\lambda_\kappa = \frac{\alpha.\beta_t\,[\bar{v} - \mathrm{LB}_\kappa]}{\|\xi_\kappa\|} \tag{8}$$

Compute the new iterate

$$\delta_{\kappa+1} = \delta_\kappa + \lambda_k \frac{d_\kappa}{\|d_\kappa\|}$$

and proceed to Step 6.

Step 6 : Increment the iteration counter κ by 1. If $\kappa > N$, then STOP. If $\kappa = \sum_{\tau=1}^{t} N_\tau + 1$, then we are at the beginning of a new block, and so, increment t by 1, and set $\alpha = 1.0$. Return to Step 1.

The above algorithm is based on the average direction conjugate subgradient algorithm (ADS) developed by Sherali and Ulular (1989). To start with we used the following parameter values :

$N = 200$, $N_f = 10$, $T = 3$, $N_1 = 75$, $N_2 = 75$, $N_3 = 50$, $\beta_1 = 0.75$, $\beta_2 = 0.5$,, and $\beta_3 = 0.25$, recommended by them. Based on preliminary runs, we found that $N = 175$, with $N_3 = 25$ was more suitable to our problem. Additionally we experimented with the *Block Halving* scheme suggested in their paper. This scheme calculates a common step length for each block, and halves it every N_f consecutive failures within the block. However, we found that calculating the step length at each iteration via (8) enhanced the performance of our algorithm.

3.7 Model Size Reduction via Logical Tests on the Strongest Surrogate Constraint

Let the dual variables associated with (5b), (5c), (5d), (5e), and (5f) be λ_{ik}, π_{ijk}, α_i, β_i, and γ_j, respectively,$\forall$ (i, j, k). The dual to LP$_{\text{DLAP}}$ can be written as follows :

$$\text{D(LP}_{\text{DLAP}}\text{)} \; : \; \text{maximize} \sum_{i=1}^{m} \alpha_i + \sum_{i=1}^{m} s_i\beta_i + \sum_{j=1}^{n} d_j\gamma_j$$

$$\text{Subject to} \quad \delta_{ij} + \lambda_{ik} - \pi_{ijk} \leq \overline{C}_{ijk} \quad \forall \; (i, j, k) \tag{9a}$$

$$-\delta_{ij} + \beta_i + \gamma_j \leq 0 \quad \forall \; (i, j) \tag{9b}$$

$$-s_i\lambda_{ik} + \sum_j u_{ij}\pi_{ijk} + \alpha_i \leq 0 \quad \forall \; (i, k) \tag{9c}$$

$$\pi_{ijk} \geq 0 \;\; \forall \; (i, j, k) \;\; \text{with } \pi_{ijk} = 0 \; \forall k \text{ if } s_i < d_j, \;\; \forall (i, j)$$

$$\alpha, \; \beta, \; \gamma, \; \delta, \; \lambda \text{ unrestricted}$$

where (9a), (9b), (9c) are the constraints written with respect to the X_{ijk}, w_{ij}, and z_{ik} variables, respectively, $\forall$ (i, j, k).

Suppose that the Conjugate Subgradient Algorithm described earlier has been used to solve the Lagrangian dual, perhaps approximately, and a lower bound $\theta(\delta^*)$ is at hand along with an incumbent partial dual solution δ^* and an associated binary z^*. The best dual completion with respect to δ^* can then be determined as follows :

For each $i \in \{1, \ldots, m\}$, solve problem P_{ik} $\forall\, k \in \{1, \ldots, K\}$ given by (6) with $\hat{\delta}$ replaced by δ^*. The corresponding optimal value of the dual variable λ_{ik} equals λ^*_{ik}, where,

$$\lambda^*_{ik} = \overline{C}_{ij^*k} - \delta^*_{ij^*} \quad \forall\, (i,\ k)$$

and where X_{ij^*k} is the index of the optimal basic variable, i.e., the last variable to be given a positive value while solving P_{ik}. Accordingly, we then compute the values π^*_{ijk} for the dual variables π_{ijk} as

$$\pi^*_{ijk} = \text{maximum}\left\{0,\ \delta^*_{ij} + \lambda^*_{ik} - \overline{C}_{ijk}\right\} \quad \forall (i,\ j,\ k).$$

The values α^*_i for the dual variables α_i as are now obtained from (9c) as

$$\alpha^*_i = \underset{k = 1, \ldots, K}{\text{minimum}} \left\{ s_i \lambda^*_{ik} - \sum_{j=1}^{n} u_{ij} \pi^*_{ijk} \right\} \quad \forall\, i.$$

The above values of the dual variables can be used to derive a strongest surrogate type constraint on which logical tests can be performed. One such test is the objective function reduced cost logical test (see Parker and Rardin (1988)).

Surrogating (5a), (5b), (5c), (5d), (5e), and (5f) using their respective dual multipliers and grouping the terms of z, X, and the w variables, we get

$$\sum_{i=1}^{m}\sum_{k=1}^{K}\left(-s_i\lambda_{ik}+\sum_{j=1}^{n}u_{ij}\pi_{ijk}+\alpha_i\right)z_{ik}+\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K}(\delta_{ij}+\lambda_{ik}-\pi_{ijk})X_{ijk}+ \\ \sum_{i=1}^{m}\sum_{j=1}^{n}(\beta_i+\gamma_j-\delta_{ij})w_{ij} \ \geq \ \sum_{i=1}^{m}\alpha_i+\sum_{i=1}^{m}s_i\beta_i+\sum_{j=1}^{n}d_j\gamma_j \tag{10}$$

Note that the right hand side of the above inequality is the objective function of D(LP_{DLAP}). Problem DLAP can be rewritten as follows :

minimize Z

subject to $Z-\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K}\overline{C}_{ijk}X_{ijk} = 0$ (11)

{ and the constraints of LP_{DLAP} }

z binary.

Adding (10) and (11), and imposing $Z<\overline{v}(1-\varepsilon)$ we get the following valid inequality for DLAP based on *ε-optimality.*

$$\overline{v}(1-\varepsilon) \ > \ \left(\sum_{i=1}^{m}\alpha_i+\sum_{i=1}^{m}s_i\beta_i+\sum_{j=1}^{n}d_j\gamma_j\right)+\sum_{i=1}^{m}\sum_{k=1}^{K}\left(s_i\lambda_{ik}-\sum_{j=1}^{n}u_{ij}\pi_{ijk}-\alpha_i\right)z_{ik}+ \\ \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K}(\overline{C}_{ijk}-\delta_{ij}-\lambda_{ik}+\pi_{ijk})X_{ijk}+\sum_{i=1}^{m}\sum_{j=1}^{n}(\delta_{ij}-\beta_i-\gamma_j)w_{ij} \tag{12}$$

Now, for the given dual feasible solution $(\delta^*, \lambda^*, \pi^*, \alpha^*, \beta^*, \gamma^*)$, the coefficients of the X and w variables in (12) are nonnegative, and the first term in (12) equals $\theta(\delta^*)$. For convenience let us denote the coefficients of z_{ik} in (10) by

$$\theta_{ik} = s_i\lambda_{ik}-\sum_{j=1}^{n}u_{ij}\pi_{ijk}-\alpha_i, \quad \forall (i,k)$$

Then, (10) implies that the following constraint is valid

$$\sum_{i=1}^{m} \sum_{k=1}^{K} \theta_{ik} z_{ik} \; < \; \bar{v}(1-\varepsilon) - \theta(\delta^*)$$

The logical test can then be done as follows :

Logical Tests

For a given δ^*, determine the best dual completion (δ^*, λ^*, π^*, α^*, β^*, γ^*) as described earlier. For each variable z_{ik} which is zero in the solution z^*, compute :

$$\theta^*_{ik} = s_i \lambda^*_{ik} - \sum_{j=1}^{n} u_{ij} \pi^*_{ijk} - \alpha^*_i$$

If

$$\theta^*_{ik} \geq \bar{v}(1-\varepsilon) - \theta(\delta^*)$$

then we can permanently fix the corresponding variable z_{ik} at zero, along with the associated variables X_{ijk} $\forall\, j$. The reduced problem is then ready to be used in the branch and bound algorithm.

3.8 Computational Experience with the Conjugate Subgradient Algorithm

In this section we present some preliminary computational experience with the Conjugate Subgradient Algorithm (henceforth referred to as **Algorithm CSA**), and offer comparisons with the quality of bounds it produces and the bounds obtained using a straightforward simplex based approach (using MINOS 5.1) to solve LP_{DLAP}. We also report on the effectiveness of the logical tests of Section 3.7. Note that a primal incumbent solution value, $\bar{v}$, is required while calculating the step length in Algorithm CSA. Also note that a tighter (lower) upper bound, $\bar{v}$, will cause more z variables to be fixed at zero during the model reduction stage. Toward this end we generate a maximum of two distinct WPGM, using the HUB procedure, before executing Step 0 of Algorithm CSA. In addition, we attempt to improve the incumbent solution as Algorithm CSA progresses by periodically searching for a fixed point solution as in Step 1 of HUB. A third WPGM is generated only if the incumbent solution was updated during Algorithm CSA. Upon termination of Algorithm CSA, if an *ε-optimal* solution has not been provably identified, we enter the model reduction stage where logical tests are performed on the strongest surrogate constraint as described in Section 3.7.

Table 2 summarizes the results of the test runs. We used $\varepsilon = 0.01$ as our stopping criterion as well as for the logical tests. Note the significant reduction in effort to obtain the required lower bound, while the quality of the bounds remains fairly close to $v(LP_{DLAP})$. For this reason we chose to implement Algorithm CSA in a branch and bound framework.

Table 2. Computational Experience with the Conjugate Subgradient Algorithm

m n	$v(LP)$	$\theta(\delta^*)$	$\bar{v}$	t_{heur}	t_{CSA}	t_{tot}	% fixed
2 4	168.00	166.72	168.00	0.03	0.01	0.04	–
2 4	80.00	79.63	80.00	0.03	0.02	0.05	–
2 4	101.00	99.19	101.00	0.03	0.03	0.06	–
3 5	65.00	64.01	65.00	0.05	0.14	0.19	–
3 5	258.00	255.07	258.00	0.09	0.01	0.10	–
3 5	109.66	109.56	113.00	0.04	0.35	0.39	60.00
3 8	324.00	320.03	324.00	0.39	1.11	1.50	–
3 8	334.00	326.78	331.00	0.25	1.12	1.37	88.41
3 8	201.65	200.99	208.00	2.46	1.05	1.51	78.43
4 10	197.49	192.55	201.00	3.37	3.24	6.61	90.76
4 10	243.45	239.25	259.00	1.77	2.52	4.29	59.13
4 10	304.00	301.37	305.00	1.30	0.76	2.06	–
4 12	467.25	465.59	480.00	3.79	5.19	8.98	86.51
4 12	458.31	457.19	484.00	2.56	3.70	6.26	60.47
4 12	421.33	418.05	423.00	2.82	3.47	6.29	–
5 12	274.39	262.99	284.00	6.26	6.19	12.45	72.06
5 12	305.00	301.30	305.00	3.02	1.88	4.90	–

$v(LP)$ is the value of LP_{DLAP}
$\theta(\delta^*)$ is the lower bound obtained at the termination of Algorithm CSA
$\bar{v}$ is the incumbent solution value
t_{heur} is the total time (in cpu secs.) to search for heuristic solutions
t_{CSA} is the time (in cpu secs.) to optimize the Lagrangian dual using Algorithm CSA
t_{tot} is the total time taken (in cpu secs.)
% fixed is the percentage of binary vars. fixed at 0 via the logical tests of Section 3.7
a – in the last column indicates that an *ε-optimal* solution was obtained

Chapter IV

An Implicit Enumeration Algorithm

Recall that problem DLAP is an equivalent representation of Problem RDLAP. Furthermore, at optimality to RDLAP we have :

$$x_i = a_j \text{ for some } j \in \{1, \dots, n\}$$

$$y_i = b_j \text{ for some } j \in \{1, \dots, n\}.$$

In this chapter, we present a branch and bound algorithm for Problem DLAP which uses the above optimality characterization for partitioning the problem in the space of the z variables. A branch and bound tree is developed where each node represents a partial solution for which the z variables can be divided into two disjoint sets, namely, $\mathbb{Z}_f = \{ (i,\ k) : 0 \le z_{ik} \le 1 \}$, and $\mathbb{Z}^0 = \{ (i,\ k) : z_{ik} = 0 \}$. Henceforth, we will refer to variable z_{ik} as a *free variable* if $(i,\ k) \in \mathbb{Z}_f$, and as *zero fixed* if $(i,\ k) \in \mathbb{Z}^0$. We will also denote $\mathbb{Z} = \mathbb{Z}_f \bigcup \mathbb{Z}^0$.

We now describe the principal components of the branch and bound algorithm, and then present the overall procedure.

4.1 Bounding Step

Given the sets $\mathbb{Z}_f$ and $\mathbb{Z}^0$ at some current node of the branch and bound tree, let

$$K'_i = \{ k : z_{ik} \in \mathbb{Z}_f \}.$$

Then the corresponding restricted version of $\mathrm{LP_{DLAP}}$ can be written as follows :

$$\begin{aligned}
\mathrm{RLP_{DLAP}}\text{:}\quad & \text{minimize} \quad \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k \in K'_i} \overline{C}_{ijk} X_{ijk} \\
& \text{subject to} \quad \sum_{k \in K'_i} X_{ijk} - w_{ij} = 0 \quad i = 1, \dots, m \;\; j = 1, \dots, n \\
& \sum_{j=1}^{n} X_{ijk} - s_i z_{ik} = 0 \quad i = 1, \dots, m \;\; \forall k \in K'_i \\
& - X_{ijk} + u_{ij} z_{ik} \geq 0 \;\; \forall (i, j) \ni u_{ij} = d_j, \;\; \forall k \in K'_i \\
& \sum_{k \in K'_i} z_{ik} = 1 \qquad i = 1, \dots, m \\
& \sum_{j=1}^{n} w_{ij} = s_i \qquad i = 1, \dots, m \\
& \sum_{i=1}^{m} w_{ij} = d_j \qquad j = 1, \dots, n \\
& w_{ij} \geq 0 \qquad i = 1, \dots, m \;\; j = 1, \dots, n \\
& z_{ik} \geq 0 \qquad i = 1, \dots, m \;\; \forall \; k \in K'_i \\
& X_{ijk} \geq 0 \;\; \forall \; (i, j) \;\; \forall k \in K'_i
\end{aligned}$$

Algorithm CSA can be used to optimize the Lagrangian dual for RLP$_{\text{DLAP}}$, in order to generate the required lower bound. The restrictions $z_{ik} = 0$, $X_{ijk} = 0 \ \forall j, \ \forall (i, k) \in \mathbb{Z}^0$ imposed on LP$_{\text{DLAP}}$ to obtain RLP$_{\text{DLAP}}$, can be handled implicitly as follows.

At Step 2 of the solution procedure for the Lagrangian dual subproblem, we solve Problem P_{ik} given by (6) only if $(i, k) \in \mathbb{Z}_f$. This ensures that the optimal solution z^* satisfies $z_{ik} = 0 \ \forall (i, k) \in \mathbb{Z}^0$. In addition, $X_{ijk} = 0 \ \forall j, \ \forall (i, k) \in \mathbb{Z}^0$ holds automatically. Step 1, which involves solving for a $w \in W$, remains unaffected.

4.2 Partitioning and Branching Scheme

A partitioning of the problem is performed on the (x, y) location variable space, in which, given some current bounds (as below) on the variables, some x or y variable is selected, and its interval is split into two subintervals. Each subinterval then represents the added restriction on the branch leading to each of the corresponding two subnodes generated.

Let $\tilde{\mathbb{Z}}_f$ and $\tilde{\mathbb{Z}}^0$ denote the index sets of the *free* and the *zero fixed z* variables after the model reduction stage, at node zero, and let the location variable bounds at the current node in Problem RDLAP be

$$\begin{aligned} l_i &\le x_i \le u_i & i &= 1, \dots, m \\ L_i &\le y_i \le U_i & i &= 1, \dots, m. \end{aligned}$$

Without loss of generality, we assume that the x and y coordinate values of the existing facility locations are nonnegative. To start with, at node zero, the above bounds can be initialized as $l_i = L_i = 0$, $u_i = a_i^{\max}$, and $U_i = b_i^{\max}$, where,

$$a_i^{\max} = \underset{k \in |K'_i|}{\text{maximum}} \{a_k\}$$

$$b_i^{\max} = \underset{k \in |K'_i|}{\text{maximum}} \{b_k\}$$

Note that given any explicitly imposed bounds on the x_i, y_i variables, the current set of free variables, $\mathbb{Z}_f$, and the current set of zero fixed variables, $\mathbb{Z}^0$, can be constructed as

$$\mathbb{Z}_f = \mathbb{Z}_f^+ \cap \tilde{\mathbb{Z}}_f, \quad \mathbb{Z}^0 = \mathbb{Z} - \mathbb{Z}_f$$

where,

$$\mathbb{Z}_f^+ = \{ (i,\ k) : l_i \le a_k \le u_i,\ L_i \le b_k \le U_i \} \tag{13}$$

and where $(a_k,\ b_k)$ are the x, y coordinate values of point k. As the algorithm progresses, we will be modifying the bounds on the x_i and the y_i variables, and constructing the sets $\mathbb{Z}_f$ and $\mathbb{Z}^0$ in order to derive the restricted lower bounding problem via the sets $K'_i = \{ k : (i,\ k) \in \mathbb{Z}_f \}\ \forall i$.

4.3 Branching Variable Selection

Algorithm CSA does not necessarily yield a primal feasible solution to LP_{DLAP}, which could have been used in the traditional manner to select a branching variable in the context of a branch and bound procedure. However, it does provide an incumbent dual solution δ^* and a corresponding binary solution z^*, based on which, the problem can be partitioned as described below.

For each $i = 1, \dots, m$, note that some $z^{*}_{ip(i)} = 1$ is basic, for an index $p(i) \in \{1, \dots, K\}$, and $z^{*}_{ik} = 0 \quad \forall\ k \neq p(i)$. The corresponding x_i, y_i solution is given by :

$$\begin{aligned} x_i &= a_{p(i)} \\ y_i &= b_{p(i)}. \end{aligned}$$

Let $v(ik)$ denote the value of problem P_{ik}, $i = 1, \dots, m$, $k \in K'_i$ when $\hat{\delta} \equiv \delta^{*}$. A branching variable is then selected as follows.

Step 1 : Determine

$$q \in \underset{i\,:\,|K'_i| \geq 2}{\operatorname{argmin}} \ [\ v(ip(i))\ ] \tag{14}$$

Let

$$I_1 = \left\{ i \ :\ |K'_i| \geq 2, \text{ and } v(ip(i)) = v(qp(q)) \right\}$$

If $|I_1| = 1$, then q is uniquely determined, and so, proceed to Step 2. Otherwise, to break the existing ties, let

$$I_2 = \left\{ \bar{i} \in I_1 : |K'_{\bar{i}}| = \underset{i \in I_1}{\text{maximum}} \quad |K'_i| \right\}$$

If $|I_2| = 1$, let q be the index in I_2 and proceed to Step 2. Otherwise, arbitrarily select $q \in$ argmax $\{ s_i : i \in I_2 \}$, and proceed to Step 2.

Step 2 : For the selected $q \in \{1, \dots, m\}$, let the bounds on x_q and y_q at the current node be given by :

$$l_q \le x_q \le u_q$$
$$L_q \le y_q \le U_q.$$

We now partition the bound interval for the x_q or the y_q variable, by choosing x_q if $(u_q - l_q) \ge (U_q - L_q)$, and choosing y_q otherwise. Proceed to Step 3.

Step 3 : Suppose that we picked x_q as the branching variable, at Step 2 (The case for y_q is symmetric). In order to partition the problem associated with the current node of the branch and bound tree into two subproblems, we impose the restrictions

$$l_q \le x_q \le u_q^{\text{new}}$$

along one branch, and

$$l_q^{\text{new}} \le x_q \le u_q$$

along the other branch, where l_q^{new} and u_q^{new} are determined as follows. Let

$$a_q^- < a_{p(q)} < a_q^+,$$

where a^-_q and a_q^+ are adjacent a_j coordinates from the set K'_q , to the immediate left and right of $a_{p(q)}$, respectively, if they exist. Then

$$\begin{aligned} u_q^{\text{new}} &= a_{p(q)}, \ l_q^{\text{new}} = a_q^+, \ \text{ if } \ a_{p(q)} - l_q \le u_q - a_{p(q)} \\ u_q^{\text{new}} &= a_q^- \ , \ l_q^{\text{new}} = a_{p(q)}, \quad \text{otherwise} \end{aligned}$$

4.4 Other Features of the Algorithm

4.4.1 Logical Tests :

Suppose that the restricted problem RLP_{DLAP} at some current node has been solved, and a δ^* is at hand along with a corresponding z^*. The best dual completion with respect to δ^* can be found as in Section 3.7, and the objective function reduced cost logical test can be performed to further restrict the problem. Note that if $|K'_i| = 0$ for any $i \in \{1, \ldots, m\}$ at a given node, it can be fathomed since (5a) is violated. Also note that at any given node the number of transportation problems required for total enumeration is given by

$$NT = \prod_{i=1}^{m} |K'_i| \tag{15}$$

Whenever this number is sufficiently small, we totally enumerate all feasible solutions to the current node subproblem, and fathom this node.

4.4.2 Upper Bounds via an Imbedded Heuristic :

An important issue in branch and bound algorithms is to obtain good quality incumbent solutions early on in the search process. In our case, the solution to any Lagrangian dual subproblem yields a binary z which can be used as a starting solution for the "alternating location-allocation" heuristic, in order to obtain a fixed point solution. Periodically (every five iterations) check if the incumbent dual solution has been updated, and if so, we use the z^* solution corresponding to this incumbent δ^* and find a fixed point solution to possibly update the incumbent solution. Note that the HUB scheme could be used instead. However, since the effort involved at Step 2 of HUB is considerable, we use HUB only at node 0 at which the cutting plane of Section 3.5 is also generated.

4.4.3 Search Strategy :

A depth first (LIFO) strategy is used in the branch and bound algorithm. A partial solution list PS keeps track of the branch and bound tree using the framework due to Geoffrion (1967). Each entry i in PS has four attributes. If $i \leq m$ then it represents the variable x_i, and otherwise, it represents the variable y_{i-m}. The other three attributes are recorded in additional arrays, say PSB, PSLU, and PSF that accompany PS. They respectively carry the explicitly imposed lower or upper bound on a variable, and an entry which is a $+1$ if this is an imposed upper bound and a -1 if it is a lower bound, and an entry which is a $+1$ if this node is "underlined", a zero, otherwise, where a node is underlined, if the descendents of its complementary branch have been fathomed. The bounds on the opposite side of the current branch are stored in the list PSNB. For example if $m = 4$ and we have imposed the following restrictions in the stated order :

$$\begin{aligned} x_3 &\leq 8 \\ y_1 &\leq 9 \\ y_2 &\geq 20 \end{aligned}$$

with the complementary branch on the first two nodes having the restrictions $x_3 \geq 10$, and $y_1 \geq 12$, respectively, and that on the final node being fathomed, then the lists corresponding to the above partial solution are as follows :

$$\begin{aligned} \text{PS} &= \{\,3,\ 5,\ 6\} \\ \text{PSB} &= \{\,8,\ 9,\ 20\} \\ \text{PSLU} &= \{\,1,\ 1,\ -1\} \\ \text{PSF} &= \{\,0,\ 0,\ 1\}. \\ \text{PSNB} &= \{\,10,\ 12,\ \phi\}. \end{aligned}$$

4.4.4 Optimality Criterion :

In order to avoid undue excessive computations involved in sifting through alternative optimal solutions or close to global optimal solutions, the fathoming criterion used is as follows :

$$\lceil \text{LB} \rceil \geq (1 - \varepsilon)\,\bar{v} \tag{16}$$

where $0 \leq \varepsilon \leq 1$, LB is a valid lower bound at the current node of the branch and bound tree, and $\bar{v}$ is the objective function value of the incumbent solution to DLAP. Note that in (16), the lower bound LB has been rounded up to the lowest integer greater than or equal to it, assuming all integer data for Problem RDLAP. Hence, when the algorithm stops, we can claim that the global minimum is within $100 \times \varepsilon$ % of the current best solution value. Also, whenever the restricted problem associated with any node is to be solved to obtain a lower bound, the incumbent dual solution δ^* corresponding to its parent node is used to provide an advanced start for Algorithm CSA.

4.5 Summary of the Algorithm

Step 0 : ***Model Reduction.*** Initialize the bounds on the x_i and the y_i variables. Solve $\mathrm{LP_{DLAP}}$ using Algorithm CSA and go through the model reduction stage of Section 3.7 to determine the sets $\tilde{\mathbb{Z}}_f$ and $\tilde{\mathbb{Z}}^0$. Transfer to Step 3.

Step 1 : ***Logical Tests.*** Given the current bounds on the (x, y) variables, construct $\mathbb{Z}_f^+$ as

$$\mathbb{Z}_f = \mathbb{Z}_f^+ \cap \tilde{\mathbb{Z}}^+$$

where $\mathbb{Z}^+$, is given by (13). If for any i, $|K'_i| = 0$, go to Step 4. Calculate NT, as given by (15), if NT $\leq mn$ enumerate all remaining feasible completions, update the incumbent solution if necessary and go to Step 4. Otherwise, using the given δ^*, generate the corresponding strongest surrogate constraint and conduct the logical test of Section 3.1 to further restrict the set of free variables. Proceed to Step 2.

Step 2 : ***Bounding Step.*** Using the available δ^* to provide an advanced start for Algorithm CSA, derive a lower bound LB on the restricted problem corresponding to $\mathbb{Z}_f$. Update the incumbent solution value $\bar{v}$ using the imbedded heuristic, if possible. If $\lceil \mathrm{LB} \rceil \geq (1-\varepsilon)\bar{v}$, go to Step 4. Otherwise, proceed to Step 3.

Step 3 : ***Branching Step.*** Let z^* be the optimal z solution obtained from the most recent lower bounding problem. Select a branching index q, and a branching variable x_q or y_q using the rules of Section 4.3, update PS by adding q (or $q+m$) to it and setting the bounds on the corresponding variable. Update the lists PSB, PSLU, PSF, and PSNB accordingly, and return to Step 1.

Step 4 : ***Fathoming Step.*** Let $\bar{i}$ be the rightmost nonunderlined entry in PS. If such an entry does not exist then STOP; the incumbent solution is *ε-optimal.* Otherwise, set $\mathrm{PSF}(\bar{i}) = 1$ and delete all the entries to the right of $\bar{i}$ in each of the lists. Update the lists in the following manner :

$$\begin{aligned} \mathrm{PSB}(\bar{i}) &= \mathrm{PSNB}(\bar{i}) \\ \mathrm{PSLU}(\bar{i}) &= -\mathrm{PSLU}(\bar{i}). \\ \mathrm{PSNB}(\bar{i}) &= \phi \end{aligned}$$

Determine the current bounds on the (x, y) variables as given by the partial solution list. Since the δ^* values corresponding to each node are not being stored, set $\delta^* \equiv \delta_0^*$, where δ_0^* is the δ^* solution corresponding to node 0 and return to Step 1.

4.6 An Illustrative Example

Consider the following problem RDLAP, with $m = 4$ and $n = 10$. Let

$$s_i = 16,\ 23,\ 22,\ 26, \text{ for } i = 1, \dots, 4$$

and

$$d_j = 9,\ 14,\ 5,\ 14,\ 4,\ 1,\ 11,\ 14,\ 1,\ 14 \text{ for } j = 1, \dots, 10$$

The existing facility locations are taken as

$$(a_j,\ b_j) = (24,0),\ (5,15),\ (18,12),\ (13,4),\ (20,5),\ (0,5),\ (24,2),\ (14,10),\ (22,8),\ \text{and}\ (8,3)\quad \text{for}\ j = 1, \ldots, 10$$

Enumerating the "intersection points" belonging to the convex hull of the above 10 existing facility locations we have a total of 46 candidate source locations. Figure 1 shows the enumeration tree for this problem. The nodes are numbered in the order in which they were generated. The lower bound corresponding to each node k denoted by LB^k has been rounded to the next highest integer and the explicitly imposed restrictions on $(x,\ y)$ variables are shown on the branches. In this example the optimality tolerance ε was taken to be 0.01. The nodes 4, 6, 7 and 8 were fathomed as their corresponding lower bound (when rounded up) was within 1% away from the incumbent solution value, $\bar{v} = 201$. Node 3 was fathomed by enumerating all its remaining feasible completions. The optimal locations of the new facilities are

$$(x_i,\ y_i) = (5,12),\ (24,2),\ (14,10),\ (5,15)\quad i = 1, \ldots, 4$$

The optimal allocations are

$$\begin{aligned}
&w_{1\,2} = 14,\ w_{1\,3} = 1,\ w_{1\,6} = 1,\ w_{2\,1} = 9,\ w_{2\,5} = 3,\ w_{2\,7} = 11\\
&w_{3\,3} = 4,\ w_{3\,4} = 2,\ w_{3\,5} = 1,\ w_{3\,8} = 14,\ w_{3\,9} = 1\\
&w_{4\,4} = 12,\ w_{4\,10} = 14,\ w_{ij} = 0,\ \text{otherwise.}
\end{aligned}$$

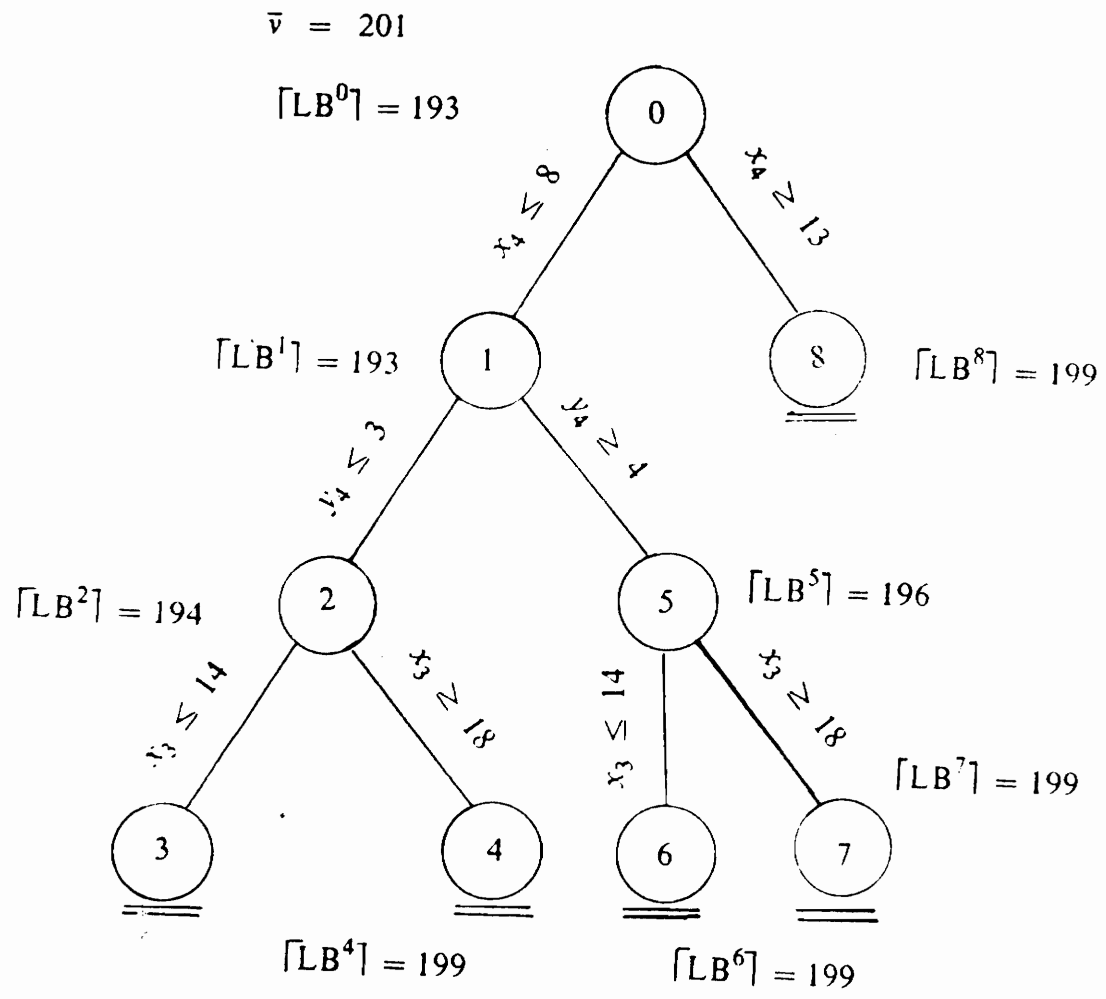
$\bar{v} = 201$
$\lceil LB^0 \rceil = 193$
0
$x_4 \le 8$
$x_4 \ge 13$
$\lceil LB^1 \rceil = 193$
1
8
$\lceil LB^8 \rceil = 199$
$y_4 \le 3$
$y_4 \ge 4$
$\lceil LB^2 \rceil = 194$
2
5
$\lceil LB^5 \rceil = 196$
$x_3 \le 14$
$x_3 \ge 18$
$x_3 \le 14$
$x_3 \ge 18$
$\lceil LB^7 \rceil = 199$
3
4
6
7
$\lceil LB^4 \rceil = 199$
$\lceil LB^6 \rceil = 199$

Figure 1. Implicit Enumeration Tree for the Illustrative Example

Chapter V

Computational Experience

In this chapter we present our computational experience with the proposed implicit enumeration algorithm. Some algorithmic refinements and modifications which were tested are also referred to in this discussion. We first describe the problem generation scheme, and then present computational results along with a discussion on the computational aspects of the algorithm.

5.1 Problem Generation

The following problem generation scheme is due to Sherali and Tuncbilek (1990). Data for the problems were randomly generated by using a uniformly distributed pseudo random number generator. The costs on the shipment routes were assumed to be directly proportional to the rectilinear distance and the quantity shipped. Distinct sink locations, (a_j, b_j), for $j = 1,\ldots, n$, were also randomly generated with each component being an integer lying in the interval [0, 25]. After generating a valid set of sink locations , the point $(a_{\min}, b_{\min})$ was taken as the origin, where $a_{\min} = \min_{j=1,\dots,n} a_j$, and $b_{\min} = \min_{j=1,\dots,n} b_j$, and all the locations were adjusted accordingly. Integer customer demands were generated over the interval [1, 15]. To generate the capacity of the m supply centers such that the total supply is equal to the total demand, the following iterative procedure was used.

Step 0. Set the counter $k = 0$. Generate initial integer supplies, $s_i^{(k)}$, for $i = 1,\dots, m$, over the interval $[1, \lfloor d_{\max} n/m \rfloor]$, where $d_{\max} = \max_{j=1,\dots,n} d_j$. Define $t_s^{(k)} = \sum_{i=1}^{m} s_i^{(k)}$, and let $t_d = \sum_{j=1}^{n} d_j$. Proceed to Step 1.

Step 1. If $t_s^{(k)} = t_d$, then stop and use the current supplies. Otherwise, compute $r = t_d / t_s^{(k)}$, and set $s_i^{(k+1)} = \max\{1, \lfloor r s_i^{(k)} \rfloor\}$ $\forall\, i = 1,\dots, m$. Increment the counter k by one, and proceed to Step 2.

Step 2. Compute $t_s^{(k)} = \sum_{i=1}^{m} s_i^{(k)}$. If $t_s^{(k)} < t_d$, then noting that $t_d - t_s^{(k)} \le m$, set $s_i^{(k)} \leftarrow s_i^{(k)} + 1$, for each $i \in \{1,\dots, t_d - t_s^{(k)}\}$, and terminate, using these revised supply values. Otherwise, return to Step 1.

5.2 Computational Results

The proposed algorithm was coded in FORTRAN and was implemented on an IBM 3090 computer. The transportation problems were solved using an improved specialized version of NETFLO. During the bounding step, if the cumulative execution time exceeded 370 cpu seconds, the execution of the algorithm was halted prematurely. The execution time for the input of the data for the problems and the output of statistics are not included in the reported total execution time.

Table 3 presents the computational results obtained. For problems with $mn \leq 80$, the optimality criterion ε was taken as 0.01, whereas for problems with $mn \geq 100$ it was taken as 0.05. With our time restriction, the largest problem solved to completion has $(m, n) = (5, 20)$ for which problem RDLAP has 110 variables and 25 constraints and the initial lower bounding problem LP_{DLAP} has over 20000 variables and approximately that many constraints. However, for the given upper bound on the cumulative execution time the same problem could not be solved to completion with $\varepsilon = 0.01$. Note that for problems where m is "small" compared to n, for example $(m, n) = (3,20)$ the initial RLT based lower bound is within 1% away from the optimum thereby leading to a successful termination at node zero itself. Also note that for larger sized problems, in particular, fluctuations in computational times for the same sized problems, e.g. for problems 26 and 27, are due to the amount of time required to perform the implicit enumeration when fathoming does not occur early on in the process, and moreover, the number of possible source locations K is different. Note the significant increase in computational effort for problems with $m = 5$, this is due to the combinatorial nature of the problem, with the total number of feasible binary solutions being K^m.

5.3 Implementation Comments

While solving the restricted problem at each node using Algorithm CSA, we check every 25 iterations for an improvement in the lower bound of at least 1 % of the incumbent solution value. If such an improvement is not detected then we terminate the solution

Table 3. Computational Experience with the Overall Algorithm

Problem	m	n	K	v^0	v^1	n_g	n_f	n_{opt}	t_{CSA}	t_{tot}	n_{CSA}	n_f^{CSA}	P_f	ε
1	2	4	7	0.9942	0.9942	1	1	1	0.01	0.04	1	1	–	0.01
2	2	4	7	1.0000	1.0000	1	1	1	0.02	0.05	1	1	–	0.01
3	2	4	6	0.9909	0.9909	1	1	1	0.03	0.06	1	1	–	0.01
4	3	5	7	1.0000	1.0000	1	1	1	0.14	0.19	1	1	–	0.01
5	3	5	11	0.9919	0.9919	1	1	1	0.01	0.10	1	1	–	0.01
6	3	5	5	0.8283	0.8283	1	1	1	0.35	0.39	1	1	–	0.01
7	3	8	30	0.9908	0.9908	1	1	1	1.50	1.11	1	1	–	0.01
8	3	8	23	0.9879	0.9879	5	3	1	1.49	1.24	3	1	88.41	0.01
9	3	8	17	0.9664	0.9664	1	1	1	1.41	1.62	1	1	–	0.01
10	3	12	46	0.9939	0.9939	1	1	1	0.24	1.66	1	1	–	0.01
11	3	12	33	1.0000	1.0000	1	1	1	0.01	0.63	1	1	–	0.01
12	3	12	39	0.9992	0.9992	1	1	1	0.62	1.75	1	1	–	0.01
13	3	15	116	0.9909	0.9909	1	1	1	2.53	6.29	1	1	–	0.01
14	3	20	190	0.9906	0.9906	1	1	1	3.00	9.31	1	1	–	0.01
15	4	10	46	0.9062	0.9062	9	5	1	5.94	9.42	8	4	90.76	0.01
16	4	10	33	0.9266	0.9266	73	37	1	26.05	28.43	73	23	59.13	0.01
17	4	10	39	0.9902	0.9902	1	1	1	0.76	2.06	1	1	–	0.01
18	4	12	63	0.9709	0.9709	25	13	1	12.66	16.79	24	9	86.51	0.01
19	4	12	43	0.9442	0.9442	19	10	1	6.95	12.98	18	10	60.47	0.01
20	4	12	65	0.9902	0.9902	1	1	1	3.47	6.30	1	1	–	0.01

Table 3. continued

Problem	m	n	K	v^0	v^1	n_g	n_f	n_{opt}	t_{CSA}	t_{tot}	n_{CSA}	n_f^{CSA}	P_f	ε
21	4	15	116	0.9880	0.9880	3	2	1	14.28	22.83	2	2	98.06	0.01
22	4	20	190	0.9951	0.9951	1	1	1	13.69	28.83	1	1	–	0.01
23	5	8	30	0.9369	0.9369	85	43	1	23.64	26.34	84	34	56.00	0.01
24	5	8	23	0.9047	0.9560	13	7	4	3.87	5.15	23	7	59.1	0.01
25	5	8	17	0.9930	0.9930	1	1	1	1.12	2.59	1	1	–	0.01
26	5	12	63	0.9261	0.9261	277	139	1	193.1	203.08	276	117	72.06	0.01
27	5	12	43	0.9902	0.9902	1	1	1	1.88	4.90	1	1	–	0.01
28	5	12	65	0.9688	0.9688	21	11	1	18.57	25.85	20	9	80.92	0.01
29	5	15	116	0.9794	0.9794	233	117	1	287.1	310.28	232	114	78.79	0.05
30	5	20	190	0.9533	0.9533	1	1	1	8.44	35.02	1	1	–	0.05

Legend: v^0 = initial lower bound / initial incumbent, v^1 = initial lower bound / final incumbent,

n_g = number of branch and bound nodes generated, n_f = number of nodes fathomed,

n_f^{CSA}= number of nodes fathomed by Algorithm CSA, n_{opt} = number of nodes generated before the final incumbent $\bar{v}$ is found,

K = number of candidate source locations, P_f is the percentage of binary variables fixed at zero via Section 3.1

t = total execution time in cpu seconds, t_{CSA} = total time spent in Alg. CSA,

n_{CSA}= number of node subproblems is solved, ε = optimality criterion as defined in Section 4.4.

of the current node problem, and branch on this node using the rules of Section 4.3. Note that although this method increases the number of nodes generated, the effort at each node is reduced considerably thereby not affecting the overall computational effort. Furthermore, we compared this method versus performing a fixed number of iterations of Algorithm CSA at each node, and detected that the latter strategy increased the computational burden significantly without improving the quality of the bounds.

At any node of the branch and bound tree, the solution δ^* corresponding to its parent node is used to provide an advanced start for Algorithm CSA. This is continued until some node is fathomed. Upon fathoming, the most recent unexplored complementary branch of the node is explored, and in this case, the solution δ^* corresponding to node 0 is used to provide an advanced start. Note that this scheme obviates the need for storing the solution δ^* corresponding to each node, thereby keeping storage requirements at a minimum.

In order to tighten the underlying lower bounding problem, we generated a polar cut from the current best WPGM available after the heuristic upper bounding phase of our algorithm. This cut was added to the problem and dualized along with the constraints of (5a). However, in our computational experiments with this strategy, the performance of the algorithm was not improved significantly, except for some strange instances. Hence, we did not adopt it for implementation. Recall that the generation of such a polar cut involves solving $m(K-1)$ parametric transportation problems and is a computationally expensive task. We also explored the possibility of tightening the lower bounding problem using more easily available **disjunctive cuts**. Given a binary solution z^*, for which some $z_{ip(i)} = 1$ is basic $\forall i$, the disjunctive cut is given by

$$\sum_{i=1}^{m} z_{ip(i)} \leq (m-1).$$

We obtained a collection of these cuts "generated" from the heuristic solutions encountered, and dualized them along with the constraints of (5a). In our computational experiments with this method the values of the dual multipliers associated with the disjunctive cuts nearly always turned out to be zero, hence not improving the performance of the algorithm.

Chapter VI

Conclusions and Recommendations for Further Research

In this thesis, we have studied the rectilinear distance location allocation problem subject to transportation constraints. The optimality characterization of the problem was exploited in order to provide an alternative equivalent formulation of the problem. Using the reformulation-linearization technique, tight linear programming relaxations were constructed. Here, various nonlinear implied constraints were first generated using products of original constraints with the variable bound factors, and the resulting nonlinear terms were subsequently linearized using an appropriate variable substitution strategy in order to obtain a relaxed linear program. The lower bounds obtained via the alternative formulations were compared, and the formulation giving tighter lower bounds was selected based on computational experimentation.

The special structure of the lower bounding problem was exploited in order to derive Lagrangian dual based lower bounds. This was shown to be dominant over a straightforward simplex based approach for solving the lower bounding problem, and was hence adopted for implementation. Also, the incumbent dual solution was used to generate a strongest surrogate type constraint. Logical tests were performed on this in order to permanently fix a number of binary variables at zero. At node zero, this can be interpreted as a model reduction phase. Using the proposed lower bounding scheme, an implicit enumeration algorithm was developed by partitioning the problem in the space of the original location variables in Problem RDLAP.

We solved problems of size $(m, \ n) = (4, 20)$ to within 1% of optimality and problems of size $(5, 20)$ to within 5% of optimality. We observed that our algorithm is more suitable for problems in which m is small compared to n. This is typical of real world problems.

Our algorithm in its proposed form handles the single product formulation of the rectilinear distance location allocation problem. A direct extension of this work would be to generalize the solution concept for the multi-product case, allowing for interactions among the sources as well, as in Shetty and Sherali (1980).

Appendix A

Selected Test Problems

This section presents some test problem data along with their corresponding solutions.

Problem 16 $m = 4, n = 10$.

$$s_i = 28,\ 18,\ 22,\ 22 \text{ for } i = 1, \dots, 4$$

and

$$d_j = 8,\ 11,\ 8,\ 7,\ 3,\ 13,\ 10,\ 13,\ 5,\ 12 \text{ for } j = 1, \dots, 10$$

$$(a_j,\ b_j) = (10,4), (2,17), (13,14), (0,16), (17,0), (21,14), (15,13), (10,0), (2,0), \text{ and } (0,2) \text{ for } j = 1, \dots, 10$$

Enumerating the "intersection points" belonging to the convex hull of the above 10 existing facility locations we have a total of 33 candidate source locations. The value of the final incumbent soluton $\bar{v}$ is 259. The optimal locations of the new facilities are

$$(x_i, y_i) = (15,14), (0,2), (2,16), (10,0) \quad i = 1, \dots, 4$$

The optimal allocations are

$$w_{1\,1} = 8,\ w_{1\,3} = 8,\ w_{1\,5} = 3,\ w_{1\,7} = 1,\ w_{1\,8} = 8,\ w_{2\,2} = 11,\ w_{2\,4} = 7$$
$$w_{3\,6} = 13,\ w_{3\,7} = 9,\ w_{4\,8} = 5,\ w_{4\,9} = 5,\ w_{4\,10} = 12,\ w_{ij} = 0,\ \text{otherwise.}$$

Problem 23 $m = 5,\ n = 8$.

$$s_i = 7,\ 3,\ 22,\ 26,\ 1 \text{ for } i = 1, \dots, 5$$

and

$$d_j = 2,\ 6,\ 6,\ 3,\ 9,\ 14,\ 5,\ 14 \text{ for } j = 1, \dots, 8$$

$$(a_j, b_j) = (24,23), (5,9), (18,0), (13,15), (20,12),$$
$$(0,4), (24,5), \text{ and } (14,5) \text{ for } j = 1, \dots, 8$$

Enumerating the "intersection points" belonging to the convex hull of the above 8 existing facility locations we have a total of 30 candidate source locations. The value of the final incumbent soluton $\bar{v}$ is 238. The optimal locations of the new facilities are

$$(x_i, y_i) = (20,12), (24,23), (0,4), (14,5), (20,12) \quad i = 1, \dots, 5$$

The optimal allocations are

$w_{1\,2} = 6,\ w_{1\,8} = 1,\ w_{2\,3} = 3,\ w_{3\,1} = 2,\ w_{3\,3} = 3,\ w_{3\,4} = 3,\ w_{3\,5} = 9$
$w_{3\,7} = 5,\ w_{4\,6} = 14,\ w_{4\,8} = 12,\ w_{5\,8} = 1,\ w_{ij} = 0$, otherwise.

<u>Problem 26 $m = 5,\ n = 12$.</u>

$$s_i = 27,\ 8,\ 20,\ 25,\ 27 \text{ for } i = 1, \dots, 5$$

and

$$d_j = 4,\ 1,\ 11,\ 14,\ 1,\ 14,\ 8,\ 11,\ 10,\ 13, 7,\ 13, \quad \text{for } j = 1, \dots, 12$$

$$\begin{aligned}(a_j,\ b_j) \ = \ & (25,10), (6,2), (19,3), (14,3), (21,0), (1,8), (25,6) \\ & (15,1), (23,12), (9,21), (0,5) \text{ and } (15,21) \ \text{ for } j = 1, \dots, 12\end{aligned}$$

Enumerating the "intersection points" belonging to the convex hull of the above 12 existing facility locations we have a total of 63 candidate source locations. The value of the final incumbent soluton $\bar{v}$ is 284. The optimal locations of the new facilities are

$$(x_i,\ y_i) \ = \ (14,3), (19,3), (1,8), (23,10), (9,21) \ \ i = 1, \dots, 5$$

The optimal allocations are

$w_{1\,2} = 1,\ w_{1\,3} = 1,\ w_{1\,4} = 14,\ w_{1\,8} = 11,\ w_{2\,6} = 8,\ w_{3\,10} = 13,\ w_{3\,12} = 7$
$w_{4\,1} = 4,\ w_{4\,5} = 1,\ w_{4\,7} = 8,\ w_{4\,9} = 10,\ w_{4\,12} = 2$
$w_{5\,3} = 10,\ w_{5\,6} = 6,\ w_{5\,11} = 7,\ w_{5\,12} = 4,\ w_{ij} = 0$, otherwise.

Problem 29 $m = 5, n = 15$.

$$s_i = 30,\ 37,\ 15,\ 48,\ 2 \text{ for } i = 1, \ldots, 5$$

and

$$d_j = 8,\ 11,\ 10,\ 13,\ 7,\ 13,\ 10,\ 3,\ 8,\ 10, 11,\ 11,\ 8,\ 1,\ 8 \text{ for } j = 1, \ldots, 15$$

$$\begin{aligned}(a_j,\ b_j) \ = \ & (25,5), (6,2), (19,10), (14,8), (21,3), (1,14), (25,23), (15,7)\\ & (23,23), (9,5), (0, 1), (15, 18), (12,22), \ (4, 0) \text{ and } (5,23) \ \text{ for } j = 1, \ldots, 15\end{aligned}$$

Enumerating the "intersection points" belonging to the convex hull of the above 15 existing facility locations we have a total of 116 candidate source locations. The value of the final incumbent soluton $\bar{v}$ is 729. The optimal locations of the new facilities are

$$(x_i,\ y_i) \ = \ (6,2), (19,8), (1,14), (15,23), (25,5) \ \ i = 1, \ldots, 5$$

The optimal allocations are

$$\begin{aligned}& w_{1\,2} = 11,\ w_{1\,10} = 10,\ w_{1\,11} = 8,\ w_{1\,14} = 1,\ w_{2\,1} = 8,\ w_{2\,3} = 10,\ w_{2\,4} = 9,\ w_{2\,5} = 7\\ & w_{2\,8} = 3,\ w_{3\,6} = 11,\ w_{3\,11} = 3\,, w_{3\,15} = 1,\ w_{4\,4} = 4,\ w_{4\,7} = 10\\ & w_{4\,9} = 8,\ w_{4\,12} = 11,\ w_{4\,13} = 8,\ w_{4\,15} = 7,\ w_{5\,6} = 2,\ w_{ij} = 0, \text{ otherwise.}\end{aligned}$$

Problem 30 $m = 5, n = 20$.

$$s_i = 29,\ 34,\ 32,\ 31,\ 33 \text{ for } i = 1, \ldots, 5$$

and

$$d_j = 11,\ 11,\ 8,\ 1,\ 8,\ 9,\ 11,\ 5,\ 15,\ 1,$$
$$3,\ 5,\ 14,\ 3,\ 12,\ 8,\ 15,\ 15,\ 2,\ 2 \text{ for } j = 1, \dots, 20$$

$$(a_j,\ b_j) = (25,14), (6,23), (19,7), (14,23), (21,5), (1,1)$$
$$(25,18), (15,22), (23,0), (9,23), (0,12), (15,17)$$
$$(12,17), (4,21), (5,11), (5,21), (2,17), (10,4)$$
$$(8,13), \text{ and } (3,16) \text{ for } j = 1, \dots, 20$$

Enumerating the "intersection points" belonging to the convex hull of the above 20 existing facility locations we have a total of 190 candidate source locations. The value of the final incumbent soluton $\bar{v}$ is 746. The optimal locations of the new facilities are

$$(x_i,\ y_i) = (25,17), (5,4), (2,17), (21,5), (6,22) \quad i = 1, \dots, 5$$

The optimal allocations are

$$w_{1\,1} = 11,\ w_{1\,4} = 1,\ w_{1\,7} = 11,\ w_{1\,8} = 5,\ w_{1\,12} = 1,\ w_{2\,6} = 9,\ w_{2\,15} = 11,\ w_{2\,18} = 14$$
$$w_{3\,3} = 8,\ w_{3\,5} = 8,\ w_{3\,9} = 15,\ w_{3\,18} = 1,\ w_{4\,10} = 1$$
$$w_{4\,11} = 3,\ w_{4\,12} = 4,\ w_{4\,13} = 14,\ w_{4\,17} = 5,\ w_{4\,19} = 2,$$
$$w_{4\,20} = 2,\ w_{5\,2} = 11,\ w_{5\,14} = 3,\ w_{5\,15} = 1,\ w_{5\,16} = 8,\ w_{5\,17} = 10,\ w_{ij} = 0, \text{ otherwise.}$$

References

Aikens, C. H. (1985) : "Facility Location Models For Distribution Planning", **European Journal of Operational Research,** 22:263-279.

Alameddine, A. R. (1990) : "A New Reformulation-Linearization Technique For The Bilinear Programming and Related Problems, With Applications To Risk Management", (unpublished) Ph.D. Dissertation, Department of Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, Virginia 24061.

Bazaraa, M. S., J. J. Jarvis, and H. D. Sherali (1990) : **Linear Programming and Network Flows,** John Wiley & Sons, New York.

Cooper, L. (1963) : "Location-Allocation Problems", **Oper. Res.** , 11: 331-343.

Cooper, L. (1964) : "Heuristic methods for Location-Allocation Problems", **SIAM review** , 6: 37-53.

Cooper, L. (1972) : "The Transportation Location Problem", **Oper. Res.,** 20: 94-108.

Francis, R. L. and J. A. White (1974) : **Facility layout and location: An Analytical Approach,** Prentice-Hall, Englewood Cliffs, N.J..

Gallo, G. and A. *Ülkücü* (1977) : "Bilinear Programming: An Exact Algorithm", **Math. Prog.** , 12: 173-194.

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

Handler, G. Y., and P.B. Mirchandani (1979) : **Location on Networks,** The MIT press, Cambridge, Massachusetts.

Hansen, P., M. Labbe', D. Peeters, and J. Thisse (1987) : Facility Location Algorithm, **Fundamentals of Pure and Applied Economics** , Systems of Cities and Facility Location, 22: 1-70.

Kennington, J. L., and R. V. Helgason (1980) : **Algorithms for Network Programming** , Wiley, New York.

Konno, H. (1976) : "A Cutting Plane Algorithm For Solving Bilinear Programs", **Math. Prog.** , 11 : 14-27.

Love, R. F., and H. Juel (1982) : "Properties and Solution Methods for Large Location-Allocation Problems", **Journal of Operational Research Society** , 33: 443-452.

Love, R. F., and J. G. Morris (1975) : "A Computational Procedure For the Exact Solution of Location-Allocation Problems with Rectangular Distances", **Naval Res. Logist. Quart.,** 22: 441-453

Love, R. F., J. G. Morris and G. O. Wesolowsky (1988) : **Facilities Location : Models & Methods**, Elsevier Science Publishers, Amsterdam, The Netherlands

Maruchek , A. S., and A. A. Aly (1987) : "An Efficient Algorithm for the Location-Allocation Problem With Rectangular Regions", **Naval Res. Logist. Quart.,** 28: 309-323.

Murtagh, B. A., and S. R. Niwattisyawong (1982) : "An Efficient Method for the Multi-Depot Location-Allocation Problem", **Journal of Operational Research Society** , 33: 629-634.

Murtagh, B. A., and M. A. Saunders (1987) : MINOS 5.1 User's Guide, Technical Report SOL 83-20R, Stanford University.

Murty, K. G. (1968) : "Solving the Fixed-Charge Problem by Ranking the Extreme Points", **Oper. Res.,** 16: 268-279.

Pardalos, P. M., and J. B. Rosen (1987) : **Constrained Global Optimization: Algorithms and Applications,** Springer-Verlag, Berlin.

Parker, R.G., and R. L. Rardin (1988) : **Discrete Optimization,** Academic Press, Inc., San Diego, CA 92101.

Selim, S. S. (1979) : "Biconvex Programming and Deterministic and Stochastic Location Allocation Problems", (unpublished), Ph.D. Dissertation, Georgia Institute of Technology.

Sherali, A. D., and C. M. Shetty (1977) : "The Rectilinear Distance Location-Allocation Problem", **AIIE Transactions,** 9: 136-143.

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

Sherali, H. D., and W. P. Adams (1990) : "Mixed-Integer Bilinear Programming ", Working Paper, Department of Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, Virginia 24061.

Sherali, H. D., and A. R. Alameddine (1990) : "A New Reformulation-Linearization Technique for the Bilinear Programming Problems", Paper, Department of Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, Virginia 24061.

Sherali, H. D., and 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., and F. Nordai (1988) : "NP-Hard, Capacitated, Balanced p-Median Problems on a Chain Graph With A Continuum of Link Demands", **Math. of Oper. Res.**, 13: 32-49.

Sherali, H. D., and C. M. Shetty (1980) : "A Finitely Convergent Algorithm for Bilinear Programming Problems using Polar Cuts and Disjunctive Face Cuts", **Math. Prog.**, 19: 14-31.

Sherali, H. D., and C. H. Tuncbilek (1990) : "A Squared Euclidean Distance Location-Allocation Problem", Paper, Department of Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, Virginia 24061.

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

Shetty, C. M., and H. D. Sherali (1980) : "Rectilinear Distance Location-Allocation Problem: A Simplex Based Algorithm", **Lecture notes in Economics and Mathematical Systems**, Extremal Methods and System Analysis, 174: 442-464.

Tuncbilek, C. H. (1990) : "A Squared Euclidean Distance Location-Allocation Problem", (unpublished), Master's Thesis, Department of Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, Virginia 24061.

Tuy, H. (1964) : "Concave Programming Under Linear Constraints", **Soviet Math.**, 5:1437-1440.

Vaish, H., and C. M. Shetty (1976) : "The Bilinear Programming Problem", **Naval Res. Logist. Quart.**, 23: 303-309.

Vaish, H., and C. M. Shetty (1977) : "A Cutting Plane Algorithm for the Bilinear Programming problem", **Naval Res. Logist. Quart.**, 24: 83-94.

Wendell, R. E., and A. P. Hurter Jr. (1973) : "Location Theory, Dominance, and Convexity", **Oper. Res.**, 21: 314-320.

Vita

Sridhar Ramachandran was born in New Delhi, India on December 16, 1967. He graduated with honors from Regional Engineering College, Trichy, India where he received a Bachelor of Engineering degree in Production Engineering, in 1988. He attended Virginia Polytechnic Institute and State University where he received a Master of Science degree in Industrial and Systems Engineering in May 1991. He would like to work in the industry in the field of applied operations research.

[illegible]