The Management School
Imperial College of Science and Technology,
Exhibition Road,
London SW7 2BX.

Optimisation of an Integrated Transport and Distribution System

by

Jalal Khoylou, B.Sc., M.Sc.

A thesis submitted for the degree of
Doctor of Philosophy of the University of London
and for the
Diploma of Imperial College

December 1989

ABSTRACT

This thesis is concerned with an Integrated Transport and Distribution System and exact solution approaches for it.

An important problem in the distribution industry is the optimal design of service networks with intermediate depots, subject to certain operational constraints. The version of the problem considered here is a three-stage trunking problem (origin city - collection depot - delivery depot - destination city) in which what is required is to determine the least cost system design for given constraints relating to customer service.

The problem has been formulated in a number of ways using mixed integer linear programming models that minimize the total costs when the structure of the distribution network is given. These models were then used to determine optimal locations for the depots.

The first algorithm developed for the problem was based on a branch and bound procedure. Lower bounds were obtained through a lagrangean relaxation which requires the solution of two generalized assignment problems. A subgradient optimization procedure was then used to update the lagrange multipliers and maximize the lower bounds. Subsequently reduction tests were used to increase the lower bounds further and to reduce the problem size.

The model has an imbedded structure which makes it an attractive candidate for integer programming decomposition. The second mixed integer programming model studies the application and acceleration of Benders decomposition for the above distribution system, and illustrates the potential flexibility of the Benders solution technique. This method is an improvement over the lagrangean approach.

Computational results are presented for medium size problems of up to 25 origin and destination cities, with 5 collection and distribution depots for both the branch and bound algorithm and the Benders decomposition method.

ACKNOWLEDGEMENTS

I am particularly grateful to my supervisor Professor Nicos Christofides. He was unfailingly generous with his time, his knowledge of combinatorial optimization and his friendship at difficult moments of my research.

Patrick Collins shared the same office during the early part of my research, and he became a close friend whose judgment and advice I valued. He was generous with his time in reviewing the manuscript and offering comments and suggestions at various stages, and he encouraged me at critical times.

Many other colleagues and friends who provided interesting ideas and suggestions, or merely leant a patient ear, are too numerous to list.

I am also grateful to my brother Farrokh, for allowing me to use his excellent word processing facilities.

This thesis is dedicated to my wife Fariba, who spent her early years of our marriage with my seemingly endless research, and then with the seemingly endless writing. My parents have followed all of this from a distance in Teheran, with a mixture of interest and scepticism. I have no idea what effect it may have had on them, but it is my impression that they both found it almost as interesting and exciting as I did. Together, this wise and loving group of people made even the worst of times the best of times.

to Fariba

Contents

Tables

Illustrations

CHAPTER 1

Introduction

Twenty five years ago Drucker [1962] described distribution as the "Economy's dark continent", and in the same year Parker [1962] summarized the area as "the last frontier for cost economies". These sentiments are, unfortunately, just as true for many companies today. Duffy (Financial Times [1982]) reports that

> "Transport and distribution activities account for up to 16 per cent of sales value in many U.K. companies, and industry could save about £2bn annually by improving management techniques".

A greater awareness of the need to look at the field of transportation and distribution as a management function is growing in certain sectors, and it has been given increasing attention in recent years (Duffy [1982]). However, the literature contains little information about strategic system design, or about the operation of

physically integrated transport and distribution systems where the service comprises multi-echelon collection and delivery.

1.1 The Parcel Distribution Problem

An important problem in the distribution industry is the optimal design of service networks with intermediate depots, subject to certain operational constraints. The version of this problem considered in this thesis is as follows:

There are N cities to be serviced by a carrier, and in each city there is a demand for commodities to be transported to each of the other N-1 cities. The commodities from the cities are taken to M collection depots ($M < N$) and after sorting according to their destination, they are dispatched in bulk to M local delivery depots, whereupon they are delivered to their final destination cities. This can be made more specific by reference to Figure 1.1, which gives a schematic checklist of the various kinds of data required to specify the model fully, In these terms, the objective is to determine:

- how many and which of the alternative collection and delivery depots to open
- which collection depots should be assigned to serve which origin cities
- which delivery depots should be assigned to serve which destination cities

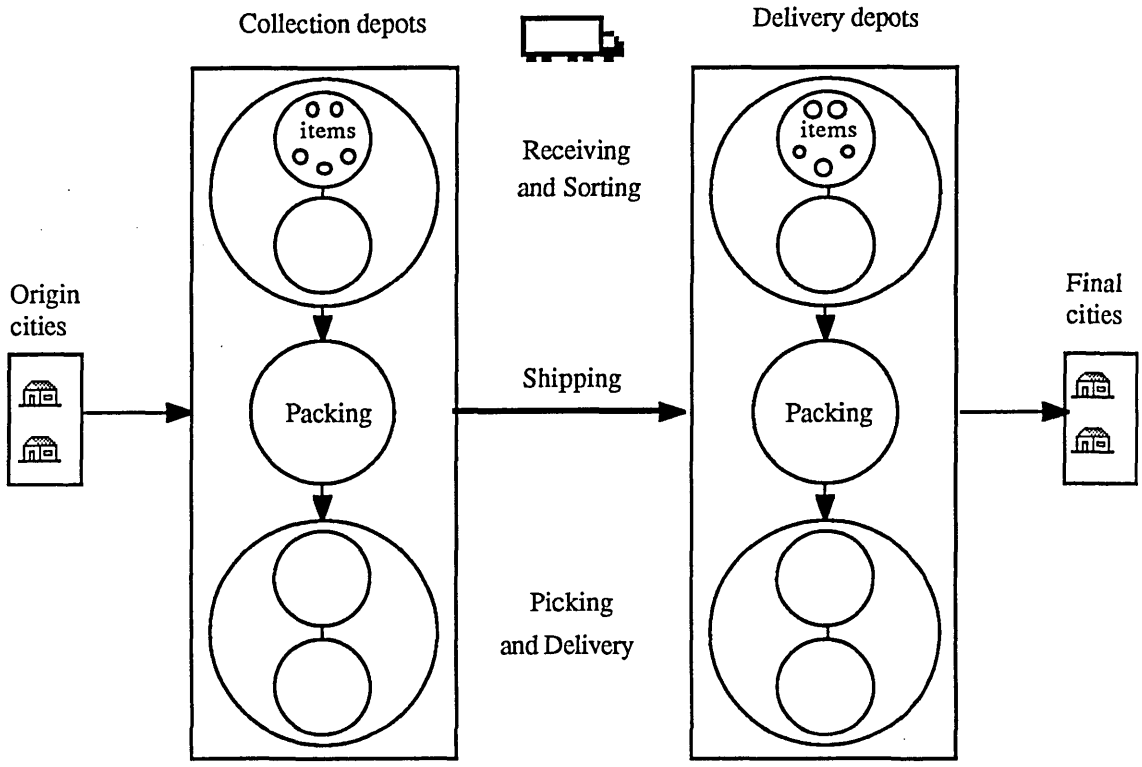
so as to minimize the total costs associated with

- collection
- delivery
- mass transportation

subject to constraints on

- commodity demands to be met in a given time horizon

- allowable throughput of collection depots
- allowable throughput of delivery depots
- each origin city to send a "commodity bundle" to only one collection depot
- each destination city to receive a "commodity bundle" from only one delivery depot.



(1) List of commodities
(2) List of origin cities
(3) Estimated shipment from origin city in a given time horizon.
(4) fixed cost of opening a route between origin city and collection depot
(5) List of candidate locations for depots
(6) Maximum allowable throughput for each collection and delivery depot
(7) Variable cost of transportation £/unit (origin city - collection depot - delivery depot - final city)
(8) List of final cities
(9) fixed cost of opening a route between final city and delivery depot

Fig. 1.1 A general three-stage parcel distribution system model. Schematic checklist of required data.

Thus, the problem is a three-stage parcel distribution problem (PDP) (origin city - collection depot - delivery depot - destination city) in which what is required is to choose that service network which minimizes the total distribution cost over a specific time horizon.

A brief discussion is now given of each of the 9 items shown in Figure 1.1. It is assumed that the model's time horizon is typically a day in duration.

1. A list of commodities

This is the final level of aggregation of a much larger number of individual parcels.

2. A list of origin cities

This is a list of cities requiring collection - of - parcels service. It is permissible, and usually desirable, to treat cities with a demand for collection service that happen to be geographically proximate to one another as a single city, and transport distances are measured to the centroid of demand in the region, or possibly to the largest city in the region.

3. Estimated shipment from origin cities

This is the estimated demand for shipment on a typical day from origin cities to destination cities.

4. Fixed cost of "opening" a route between origin city and collection depot

If a collection depot is assigned to serve an origin city, then there is a fixed cost associated with "opening" of that route, which includes annual running costs, if any, as well as fixed portions of operating costs. Some components of the fixed cost may arise

from linear approximation to nonlinear cost curves.

5. A list of candidate collection and delivery depot locations

Collection and delivery depot locations will be restricted to this list, which includes existing depots as well as promising new locations. In each computer run selected locations of depots can be prespecified as "open" or "closed".

6. The allowable throughput for collection and delivery depots

The need for throughput limits arises from two different kinds of consideration: practical and economic. Nearly all depots have a limited practical throughput capacity imposed by available physical structures and systems for commodities handling. Economic considerations, on the other hand, concern the need to restrict the size of collection and delivery depots to the range over which the associated variable and fixed cost estimates are valid.

7. Variable transportation costs

For each pair of origin and destination cities, there is a variable transportation cost via intermediate depots, which normally is proportional to the quantity of shipment and the distance. This cost can also include handling, and so on.

8. A list of destination cities

This is the final geographic aggregation of destination cities which receive the transported commodities. The aggregation should be to a level where there will be no necessity for different cities to be assigned to separate delivery depots.

9. Fixed cost of "opening" a route between delivery depots and destination cities

The model requires that no destination city is allowed to receive goods from more

than one delivery depot. Thus, each destination city must receive all its commodities from a single delivery depot. In order to "open" such a route a fixed cost is incurred, which is independent of the level of flow.

1.2 Historical Background

The aim of this review is to describe how the parcel distribution problem discussed in this thesis fits into the existing literature on location/distribution problems. Thus, it is not intended to provide a comprehensive survey of the location/distribution problems that have appeared in the literature over the years. The only contributions that are mentioned are ones relating to the development of the parcel distribution problem presented in this thesis. Its chief ancestors are, of course, the well known and much simpler "plant location" problems. If seminal works in location problems such as Fermat's from the early 1600's, Cavaleri's [1647], Sylvester's [1857], and in the 1800s by Steiner are disregarded, the simple plant location problems entered the stage of their present form in the period 1957-64. These problems are basically to locate plants, in such way as to minimize the cost of satisfying the demand for some commodity. In general there are fixed costs for locating the plants and transportation costs for distributing commodities between the plants and customers.

1.2.1 Simple Plant Location Problems

This problem is often referred to as the plant, depot, warehouse, site or facility location problem, and it has inspired an extensive literature in the last two decades. Krarup and Pruzan [1983] give a selective bibliography consisting of over 150 papers. Early reviews and shorter summaries of the state of the art can also be found in Balinski and Spielberg [1969], ReVelle, Marks and Liebman [1970], Eilon, Watson-Gandy and Christofides [1971], Hansen [1972], Lea [1973], Francis and Goldstein [1974], Salkin [1975], Guignard and Spielberg [1977], Cornuéjols [1978].

(1.1) x_{ij} amount
(1.2) proportions
THIS PROBLEM RECURS

ıematical formulation of these problems as an integer program has proved
ı the derivation of solution methods. The simple plant location problem
lized as follows: Consider a set of customers $I = \{1, \ldots, M\}$ with a
l for a single commodity. Let $J = \{1, \ldots, N\}$ be a set of sites where
ɔotentially located. Let f_j be the fixed cost of opening plant j and assume
cost per unit of transportation of goods from j to i. The simple plant
em is to "open" a subset of facilities in order to minimize the total cost
ıll the demands have to be satisfied.

ger linear programming formulation is obtained by introducing the
iables. Let $y_j = 1$ if plant j is "open" and $y_j = 0$ otherwise; x_{ij} is the
ıstomer i's demand which is satisfied from plant j. The integer linear

$$\text{inimize} \quad \sum_{j=1}^{N} f_j y_j + \sum_{i=1}^{M}\sum_{j=1}^{N} c_{ij} x_{ij} \tag{1.1}$$

$$\text{ɔject to} \quad \sum_{j=1}^{N} x_{ij} = 1 \qquad i = 1, \ldots, M \tag{1.2}$$

$$\sum_{i=1}^{M} x_{ij} \le M y_j \qquad j = 1, \ldots, N \tag{1.3}$$

$$x_{ij} \ge 0 \qquad \begin{matrix} i = 1, \ldots, M \\ j = 1, \ldots, N \end{matrix} \tag{1.4}$$

$$y_j \in \{0, 1\} \qquad j = 1, \ldots, N \tag{1.5}$$

Constraint (1.2) ensures that the demand of every customer is satisfied. Constraint (1.3) ensures that customers are supplied only from open depots.

Another widely used integer linear program in the literature that is equivalent to

(1.1) - (1.5) is obtained by replacing constraint (1.3) by the disaggregated set constraints

$$x_{ij} \leq y_j \qquad \begin{matrix} i = 1, \ldots, M \\ j = 1, \ldots, N \end{matrix} \qquad (1.6)$$

1.2.2 Algorithms for Simple Plant Location Problems

Numerous algorithms have been proposed for simple plant location problems. Krarup and Pruzan [1983] present a bibliography of this problem. The most significant approaches can be broadly classified into four groups.

- heuristic methods
- Benders decomposition methods
- branch and bound methods
- dual based methods

1.2.2.1 Heuristic Methods

The earliest attempts to tackle the problem were through the use of heuristic methods. One of the earliest heuristics for the simple plant location problem is due to Kuehn and Hamburger [1963] who actually present it for a wider class of location problems. It consists of two routines. An "add routine" selects locations sequentially in an order that maximizes the decrease of the objective function at each step. It stops when adding a new plant could only increase the objective. The second, pair-wise interchange, or "bump and shift" routine, eliminates (bumps) any plant that has become uneconomical because of the presence of other plants located subsequently by the "add routine". Then from this feasible solution it considers inter-changing (shifting) a location in the solution with one that is not. The procedure stops when it cannot be improved by any such inter-changes.

Kuehn - Hamburger's procedure, although over two decades old, still provides a kind of generic standard against which algorithm designers compete in computational

efficiency. In the recent literature Kuehn-Hamburger's procedure is referred to as a "greedy heuristic", and the shifting procedure as an "interchange heuristic". Reported computational experiments by Cornuéjols, Fisher and Nemhauser [1977], Rosing and Hillsman [1979], and Peeters [1980] suggest that such procedures work quite well.

Cornuéjols, Fisher and Nemhauser [1977] have studied the worst-case performance of location heuristics. They analyzed the "greedy", and the "interchange" heuristics in connection with Kuehn-Hamburger's procedure, and the "greedy-interchange" heuristic obtained by applying the two previous heuristics sequentially. The most significant results achieved are that the greedy heuristic has an error bound that varies between 0 and 1/e (e is the base of the natural logarithm) of the optimal solution value. The interchange heuristic (considered only for cases with all fixed costs equal to zero), on the basis of worst-case analysis, does not perform as well as the greedy heuristic. Moreover, it is not known to be polynomially time bounded. The greedy-interchange heuristic will never do worse than the greedy heuristic, but on the other hand it has no better error bound in the worst case.

Cornuéjols, Nemhauser, and Wolsey [1980], and Fisher and Hochbaum [1980] have carried out a probabilistic analysis of heuristics.

1.2.2.2 Benders Decomposition Methods

The formulation (1.1) - (1.5) is a mixed integer linear program and it can be solved using Benders decomposition [1962]. This approach was proposed by Balinski and Wolfe [1963] and appears to have been the first attempt to solve the simple plant location problem to optimality, other than the total enumeration approach in Stollsteimer [1963]. However, as reported in Balinski [1965], the computational experiments were discouraging and this method was abandoned until Geoffrion and Graves [1974] reported successful computational results for a more general location problem. In very

recent years Magnanti and Wong [1981] developed techniques to accelerate the convergence of Benders decomposition. They generated strong cuts from the set of feasible Benders cuts, and by so doing they were able to reduce the number of integer programs to be solved. This (accelerated) Benders decomposition and its successful application to Geoffrion-Graves's model will be fully discussed later on in this chapter.

1.2.2.3 Branch and Bound Methods

The first branch and bound algorithm to solve the simple plant location problem to optimality is due to Efroymson and Ray [1966]. Branchings are performed on the strategic y-variables in the enumeration tree by setting selected free variables y_j either to 0 or 1, thus creating two new branches in the tree (Figure 1.2). At each node of the tree the set J of locations is partitioned into three subsets:

$$K_0 = \{j \mid y_j = 0\},$$
$$K_1 = \{j \mid y_j = 1\},$$
$$K_2 = \{j \mid y_j \text{ is free}\}$$

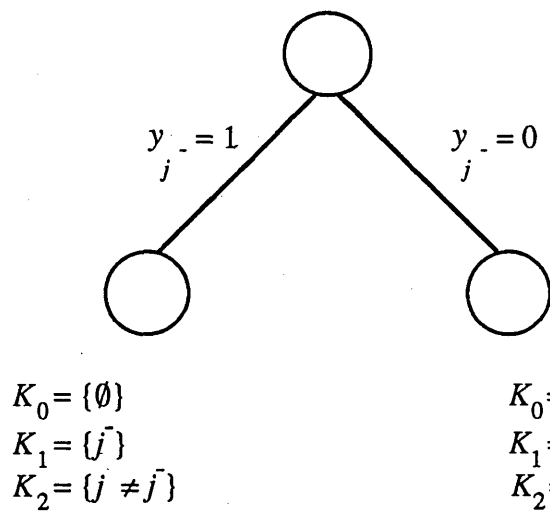


Fig. 1.2 Branching strategy

An upper bound on the value of the corresponding integer program where y_j is fixed for $j \in K_0 \cup K_1$, can be obtained by solving the linear programming relaxation (1.1) - (1.4) and

$$0 \leq y_j \leq 1 \tag{1.7}$$

The main result of Efroymson and Ray is a simple analytic solution of this linear program for every partition of J into $K_0 \cup K_1 \cup K_2$. Thus, the bound at each node of the enumeration tree can be computed very fast by inspection. The report on the computational efficiency is somewhat brief. Efroymson and Ray state that " a number of simple plant location problems ($M = 50, N = 200$) have been solved in an average of 10 minutes on an IBM 7094". These results are impressive even on an early generation computer. Improvements on the algorithm of Efroymson and Ray were made by Spielberg [1969], using an implicit enumeration procedure, but from a different perspective. The linear programming relaxation and its dual are solved for a sequence of fixed y_j, generated by a search algorithm. This is a single branch scheme in contrast to Efroymson and Ray. It starts with all y_j equal to either zero or one. At each node two solutions can be generated which will always be feasible. One solution is obtained by dropping the fixed charges for any plant not used in the sub-problem solution. A second feasible solution is obtained by solving a linear programming problem with all free variables set to 1. The minimum of these two solutions is compared with the current lower bound on the tree search, and if it is less, then this value will be the new bound. Spielberg reported computational results on a range of problem sizes. One example comprised 60 plant locations and 80 customers. The reported results for three problems of this size were: "150 CPU minutes on an IBM 360/50 halted short of optimality at 12,200 iterations from a cold start; 60 CPU minutes

on an IBM 360/50 solved to optimality with 9975 iterations, starting with an initial incumbent; and 60 CPU minutes on an IBM 7094 halted short of optimality after 103,000 iterations".

Two notable algorithms appeared in the same year, namely Khumawala [1972], and Hansen [1972]. The results achieved by Khumawala and Hansen can be regarded as significant improvements in the algorithm of Efroymson-Ray with respect to storage requirements and computing time. Khumawala investigated four criteria of branch selection based on three simplified partitions proposed by Efroymson and Ray. Each criterion embraced a pair of Largest and Smallest rules, namely Omega, Delta, Demand, and Z Integrality rules. Details on how these rules operate can be found in Khumawala [1972]. These branch selection rules were tested on a CDC 6500 computer. The Largest Omega rule, based on the maximum net gain achieved by setting a free variable y_j to 1, appears to perform best in efficiency, and Smallest Omega to be the poorest. Both the Largest and Smallest Delta and Demand rules generally performed poorly. Largest Z worked better than the Smallest Z rule. The average computing time with Largest Omega for the 16 test problems was approximately 3.8 seconds.

Hansen [1972] deals with two implicit enumeration algorithms (akin to the algorithm introduced by Spielberg) exploiting the concepts of "additive penalties". Recalling the definition of the three disjoint index sets (K_0, K_1, K_2) at each node of the tree, a penalty can be viewed as an increase of the lower bound when a free variable y_j, is fixed at either 0 or 1. If the values of a subset of free variables are fixed simultaneously, some at 0 and some at 1, and if the "addition" of the corresponding penalties to the lower bound can be shown to constitute a new valid bound, then these penalties are said to be "additive".

The common feature of all these algorithms is that the computation of bounds is based on the linear programming relaxation (1.1) - (1.4) and (1.7). This relaxation is known as the weak linear programming relaxation whereas (1.1), (1.2), (1.4), (1.6)

and (1.7) are known as the strong linear programming relaxation. These adjectives, weak and strong, emphasize the fact that the latter relaxation is tighter than the former, i.e. its set of feasible solutions is, in general, smaller. This follows since constraint (1.3) is a linear combination of constraints (1.6), but conversely, constraints (1.6) are not, in general, implied by (1.2), (1.3), (1.4) and (1.7).

It has been observed by ReVelle and Swain [1970] and other authors that the strong linear programming relaxation is so tight that its solution often yields an integer optimum for a large fraction of test problems proposed in the literature or at least a tight lower bound on it. Thus, a branch and bound algorithm based on the strong linear programming relaxation to compute the bounds is likely to perform well. Unfortunately, because the number of variables and constraints grows so fast with the size of the problem, it is not efficient to apply the standard simplex method directly. There have been various attempts to over-come this drawback. Marsten [1972] used parametric linear programming and a special implementation of the simplex method. Garfinkel, Neebe and Rao [1974] used the Dantzig-Wolfe decomposition. Schrage [1975] devised a generalized simplex method to deal with the variable upper bounds (1.6). Guignard and Spielberg [1977] suggested a version of the polytope (1.2), (1.4), (1.6), and (1.7). Finally Rosing and ReVelle [1978] reduce the number of rows and columns of the fully specified model.

1.2.2.4 Dual-Based Methods

The bounds obtained from the dual problem are theoretically as good as those obtained by the strong linear programming relaxation. This approach has the advantage that, in a branch and bound algorithm, bounds follow from any feasible solution of the dual of the linear program at the nodes of the enumeration tree. Thus, it might not be necessary to solve the dual to optimality.

The earliest work in this area was done by Bilde and Krarup [1967] but

unfortunately their work has not been translated into English for a full decade (Bilde and Krarup [1977]). A similar approach has been followed independently by Erlenkotter [1978]. The dual of the linear programming relaxation provides a lower bound, and heuristic methods are used in an attempt to maximize this bound. A branch and bound algorithm is then used to solve the problem to optimality. Erlenkotter's procedure was so effective that in 45 out of 48 problems that he tested, optimality was reached at the first node of the branch and bound algorithm.

Boffey [1978] gives a heuristic for solving the dual of the linear program; his method is related to that devised by Erlenkotter. Galvao [1980] has also reported a branch and bound algorithm incorporating a dual-based bound.

Slightly better perhaps are the results obtained with lagrangean duality concepts. Interest in lagrangean relaxation was aroused by the success of the technique in solving travelling salesman problems. Geoffrion [1974] has proposed the lagrangean dual formulation for the general integer programming problem and solved it using subgradient optimization. Cornuéjols, Fisher and Nemhauser [1977], Narula, Ogbu and Samuelson [1977], and Hanjoul and Peeters [1985] suggest performing lagrangean relaxation of the set of constraints (1.2), and report some computational results. An alternative lagrangean dual obtained by relaxation of the set of constraints (1.6) (instead of (1.2)) is mentioned by Krarup and Pruzan [1983]. Further interesting discussion is provided in Fisher [1981].

1.2.3 Capacitated Plant Location Problems

In the simple plant location problem, it is assumed that the plants have unlimited capacity such that, in principle, any plant can satisfy all demands. A natural extension to this problem is when each potential plant has a capacity, which is the maximum demand that it can supply. This problem is known as the capacitated plant location problem and it can be formulated as a mixed integer linear program:

$$\text{Minimize} \sum_{j=1}^{N} f_j y_j + \sum_{i=1}^{M}\sum_{j=1}^{N} c_{ij} x_{ij} \tag{1.8}$$

$$\text{Subject to} \sum_{j=1}^{N} x_{ij} = 1 \qquad i = 1, \ldots, M \tag{1.9}$$

$$\sum_{i=1}^{M} d_i x_{ij} \leq S_j y_j \qquad j = 1, \ldots, N \tag{1.10}$$

$$x_{ij} \geq 0 \qquad \begin{matrix} i = 1, \ldots, M \\ j = 1, \ldots, N \end{matrix} \tag{1.11}$$

$$y_j \in \{0, 1\} \qquad j = 1, \ldots, N \tag{1.12}$$

There are N possible plant locations to supply commodities for transportation to M customers. The binary variable y_j is either 1 or 0 according to whether plant location j is "opened" or not. The continuous variable x_{ij} represents the proportion of demand d_i of customer i which is supplied by a plant at location j. All demand must be met (constraint (1.9). Constraint (1.10) prevents upper bound violations of supply for open plants, where S_j is the capacity of plant j.

This problem is also known as the fixed charge transportation problem in the literature, since if a set of open plants are decided upon, the problem reduces to one of deciding the assignment of the customers to these plants. This problem is a transportation problem and can be easily solved (Fulkerson [1961]).

1.2.4 Algorithms for Capacitated Plant Location Problems

Branch and bound has been widely applied to the solution of capacitated plant location problems. However, the major differences between these methods concern the

bound computation, that is

- the type of relaxation selected,
- the procedure used to solve the relaxation,
- the techniques used for the strengthening of the bounds.

Sa [1969] extended the simple plant location problem developed in Efroymson and Ray [1966] to a capacitated plant location problem. The solution procedure proposed by Sa was based on obtaining bounds by replacing y_j variables by

$$1/S_j \sum_{i=1}^{M} d_i x_{ij}$$

The resulting relaxation is a transportation problem and it can be solved by a network flow algorithm [Fulkerson, 1961]. Bounds obtained from this relaxation become weak when the supply capacities get large. Rardin and Unger [1976b] proposed group-theoretical results to strengthen these bounds in the context of the more general fixed charge network flow problem. Aking and Khumawala [1977] generalized Khumawala's [1972] bounding rules to the capacitated case and, additionally, proposed a hybrid node selection rule. The node selection rule makes use of two parameters, α and β. Specifically, when a node is fathomed, the next node evaluated is selected to be the one with the least lower bound (breadth-first method). This procedure will eventually result in a large number of non-terminal nodes in the search tree. A depth-first method generally leaves few non-terminal nodes since the node selection priority favors any completion of lower level nodes, if indicated, before backtracking to higher level nodes. Aking-Khumawala's procedure implements a breadth-first method and continues until the number of non-terminal nodes reaches the level β, at which time a switch is made to depth-first to "clean up" some of the non-terminal nodes. When the number of non-terminal nodes is reduced to the level α, the procedure reverts to the

breadth-first method.

Ross and Soland [1977] show how the capacitated plant location problem can be formulated as a generalized assignment problem (GAP). This can then be solved using bounds obtained from lagrangean relaxation of the generalized assignment problem, which can be solved as a series of binary knapsack problems. No computational results are reported for this problem. However, results obtained for uncapacitated problems with 25 possible plant locations and 50 customers required less than 5 CPU seconds solution time on a CDC 70-94 computer.

Another approach was initiated by Davis and Ray [1969], and pursued by Schrage [1975], Geoffrion and McBride [1978], Nauss [1978], Guignard and Spielberg [1979], and Christofides and Beasley [1983]. They introduced the following set of disaggregated and redundant constraints to the original problem.

$$x_{ij} \leq y_j \qquad \begin{matrix} i = 1, \ldots, M \\ j = 1, \ldots, N \end{matrix} \qquad (1.13)$$

Their experience shows that inclusion of the disaggregated constraints gave very tight bounds and hence a small enumeration tree. Davis and Ray employed Benders decomposition to compute the lower bounds so as to exploit the fact that the linear programming relaxation to be solved reduces to a transportation problem when the y-variables are temporarily fixed. Schrage developed a simplex-based algorithm with specific pivoting rules and a compact basis inversion representation. Lagrangean relaxation was proposed by Geoffrion and McBride. The lagrangean problem is obtained by relaxing constraint (1.9) which means that the remaining problem decomposes into N smaller problems, one for each plant. The main thrust of their work is towards understanding lagrangean relaxation and only a small amount of computational experience on problems drawn from the literature is reported. Nauss

applied the same lagrangean relaxation augmented by the constraint

$$\sum_{j=1}^{N} S_j y_j \geq \sum_{i=1}^{M} d_i \qquad (1.14)$$

which ensures that the total supply capacity can meet total demand in any sub-problem solution. The lagrangean relaxation still decomposes into N smaller problems but, in addition, a knapsack problem must be solved. Nauss implemented subgradient optimization to obtain good lagrangean multipliers and tighter lower bounds. A branch and bound algorithm embeds these lower bounds, in conjunction with "pegging rules" which significantly reduce the number of branching operations, by invoking certain plants to be "pegged" open or closed. Nauss's method is reported to be computationally three times faster than Aking and Khumawala's, which is claimed to have outperformed many other existing algorithms. In a series of test runs with 25 plants and 50 customers conducted on an IBM 370/168, all but one problem were solved in less than 30 CPU seconds. A dual ascent method was successfully developed and implemented by Guignard and Spielberg, but no computational comparisons with other approaches were reported. Christofides and Beasley developed a similar approach to Nauss's with a slight improvement in their results.

The capacitated location problem surveys also include the monograph by Elshafei and Haley [1974] and the textbook by Salkin [1975]. Heuristic algorithms for the problem are described in Eilon, Watson-Gandy, and Christofides [1971].

1.2.5 Multi-Commodity Uncapacitated Plant Location Problems

All the plant location models discussed so far were assumed to be single commodity problems. The multi-commodity uncapacitated plant location problem is an extension of the plant location problem. There are few models in the literature on

location problems that deal explicitly with several commodities. This is probably because problems involving more than one commodity can often be handled by considering each commodity independently, or by not distinguishing which commodity is supplied to each customer. These approaches are not versatile when there are restrictions on the number of commodities that may be supplied from each location. These restrictions are modelled in multi-commodity plant location problems.

The multi-commodity uncapacitated plant location problem was first addressed by Warszawski and Peer [1973], who were motivated by a large construction project which necessitated modelling the location on a building site, of supply centers for three major commodities: concrete, reinforcing steel and building blocks. Each plant was capable of producing only one of the three commodities and had to be sited in such way that 38 customers could be most efficiently serviced.

Given a set of N possible locations for establishing plants, the multi-commodity plant location problem deals with the supply of P commodities from a subset of these plants to a set of M customers. For given positive costs c_{ijk} associated with the cost of supplying the demand for commodity k from location j to the customer i, and positive fixed costs f_{jk} representing the cost of opening a plant at location j to supply commodity k, the multi-commodity plant location problem seeks to minimize the total costs knowing that all the demands have to be satisfied. Let,

x_{ijk} be the continuous variable representing the fraction of customer i's demand for commodity k that is delivered from location j,

$y_{jk} = 1$ if plant j is open to supply commodity k,
$= 0$ otherwise.

The multi-commodity plant location problem can be formulated as a mixed integer

linear program:

$$\text{Minimize} \sum_{j=1}^{N}\sum_{k=1}^{P} f_{jk}\, y_{jk} + \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{k=1}^{P} c_{ijk} x_{ijk} \tag{1.15}$$

$$\text{Subject to} \sum_{j=1}^{N} x_{ijk} = 1 \qquad \begin{matrix} i = 1, \ldots, M \\ k = 1, \ldots, P \end{matrix} \tag{1.16}$$

$$x_{ijk} \le y_{jk} \qquad \begin{matrix} i = 1, \ldots, M \\ j = 1, \ldots, N \\ k = 1, \ldots, P \end{matrix} \tag{1.17}$$

$$\sum_{k=1}^{P} y_{jk} \le 1 \qquad j = 1, \ldots, N \tag{1.18}$$

$$x_{ijk} \ge 0 \qquad \begin{matrix} i = 1, \ldots, M \\ j = 1, \ldots, N \\ k = 1, \ldots, P \end{matrix} \tag{1.19}$$

$$y_{jk} \in \{0, 1\} \qquad \begin{matrix} j = 1, \ldots, N \\ k = 1, \ldots, P \end{matrix} \tag{1.20}$$

Constraint (1.16) ensures that the demand of every customer is satisfied for each commodity. Constraint (1.17) ensures that a plant is opened at each location if a customer is supplied with a commodity from that location, and constraint (1.18) requires that no more than one plant can supply commodity *k*.

Warszawski's paper discussed both a branch and bound procedure and a heuristic for solving the multi-commodity plant location problem. No computational results are provided for the branch and bound algorithm, as Warszawski concluded that large problems would consume excessive computer time.

Karkazis and Boffey [1981] combined and extended the work of Bilde and Krarup [1977] and Erlenkotter [1978], to develop two dual-based algorithms for solving multi-commodity plant location problems, both of which appear to outperform Warszawski's procedure. Neebe and Khumawala [1981] calculated bounds from the solutions to the simple plant location problem by removing constraint (1.18). The simple plant location problem was solved using the algorithm developed by

Khumawala [1972]. The branch and bound algorithm was then used to obtain the optimal solution. Laundy [1985] presented an interchange heuristic for producing a good solution to the multi-commodity location problem. The optimality of this solution was tested by using a dual heuristic to construct a solution to the dual of an LP relaxation of the problem. A tree search was then used if these heuristics failed to produce an optimal solution. Lower bounds for this tree search were obtained by applying lagrangean relaxation in conjunction with subgradient optimization. The reported results indicate that this algorithm outperformed the Karkazis-Boffey and Neebe-Khumawala algorithms. The total time required to solve a 3 commodity, 38 customer, and 9 possible plant location problem was 0.93 CPU seconds on an ICL 2970 computer.

1.2.6 Multi-Commodity Capacitated Single-Echelon Plant Location Problems

Marks, Liebman and Bellmore [1970] presented a more elaborate extension of the capacitated plant location problem to the optimal location of intermediate facilities in multi-echelon systems. They report reasonably good computational experience with a conventional branch and bound algorithm in which the linear programs, which specialize in capacitated trans-shipment problems, are solved by an out-of-kilter routine. Ellwein and Gray [1971] briefly considered the same model and indicated that their algorithm for the plant location problem can be extended to this case. However no computational experience was reported.

The more general multi-commodity feature of the model was introduced by Bartakke et al. [1971] who describe an application of Bonner and Moore's Functional Mathematical Programming System for the Univac 1108 to an industrial problem with 4 plants, 4 commodities, 10 intermediate distribution sites, and 39 customer zones. Elson [1972] also discussed the multi-commodity capacitated plant location problem,

concentrating on a single echelon of trans-shipment stocking points. Elson's model recognized the management option to expand existing plants, as well as to open new ones, and the need to provide customer service at different levels. Elson applied a mixed integer programming code (Ophelie Mixed System) to the solution of the multi-commodity problem. A test problem with 3 commodities, 5 plants, 14 customer zones, 22 distribution sites, and 2 service levels was solved to optimality in 174 CPU seconds on a CDC 6600 computer.

Geoffrion and Graves [1974] propose a multi-commodity, capacitated, mixed integer linear programing problem formulation focusing upon the optimal location of intermediate delivery depots between plants and customers. Geoffrion and Graves's model not only deals with plant location and commodity flows, but with customer assignment as well. In their model, sole-sourcing of customers is mandatory, and transportation costs are determined by the plant to customer route and the distance traveled.

The mathematical formulation of the problem has the following notation: Consider a set of customers $L = \{1, \ldots, M\}$ with a given demand for a set of commodities $I = \{1, \ldots, M\}$. Let $J = \{1, \ldots, N\}$ be a set of possible locations for establishing plants, and $K = \{1, \ldots, N\}$ be a set of sites where delivery depots can be potentially located. Let c_{ijkl} be the positive unit production and delivery cost associated with shipping commodity i from plant j to customer l through delivery depot k, and f_k be the fixed cost of establishing delivery depot k. The unit variable cost of throughput for delivery depot k is assumed to be v_k. The supply of commodity i at plant j is S_{ij}, and the demand of customer l for commodity i is D_{il}. Define $\underline{V}_k$, $\overline{V}_k$ to be the minimum and maximum allowable throughputs for delivery depot k.

The mixed integer linear programming formulation is obtained by introducing the following variables. Let,

x_{ijkl} be the number of units of commodity i shipped from plant j to customer l through delivery depot k,

y_{kl} $= 1$ if customer l is assigned to delivery depot k,

$= 0$ otherwise.

z_k $= 1$ if a delivery depot is established at site k,

$= 0$ otherwise.

The mixed integer linear program is:

$$\text{Minimize} \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{M} c_{ijkl}\, x_{ijkl} + \sum_{k=1}^{N}\left[f_k\, z_k + v_k \sum_{i=1}^{M}\sum_{l=1}^{M} D_{il}\, y_{kl}\right] \qquad (1.21)$$

$$\text{subject to} \sum_{k=1}^{N}\sum_{l=1}^{M} x_{ijkl} \le S_{ij} \qquad \begin{matrix} i = 1, \ldots, M \\ j = 1, \ldots, N \end{matrix} \qquad (1.22)$$

$$\sum_{j=1}^{N} x_{ijkl} = D_{il}\, y_{kl} \qquad \begin{matrix} i = 1, \ldots, M \\ l = 1, \ldots, M \\ k = 1, \ldots, N \end{matrix} \qquad (1.23)$$

$$\sum_{k=1}^{N} y_{kl} = 1 \qquad l = 1, \ldots, M \qquad (1.24)$$

$$\underline{V}_k\, z_k \le \sum_{i=1}^{M}\sum_{l=1}^{N} D_{il}\, y_{kl} \le \bar{V}_k\, z_k \qquad k = 1, \ldots, N \qquad (1.25)$$

$$y_{ij},\ z_{kl} \in \{\, 0, 1 \,\} \qquad \text{for all } i, j, k, l \qquad (1.26)$$

$$x_{ijkl} \ge 0 \qquad \text{for all } i, j, k, l \qquad (1.27)$$

Constraint (1.22) is the supply constraint. Constraint (1.23) requires that if a customer is assigned to a particular delivery depot then the demand must be satisfied via that depot. Constraint (1.24) ensures that each customer is supplied via a single

delivery depot. Constraint (1.25) enforces that the total throughput of an "open" delivery depot does not exceed the upper and lower limits.

A characteristic of Geoffrion-Graves's model is that commodities do not "forget" their source identity when traversing delivery depots. This is achieved by using the quadruply subscripted variable x_{ijkl}. Geoffrion and Graves were the first to introduce optimization over the entire path for commodity flows; previous models such as Elson's [1972] model employ two sets of triply subscripted variables x_{ijk}, and x_{jkl}. That is, the optimization is plant-to-delivery depot and then delivery depot-to-customer. The other advantage of quadruply subscripted notation is that direct plant-to-customer transportation can be easily accommodated. For business applications where there are perishable commodities, it is important to have this direct shipment feature.

Another characteristic of the model is that each customer's demand must be satisfied by a single delivery depot. In practical applications single sourcing is convenient administratively, and tends to reduce small delivery shipments.

Geoffrion, Graves, and Lee [1978] refined the model presented in Geoffrion and Graves[1974] to a form more amenable to practical application. In the revised version, sole-sourcing is only imposed on a "bundle" of similar items, not the totality of demand for all items. Upper and lower limits on delivery depots' throughput are not strictly enforced, and violation is allowed at a penalty cost. Lower as well as upper limits are imposed on plant capacity to enable some control over economies of scale. Throughput is computed as a weighted sum of items shipped through delivery depots, with each commodity having a distinct weight. Finally, the refinement permits the unit variable cost of throughput to differ by commodity. All of these refinements seem realistic in the context of real world problems. The Geoffrion-Graves-Lee model is the product of many man-years of empirical refinement, having been validated over a wide cross-section of industrial applications, and it would thus appear to represent the state-of-the-art for multi-commodity capacitated location problems. The implementation vehicle is a

user-oriented management support system known as ODS. The optimization procedure employed by ODS is based on the decomposition theory of Benders [1962]. Geoffrion [1976] reported the computational results of a large-scale distribution warehouse location analysis, using the Benders decomposition method, for Hunt-Wesson Foods, Inc., a firm which produces several hundred distinct food products at fourteen plants, and delivers these products to customers throughout the USA through a network of twelve intermediate delivery depots. Computational results on an IBM 360/91 computer ranged from just 16 CPU seconds in the case of a fixed configuration of 16 delivery depots, and solving for 249 assignment variables only; to 191 CPU seconds in the case of 30 delivery depots, and 513 binary variables.

1.3 Basic Notation

In this thesis several mixed-integer linear programming formulations of the PDP are given. All of these formulations are new and lend themselves to new solution techniques. However, before discussing any of the models, some basic common notation which is used in all of these models, and in the remainder of this thesis, is introduced.

Let model parameters be:

N Total number of cities (positive integer),

M Total number of candidate depots (positive integer),

i index for origin cities,

j index for collection depots,

k index for delivery depots,

l index for destination cities,

q_{il} required quantity for shipment from city i to city l,

Q_j^c handling capacity of the collection depot j (i.e. the limit on the total throughput which can be allocated to a collection depot located at j),

Q_k^d handling capacity of the delivery depot k (i.e. the limit on the total throughput which can be allocated to a delivery depot located at k),

c_{ijkl} average unit cost of transporting goods from city i to city l, via two intermediate collection and delivery depots j and k respectively,

f_{ij} fixed operating cost of assigning city i to the collection depot j,

f_{kl} fixed operating cost of assigning city l to the delivery depot k.

1.4 Branch and Bound Algorithm for the PDP Formulation

The PDP can be formalized as a set of origin cities, $I = \{1, \ldots, N\}$ with a demand for transportation of commodities to a set of destination cities $L = \{1, \ldots, N\}$. Let $J = \{1, \ldots, M\}$ and $K = \{1, \ldots, M\}$ be a set of sites where collection and delivery depots can be potentially located, respectively. N and M are assumed to be positive and integer.

The PDP can then be viewed as two separate capacitated plant location problems together with some other side constraints. The first plant location problem is to select collection depots from a set of possible sites to which cities are to be assigned (Figure

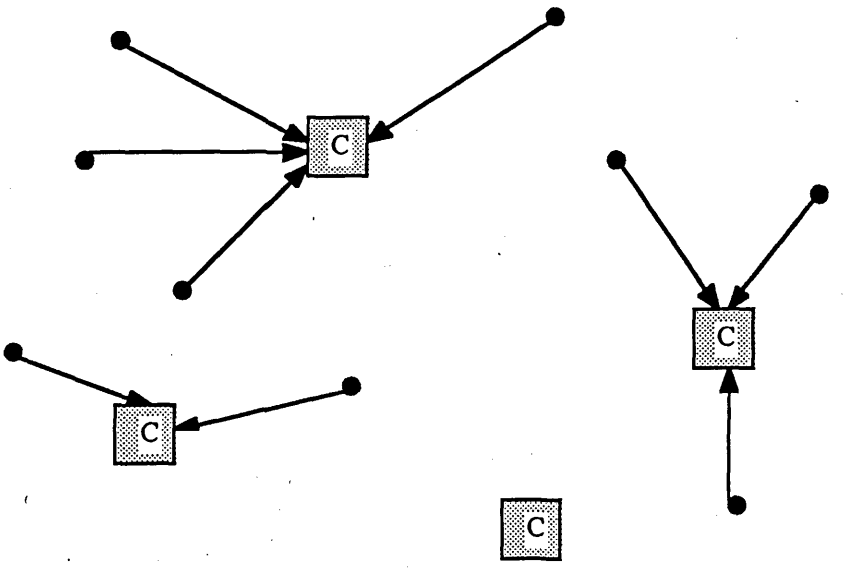


Fig. 1.3 The collection problem

1.3). These are the set of cities where the demand for the collection of commodities "originate". This problem is similar to the plant location problem in which origin cities represent plants, and the collection depots represent warehouses.

The second problem is to select delivery depots from a set of possible sites to which cities are to be assigned. These are the set of cities where the supplied commodities are delivered as their final "destination" (Figure 1.4). This problem is also similar to the plant location problem in which delivery depots represent warehouses, and the destination cities represent customers.

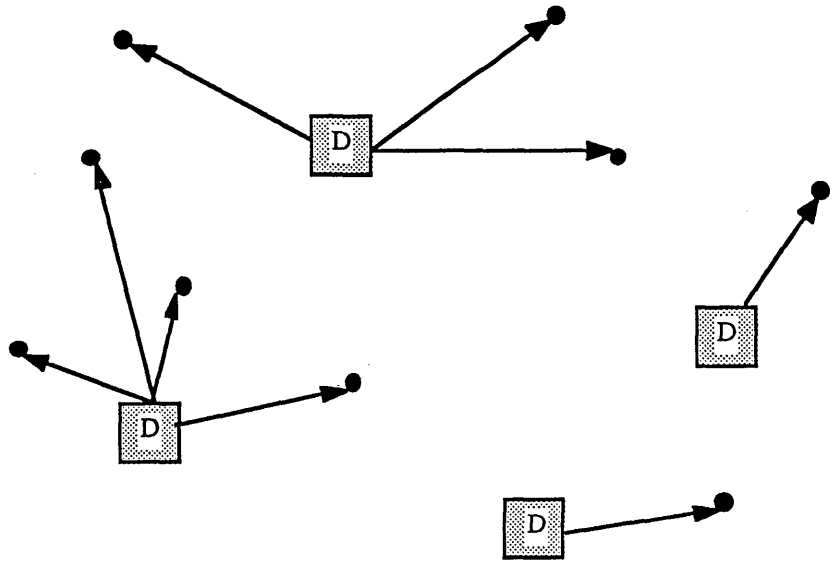


Fig. 1.4 The delivery problem

Ross and Soland [1977] were the first to propose, formulate and solve the capacitated plant location problem as a generalized assignment problem (GAP). The PDP problem has been formulated as an integer linear program in which its structure is based on the GAP. Although the GAP is an NP-complete problem (Garey and Johnson [1979]), nevertheless there exists an effective algorithm for it, such as Martello and Toth [1981]. It was hoped that the use of the GAP algorithm in exploiting the structure

of the PDP would yield a practical and general solution method for the problem.

Let,

$y_{ij} = 1$ if the origin city i is assigned to collection depot j,

$= 0$ otherwise.

$z_{kl} = 1$ if the destination city l is assigned to delivery depot k,

$= 0$ otherwise.

x_{ijkl} is the proportion of origin city i's supply which is shipped to the destination city l, via the intermediate collection depot j and delivery depot k.

The integer linear program then is

Problem P

$$\text{Minimize } Z_1 = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{M}\sum_{l=1}^{N} c_{ijkl}\, x_{ijkl} + \sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij}\, y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N} f_{kl}\, z_{kl} \qquad (1.28)$$

subject to

$$\sum_{j=1}^{M} y_{ij} = 1 \qquad i = 1, \ldots, N \qquad (1.29)$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il} y_{ij} \le Q_j^c \qquad j = 1, \ldots, M \qquad (1.30)$$

$$\sum_{k=1}^{M} z_{kl} = 1 \qquad l = 1, \ldots, N \qquad (1.31)$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il} z_{kl} \le Q_k^d \qquad k = 1, \ldots, M \qquad (1.32)$$

$$\sum_{j=1}^{M}\sum_{k=1}^{M} x_{ijkl} = 1 \qquad \begin{matrix} i = 1, \ldots, N \\ l = 1, \ldots, N \end{matrix} \qquad (1.33)$$

$$x_{ijkl} \le 1/2\ (y_{ij} + z_{kl}) \qquad \text{for all } i, j, k, l \qquad (1.34)$$

$$y_{ij},\ z_{kl},\ x_{ijkl} \in \{0, 1\} \qquad \text{for all } i, j, k, l \qquad (1.35)$$

Constraint (1.29) ensures that each origin city supplies its commodity to only one open collection depot. Similarly constraint (1.31) ensures that the demand for each destination city is supplied from only one open delivery depot. Constraints (1.30) and (1.32) may place limits on the amount of commodities that can be handled by collection and delivery depots. Constraint (1.33) follows from the fact that there is only one route between each pair of origin-destination cities. Constraint (1.34) ensures that if the origin city i is assigned to the collection depot j, and the destination city l is assigned to the delivery depot k, then there may be a flow from origin city i to destination city l via the assigned depots j and k. Constraints (1.35) are the integrality constraints.

In the above integer linear programming formulation constraints (1.29) and (1.30) represent the capacitated collection problem. The capacitated delivery problem is represented by constraints (1.31) and (1.32). The mass transportation from collection to delivery depots (Figure 1.5) is represented by the remaining constraints.

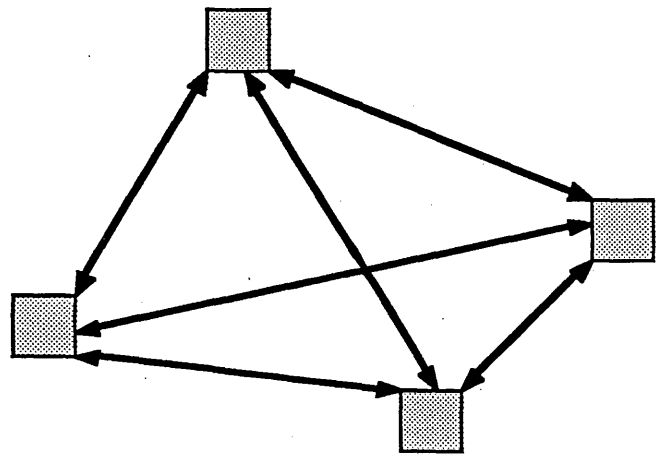


Fig. 1.5 Mass transportation problem

Many of the most successful algorithms for the simple plant location problem and

closely related problems employ branch and bound algorithms as their solution technique. Thus, as a first exact solution approach to the PDP, a branch and bound algorithm is considered. The above formulation seems to be a strong one, since the inter-linking constraint (1.34) considered has a disaggregated form. The linear program relaxation of the problem provides integer solutions in many instances. However, for a practical sized problem, the size of the PDP becomes excessive to solve by existing LP codes. If, however, constraint (1.34) is relaxed in a lagrangean fashion, then the remaining problem consists of two GAPs, so that their structure can be fully exploited by using special algorithms.

Martello and Toth [1981] suggest relaxing the assignment constraints (1.29) and (1.31) in the GAP using lagrange multipliers, and solving the remaining problems as knapsack problems. These relaxations are also implemented. The solution method based on these relaxations, and further improvement of bounds by applying reduction tests, provides sharp bounds that can be embedded into a tree search to obtain an exact optimal solution.

1.5 Benders Decomposition Method and its Application to the PDP

Problem P presented in the previous section is a large-scale integer linear programming problem and, for most practical applications, is computationally complex using a conventional branch and bound algorithm. In this section an alternative approach will be introduced which reduces the computational difficulty by decomposing the problem.

Benders decomposition is a method for solving mixed integer programming problems that has been applied successfully to a variety of applications (Florian et al. [1976], Richardson [1976], and Côté and Laughton [1982]). Geoffrion and Graves [1974] have had great success applying the algorithm to multi-commodity capacitated distribution systems. These contributions demonstrate the potential for using Benders'

method to solve specially structured mixed integer programs.

The PDP, as discussed, is in the class of location problems, as well as having an attractive feature that makes it possible to be decomposed as two GAPs and a transportation sub-problem. Thus this has been a reason to study Benders decomposition method and its application to the PDP.

1.5.1 Benders Decomposition Approach

It is perhaps helpful to give an intuitive interpretation of the Benders decomposition method so as to clarify the unjustified "mystery" that often surrounds it. This also highlights the inherent advantages of the decomposition method. The Benders decomposition can be viewed as a learning "trial and error procedure". This is shown schematically in Figure 1.6, where a trial configuration is selected and its performance evaluated. The results obtained in this trial will be the basis for another configuration selection. The iterative procedure is continued until the difference between the current trial configuration and the previous one appears small enough, when further calculations do not significantly improve the configurations.

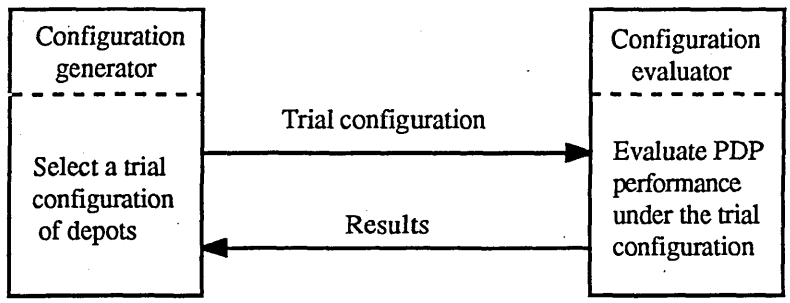


Fig 1.6 A simple configuration design for the PDP

A "trial system configuration" for the PDP was usually an assumed assignment of origin/destination cities to a set of open collection/delivery depots. An "evaluation" of

performance usually meant calculating the total fixed costs plus the total cheapest transportation flows from origin cities to destination cities, via the assigned depots, avoiding any violations of capacities of depots and other constraints.

The major short-coming of such a trial and error procedure, as Geoffrion, Graves and Lee [1978] pointed out, are:

(1) It ignores many alternative system designs.

(2) It is a very tedious and slow procedure, with many dangerous and tempting short-cut opportunities.

(3) Sensitivity analysis is extremely limited.

However the Benders decomposition method employs the refined and automated selection of trial configurations, as well as the evaluation of system performance. It uses the underlying structure of the trial configurations in the process of selection, which is an important feature. This method completely eliminates the first and second short-comings, and it significantly improves the final one.

The Benders decomposition method and its application to PDP will be discussed in the following section, where the inherent advantages of the method will become apparent.

1.5.2 PDP Weak Formulation

The PDP can be formulated as a mixed integer linear program:

Problem W

$$\text{Minimize} \quad Z_2 = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{M}\sum_{l=1}^{N} c_{ijkl}\, x_{ijkl} + \sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij}\, y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N} f_{kl}\, z_{kl} \qquad (1.36)$$

subject to

$$\sum_{j=1}^{M} y_{ij} = 1 \qquad i = 1, \ldots, N \qquad (1.37)$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il} y_{ij} \le Q_j^c \qquad j = 1, \ldots, M \qquad (1.38)$$

$$\sum_{k=1}^{M} z_{kl} = 1 \qquad l = 1, \ldots, N \qquad (1.39)$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il} z_{kl} \le Q_k^d \qquad k = 1, \ldots, M \qquad (1.40)$$

$$\sum_{k=1}^{M}\sum_{l=1}^{N} x_{ijkl} = \sum_{l=1}^{N} q_{il} y_{ij} \qquad \begin{matrix} i = 1, \ldots, N \\ j = 1, \ldots, M \end{matrix} \qquad (1.41)$$

$$\sum_{i=1}^{N}\sum_{j=1}^{M} x_{ijkl} = \sum_{i=1}^{N} q_{il} z_{kl} \qquad \begin{matrix} k = 1, \ldots, N \\ l = 1, \ldots, M \end{matrix} \qquad (1.42)$$

$$\sum_{j=1}^{M}\sum_{k=1}^{M} x_{ijkl} = q_{il} \qquad \begin{matrix} i = 1, \ldots, N \\ l = 1, \ldots, N \end{matrix} \qquad (1.43)$$

$$y_{ij},\ z_{kl} \in \{ 0, 1 \} \qquad \text{for all } i, j, k, l \qquad (1.44)$$

$$x_{ijkl} \ge 0 \qquad \text{for all } i, j, k, l \qquad (1.45)$$

Constraints (1.37) and (1.39) state that each origin/destination city may only be served by a collection/delivery depot respectively. Constraints (1.38) and (1.40) ensure that the total throughput limits through collection/delivery depots are not violated. Constraint (1.41) ensures that the total supply from the origin city must pass through

the assigned collection depot. Similarly, constraint (1.42) ensures that all the demand of a destination city is served via the assigned delivery depot. Constraint (1.43) ensures that all demands are satisfied.

Problem W is a large-scale mixed integer linear program. Benders decomposition reduces the computational difficulty in the following way.

An equivalent formulation of W is to minimize;

$$\left\{\sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij}\, y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N} f_{kl}\, z_{kl} + \min\left[\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{M}\sum_{l=1}^{N} c_{ijkl}\, x_{ijkl}\right]\right\} \qquad (1.46)$$

Subject to (1.37)-(1.43)

If y and z are held temporarily fixed, (1.46) together with (1.41)-(1.43) and (1.45) define a multi-commodity transportation problem. If a solution x to this transportation problem is held fixed temporarily, and y and z are permitted to vary, (1.46) together with (1.37)-(1.40) and (1.44) define an integer programming problem.

The Benders decomposition method is based on the convergence of upper and lower bounds obtained from alternately solving the two sub-problems. With the binary variables held fixed (that is, a fixed configuration) the commodity independent transportation sub-problems are solved for optimal transportation costs and flows. These sub-problems constitute restrictions on W since not all of the variables are free to vary. Consequently any sub-problem solution is an upper bound on W. Iteratively varying the configurations and solving the associated sub-problems produces a non-monotonic sequence of such upper bounds. Each time a sub-problem is solved, the optimal solution is used to solve the master problem for a new configuration. Specifically the master problem is:

$$\text{Minimize } Z_M = \sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij}\, y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N} f_{kl}\, z_{kl} + M_0 \qquad (1.47)$$

subject to (1.37)-(1.40), and

$$\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{l=1}^{N} \alpha_{ij}^{h} q_{il} y_{ij} + \sum_{i=1}^{N}\sum_{k=1}^{M}\sum_{l=1}^{N} \beta_{kl}^{h} q_{il} z_{kl} + \sum_{i=1}^{N}\sum_{l=1}^{N} \gamma_{il}^{h} q_{il} \le M_0$$

$$h = 1, \ldots, H \qquad (1.48)$$

$$y_{ij},\ z_{kl} \in \{ 0, 1 \} \qquad \text{for all } i, j, k, l$$

where α, β, and γ coefficients represent an optimal dual solution of the h^{th} transportation problem corresponding respectively to constraints (1.41)-(1.43).

The sense of the optimization is maximize instead of minimize since it is actually the transportation duals which are solved rather than the primals. The reason for this is that the solution space of the transportation dual is configuration-independent. Such is not the case with the primal. The master problem is a relaxation of W, and each additional transportation sub-problem solved contributes a new constraint of the form (1.48) to the master. These constraints are called Benders cuts, and because each new cut reduces the size of the solution space of the master problem, successive solutions constitute a monotonically increasing sequence of lower bounds on W. The master problem is solved for a new configuration and the procedure repeats. Termination occurs when upper and lower bounds converge to an ε gap.

1.5.3 PDP Strong Formulation

A major computational bottleneck in applying Benders decomposition is that the master problem which must be solved repeatedly has two GAPs, which are NP complete problems, plus Benders cuts. Even when the master problem is highly structured, the relaxation algorithm does not generally perform well due to its poor convergence properties (Geoffrion and Graves [1974]; Magnanti and Wong [1981]).

There are several possibilities for improvement;

- Formulating the problem in a "stronger model",
- Selecting strong cuts to add to the master when there are several choices,
- Modifying the master problem objective in order to select a different configuration.

Two different formulations of the same problem might be identical in terms of feasible solutions, but might be different in other ways such as having different relaxations of the master. Several researchers have illustrated the importance of problem formulation (Geoffrion and Graves [1974]; Cornuéjols et al. [1977]; Magnanti and Wong [1981]). Geoffrion and Graves show that proper model formulation, for multi-commodity distribution systems, can generally improve the computational efficiency of the Benders procedure. Magnanti and Wong provide theoretical insight concerning the role of formulation in accelerating Benders decomposition.

For a given model representation, it is possible in many instances to accelerate Benders decomposition by generating several different cuts, and selecting the "Pareto-optimal" cuts at each iteration (Magnanti and Wong [1981]). This selection process is accomplished by solving a linear program, starting from the multiple optimal solutions of another linear program.

There have been several proposals to alter the master problem objective function at each iteration (Nemhauser and Widhelm [1971]; Geoffrion and Graves [1974]; Marsten et al. [1975]). This is because the master problem is feasibility-seeking only. Thus it is permissible, literally, to introduce any attractive objective function. Nemhauser and Widhelm [1971] demonstrate scaling the constraints of the master problem to find the "geometrically centered" value for integers at each iteration. Geoffrion and Graves [1974] produce several different cuts in the disaggregated model of multi-commodity distribution systems. Marsten et al. [1975] had success in restricting the solution to the master at each iteration to lie within a box centered about

the previous solution.

These experiences encourage reformulation of the PDP problem in the following disaggregated form, in which the resulting multi-commodity subproblem, when binary variables are fixed, is separable by origin and destination cities and hence commodity-independent. It will be shown that this produces choices for Benders cuts at each iteration. Thus strong choice of Benders cuts limits the number of Benders iterations, and convergence to optimality is accelerated.

The PDP can then be formulated as a mixed integer linear program;

Problem S

$$\text{Minimize } Z_3 = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{M}\sum_{l=1}^{N} c_{ijkl}\, x_{ijkl} + \sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij}\, y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N} f_{kl}\, z_{kl} \qquad (1.49)$$

subject to

$$\sum_{j=1}^{M} y_{ij} = 1 \qquad i = 1, \ldots, N \qquad (1.50)$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il} y_{ij} \le Q_j^c \qquad j = 1, \ldots, M \qquad (1.51)$$

$$\sum_{k=1}^{M} z_{kl} = 1 \qquad l = 1, \ldots, N \qquad (1.52)$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il} z_{kl} \le Q_k^d \qquad k = 1, \ldots, M \qquad (1.53)$$

$$\sum_{k=1}^{M} x_{ijkl} = q_{il} y_{ij} \qquad \begin{matrix} i = 1, \ldots, N \\ j = 1, \ldots, M \\ l = 1, \ldots, N \end{matrix} \qquad (1.54)$$

$$\sum_{j=1}^{M} x_{ijkl} = q_{il} z_{kl} \qquad \begin{matrix} i = 1, \ldots, N \\ k = 1, \ldots, M \\ l = 1, \ldots, N \end{matrix} \qquad (1.55)$$

$$y_{ij},\ z_{kl} \in \{0, 1\} \qquad \text{for all } i, j, k, l \qquad (1.56)$$

$$x_{ijkl} \ge 0 \qquad \text{for all } i, j, k, l \qquad (1.57)$$

Constraints (1.50) - (1.53) are similar to (1.37) - (1.40), and represent the assignment of cities to depots. Constraint (1.54) states that if a city is assigned to a particular collection depot, then all the transportation flows must pass through that depot. Similarly, constraint (1.55) requires that all the transportation flows which are destined to a city must pass through the assigned delivery depot. Constraints (1.54) and (1.55) are disaggregated version of constraints (1.41) and (1.42).

1.6 Computational Complexity

An algorithm is a step-by-step procedure for solving a problem, which has a finite computing time for all data instances. For a given problem type (e.g. simple plant location problem) and a set of data instances of a given size, corresponding to a given input length (e.g. in simple plant location problem (M, N)), the complexity function for an algorithm expresses the largest amount of time required for solving the problem for an arbitrary data instance of that size. An algorithm is polynomial-time-bounded if for all data instances its computing time does not exceed some polynomial function of data size. Such algorithms are said to be polynomial. Otherwise the algorithm is called non-polynomial. Example algorithms in the polynomial class are sorting numbers, matching in graphs, network flow, and calculation of the shortest path and shortest spanning trees in graphs.

Cook [1971] and Karp [1972] have introduced the notion of NP-complete problems. This is a class of combinatorial problems that are equivalent in the sense that either all these problems can be solved by a polynomial time algorithm or none of them can be. Problems in this class include many classic combinatorial optimization problems such as the Clique Problem, the Steiner Network Problem, and the Vertex Covering Problem [see Maffioli, 1978]. A rigorous definition of this important notion

will not be given here, but instead the reader is referred to Aho, Hopcroft and Ullman [1974]. At present no polynomial time algorithm is known for solving any NP-complete problem, and it has been widely conjectured that none exists, although this question is still unsettled. A problem is said to be NP-hard if the existence of a polynomial time algorithm to solve it would imply that all NP-complete problems can be solved by a polynomial time algorithm. Thus to show that a problem (P) is NP-hard it suffices to find a polynomial transformation that reduces a known NP-complete problem to the problem (P) [Karp, 1972].

According to Guignard and Spielberg [1977]: "The simple plant location problem is one of the simplest mixed integer problems which exhibits all the typical combinatorial difficulties of mixed 0-1 programming and at the same time has a structure that invites the application of various specialized techniques". This statement indicates that the simple plant location problem is a hard problem to solve, or to use a more precise characterization, that is highly unlikely that an exact polynomial time bounded algorithm can ever be devised for its solution.

In Cornuéjols [1978] it is shown that the simple plant location problem is NP-hard. Then it is a simple exercise to show that other location problems mentioned in the previous section, namely the capacitated plant location problem, multi-commodity uncapacitated plant location problem, multi-commodity capacitated single-echelon plant location problem, and the parcel distribution problem which is considered in this thesis are also NP-hard.

The more detailed theory of computational complexity is closely related to the theory of computing machines and is beyond the scope of this thesis. A good survey of the area is provided by Maffioli [1978]. Karp [1975] provides the underlying mathematics as well as identifying problems in each class. An ongoing catalogue on the complexity status of various problems can be found in Johnson [1987].

1.7 Overview of the Thesis

Many of the most successful algorithms for the simple plant location problem and closely related problems depend on the solution of the strong linear programming relaxation or its dual.

These linear programs play such an important role because they actually solve the integer linear problem in many instances. Unfortunately the solution of these linear programs is difficult due to their large size. However the special structure of some problems can be exploited. In Chapter 2, a strong formulation for the PDP based on the generalized assignment problem will be presented, instead of the direct use of the linear programming relaxation of the problem. Its special structure is exploited using lagrangean relaxation to obtain lower bounds. The reduction tests are based upon these lower bounds, in order to reduce the problem size. A subgradient optimization procedure is applied to update the lagrange multipliers.

Chapter 3 introduces an alternate mixed integer linear program formulation of the PDP. This formulation has an attractive structure which can be decomposed as two separate sub-problems. Hence, the BDM and its application to this formulation of the PDP will be presented. This is followed in Chapter 4 by the development of a stronger formulation of the PDP for accelerating Benders decomposition. An acceleration technique for reducing the number of Benders iterations in the procedure is described. The structure of the transportation sub-problems are also fully exploited in this chapter.

A major computational bottleneck in applying Benders decomposition is that the master problem, which must be solved repeatedly, is an integer program. Chapter 5 describes several bounding strategies which are introduced for the solution of the master problem. Finally, a complete algorithm for the PDP based on a revised, accelerated BDM is described.

Finally, Chapter 6 presents conclusions and considers some problems suitable for further studies.

1.8 Conclusion

This chapter has reviewed the more significant work which has contributed to the solution of location problems and closely related problems of physical distribution management. It would appear that an abundance of tools is available for solving simple (uncapacitated) and capacitated plant location problems. Methods for dealing with multi-commodity problems are not abundant, however. Nevertheless some innovative branch and bound approaches and the application of Benders decomposition have made possible the optimization of some reasonably large systems. Little work has been reported in dealing with single echelon plant location problems; while work on solving multi-echelon systems is practically nonexistent. The need for the solution of such systems motivated the study of the three stage parcel distribution problem presented in this chapter. The solution methodology of this problem will be looked at in the remainder of this thesis.

CHAPTER 2

An Algorithm for the Parcel Distribution Problem Based on Lagrangean Relaxation

A branch and bound based method for solving the Parcel Distribution Problem (PDP) is developed in this Chapter. Lower bounds are obtained through a lagrangean relaxation of a 0-1 integer formulation of the problem. A subgradient optimization procedure updates the lagrange multipliers. Problem reduction tests based upon these lower bounds and the original problem are introduced, in order to reduce the problem size. The algorithm is tested on randomly generated data sets, and computational results on the performance of the developed code are presented.

2.1 Problem Formulation

The integer linear programming formulation of the PDP is finding binary variables y_{ij}, z_{kl}, and x_{ijkl} that satisfy

Problem P

$$\text{Minimize} \quad Z = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{M}\sum_{l=1}^{N} c_{ijkl}\, x_{ijkl} + \sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij}\, y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N} f_{kl}\, z_{kl} \tag{2.1}$$

$$\text{subject to} \quad \sum_{j=1}^{M} y_{ij} = 1 \qquad i = 1, \ldots, N \tag{2.2}$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il} y_{ij} \le Q_j^c \qquad j = 1, \ldots, M \tag{2.3}$$

$$\sum_{k=1}^{M} z_{kl} = 1 \qquad l = 1, \ldots, N \tag{2.4}$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il}\, z_{kl} \le Q_k^d \qquad k = 1, \ldots, M \tag{2.5}$$

$$\sum_{j=1}^{M}\sum_{k=1}^{M} x_{ijkl} = 1 \qquad \begin{matrix} i = 1, \ldots, N \\ l = 1, \ldots, N \end{matrix} \tag{2.6}$$

$$x_{ijkl} \le 1/2\ (y_{ij} + z_{kl}) \qquad \text{for all } i, j, k, l \tag{2.7}$$

$$y_{ij}, z_{kl}, x_{ijkl} \in \{0, 1\} \qquad \text{for all } i, j, k, l \tag{2.8}$$

y_{ij} is a 0-1 decision variable that is equal to one if a city i is assigned to a collection depot j, and it is equal to 0 otherwise. Similarly z_{kl} is a 0-1 decision variable for the assignment of a delivery depot k to a city l. x_{ijkl} is equal to one if (q_{il}) the quantity required for the shipment from city i to city l, passes through intermediate collection depot j and delivery depot k, and it is equal to 0 otherwise. The other variables are defined following the notation developed in Chapter 1.

Constraints (2.2) and (2.4) ensure that exactly one collection/delivery depot is allocated to each city. Constraints (2.3) and (2.5) ensure that no capacity limits are violated. Constraint (2.6) follows from the fact that there is only one route from each origin city i to each destination city l. Constraint (2.7) provides a correct logical relation between the variables. Constraint (2.8) is the integrality constraint. It is assumed that there exists a feasible solution y_{ij}, z_{kl}, x_{ijkl} to the above formulation.

Note that in this formulation it is assumed that each city requires shipment services. If at any city i there is no shipment to destination city l, then corresponding costs could be set to some large number in order to preclude the possibility of having it in an optimal solution.

2.2 Lower Bounding Procedure

To solve the above defined PDP a lower bound is derived, from a lagrangean relaxation of the problem, for use in a branch and bound procedure.

The lower bound on the optimal solution to problem P is generated by multiplying the constraints in (2.7) by the vector of positive lagrange multipliers $\lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_N\}$, and adding them to the objective function to obtain the following lagrangean dual problem:

Problem PR_λ

Minimize,

$$L_\lambda = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{M}\sum_{l=1}^{N} c_{ijkl}\, x_{ijkl} + \sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij} y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N} f_{kl}\, z_{kl} + \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{M}\sum_{l=1}^{N} \lambda_{ijkl}\,(x_{ijkl} - 1/2y_{ij} - 1/2z_{kl}) \tag{2.9}$$

subject to (2.2) - (2.6), (2.8)

After re-arrangement of terms, the lagrangean problem can be re-written as;

Minimize,

$$L_\lambda = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{M}\sum_{l=1}^{N}(c_{ijkl} + \lambda_{ijkl})\, x_{ijkl} + \sum_{i=1}^{N}\sum_{j=1}^{M}(f_{ij} - 1/2\gamma_{ij})\, y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N}(f_{kl} - 1/2\delta_{kl})\, z_{kl} \tag{2.10}$$

subject to (2.2) - (2.6), (2.8)

where $\gamma_{ij} = \sum_{k=1}^{M}\sum_{l=1}^{N} \lambda_{ijkl}$

and $\delta_{kl} = \sum_{i=1}^{N}\sum_{j=1}^{M} \lambda_{ijkl}$

The lagrangean dual problem PR_λ could be solved as two independent generalized assignment problems and a simple generalized upper bound problem.

2.3 Algorithm for Solving the Lagrangean Dual Problem

Step 1 Solve a generalized assignment problem for the collection part. This is given by

Problem C

$$\text{Minimize } C_\lambda = \sum_{i=1}^{N}\sum_{j=1}^{M} (f_{ij} - 1/2\gamma_{ij})\, y_{ij} \tag{2.11}$$

$$\text{subject to} \quad \sum_{j=1}^{M} y_{ij} = 1 \qquad i = 1, \ldots, N \tag{2.12}$$

$$\sum_{i=1}^{N} \alpha_i y_{ij} \le Q_j^c \qquad j = 1, \ldots, M \tag{2.13}$$

$$y_{ij} \in \{0, 1\} \qquad \text{for all } i, j \tag{2.14}$$

where in the above problem,

$$\alpha_i = \sum_{l=1}^{N} q_{il} \qquad i = 1, \ldots, N$$

Step 2 Similarly, solve a generalized assignment problem for the delivery part. This is given by

Problem D

$$\text{Minimize } D_\lambda = \sum_{k=1}^{M} \sum_{l=1}^{N} (f_{kl} - 1/2\delta_{kl})\, z_{kl} \tag{2.15}$$

$$\text{subject to} \quad \sum_{k=1}^{M} z_{kl} = 1 \qquad l = 1, \ldots, N \tag{2.16}$$

$$\sum_{l=1}^{N} \beta_l z_{kl} \le Q_k^d \qquad k = 1, \ldots, M \tag{2.17}$$

$$z_{kl} \in \{0, 1\} \qquad \text{for all } k, l \tag{2.18}$$

$$\text{where} \quad \beta_l = \sum_{i=1}^{N} q_{il} \qquad l = 1, \ldots, N$$

Step 3 Select a cheapest route from the modified cost matrix for each origin city i and destination city l. This part is represented by the following 0-1 dual problem.

Problem S

$$\text{Minimize } S_\lambda = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{M}\sum_{l=1}^{N} (c_{ijkl} + \lambda_{ijkl})\, x_{ijkl} \tag{2.19}$$

$$\text{subject to} \quad \sum_{k=1}^{M}\sum_{j=1}^{M} x_{ijkl} = 1 \qquad \begin{matrix} i = 1, \ldots, N \\ l = 1, \ldots, N \end{matrix} \tag{2.20}$$

$$x_{ijkl} \in \{0, 1\} \qquad \text{for all } i, j, k, l \tag{2.21}$$

This 0-1 problem could be solved by inspection since there are $N \times N$ independent problems for each pair of origin - destination cities. The solution is as follows;

(i) From a list of the $(C_{ijkl} + \lambda_{ijkl})$ for each pair of origin - destination cities, arrange them in ascending order and pick the least one.

(ii) Add this cost to the cost of S_λ (i.e. $S_\lambda$is the total cost of the cheapest route for each origin - destination pair).

Step 4 $L_\lambda = C_\lambda + D_\lambda + S_\lambda$, where $L_\lambda$is a lower bound on Z^* the optimal value of the PDP.

2.4 Details in Step 1 and 2 (Solving GAP)

In step 1 and 2 of the algorithm, it is required to solve a generalized assignment problem (GAP) with a modified cost function. The most widely known exact methods for solving GAP have been developed by Ross and Soland [1975], Martello and Toth [1981], and Fisher, Jaikumar and Van Wassenhove [1986]. Reported computational results with Martello-Toth's algorithm are comprehensive and provide satisfactory results up to date. Hence, a description of the GAP and an algorithm for solving it based on Martello and Toth [1981] will be presented in this section.

A general notation is used to describe the procedure, this can then be easily modified for each GAP at steps 1 and 2. Consider a generalized assignment problem as;

Problem G

$$\text{Minimize } Z_G = \sum_{i=1}^{N} \sum_{j=1}^{M} a_{ij}\, x_{ij} \tag{2.22}$$

$$\text{subject to} \quad \sum_{i=1}^{N} x_{ij} = 1 \qquad j = 1, \ldots, M \tag{2.23}$$

$$\sum_{j=1}^{M} w_{ij}\, x_{ij} \le b_i \qquad i = 1, \ldots, N \tag{2.24}$$

$$x_{ij} \in \{0, 1\} \qquad \text{for all } i, j \tag{2.25}$$

where $\{i | i = 1, \ldots, N\}$ is a set of depotindices, $\{j | j = 1, \ldots, M\}$ is a set of city indices, a_{ij} is the cost of assigning depot i to city j, and $w_{ij} > 0$ is the amount of a shipment from city j to depot i, and each depot has total maximum capacity b_i. Let,

$x_{ij} = 1$ if city j is assigned to depot i,

$= 0$ otherwise.

Constraints (2.23) and (2.25) specify that each city is to be assigned to exactly one depot. A depot, however, need not be assigned only to one city. Constraints (2.24) place limits on the amount of goods supplied by the cities.

The proposed algorithm for solving the above formulation requires relaxation of constraints (2.23). This relaxation would have no meaning in the formulation of problem G, since relaxation of (2.23) leads to a trivial solution with all the variables equal to zero, and an optimal value of zero. Therefore, an equivalent maximization

problem is considered by defining a constant positive vector $V = \{v_1, v_2, \ldots, v_M\}$ such that

$$v_j = \{\underset{i}{\text{Max}}\, a_{ij}\} \qquad j = 1, \ldots, M \qquad (2.26)$$

$$\text{and set} \quad P_{ij} = v_j - a_{ij} \qquad \begin{matrix} i = 1, \ldots, N \\ j = 1, \ldots, M \end{matrix}$$

Hence problem G can be re-written as

$$\text{Minimize } Z_G = \sum_{j=1}^{M} v_j \sum_{i=1}^{N} x_{ij} - \sum_{i=1}^{N} \sum_{j=1}^{M} P_{ij} x_{ij}$$

subject to (2.23) - (2.25)

Since $\sum_{i=1}^{N} x_{ij} = 1$, the above formulation becomes

Problem G'

$$\text{Maximize } Z_{G'} = \sum_{i=1}^{N} \sum_{j=1}^{M} P_{ij} x_{ij} \qquad (2.27)$$

subject to (2.23) - (2.25)

The problem G' is equivalent to G, and relaxation of constraint (2.23) would not result in a trivial problem. Thus, constraint (2.23) is removed and the remaining problem is N 0-1 single knapsack problems of the form

$$\text{Maximize } u_i = \sum_{j=1}^{M} P_{ij}x_{ij}$$

$$\text{subject to} \quad \sum_{j=1}^{M} w_{ij}x_{ij} \leq b_i \tag{2.28}$$

$$x_{ij} \in \{0, 1\} \qquad j = 1, \ldots, M$$

There are a wide range of algorithms in the literature for solving the knapsack problem, and include techniques based on:

- Dynamic programming,
- Network approaches,
- Generalized lagrangean methods,
- Implicit enumeration.

Dynamic programming algorithms are computationally efficient when the value of the capacity b_i is small. When b_i is quite large they generally tend to be very inefficient in both time and storage space requirements. Network approaches, discussed in Shapiro [1968], Shapiro and Wagner [1966], and in Frieze [1976], formulate the knapsack problem as a shortest route problem. These methods are usually inefficient because of the enormous size of the resulting networks. The performance of generalized lagrangean methods is computationally satisfactory only when approximate solutions are required; classical studies on these solution methods can be found in Everett [1963], Brooks and Geoffrion [1966], Nemhauser and Ullman [1968], Shapiro [1971]. The first implicit enumeration method was a breadth-first branch and bound procedure presented by Kolesar [1967]; the large computer memory and time requirements of Kolesar's algorithm were greatly reduced by the depth-first branch and bound method of Greenberg and Hegerich [1970]. Horwitz and Sahni [1974] proposed a highly effective branch and bound procedure, based on Greenberg-Hegerich's

algorithm. Further improvements have been presented by Ahrens and Finke [1975], Barr and Ross [1975], Fayard and Plateau [1975], Nauss [1976], and Martello and Toth [1977].

Thus, the technique selected for solving the 0-1 knapsack problem was Martello and Toth's algorithm [1977], whose computational performance is good in terms of both time and memory requirements. The upper bound obtained is

$$U = \sum_{i=1}^{N} u_i$$

If the solution of the relaxed problem (2.28) satisfies constraint (2.23), then the optimal solution is obtained for problem G'. Otherwise a depth-first branch and bound algorithm is used to obtain an optimal solution. Suppose that the solution does not satisfy constraint (2.23), then two types of infeasibility could be defined

$$J_0 = \left\{ j \mid 1 \le j \le M, \sum_{i=1}^{N} x_{ij} = 0 \right\}$$

$$J_1 = \left\{ j \mid 1 \le j \le M, \sum_{i=1}^{N} x_{ij} > 1 \right\}$$

The upper bound can be improved by calculating the penalty to be paid in order to satisfy J_0 & J_1.

$$\bar{r}_{ij} = \text{Min}\left[u_i \mid x_{ij} = 1\right] \qquad \forall\, i, j \in J_0$$

$$r_{ij} = \text{Min}\left[u_i \mid x_{ij} = 0\right] \qquad \forall\, i \in I_j, j \in J_1, \text{ where } I_j = \{s \mid x_{sj} = 1\}$$

The corresponding lower bound is

$$\bar{l}_j = \underset{i}{\text{Min}} \left[(u_i - \bar{r}_{ij}) \mid j \in J_0 \right]$$

$$l_j = \left\{ \left[\sum_{i=1}^{N} (u_i - r_{ij}) - \underset{i}{\text{Max}} (u_i - r_{ij}) \right] \mid j \in J_1, i \in I_j \right\}$$

Hence the upper bound can be improved in the following way

$$u^* = \sum_{i=1}^{N} u_i - l_{j'}$$

where $l_{j'} = \text{Max} \{\bar{l}_j, l_j\}$ $\qquad j \in J_0 \cup J_1$

The values of $l_{j'}$ can be used in order to select a branching variable. A branching is then carried out on a variable j'. If $j' \in J_0$ then for each i a node is generated by setting $x_{ij'} = 1$. But if $j' \in J_1$ then for each i for which $x_{ij'} = 0$ a node is generated. The algorithm proceeds on each node in a depth first manner until an optimal feasible integer solution is found.

2.5 Determination of a Feasible Solution

A solution to the lagrangean problem PR_λ is not necessarily a feasible solution to the PDP, and consequently a feasible solution must be generated initially. In searching for a feasible solution, of course, the better the initial solution, the shorter is the subsequent tree search. A number of different procedures were adopted to search for a initial feasible solution. The following procedure is typical.

Step 1 List the costs of all possible combinations of origin city i - collection depot j, and rank them in ascending order.

Step 2 Assign origin cities i to collection depots j in order, starting with the lowest cost and eliminating higher cost alternatives as appropriate. When a collection depot's capacity (Q_j) is reached, all further assignments to that depot are eliminated. Continue until all cities i are assigned to a depot j.

Step 3 List the costs of all possible combinations of destination city l - delivery depot k, and rank them in ascending order.

Step 4 Assign collection depots j to delivery depots k in order, starting with the collection depot j containing parcels for the delivery depot k with lowest destination city delivery cost, and eliminating higher cost alternatives as appropriate. When a delivery depot's capacity Q_k is reached, all further assignments to that depot are eliminated.

Step 5 Deliver all parcels to correct destination cities.

Step 6 Calculate the total cost of deliveries and define this as Z_{UB}, the upper bound on the solution.

Note that this procedure does not consider the cost of delivery from collection depots to delivery depots; the cost of setting up local collection and delivery routes is assumed to be greater than the cost of mass trunking between collection and delivery depots.

2.6 The Subgradient Optimization Procedure

One objective in determining a new set of lagrange multipliers λ is to increase the lower bound (L_λ), and to force the variables to more closely satisfy the constraints in (2.7). Therefore the dual phase is defined as the maximization problem:

$$\underset{\lambda}{\text{Max}}\, L_\lambda$$

The iterative method of updating the lagrange multipliers is called subgradient optimization and is described in Held, Wolfe and Crowder [1974], as well as in many other papers.

In the search for lagrange multiplier values that maximize the lagrangean expression, it has been shown (see Fisher [1981] e.g.) that a subgradient vector υ, computes a vector which "points" in the general direction of the optimal λ. The subgradient direction at the h^{th} iteration is as follows:

$$\upsilon^h_{ijkl} = x_{ijkl} - 1/2y_{ij} - 1/2z_{kl} \qquad \text{for all } i, j, k, l$$

Using a step size t_h, the method computes new values for λ^h by moving in the direction υ_{ijkl}. The new set of multipliers λ^{h+1} is

$$\lambda^{h+1}_{ijkl} = \lambda^h_{ijkl} + t_h \upsilon^h_{ijkl} \tag{2.29}$$

A commonly used step size is given by

$$t_h = \frac{\mu_h(\bar{Z} - \bar{L}^h_\lambda)}{\| \upsilon^h \|^2} \tag{2.30}$$

$\bar{Z}$ is an over-estimate of the maximum value of L_λ. If $\bar{Z}$ is equal to L^*_λ, the sequence of $\{\lambda^h_{ijkl}\}$ is guaranteed to converge to that value of λ which maximizes L_λ as $h \rightarrow \infty$. For computational convenience, however, $\bar{Z}$ is usually set equal to an upper bound on Z^*, the value of the optimal solution to the PDP. While this choice

does not guarantee convergence, no convergence problem occurred in any of the large number of problems tested.

The double bars || . || indicate the Euclidean norm. μ_h is a positive number ≤ 2 determined by the user. Typically, its starting value is initialized at 2. Since an iteration maximum is specified for a node, the value of μ_h should decrease to force smaller steps as the search nears its end. The most promising approach, discussed in Held, Wolfe, and Crowder [1974], is to decrease μ_h from 2 to 1 after half of the iterations, then to 0.5 at the three-quarter point. As L_λ increases and Z decreases, the numerator of (2.30) decreses, further reducing the step-size, and allowing the algorithm to "fine tune" λ.

2.6.1 Modification of the Subgradient Optimization Procedure

In computational experiments a modification of the standard subgradient procedure involves the use of a direction-correcting factor, proposed and tested successfully by Crowder [1976] and Mulvey and Crowder [1979]. Instead of moving along the subgradient direction in h^{th} step, as defined by equation (2.29), the following substitution was made

$$\upsilon^h = \upsilon^h + 0.75\upsilon^{h-1}$$

This modification leads to improved convergence over the standard subgradient optimization procedure. After testing various coefficients between 0 and 1, it was concluded that a smoothing factor of 0.75 gives the best convergence rate for the PDP.

Let LB_1 denote the lower bound obtained by the subgradient optimization procedure, which does not use the modification just described. LB_2 is the bound obtained with the modification. A preliminary experiment on a set of ten randomly generated problems of small size revealed that LB_2 generated the best bounds (Table 2.1). The procedure stopped whenever the step size in the direction of the subgradient was smaller than a pre-specified quantity. LB_2 was then incorporated into the branch

and bound code.

In table 2.1 the number of iterations used by LB2 was substantially less than LB1, being as low as 37% (in case 6). The average number of iterations was only 66% of the number used by LB1, and the worst case was 88% (case 5).

Likewise, the time taken by LB2 was less than LB1: the average value for the 10 cases is 71% of the time taken by LB1, the lowest being 37% (case 6), and the highest 98% (case 2).

Figure 2.1 illustrates the lower bounds obtained by LB1 and LB2 in problem number 7. It is clear that after the first 10 iterations, LB2 converges towards the optimal solution much faster than LB1, reaching optimality after only 93 iterations, whereas LB1 took 146 iterations.

Fig. 2.1 Ascent of lagrangean bounds for the problem 7

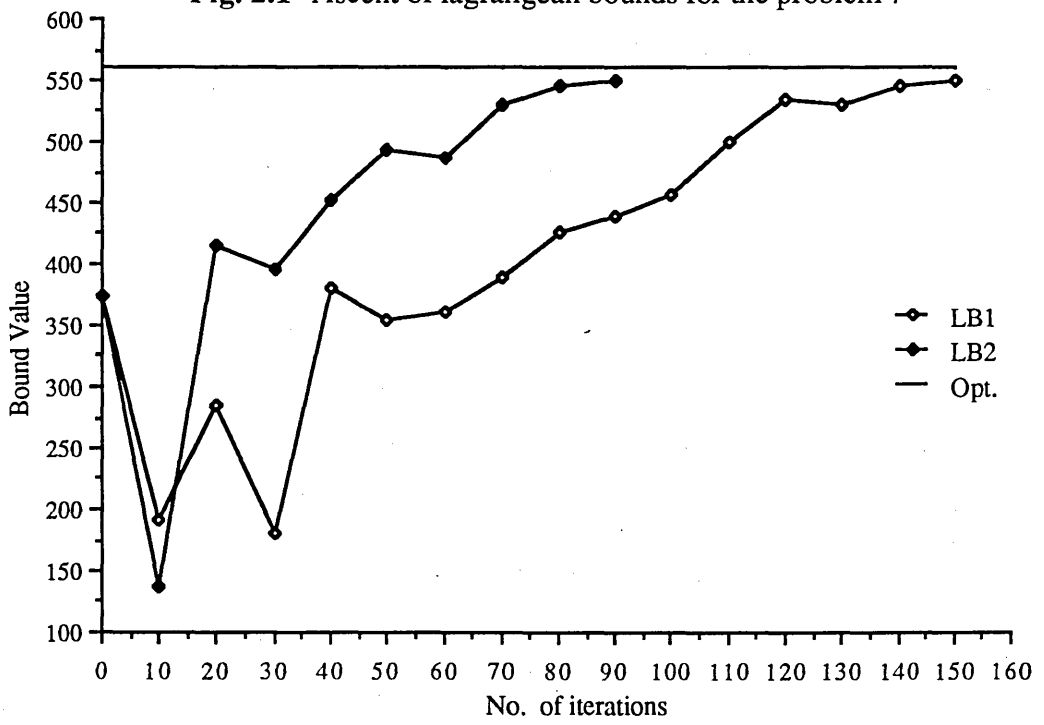


Table 2.1
Evaluation of the lagrangean bounds using two different step sizes

Problem Number	Problem Size				Lower Bound (LB1)			Lower Bound (LB2)			
					No. of		Time	No. of		Time	Optimal
	i	j	k	l	Iterations	Bound	(CPU Sec.)	Iterations	Bound	(CPU Sec.)	Solution
1	5	3	3	5	36	422.0	17.5	23	422.0	12.6	422.0
2	5	3	3	5	54	433.6	27.0	47	433.1	26.4	434.0
3	5	3	3	5	101	454.5	44.2	72	454.8	36.6	456.0
4	7	3	3	7	126	557.3	73.4	64	558.0	40.6	559.0
5	7	3	3	7	110	566.0	63.8	97	566.4	59.2	568.0
6	7	4	4	7	57	538.6	39.9	21	539.0	14.7	539.0
7	7	4	4	7	146	559.2	68.1	93	558.9	44.1	560.0
8	7	5	5	7	158	544.6	102.9	128	545.0	92.4	545.0
9	7	5	5	7	230	526.4	157.0	187	528.0	130.9	528.0
10	10	2	2	10	113	714.2	75.5	43	713.7	27.8	716.0

2.7 Reduction in Problem Size

Reduction procedures have been found to be extremely useful in some classes of combinatorial optimization problems (Crowder and Padberg [1980]; Crowder, Johnson and Padberg [1983]; Christofides and Beasley [1982] & [1983]; Gavish and Pirkal [1985]; Lucena-Filho [1986]). In this section, methods for performing sensitivity analysis on the solutions to problem (PR_λ) generated by the subgradient optimization procedure are developed and evaluated. Given an arbitrary λ vector, consider the effect on L_λ of setting to zero a variable that has a value of 1 in the optimal solution to problem (PR_λ). If the resulting change in L_λ is such that the new value of L_λ exceeds Z_{UB}, the best known upper bound Z^*, then it is evident that the optimal solution to PDP must have this variable set equal to one. Therefore, the PDP or any relaxation of it, can be solved, with variables of this type set equal to one.

At the start of the branch and bound procedure, sensitivity analysis of this type is performed on the relaxed problem (PR_λ), using λ^*. Any variables set to one as a result of this sensitivity analysis can be set to one throughout the branch and bound procedure. Sensitivity analysis is also useful at any level of the tree search, where additional sets of variables can be set to one in the appropriate sub-problems. Here, it is possible to identify variables that must be part of an optimal solution to the original PDP, with the additional branch and bound variables fixed to zero or one.

2.7.1 Penalty on Assigning an Origin City to a Collection Depot

The penalty to be paid for assigning an origin city i_0 to a collection depot j_0 in the dual solution to problem (PR_λ) is as follows;

Suppose that u_j^c are the set of upper bounds for the problem C‹. This is equivalent to problem C, having constraint (2.12) removed.

Define,

$$\bar{r}_{i_0 j} = \text{Min}\ \{u_j^c \mid y_{i_0 j} = 1\} \qquad j \in J$$

Thus, the penalty for having origin city i_0 served by collection depot j_0 (i.e. $y_{i_0 j_0} = 1$) is

$$\bar{P} = \min_{j \in J} \left\{ u_j^c - r_{i_0 j} \right\} + \theta_{i_0 j_0}$$

where $$\theta_{i_0 j_0} = \sum_{k=1}^{M} \sum_{l=1}^{N} c_{i_0 j_0 kl}\, x_{i_0 j_0 kl}$$

2.7.2 Penalty on Assigning a Destination City to a Delivery Depot

The penalty in serving destination city l_0 by a delivery depot k_0 can be computed similarly, by letting u_k^d be the upper bounds for the problem $\overline{\square}$, the equivalent GAP $\square$, where constraint (2.16) is removed.

Define,

$$\bar{r}_{kl_0} = \text{Min}\ \{u_k^d \mid z_{kl_0} = 1\} \qquad k \in K$$

Thus, the penalty to be paid is given by

$$\bar{P} = \text{Min}\ \left[u_k^d - \bar{r}_{kl_0}\right\} + \theta_{k_0 l_0}$$

where $$\theta_{k_0 l_0} = \sum_{i=1}^{N} \sum_{j=1}^{M} c_{ijk_0 l_0}\, x_{ijk_0 l_0}$$

2.7.3 Penalty on Re-Routing an Origin City to a Collection Depot

The penalty occurred in re-routing an origin city i_0 via a different collection depot

j_0 (i.e. $y_{i_0 j_0} = 0$) is as follows;

Define,

$$r_{i_0 j} = \text{Min} \left\{ u_j^c \mid y_{i_0 j} = 0 \right\} \qquad j \in J_i, \text{ where } J_i = \{ s \mid y_{is} = 1 \}$$

Thus, the penalty for closing route $y_{i_0 j_0}$ is

$$P = \sum_{j \in J_i} \left\{ u_j^c - r_{i_0 j} \right\} - \underset{j \in J_i}{\text{Max}} \left\{ u_j^c - r_{i_0 j} \right\} - \theta_{i_0 j_0}$$

where $\theta_{i_0 j_0} = \sum_{k=1}^{M} \sum_{l=1}^{N} c_{i_0 j_0 kl} \, x_{i_0 j_0 kl}$

2.7.4 Penalty on Re-Routing a Destination City to a Delivery Depot

Similarly, the penalty paid for serving a destination city l_0 via a different delivery depot k_0 is computed as follows;

Define,

$$r_{kl_0} = \text{Min} \left\{ u_k^d \mid z_{kl_0} = 0 \right\} \qquad k \in K_l, \text{ where } K_l = \{ s \mid z_{sl} = 1 \}$$

thus the penalty paid for setting $z_{k_0 l_0} = 0$ is

$$P = \sum_{k \in K_l} \left\{ u_k^d - r_{kl_0} \right\} - \underset{k \in K_l}{\text{Max}} \left\{ u_k^d - r_{kl_0} \right\} - \theta_{k_0 l_0}$$

where $\theta_{k_0 l_0} = \sum_{i=1}^{N} \sum_{j=1}^{M} c_{ijk_0 l_0} \, x_{ijk_0 l_0}$

2.8 Outline of the Lower Bounding Procedure

The lower bounding procedure incorporating the subgradient technique and the

penalty tests given in the previous section is described in detail as follows;

1. Using a heuristic procedure, generate a feasible solution to PDP. Use this value as the upper bound $\bar{Z}$ on the optimal solution value that is required for the subgradient procedure (The algorithm described in section 2.5 is used to obtain $\bar{Z}$). Initialize the lagrangean vector λ.

2. Compute the optimal solution to the lagrangean problem.

3. If in the optimal solution to the lagrangean problem, the correct relation between y's and z's is not obtained, use a heuristic procedure to generate a feasible solution to the PDP, starting from the solution to the lagrangean problem. If the associated cost of this procedure improves on the best - known feasible solution, update the value of $\bar{Z}$.

4. Terminate if $\bar{Z} - L_\lambda \leq \varepsilon$, with the increment accepted as the ε-optimal solution. Otherwise continue with step 5.

5. Update the vector λ, using the subgradient optimization procedure.

6. Repeat steps 2 through 5 until no further significant improvement in the lower bound value occurs.

7. Perform sensitivity analysis on the best lagrangean problem generated. Reduce the problem size by fixing the assignment variables to zero or one whenever possible, and similarly reduce the size of the problem by forcing the associated flow variables x_{ijkl} to zero or one.

8. If the lower bound value equals the best feasible solution value known, stop. Otherwise use a branch and bound procedure to generate and verify the optimal solution.

2.9 The Branch and Bound Procedure

The use of lagrangean relaxation in branch and bound has been described in Fisher [1981], and Geoffrion [1974]. The general flow of the procedure will now be

described in order to clarify how the different concepts, described fit together.

1. Initialize LB to be equal to any respectable lower bound on the original PDP, and $\bar{Z}$ to be equal to an artificial upper bound obtained from a heuristic procedure.

2. If the branch and bound search is completed then stop.

3. Relax problem Z by removing constraints (2.7). Choose a set of dual variables λ to represent the deleted constraints. Incorporate those into the objective function, which now becomes the lagrangean problem L_λ.

4. Solve the relaxed problem to minimize L_λ. This includes solving two GAP problems and a simple inspection problem, and a smoothing procedure to remove the infeasibility. Set LB = L_λ.

5. If LB > $\bar{Z}$, or problem Z has no feasible solution, problem Z is fathomed. Backtrack to step 2.

6. If the solution from step 4 is feasible, compute its real cost Z^*. If $Z^* < Z$ then set $Z = Z^*$ and record this solution as the incumbent. Fathom any sub-problems with LB>$\bar{Z}$.

7. Systematically update the multipliers λ and repeat steps 3 to 6 until an iteration limit is reached.

8. Perform sensitivity analysis on the best multipliers obtained.

9. If the LB value equals the best feasible solution known, stop.

10. Choose a branching variable that is not currently fixed in value. Create two new sub-problems by adding to problem Z the separation constraints.

11. Select the sub-problem with lowest lower-bound value between the two sub-problems. Go to step 2.

At the initial tree node sensitivity analysis was carried out in the attempt to reduce the problem size. The modified subgradient procedure was then carried out until termination rules were satisfied. If the subgradient failed to converge to an optimal

feasible solution then the set of the best multipliers corresponding to the highest lower bound was used to update the upper bound value of the PDP. A further twenty iterations of the subgradient procedure were then performed which takes advantage of the improved feasible solution obtained.

The subgradient iteration was also performed for twenty iterations at each node. The initial step size $\mu_h = 2$ was chosen and halved according to the rule described in section 2.6. The branching rule will be explained in the following section.

2.10 Branching Procedure

Three variable types are involved in the branch and bound procedure; y_{ij}, z_{kl}, and x_{ijkl}. These variables are nested, since $x_{ijkl} = 1$ implies that y_{ij} and z_{kl} are equal to one. This property implies that a possible branching might be to branch to M^2 new nodes. Obviously, this branching strategy will increase enormously the number of nodes in the branch and bound tree. By using another branching strategy, the number of nodes was significantly reduced. The branching variables can be selected by considering the following rules;

1. Select a variable x_{ijkl} satisfying $x_{ijkl} = 0$ and $y_{ij} = z_{kl} = 0$, and branch.
2. Alternatively, select a variable x_{ijkl} satisfying $x_{ijkl} = 1$ and $y_{ij} = z_{kl} = 1$ and branch.
3. If no such variables can be found, find a variable y_{ij} satisfying $y_{ij} = 0$ and $x_{ijkl} = 0$, for all k, l and branch.
4. Alternatively, find a variable y_{ij} satisfying $y_{ij} = 1$ and $\sum_{kl} x_{ijkl} = N$, and branch.
5. If the current solution is feasible and the above complementary conditions are not satisfied, select a variable y_{ij} / z_{kl} and branch.

The generated sub-problems with their associated data were stored in a tree. The next sub-problem for branching was selected as the one with the lowest lower bound

between the two sub-problems.

In order to improve the efficiency of the branch and bound algorithm, Rule 1 was used in the following manner: Let x_{ijkl} be selected as the branching variable. Whenever the constraint $x_{ijkl} = 0$ is added to some sub-problem, the constraints $y_{ij} = 0$ and $z_{kl} = 0$ are also added for every successor node on the tree.

Similarly, Rule 2 implies that if the constraint $x_{ijkl} = 1$ is added to some sub-problem, the successive nodes emerging from that sub-problem have the following constraints $y_{ij} = 1$ and $z_{kl} = 1$ augmented on them.

In addition, Rule 3 is used to introduce the constraint $\sum x_{ijkl} = 0$ whenever $y_{ij} = 0$ is selected. If a variable y_{ij} is selected for branching and the constraint $y_{ij} = 1$ is added to the sub-problem, Rule 4 states that since there are N destination cities with demands for shipments from origin city i, then the constraint $\sum x_{ijkl} = N$ is added to successor nodes.

This concludes the description of the algorithm. The next section reports on tests of the performance of the algorithm.

2.11 Computational results

In this section computational results using the lagrangean relaxation approach are shown in order to demonstrate the power of this method when applied to PDP. All the problems attempted had fixed operating cost coefficients generated randomly via a uniform distribution on the interval [1000,9000]. The variable operating costs were represented as a function of the distances between cities and depots and the unit costs of local and trunk transport.

The unit cost of local collection and distribution was chosen to be 1 and the unit cost of mass transportation is set at 1/4. The values of local distances were drawn from a uniform distribution between 0 and 15. Similarly, the distances between depots are drawn from a uniform distribution. Finally a uniform distribution of integers between

Table 2.2

Representative runs for 9 problem sizes

	Problem size§ OC x CD DD x DC	No. of problems			Percent Duality Gap		No. of lagrangean iterations						No. of tree nodes			CPU Time (sec.)					
							At root node			Total						At root node			Total		
		a	*s*	*b*	Ave.	Max.	Ave.	Min.	Max.	Ave.	Min.	Max.	Ave.	Min.	Max.	Ave.	Min.	Max.	Ave.	Min.	Max
1	5x3x3x5	5	5	4	0.25	0.36	86	23	168	98	23	229	1	0	6	46.4	12.6	87.4	57.9	12.6	123.6
2	7x3x3x7	5	5	3	0.16	0.31	76	29	157	87	41	278	5	1	17	49.3	18.9	97.7	61.5	26.9	175.1
3	7x4x4x7	5	5	4	0.10	0.23	83	21	172	96	21	242	3	0	12	49.1	14.7	115.2	72.2	14.7	164.5
4	7x5x5x7	5	5	3	0.05	0.33	120	39	194	225	92	397	23	5	50	90.5	28.2	138.2	163.0	66.2	297.7
5	10x2x2x10	5	5	3	0.23	0.38	65	21	169	81	21	263	10	0	32	52.2	13.6	106.4	52.6	13.6	168.3
6	10x3x3x10	5	5	3	0.17	0.24	96	18	177	136	18	302	14	0	28	73.0	12.6	131.5	97.2	12.6	211.4
7	10x4x4x10	5	4	2	0.11	0.45	134	88	208	373	88	704*	34	0	72	116.2	75.6	196.6	330.4	75.6	600.0*
8	15x2x2x15	5	5	3	0.13	0.21	110	63	186	389	63	513	8	0	24	87.0	50.4	154.2	301.2	50.4	439.2
9	15x3x3x15	5	3	2	0.07	0.56	152	97	192	429	97	692*	14	1	42	181.6	122.3	261.6	528.1	122.3	600.0*

§ OC Origin City CD Collection Depot DD Delivery Depot DC Destination City

a Attempted

s Solved to Optimality

b No. of problems requiring Branch and Bound

* Program Terminated

100 and 500 was used to create quantities of shipment from cities i to cities l.

The generation of the problems is discussed in greater detail in Section 3.6. Table 2.2 summarises the computational results using the method described. Nine different problem sizes were used, with five examples of each. Of these 45 problems, 42 were solved optimally, and 40% did not require Branch and Bound. At least one of each problem size did not require Branch and Bound.

The average duality gap at optimality was only 0.14% of the overall results, and the average for each class of problem did not appear to depend on the problem size.

The worst case for any problem size was only 0.56%, and similarly there is no apparent pattern in the worst case duality gaps of the different problem sizes.

The number of Lagrangean iterations required at the root node increased with the size of the problem; the average, minimum, and maximum number of iterations for each size being (in most cases) substantially larger for the larger problems.

2.12 Conclusion

The reduction methods and penalty routines used in connection with the PDP are the most important feature discussed in this Chapter. It is important to appreciate the strong influence that the chosen reduction method has on the solution procedure. The branch and bound algorithm, as described in this Chapter, depends on an optimal solution to the lagrangean relaxed problem in order to supply lower bounds. By introducing sensitivity analysis the algorithm may require extra computational effort. The benefits achieved by this sacrifice are good quality lower bounds which limit the tree search, and hence faster solution times for the PDP.

CHAPTER 3

A Formulation of the Parcel Distribution Problem Based on Benders Decomposition

This Chapter introduces a formulation of the Parcel Distribution Problem (PDP). The Benders decomposition method and its application to the model will be described. The behaviour of the model will be illustrated on small size problems, and further refinements will be dealt with in subsequent Chapters.

3.1 Problem formulation

The Parcel Distribution Problem can be viewed as having two sub-problems, a city-to-depot assignment sub-problem, and a transportation flow sub-problem. The

city-to-depot assignment problem can be considered as a sub-problem which assigns each origin city, with a known demand for goods to be collected, to a collection depot. Similarly, each destination city is assigned to a delivery depot. The transportation flow sub-problem ensures the integrity of the goods at intermediate stages of shipment and ensures that the correct shipment is made between corresponding origin-destination pairs. The interlinking between the assignment sub-problem and the transportation sub-problem is considered in the transportation sub-problem itself. If an origin city is assigned to a collection depot then all goods emerging from that city must be routed via that collection depot. Similarly, goods received at a final city must be delivered via a delivery depot allocated by the assignment sub-problem. Figure 3.1 shows a typical PDP in a schematic form.

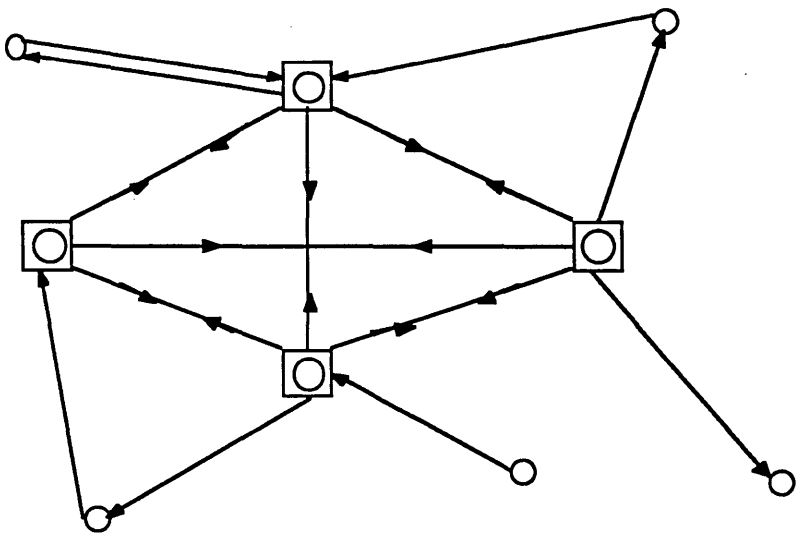


Fig. 3.1 PDP network

The PDP can be formalized as the following mixed integer program:

Define,

y_{ij} = 1 if city i is assigned to the collection depot j

0 otherwise

z_{kl} = 1 if city l is assigned to the delivery depot k

0 otherwise

x_{ijkl} is a variable denoting the amount of goods dispatched from city i to city l, via two intermediate depots j and k.

Thus, with N cities providing the supply, and N cities receiving the deliveries via M collection depots and M delivery depots, the model is:

Problem W

$$\text{Minimize} \quad Z = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{M}\sum_{l=1}^{N} c_{ijkl}\, x_{ijkl} + \sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij}\, y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N} f_{kl}\, z_{kl} \tag{3.1}$$

subject to

$$\sum_{j=1}^{M} y_{ij} = 1 \qquad i = 1, \ldots, N \tag{3.2}$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il} y_{ij} \le Q_j^c \qquad j = 1, \ldots, M \tag{3.3}$$

$$\sum_{k=1}^{M} z_{kl} = 1 \qquad l = 1, \ldots, N \tag{3.4}$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il}\, z_{kl} \le Q_k^d \qquad k = 1, \ldots, M \tag{3.5}$$

$$\sum_{k=1}^{M}\sum_{l=1}^{N} x_{ijkl} = \sum_{l=1}^{N} q_{il}\, y_{ij} \qquad \begin{matrix} i = 1, \ldots, N \\ j = 1, \ldots, M \end{matrix} \tag{3.6}$$

$$\sum_{i=1}^{N}\sum_{j=1}^{M} x_{ijkl} = \sum_{i=1}^{N} q_{il}\, z_{kl} \qquad \begin{matrix} k = 1, \ldots, N \\ l = 1, \ldots, M \end{matrix} \tag{3.7}$$

$$\sum_{j=1}^{M}\sum_{k=1}^{M} x_{ijkl} = q_{il} \qquad \begin{matrix} i = 1, \ldots, N \\ l = 1, \ldots, N \end{matrix} \tag{3.8}$$

$$y_{ij},\ z_{kl} \in \{ 0, 1 \} \qquad \text{for all } i, j, k, l \tag{3.9}$$

$$x_{ijkl} \ge 0 \qquad \text{for all } i, j, k, l \tag{3.10}$$

It is assumed that all the summations and constraints run only over the possible combinations of subscripts. All demands must be met exactly. The objective function (3.1) is made up of two parts: a transportation cost part, and the fixed cost of operating a possible route between a city and a depot.

Constraints (3.2) and (3.4) specify that each city must be assigned to only one collection/delivery depot.

Constraints (3.3) and (3.5) enforce the limits on total throughput through depots j and k, namely Q_j^c and Q_k^d.

Constraint (3.6) enforces the correct relationship between the x's and y's. When $y_{ij} = 1$ then the total flows from city i must be through collection depot j. When $y_{ij} = 0$ then all the corresponding x_{ijkl} flows must be equal to zero.

Constraint (3.7) similarly enforces the same interlinking condition for x's and z's.

Constraint (3.8) ensures that the demand for transportation between two cities must be met via two intermediate depots.

3.2 Application of the Benders decomposition method

Consider the decomposing of the variables for the PDP in two sets X and W given by

$$X = \{ x_{ijkl} \} \qquad \text{(transportation variables)}$$

$$W = \{ y_{ij} , z_{kl} \} \qquad \text{(assignment variables)}$$

The PDP can then be re-written (for the sake of convenience) in matrix form as

$$
\begin{aligned}
&\text{Minimize} && C_1^T X + C_2^T W \\
&\text{subject to} && DW \geq B_1 \\
& && A_1 X + A_2 W \geq B_2 \\
& && X \geq 0 \\
& && W \in \{0, 1\}
\end{aligned}
\tag{3.11}
$$

where D is the matrix of coefficients for equations (3.2) - (3.5), and A_1 and A_2 are the coefficient matrices of equations (3.6) - (3.8).

An observation regarding the structure of the matrix D reveals that it represents two separate and independent generalized assignment problems (GAP), and these can be solved "effectively" using existing algorithms, (Martello and Toth [1981]; Fisher, Jaikumar and Van Wassenhove [1986]).

For any given values of variables W which satisfy the constraints of the GAP, the remaining problem in X is:

$$
\begin{aligned}
&\text{Minimize} && C_1^T X \\
&\text{subject to} && A_1 X \geq B_2 - A_2 W \\
& && X \geq 0
\end{aligned}
\tag{3.12}
$$

which is a linear programming problem. Thus, this fact suggests the applicability of the Benders decomposition method which was developed for such types of mixed integer linear programs (Benders [1962]; Hu [1969]; Lasdon [1970] & [1972]; Zionts

[1974]; Balas and Bergthaller [1977]; Mc Daniel and Devine [1977]; Magnanti and Wong [1981]).

3.3 The Benders decomposition method

The Benders decomposition method can provide an efficient primal approach for solving problems with structures similar to (3.11). When applying BDM the focus is on a problem which is "equivalent" to the one stated in (3.11). In this equivalent problem, the assignment variables are suppressed and dual variables of the transportation sub-problem are brought explicitly into play to "represent" the assignment sub-problem. The discussion that follows is based on Geoffrion [1972]. For further details, the reader is referred to the original paper by Benders [1962] and Lasdon [1970].

3.3.1 Theory

To emphasize its special structure the PDP (3.11) can be re-written as:

$$\underset{W \in S(W)}{\text{Minimize}} \; C_2^T W + [\underset{x \geq 0}{\text{Minimize}} \; C_1^T X \mid A_1 X \geq B_2 - A_2 W \;] \tag{3.13}$$

where $S(W) = \{W \mid DW \geq B_1,\ W \in \{0, 1\}\}$

If the assignment variable W is fixed in (3.13) the problem reduces to:

$$\begin{aligned} &\text{Minimize} && C_1^T X \\ &\text{subject to} && A_1 X \geq B_2 - A_2 W \\ & && X \geq 0 \end{aligned} \tag{3.14}$$

Introducing the non-negative vector U, the dual of (3.14) can be written as:

$$\begin{aligned} &\text{Maximize} && (B_2 - A_2W)^{\mathsf{T}}U \\ &\text{subject to} && A_1^{\mathsf{T}}U \leq C_1 \\ & && U \geq 0 \end{aligned} \tag{3.15}$$

A superficial study of problem (3.15) indicates that the set of feasible solutions to the dual does not depend on the value of W whereas the right-hand-side of the primal is parametrized in W. It will be assumed for simplicity that problem (3.14) is both feasible and bounded for any value of $W \in S(W)$. In order to ensure feasibility it is sufficient to introduce a dummy collection and delivery depot with an unlimited handling capacity and an artificially very high fixed cost. Although these depots will not be utilized at optimality their presence guarantees the feasibility of the problem. Under these conditions the dual (3.15) is also feasible and bounded. Denoting an optimal solution to (3.14) and (3.15) as X^* and U^*, respectively, we must have, by the duality theorem (Hu [1969]) that

$$C_1^T X^* = (B_2 - A_2W)\, U^* \tag{3.16}$$

Further, from (3.13) and the equivalence of (3.14) and (3.15), the following must hold

$$\text{Min. } C_2^T W + C_1^T X = \underset{W \in S(W)}{\text{Minimize}}\ C_2^T W + [\underset{U \geq 0}{\text{Maximize}}\ (B_2 - A_2W)^{\mathsf{T}}U \mid A_1^{\mathsf{T}}U \leq C_1] \tag{3.17}$$

It is well known from the theory of linear programming that under these conditions the set of feasible solutions to any linear program forms a convex polytope which can be described by the finite set of its extreme points. It has been established as well that for such a linear program, the objective function will achieve its optimal value at one or more of these extreme points. Hence, let $L = \{ U^h \}$, for $h = 1, \ldots, H$ to represent the set of all extreme points of the solution space of (3.15). Then problem (3.15) can be written as:

$$\underset{\forall U^h \in L}{\text{Minimize}} \ (B_2 - A_2 W)^{\mathsf{T}} (U^h) \tag{3.18}$$

Then, the sub-problem in (3.17) can be replaced by (3.18) to become:

$$\underset{W \in S(W)}{\text{Minimize}} \ C_2^T W + [\underset{U^h \in L}{\text{Maximize}} \ (B_2 - A_2 W)^{\mathsf{T}} (U^h) \ ] \tag{3.19}$$

or, after introduction of a scalar unbounded variable M_0 and a few manipulations:

$$\begin{aligned} &\text{Minimize} && C_2^T W + M_0 \\ &\text{subject to} && DW \geq B_1 \\ & && (B_2 - A_2 W)^{\mathsf{T}} (U^h) \leq M_0 && \forall U^h \in L \\ & && W \in \{0, 1\} \\ & && M_0 \quad \text{un-bounded} \end{aligned} \tag{3.20}$$

Each extreme point U^h gives one constraint in (3.20) which restricts the solution space of the problem. These constraints are often referred to as Benders cuts in the literature (Lasdon [1970]).

3.3.2 A solution procedure

The solutions of problems (3.11) and (3.20) are equivalent. However, because the number of dual solutions in set L is very large, and because there exists a Benders cut associated with each solution, problem (3.20) is not tackled directly. Instead a significant subset $\overline{L}$ of L is constructed in a two stage iterative procedure. In the first stage of each iteration, a relaxed version of the equivalent problem (3.20) is solved. Each time the transportation sub-problems are solved under some assignment of cities to depots, it is possible to generate one of the basic dual variable solutions belonging to L. Hence, it is possible to construct a subset $\overline{L}$ of L and to solve a relaxed equivalent problem (3.20) over this subset.

The relaxed problem is normally referred to as the master problem, and solving it delivers both a feasible assignment of cities to depots and a lower bound on the minimum cost of the master problem.

In the second stage, the transportation sub-problem (3.14) is solved under the assignment decision just obtained in stage one. Stage two yields not only the total cost for this parcel distribution problem but also another basic dual feasible solution to further add to the set $\overline{L}$.

As the two-stage process is repeated, the cost of the lower bound converges to the minimum cost of the original problem and the optimal parcel distribution plan is eventually obtained. In normal practice with BDM, the computations are stopped when the lower bound on the minimum cost is within a pre-specified error tolerance of the cost of the best distribution plan obtained up to that point.

3.3.3 A specialized procedure for the PDP

The Benders decomposition method, formally described in the previous section, applied to the parcel distribution problem (3.1) - (3.10) leads to the following algorithm:

step 0 Initialize $h = 0$, UB = ∞, LB = $-\infty$.

Let $\varepsilon \geq 0$ to be a convergence tolerance parameter.

step 1 Increment h by 1.

Solve the current relaxed master problem.

$$\text{Minimize } Z_M = \sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij}\, y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N} f_{kl}\, z_{kl} + M_0$$

$$\text{subject to} \quad \sum_{j=1}^{M} y_{ij} = 1 \qquad i = 1, \ldots, N$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il} y_{ij} \leq Q_j^c \qquad j = 1, \ldots, M$$

$$\sum_{k=1}^{M} z_{kl} = 1 \qquad l = 1, \ldots, N \qquad (3.21)$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il}\, z_{kl} \leq Q_k^d \qquad k = 1, \ldots, M$$

$$\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{l=1}^{N} \alpha_{ij}^h q_{il} y_{ij} + \sum_{i=1}^{N}\sum_{k=1}^{M}\sum_{l=1}^{N} \beta_{kl}^h q_{il} z_{kl} + \sum_{i=1}^{N}\sum_{l=1}^{N} \gamma_{il}^h q_{il} \leq M_0$$

$$h = 1, \ldots, H$$

y_{ij}, $z_{kl} \in \{0, 1\}$ for all i, j, k, l

M_0 un-bounded

Let $(y_{ij}^{h+1}, z_{kl}^{h+1}, M_0^{h+1})$ be the optimal solution and Z_M^* be the optimal value. Z_M^* is the new lower bound, i.e. put LB = Z_M^*. If UB - LB ≤ ε, stop with the increment accepted as the ε-optimal solution.

Step 2 Using $(y_{ij} = y_{ij}^{h+1}, z_{kl} = z_{kl}^{h+1})$, solve the transportation sub-problem.

$$\text{Minimize} \quad Z_S = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{M}\sum_{l=1}^{N} c_{ijkl}\, x_{ijkl}$$

$$\text{subject to} \quad \sum_{k=1}^{M}\sum_{l=1}^{N} x_{ijkl} = \sum_{l=1}^{N} q_{il}\, y_{ij} \qquad \begin{matrix} i = 1, \ldots, N \\ j = 1, \ldots, M \end{matrix}$$

$$\sum_{i=1}^{N}\sum_{j=1}^{M} x_{ijkl} = \sum_{i=1}^{N} q_{il}\, z_{kl} \qquad \begin{matrix} k = 1, \ldots, N \\ l = 1, \ldots, M \end{matrix} \qquad (3.22)$$

$$\sum_{j=1}^{M}\sum_{k=1}^{M} x_{ijkl} = q_{il} \qquad \begin{matrix} i = 1, \ldots, N \\ l = 1, \ldots, N \end{matrix}$$

$$x_{ijkl} \geq 0 \qquad \text{for all } i, j, k, l$$

Denote the optimal solution by x_{ijkl}^{h+1} and its objective value by Z_S^{h+1}. If,

$$Z_T = Z_S^{h+1} + \left[\sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij}\, y_{ij}^{h+1} + \sum_{k=1}^{M}\sum_{l=1}^{N} f_{kl}\, z_{kl}^{h+1}\right] \leq \text{UB}, \text{ put UB} = Z_T,$$

and update the incumbent solution, i.e. put ,

$$\{y_{ij}^*, z_{kl}^*, x_{ijkl}^*\} = \{y_{ij}^{h+1}, z_{kl}^{h+1}, x_{ijkl}^{h+1}\}.$$

If UB - lB $\leq \varepsilon$ stop; $\{y_{ij}^{*}, z_{kl}^{*}, x_{ijkl}^{*}\}$ is a ε-optimal solution.

If not, determine an optimal dual solution for the transportation sub-problem, denote it by

$$\alpha_{ij}^{h+1},\ \beta_{kl}^{h+1},\ \gamma_{il}^{h+1}$$

corresponding respectively to constraints (3.6), (3.7), and (3.8), and go to step 1.

Note that if an initial value of $\{y_{ij}, z_{kl}\}$ is available, then the procedure can be started in step 2. Practical experience may guide the choice of a good initial solution. Finite convergence of the procedure is a direct consequence of the finiteness of the extreme points.

3.4 Application of the procedure to the PDP

A successful implementation of BDM depends on the resolution of several key factors. Firstly, Benders established that the convergence of the method towards the optimal solution is theoretically guaranteed, nonetheless this does not give indication as to the rate of convergence i.e. how many times step 1 and 2 must be solved before the optimal solution is reached. Secondly, the transportation sub-problem can be relatively easy to solve, the procedure for it must produce optimal multipliers. Thirdly, the master problem can be solved with a high level of efficiency. It will be shown in the following that these factors account for the great flexibility of the method, which allows the application of a specifically tailored procedure to different problems.

3.4.1 Solution of the master Problem

An advantage of BDM is the fact that no restriction is imposed on the type of algorithm used for solving the master problem. In this way any appropriate algorithm which exploits any underlying special structure to the full, may be used. The master

problem (3.21) at first iteration ($h = 0$) has a special structure and can be solved as two separate generalized assignment problems. However, for $h > 0$, the master problem has augmented Benders cuts. This destroys its special structure. Thus, the greatest limitation of the approach lies in the necessity to solve a sequence of difficult mixed integer linear programs. This is the main obstacle that arises in most applications of BDM. The PDP is no exception to the rule, and the procedure presented is designed to alleviate this problem.

The solution method is a depth-first, tree-search procedure where the two branches from any node of the tree represent assignment and non-assignment of a city to a particular depot. Lower bounds are calculated by relaxing the integrality condition in the master problem. The relaxation exploits the efficiency of modern LP codes. It also provides a good lower bound on the optimal value of the PDP. The conventional branch and bound algorithm is outlined in detail below.

The following notation is adopted for the purpose of describing the procedure.

I_n The set of assignments of cities to depots which are explicitly included in the solution at node n to effect branching (i.e. variables, y_{ij} & z_{kl} whose values are fixed at 1 at node n).

E_n The set of assignments of cities to depots which are explicitly excluded in the solution at node n to effect branching (i.e. variables, y_{ij} & z_{kl} whose values are fixed at 0 at node n).

$I_n \cup E_n$ The union I_n and E_n , the set of assignments which form the partial solution at node n, (i.e. variables whose values are fixed at node n).

$(I_n \cup E_n)^c$ The complement of (I_nU E_n), i.e. the set of free items at n, (i.e. variables that are free to take on either 1 or 0 at node n).

F_n The set of fractional assignments of each city to a particular depot at node n, (i.e. variables whose values are fractional at node n).

Step 1 Initializing and establishing the starting node (root node).

Set $n = 0$.

Set the lower bound LB to be equal to the current best lower bound computed from the previous Benders cut.

Set the upper bound UB to be equal to the current best upper bound found for the PDP.

Solve the LP relaxation of the master problem (3.21). Let F_{n+1} be equal to the set of fracti nal variables (y_{ij} and z_{kl}), and go to step 2.

Step 2 Branching and fea ibility testing

Set $n = n + 1$, if F_n empty; set $n = n - 1$, if $n = 0$, stop; else repeat step 2.

Select an item from the set F_n in numerical order, (say $y_{i_1 j'}$). This is a simple and successful selection strategy. A more complex procedure will be given in the following Chapter. Execute branching on a city i_1, for which the integrality conditions are violated and generate following branchings:

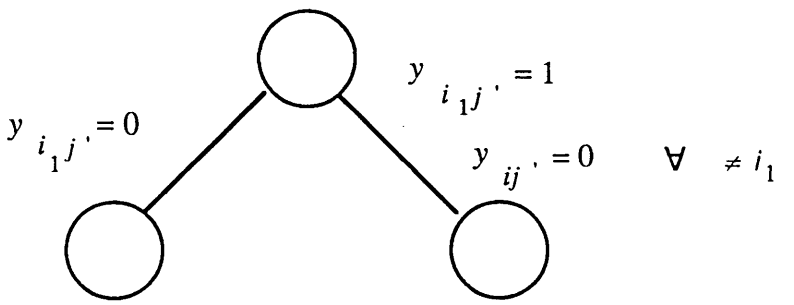


Fig. 3.2 branching strategy

Update I_n, E_n. Test feasibility of the solution corresponding to $I_n \cup E_n$. If infeasible go to step 4; else go to step 3.

Step 3 Calculation of lower bound

Compute a lower bound LB with respect to current state.

If LB > z^* go to step 5; else go to step 4.

Step 4 If F_n empty, then a better solution has been found. Update z^*, and go to step 5; else go to step 2.

Step 5 Backtrack

Remove y_{ij}, by updating I_n, E_n. Go to step 2.

3.4.2 Second solution approach to the master problem (Alternative Strategy)

It has been realized that it is not critical to the application of BDM that the optimal solution must be obtained for the current master problem at each iteration, (Geoffrion [1974]; Mc Daniel and Devine [1977]; Magnanti and Wong [1981]). This is due to the fact that the initial master problems contain too little information about the transportation flow costs to be worth optimizing very strictly. Hence, it might be better to search for a feasible solution with cost ≤ UB - ε, where UB represents the best upper bound obtained so far, and to generate a cut at this point by solving the transportation sub-problem. The algorithm terminates due to the fact that the number of dual solutions to the transportation sub-problem is finite, and by the realization that if a dual solution is generated twice, then the master solution must be improved by at least ε.

The new strategy can be written more formally once M_0 is eliminated from the master problem (3.20) and the upper bound UB is introduced:

$$\begin{aligned} &\text{Minimize} && C_2^T W \\ &\text{subject to} && DW \geq B_1 \\ & && (B_2 - A_2 W)^{\mathsf{T}} (U^h) \leq \text{UB} - \varepsilon \qquad \forall U^h \in L \\ & && W \in \{0, 1\} \end{aligned} \tag{3.23}$$

The master problem (3.23) no longer produces a lower bound on the optimal value of PDP. Thus the algorithm introduced at section (3.3.3) should be modified accordingly:

Step 0 Initialize $h = 0$, UB $= \infty$.

Step 1 Increment h by 1.

Solve the current relaxed master problem.

$$\text{Minimize } Z_M = \sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij}\, y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N} f_{kl}\, z_{kl}$$

$$\text{subject to} \quad \sum_{j=1}^{M} y_{ij} = 1 \qquad i = 1, \ldots, N$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il} y_{ij} \le Q_j^c \qquad j = 1, \ldots, M$$

$$\sum_{k=1}^{M} z_{kl} = 1 \qquad l = 1, \ldots, N \qquad (3.24)$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il}\, z_{kl} \le Q_k^d \qquad k = 1, \ldots, M$$

$$\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{l=1}^{N} (f_{ij} + \alpha_{ij}^h\, q_{il})\, y_{ij} + \sum_{i=1}^{N}\sum_{k=1}^{M}\sum_{l=1}^{N} (f_{kl} + \beta_{kl}^h\, q_{il})\, z_k + \sum_{i=1}^{N}\sum_{l=1}^{N} \gamma_{il}^h\, q_{il} \le \text{UB-}\varepsilon$$

$$h = 1, \ldots, H$$

$$y_{ij},\ z_{kl} \in \{0, 1\} \qquad \text{for all } i, j, k, l$$

Let $(y_{ij}^{h+1}, z_{kl}^{h+1})$ be a feasible solution. If no feasible solution exists, then terminate. The current UB is ε-optimal.

Step 2 Using $(y_{ij}^{h+1}, z_{kl}^{h+1})$ solve the transportation sub-problem (3.22), determining an optimal solution x_{ijkl}^{h+1} and its objective value Z_S^{h+1}. If,

$$Z_T = Z_S^{h+1} + \left[\sum_{i=1}^{N} \sum_{j=1}^{M} f_{ij}\, y_{ij} + \sum_{k=1}^{M} \sum_{l=1}^{N} f_{kl}\, z_{kl} \right] \leq \text{UB}, \quad \text{put UB} = Z_T,$$

and update the incumbent solution. If not, determine the corresponding optimal dual solution $(\alpha_{ij}^{h+1}, \beta_{kl}^{h+1}, \gamma_{il}^{h+1})$; and go to step 1.

Notice that the master problem has become a pure 0-1 integer problem. The effort needed to find a feasible solution to this problem increases as new cuts are added. Geoffrion and Graves [1974] who introduced this approach point out that the master problem (3.24) is needed only in order to produce a feasible solution, so that any objective function could be used instead of the one described above. Hence, a new objective was developed to encourage the production of useful feasible solutions. This involves constructing a surrogate constraint (Glover [1968]) from the Benders cuts.

Suppose $\lambda_h \geq 0$ is determined for $h = 1, \ldots, H$ such that :

$$\sum_{h=1}^{H} \lambda_h = 1$$

Multiply the h^{th} Benders cut by λ_h for $h = 1, \ldots, H$ and sum the results to obtain the surrogate constraint

$$\phi(\lambda)^h = \sum_{h=1}^{H} \lambda_h (B_2 - A_2 W)^T (U^h) \quad < \text{UB} - \varepsilon \tag{3.25}$$

The left hand side of function ϕ is then used to build the objective function.

3.4.3 Solution of the transportation sub-problem

At step 2 of the Benders decomposition problem, it is required to solve the following transportation problem given that at iteration h, $y_{ij} = y_{ij}^{h+1}$ and $z_{kl} = z_{kl}^{h+1}$.

$$\text{Minimize} \quad Z_S = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{M}\sum_{l=1}^{N} c_{ijkl}\, x_{ijkl}$$

$$\text{subject to} \quad \sum_{k=1}^{M}\sum_{l=1}^{N} x_{ijkl} = \sum_{l=1}^{N} q_{il}\, y_{ij} \qquad \begin{matrix} i = 1, \ldots, N \\ j = 1, \ldots, M \end{matrix}$$

$$\sum_{i=1}^{N}\sum_{j=1}^{M} x_{ijkl} = \sum_{i=1}^{N} q_{il}\, z_{kl} \qquad \begin{matrix} k = 1, \ldots, N \\ l = 1, \ldots, M \end{matrix}$$

$$\sum_{j=1}^{M}\sum_{k=1}^{M} x_{ijkl} = q_{il} \qquad \begin{matrix} i = 1, \ldots, N \\ l = 1, \ldots, N \end{matrix}$$

$$x_{ijkl} \geq 0 \qquad \text{for all } i, j, k, l$$

The above model represents a linear programming problem which can be solved by any LP code. It is evident that even a commercial LP code (for large a scale problem) requires much computer time and considerably more space for data storage than the special algorithms. However, the primary interest was to develop the

framework of a general problem in this Chapter. The more specialized algorithm will be developed in the following Chapter to solve the transportation problem.

Marsten's [1981] XMP code is used to solve the above problem, and the dual obtained at each iteration was used to derive the cuts of the master problem at step 1 of the procedure.

3.5 Numerical example

To illustrate the solution method in the proceeding sections the PDP has been solved for a small model (figure 3.3). The model has 2 origin cities with demand for deliveries of goods, 2 receiving (destination) cities, and 2 collection and delivery depots. In order to present the model in a simple form, the number of origin, destination cities and collection, delivery depots is chosen to be equal, without loss of generality.

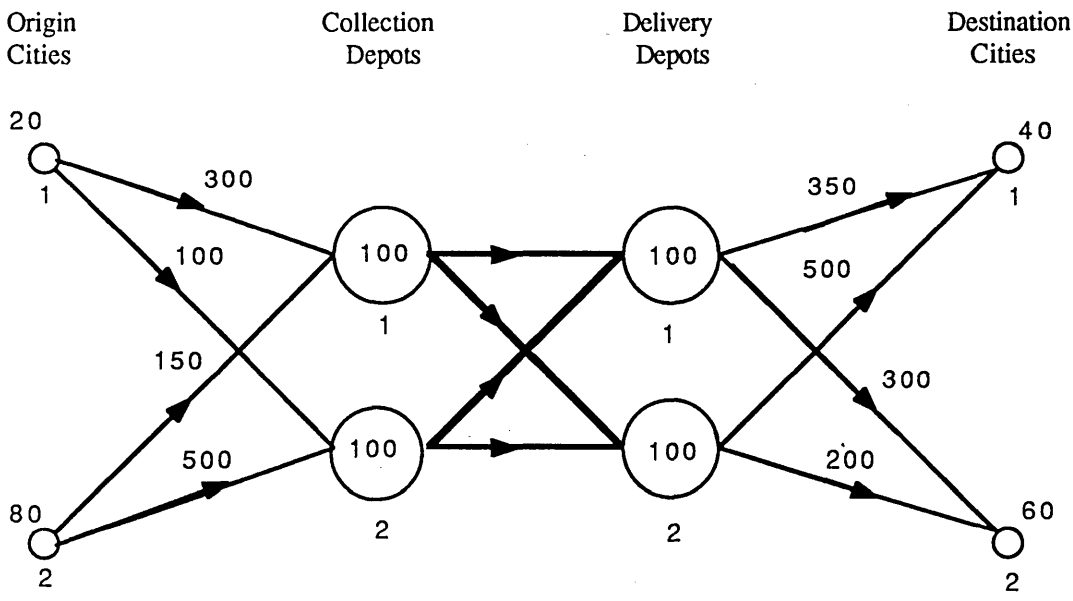


Fig. 3.3 A PDP network example

The numbers above the arcs give the fixed cost of establishing that link. The

numbers inside the (collection/delivery) nodes give the maximum handling capacities of those nodes. The numbers beside the (origin/destination) nodes give the total quantity of shipment from/to those nodes. The unit variable costs of shipment from origin to destination cities via two intermediate depots are given in Table 3.1. The desired quantities of shipment from origin to destination cities are given in Table 3.2.

Table 3.1
Origin-destination unit cost matrix

	l	1		2	
i	k / j	1	2	1	2
1	1	3	1	1	5
	2	9	4	8	2
2	1	8	2	3	8
	2	2	3	5	4

Table 3.2
Quantity of shipment (q_{il}) matrix

i / l	1	2
1	15	5
2	25	55

The BDM applied (step by step) to the above example is as follows:

Step 0 Initialization.

The iteration step h set to 0.

The upper bound (UB) set to ∞. The lower bound (LB) set to $-\infty$.

The convergence tolerance parameter (ε) set to 0.

Step 1 $h = h + 1 = 1$

Initially the master problem is decomposed into two generalized assignment problems. The first one assigns origin cities to collection depots. The second one assigns destination cities to distribution depots.Two GAPs are solved to obtain,

$$y_{12}^1 = y_{21}^1 = z_{11}^1 = z_{22}^1 = 1 \quad \text{and all other} \quad y_{ij}^1 = z_{kl}^1 = 0.$$

The lower bound on the optimal solution to PDP is provided by the summation of optimal solutions to two GAPs; that is LB = 800.

Step 2 Using the fixed cost solutions for y_{ij} and z_{kl} from step 1, the transportation sub-problem was solved as a linear programming problem. The results obtained are $x_{1211} = 15$, (i.e. the desired quantity of 15 units from origin city 1 is shipped to destination city 1 via collection depot 2 and delivery depot 1). Similarly, $x_{1222} = 5$, $x_{2111} = 25$, and $x_{2122} = 55$.

The total variable cost (TVC) is 785 and UB to the PDP is equal to total fixed cost (TFC) plus (TVC); i.e. UB = 800 + 785 = 1585.

The dual variables obtained are as follows:

$$\alpha_{11}^1 = -6\ ,\ \alpha_{22}^1 = -6\ ,\ \beta_{21}^1 = -6\ ,\ \beta_{12}^1 = -5,\ \gamma_{11}^1 = 9\ ,\ \gamma_{12}^1 = 2\ ,\ \gamma_{21}^1 = 8\ ,\ \text{and}\ \gamma_{22}^1 = 8.$$

Thus the first Benders cut derived is as follows:

$$-120\, y_{11} - 480\, y_{22} - 240\, z_{21} - 300\, z_{22} + 785 \le M_0$$

Step 1 $h = h + 1 = 2$

The integrality constraint in the master problem is relaxed and the problem

solved as an LP problem. The solutions obtained are all integer,

$y_{12}^2 = y_{22}^2 = z_{21}^2 = z_{12}^2 = 1$ and all other $y_{ij}^2 = z_{kl}^2 = 0$.

Thus there is no requirement for a branch and bound procedure at this iteration. The optimal solution to the master problem gives the new lower bound to the PDP, which is LB = 1165.

Step 2 Using the recent fixed cost values, the following solution is obtained to x's:

$x_{1221} = 15$, $x_{1212} = 5$, $x_{2221} = 25$, and $x_{2212} = 55$.

The TVC = 450. The updated upper bound remains the same as at the previous iteration. The new dual variables are as follows:

$\alpha_{11}^2 = -7$, $\alpha_{21}^2 = -2$, $\beta_{11}^2 = -1$, $\beta_{22}^2 = -6$, $\gamma_{11}^2 = 4$, $\gamma_{12}^2 = 8$, $\gamma_{21}^2 = 3$, and $\gamma_{22}^2 = 5$.

Thus, the second Benders cut derived is as follows:

$-140\, y_{11} - 160\, y_{21} - 40\, z_{11} - 360\, z_{22} + 450 \leq M_0$

Step 1 $h = h + 1 = 3$

The second cut is also augmented to the master problem. The integrality condition is relaxed and the master problem solved as an LP problem. The following result is obtained:

$y_{12}^3 = y_{21}^3 = z_{21}^3 = 1$, $z_{12}^3 = 0.93$, $z_{22}^3 = 0.07$, and all other $y_{ij}^3 = z_{kl}^3 = 0$.

The above solution to the master problem is not integer and requires a branch and bound procedure to obtain a feasible integer solution. The complete tree search for this procedure is illustrated in Figure 3.4. The top numbers inside each node indicate the order in which the nodes were examined, and the other value designates the lower bound (LB) on the node.

The summary of the solutions at each node are as follows:

node 1 $y_{12}^3 = z_{21}^3 = z_{22}^3 = 1$, $y_{21}^3 = 0.04$, $y_{22}^3 = 0.96$, and $M_0 = 837.50$

node 2 $y_{12}^3 = y_{21}^3 = z_{12}^3 = 1$, $z_{11}^3 = 0.16$, $z_{21}^3 = 0.83$, and $M_0 = 283.57$

node 3 $y_{12}^3 = y_{21}^3 = z_{21}^3 = z_{12}^3 = 1$, and $M_0 = 315$

node 4 $y_{12}^3 = z_{11}^3 = z_{12}^3 = 1$, $y_{21}^3 = 0.63$, $y_{22}^3 = 0.37$, and $M_0 = 308.75$

node 5 $y_{12}^3 = y_{22}^3 = z_{11}^3 = z_{12}^3 = 1$, and $M_0 = 410$

node 6 $y_{12}^3 = y_{21}^3 = z_{11}^3 = z_{12}^3 = 1$, and $M_0 = 485$

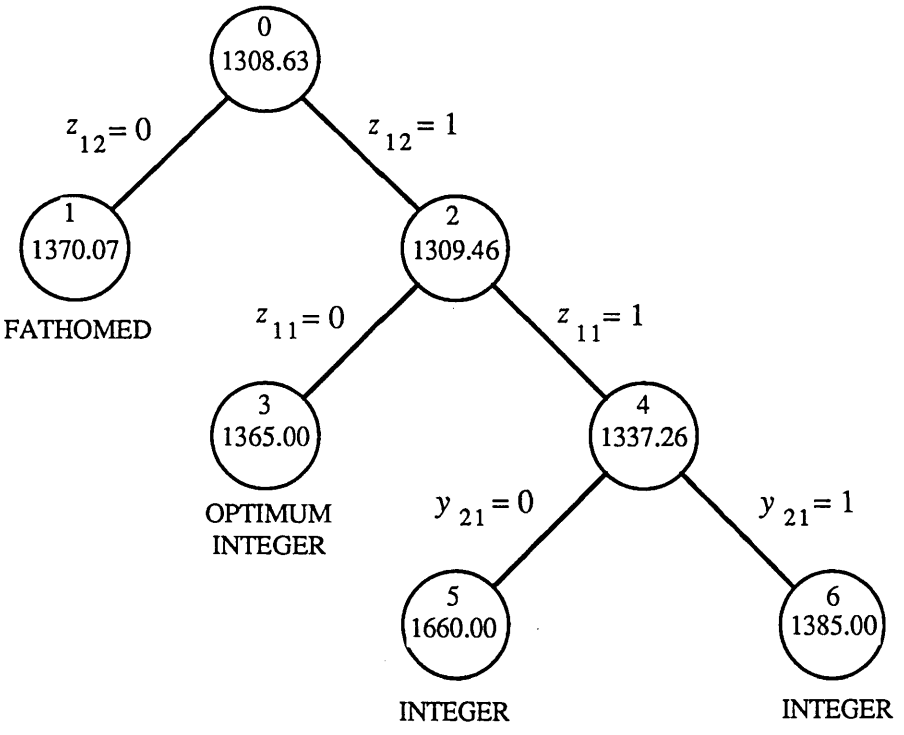


Fig. 3.4 Branch and bound tree for the example.

The new lower bound obtained from the feasible integer solution at node 3, i.e. LB = 1365.

Step 2 Using the solution obtained at node 3 of the branch and bound procedure, the transportation problem was solved and the following solution was obtained: $x_{1221} = 15$, $x_{1212} = 5$, $x_{2121} = 25$, and $x_{2112} = 55$.

The TVC = 315. The updated upper bound is UB = 1050 + 315 = 1365. At this iteration UB - LB = 0. Therefore, the optimum solution to the PDP is obtained and the procedure terminated.

The optimum solution network is shown in figure 3.5.

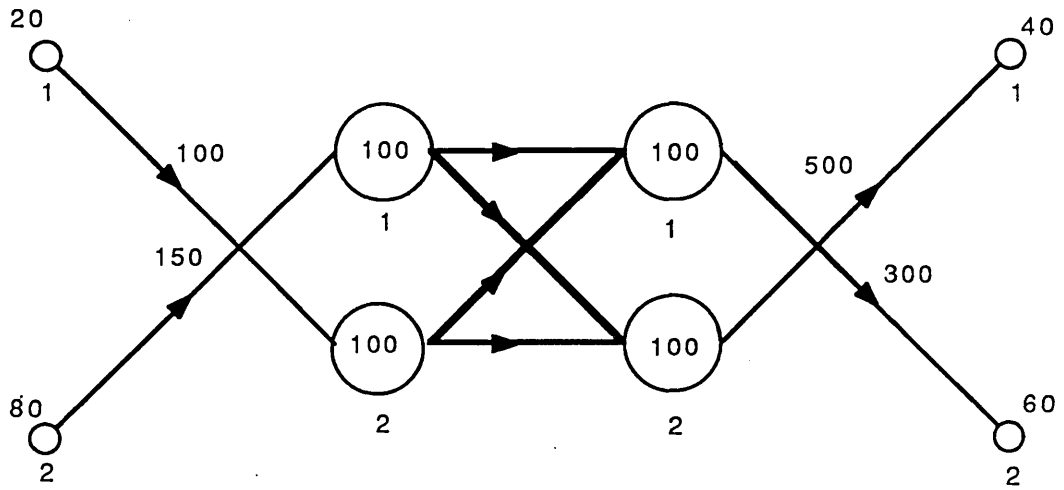


Fig. 3.5 Optimum network solution for the example

3.6 Computational results

In this section the computational results with the proposed approach have been provided to demonstrate the power of the BDM when applied to PDP. All the problems attempted had fixed operating cost coefficients f_{ij} and f_{kl} generated randomly via a uniform distribution on the interval [1000,9000]. The variable operating cost coefficients c_{ijkl} were represented by the following formula:

$$c_{ijkl} = d_{ij}\, u_{ij} + d_{jk}\, u_{jk} + d_{kl}\, u_{kl}$$

where d_{ij}, d_{jk} and d_{kl} represent the distances between the cities and collection/delivery depots. The unit cost of local transport between cities and collection/delivery depots is represented by u_{ij} and u_{kl} respectively. The u_{jk} is the

unit cost of mass transportation between collection and delivery depots.

The unit cost of local collection and distribution was chosen to be 1 ($u_{ij} = u_{kl} = 1, \forall\ i, j, k, l$). The unit cost of mass transportation is set to be 1/4 ($u_{jk} = 1/4, \forall\ j, k$). The values of local distances d_{ij} and d_{kl} were drawn from a uniform distribution between 0 and 15. Similarly, d_{jk} is drawn from a U[100,700]. Further, a uniform distribution of integers between 100 and 500 was used to create quantities of shipment from cities i to cities l.

It has been pointed out that the master problem without an augmented Benders cut represents the GAP. The computational experience of Martello and Toth [1981] shows that for generalized assignment problems if the capacities are chosen to be comparatively small, then the solution time tends to become very high. Thus, the following formula was used to generate data for the handling capacities Q_j^c and Q_k^d in order to test the PDP.

A. The values of Q_j^c and Q_k^d were determined by:

$$Q_j^c = \lfloor (0.6 * (\operatorname{Max}_i \sum_{l=1}^{N} q_{il} - \operatorname{Min}_i \sum_{l=1}^{N} q_{il})) * (\frac{N}{M}) + 0.4 * R \rfloor$$

$$\text{where} \quad R = \operatorname{Max}_{i \in I} \{ \sum_{l=1}^{N} q_{il} y_{ij} \}$$

Similarly,

$$Q_k^d = \lfloor (0.6 * (\operatorname{Max}_l \sum_{i=1}^{N} q_{il} - \operatorname{Min}_l \sum_{i=1}^{N} q_{il})) * (\frac{N}{M}) + 0.4 * R \rfloor$$

$$\text{where} \quad R = \operatorname{Max}_{l \in L} \{ \sum_{i=1}^{N} q_{il} z_{kl} \}$$

$\lfloor . \rfloor$ represents the "integer part of", y_{ij} and z_{kl} are the solutions to the initial

relaxation. The data generated above are similar to Ross and Soland's [1975] expression. All the problems generated were feasible.

B. The same formula was used for generating the data, but Q_j^c and Q_k^d values were chosen to be 70 percent of the generated values.

The Benders decomposition procedures described were implemented in Fortran IV and run on a CDC 7600 using the FTN5 compiler. The master problems were solved using rudimentary linear programming based on a branch and bound algorithm (Land and Powell [1973]). The implementation used the XMP code (Marsten [1981]) to solve LP relaxation of the master and transportation sub-problems. The generalized assignment code (Martello and Toth [1981]) was also used to solve GAP problems.

Table 3.3 gives the computational results for 52 test problems all solved to optimality (allowing for the usual convergence tolerance). Only one problem of type B terminated because of a time limit. At termination the solution was within 2.3% of the optimum. In Table 3.3 the average running time for each class of problem is given. The minimum and maximum times are also shown. It also shows the average number of major iterations of Benders, as well as the minimum and maximum iterations.

It should be noted that BDM has a better performance for data set A than for the "hard" data set B. This is due to the difficulty of solving the master problem of set B at each iteration, which reflects the tightness of their handling capacities. This feature is in line with the experience of Martello and Toth [1981]. Furthermore, for each data set the problem difficulty increases more with the number of depots M than with cities N because the number of feasible assignments is M!. The average number of iterations on both data sets is approximately the same.

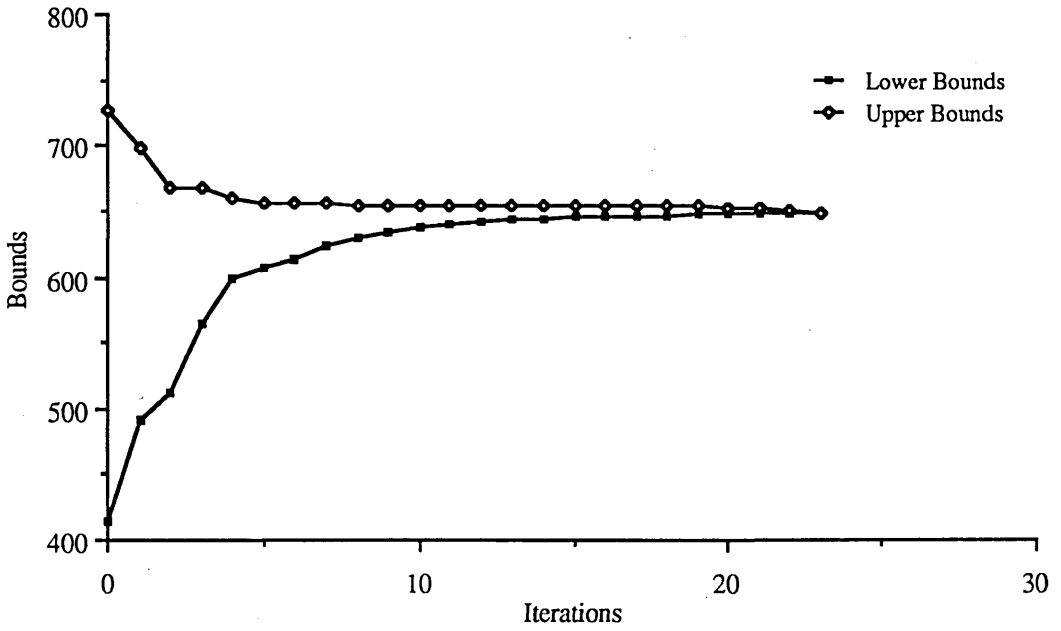
Table 3.3
Representative runs

Data Set	Problem Size				No. of 0-1 Variables	No. of Other Variables	No. of problems solved	Major Iterations			Total execution time (Sec.)		
	i	*j*	*k*	*l*				Ave.	Min.	Max.	Ave.	Min.	Max.
A	5	2	2	5	20	100	5	6	3	9	6.534	3.758	9.529
A	5	3	3	5	30	225	5	9	6	10	41.519	22.938	48.934
A	8	3	3	8	48	576	4	19	13	22	81.271	72.664	87.390
A	10	2	2	10	40	400	5	14	10	19	62.495	53.107	68.792
A	10	3	3	10	60	900	5	18	14	24	90.440	75.514	98.573
A	10	4	4	10	80	1600	2	23	18	27	180.362	145.904	214.827
B	5	2	2	5	20	100	5	6	2	10	7.282	2.536	10.385
B	5	3	3	5	30	225	5	6	4	12	48.594	28.924	60.291
B	8	3	3	8	48	576	4	19	16	23	96.281	86.877	105.120
B	10	2	2	10	40	400	5	16	13	20	79.812	66.373	90.742
B	10	3	3	10	60	900	5	20	16	22	113.421	101.726	124.691
B	10	4	4	10	80	1600	2	-	19	28*	-	196.459	300.00*

* program terminated without convergence for one of the two problems

The convergence properties of the BDM are illustrated in figure 3.6 for a problem size of 10x3x3x10. It can be seen that the gap between upper and lower bounds on the solution is reduced very sharply in the first iterations. This remains in line with past experience of the method (Geoffrion and Graves [1974]; Sherali and Adams [1984]; Côté and Laughton [1984]). It should be noted that after only 5 iterations this gap (as a percentage of the lower bound value) has fallen from 42.6% to 8.7%. At iteration 10 this gap was 2.4%, and an additional 14 iterations were run to reduce this gap to zero. The fact that the exact optimal solution required so many additional iterations suggests that many solutions are near-optimal. The ability of the algorithm to generate such solutions will be useful for sensitivity analysis.

Fig. 3.6 Convergence of the BDM



The other element which testifies to the efficiency of the BDM, i.e. the ability to solve master and transportation sub-problems efficiently, is best illustrated by considering the execution time. Table 3.4 gives details of the execution time for the problem of size 10x3x3x10 of sets A and B. For the problem of type A, the solution

presented was obtained in 93.83 seconds, of which 77% was spent solving the sequence of master problems, and 23% the transportation sub-problems and overheads. Similarly, for type B, total execution time was 115.07 seconds of which 83% was taken by masters and 17% by the rest of the problem. The fact that the optimal solution was obtained with fewer iterations for problem type B indicates the tightness of capacities in this class. However, the "hardness" of this type of problem was reflected by the greater execution time required.

Table 3.4
Detailed run time for a 10x3x3x10 problem

Major Iteration	DATA SET A			DATA SET B		
	LP bound on root node %	Execution Time		LP bound on root node %	Execution Time	
		Master	Total		Master	Total
1	94.97	1.66	4.93	90.48	2.45	5.92
2	94.84	1.64	2.48	90.02	2.53	3.28
3	94.93	1.72	2.46	89.89	2.81	3.79
4	93.95	1.77	2.67	89.33	3.36	4.10
5	93.64	2.10	2.84	88.68	3.52	4.37
6	92.86	2.30	2.87	88.27	3.72	4.41
7	92.62	2.53	3.44	88.10	3.91	4.67
8	91.41	2.87	3.56	87.30	4.47	5.34
9	90.68	2.93	3.74	86.47	4.67	5.65
10	90.80	2.85	3.63	86.00	4.86	5.69
11	90.63	2.88	3.59	85.84	4.11	5.90
12	90.54	2.81	3.51	85.83	5.22	5.80
13	90.02	2.93	3.72	84.36	5.55	6.19
14	90.00	3.05	3.79	83.73	5.42	6.03
15	89.96	3.02	3.64	83.15	5.44	6.17
16	89.90	3.12	3.98	82.34	5.91	6.55
17	89.41	3.38	4.33	81.51	6.36	7.09
18	89.25	3.74	4.71	80.86	6.94	7.56
19	89.14	3.82	4.56	79.07	7.34	8.05
20	88.61	3.91	4.94	78.53	7.76	8.51
21	88.35	4.09	4.98			
22	87.75	4.13	4.86			
23	87.38	4.35	5.27			
24	87.18	4.42	5.33			
Total Time		72.02	93.83		96.35	115.07

It should be noted that the initial iterations take less time to solve than the last ones in both types. This is because the number of constraints in the master is smaller at the beginning (one element is added at each iteration) and also because the problem is more highly structured. However, it can be seen that sub-optimizing the master as described in section 3.4.2 is generally successful in helping to keep the time spent on it small. It should also be emphasized that transportation sub-problems have little variation on their run time at each sequence of iteration.

Moreover, at each iteration the previous master solution could be used as an initial solution to the present one by simply updating the free variable M_0 in (3.20). Because it is a very primitive approach employed to solve relaxed LP problems, this was not implemented in the present version. Thus, this also partly accounts for the high proportion of time spent on the masters.

Table 3.5 gives average, minimum and maximum LP bounds at the root node, as a percentage of the lower bound to the master problem. It details each major iteration of problem size 10x3x3x10. The average LP bounds for data set A are better than data set B. In general, LP bounds are satisfactory for the given formulation.

It should be noted that although the quality of bounds improves as each Benders cut is augmented to the master, nevertheless the LB of the PDP increases at a rapid rate. Therefore the percentage of LP bounds decreases with an increase in the number of iterations. This indicates the rapid convergence of the BDM during early iterations, and that it slows during further steps. It also shows that as the number of iterations increases, greater time should be spent to obtain an optimal solution to the master. Therefore, it appears worthwhile given the experience of Mc Daniel and Devine [1977] to generate several cuts from the initial linear programming solution.

Table 3.5
LP bounds at root node (%) for a set of 10x3x3x10 problems

Major Iteration	DATA SET A				DATA SET B			
	No. of Problems	LP bound at root node (%)			No. of Problems	LP bound at root node (%)		
		Ave.	Min.	Max.		Ave.	Min.	Max.
1	5	94.30	93.32	95.74	5	90.39	89.92	91.16
2	"	93.98	92.73	95.71	"	90.02	89.42	90.80
3	"	93.61	92.58	95.19	"	89.44	88.59	90.04
4	"	93.29	91.99	95.10	"	88.81	87.86	89.71
5	"	92.79	91.66	94.69	"	88.20	87.36	89.45
6	"	92.22	91.23	94.15	"	88.05	87.12	89.41
7	"	91.47	90.60	94.08	"	87.68	86.71	89.06
8	"	91.24	90.05	93.68	"	87.14	86.56	88.48
9	"	90.81	89.53	93.52	"	86.63	86.06	87.85
10	"	90.23	88.70	92.73	"	85.80	85.03	86.81
11	"	89.90	88.66	92.10	"	85.25	84.14	86.17
12	"	89.63	88.48	91.90	"	84.86	84.09	85.84
13	"	89.34	88.15	91.68	"	84.25	83.89	85.12
14	"	89.17	88.00	91.33	"	83.67	82.97	84.58
15	4	89.18	88.07	90.44	"	83.11	82.22	84.48
16	"	89.09	88.02	90.33	"	82.45	81.47	83.90
17	2	88.59	87.77	89.41	4	81.42	80.37	82.37
18	"	88.50	87.75	89.25	"	81.17	80.52	82.06
19	"	88.37	87.59	89.14	3	80.32	79.07	81.53
20	"	87.87	87.12	88.61	"	79.58	78.53	80.37
21	1	88.35	88.35	88.35	2	79.59	79.16	80.03
22	"	87.75	87.75	87.75	"	79.09	78.53	79.66
23	"	87.38	87.38	87.38				
24	"	87.18	87.18	87.18				

3.7 Conclusion

The purpose of this Chapter has been to introduce the BDM and to establish that it offers a sound approach to solving the PDP. A practical implementation of the method suggests that the approach can work efficiently and with a reasonable degree of accuracy. The method has displayed good convergence characteristics. The time needed to solve the linear programming relaxation of the master and, thereby, to generate the initial set of cuts was not very substantial. Thus, in situations where near-optimal solutions are satisfactory, the method provides planners with a more powerful tool in

order to arrive at meaningful decisions for PDP, contrary to the previous approach (lagrangean relaxation) where no judgement could be made when it terminated prematurely. Admittedly the solution times reported may be rather conservative since the present implementation does not use the more efficient codes. However, in the next Chapters this formulation will be redefined to further exploit the BDM and available codings, and the model will also be enlarged to include more complex problems. It will be illustrated that BDM will be most appropriate for handling these extensions.

CHAPTER 4

A Strong Formulation of the Parcel Distribution Problem Based on Benders Decomposition (Model 2)

The Parcel Distribution Problem has been shown in Chapter 1 to be NP hard. However, it has been pointed out that constraints and variables, in the mixed integer linear programming formulation of the problem, naturally decompose into binary and continuous variables. Furthermore, the problem formulation has an imbedded generalized assignment structure. These encourage the use of the Benders decomposition procedure for obtaining an optimal solution.

In this Chapter, an alternative "strong" formulation of the problem will be studied. It will be shown that if the binary variables are fixed in PDP then the resulting continuous sub-problem is separable by origin and destination cities.

4.1 An alternative formulation for the PDP

The notation developed in Chapter 1 will be used. The problem can be written as the following mixed integer linear program,

Problem S

$$\text{Minimize} \quad Z = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{M}\sum_{l=1}^{N} c_{ijkl}\, x_{ijkl} + \sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij}\, y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N} f_{kl}\, z_{kl} \tag{4.1}$$

$$\text{subject to} \quad \sum_{j=1}^{M} y_{ij} = 1 \qquad i = 1, \ldots, N \tag{4.2}$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il} y_{ij} \le Q_j^c \qquad j = 1, \ldots, M \tag{4.3}$$

$$\sum_{k=1}^{M} z_{kl} = 1 \qquad l = 1, \ldots, N \tag{4.4}$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il}\, z_{kl} \le Q_k^d \qquad k = 1, \ldots, M \tag{4.5}$$

$$\sum_{k=1}^{M} x_{ijkl} = q_{il} y_{ij} \qquad \begin{array}{l} i = 1, \ldots, N \\ j = 1, \ldots, M \\ l = 1, \ldots, N \end{array} \tag{4.6}$$

$$\sum_{j=1}^{M} x_{ijkl} = q_{il} z_{kl} \qquad \begin{array}{l} i = 1, \ldots, N \\ k = 1, \ldots, M \\ l = 1, \ldots, N \end{array} \tag{4.7}$$

$$y_{ij},\, z_{kl} \in \{0, 1\} \qquad \text{for all } i, j, k, l \tag{4.8}$$

$$x_{ijkl} \ge 0 \qquad \text{for all } i, j, k, l \tag{4.9}$$

In this formulation, c_{ijkl} is the non-negative per unit cost for shipping goods from origin city i to destination city l, via intermediate depots k and l. The f_{ij} and f_{kl} are the fixed charge assignment cost for arcs $\{i, j\}$ and $\{k, l\}$. Constraints (4.2) - (4.5) are similar to (3.2) - (3.5) and relate to assignment decisions.

The "forcing" constraint (4.6) states that if city i is assigned to collection depot

j, i.e. if $y_{ij} = 1$, then total flow from origin city i to destination city l via intermediate collection depot j and all depots k must be equal to the required quantity of shipment, q_{il}. Similarly, constraint (4.7) is a "forcing" constraint for the delivery part. However, constraints (4.6) and (4.7) jointly impose a logical relation between y's and z's, i.e. if $y_{ij} = 1$ and $z_{kl} = 1$ then $x_{ijkl} = q_{il}$. The "forcing" constraints (4.6) and (4.7) are disaggregate versions of constraints (3.6) and (3.7),

$$\sum_{k=1}^{M}\sum_{l=1}^{N} x_{ijkl} = \sum_{l=1}^{N} q_{il}\, y_{ij} \qquad \begin{matrix} i = 1, \ldots, N \\ j = 1, \ldots, M \end{matrix} \qquad (4.10)$$

$$\sum_{i=1}^{N}\sum_{j=1}^{M} x_{ijkl} = \sum_{i=1}^{N} q_{il}\, z_{kl} \qquad \begin{matrix} k = 1, \ldots, N \\ l = 1, \ldots, M \end{matrix} \qquad (4.11)$$

Both versions of these constraints impose that $x_{ijkl} = q_{il}$ if $y_{ij} = z_{kl} = 1$, otherwise to be zero.

This disaggregation substantially increases the number of constraints in the formulation for even medium sized problems: with 30 origin and destination cities and 5 collection and delivery depots, the disaggregate formulation (4.1) - (4.9) contains 9000 forcing constraints of type (4.6) and (4.7); whereas the aggregate version contains only 300 constraints of type (4.10) and (4.11).

Note that in the formulation (4.1) - (4.9) the arcs are directed. Thus, each city, which can receive goods as well as sending goods, is split into two distinct origin and destination cities. Similarly, depots are separated into two sets of collection and delivery depots (Figure 4.1). In the problem where directed arcs $\{i, j\}$ and $\{k, l\}$ can be denoted by undirected arc $\{i, j\}$ (Figure 4.2) (i.e. each origin and destination city is assigned to a depot, and it sends as well as receives its commodities from that depot), then the modified formulation would reduce the number of continuous variables, for a problem of 30 cities and 5 depots, from 22,500 to 5,625.

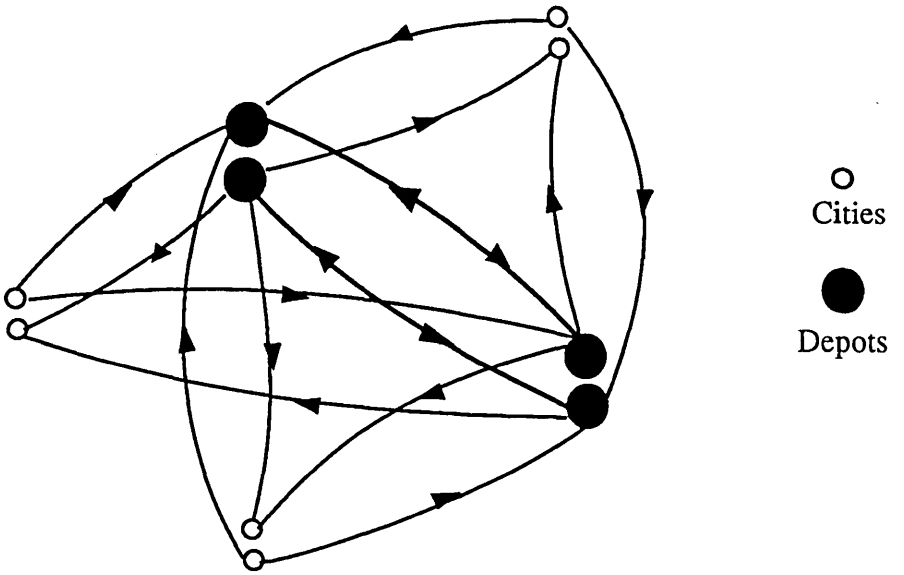


Fig. 4.1 Arcs directed PDP graph

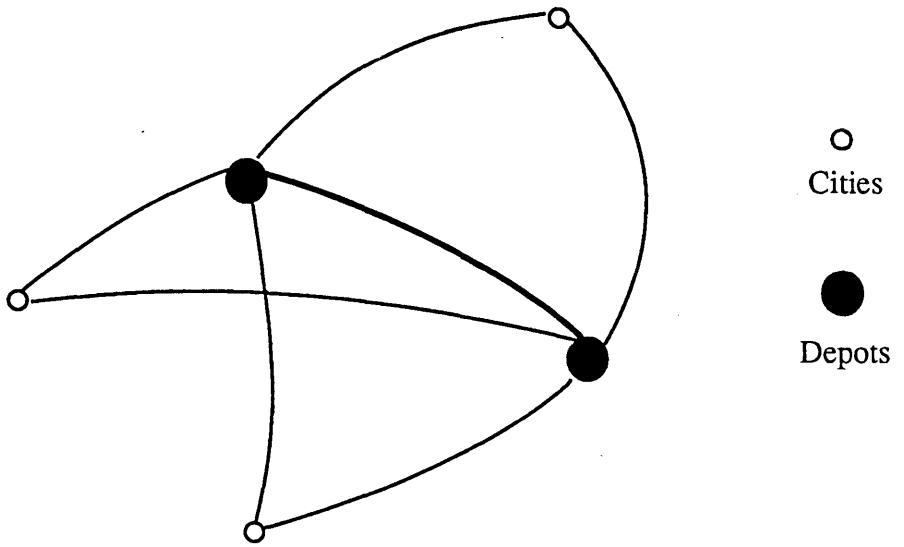


Fig. 4.2 An undirected PDP graph

Surprisingly, the seemingly less efficient and substantially larger disaggregate formulation with forcing constraints (4.6) and (4.7) instead of (4.10) and (4.11), and with transport flow variables for each origin-destination rather than for each origin, leads to a stronger formulation and effective algorithms. The disaggregated model is preferred computationally for the following reasons:

Many techniques such as LP based branch and bound, for solving problems of this kind require the solution of linear programming relaxations of the model. The disaggregate linear program provides a sharper lower bound on the value of the integer programming formulation, because the linear programming version of the disaggregate formulation is much more tightly constrained than the linear programming version of the aggregate formulation. Thus, the common practice of dropping integrality requirements in order to produce an LP relaxation, in disaggregated formulation, yields a tighter relaxation than in aggregate formulation. The important advantages of using "tight" linear programming relaxation have been pointed out by a number of authors (Beale and Tomlin [1972]; Cornuéjols, Fisher and Nemhauser [1977]; Davis and Ray [1969]; Geoffrion and Graves [1974]; Moris et al. [1978]; Magnanti and Wong [1981]; Magnanti et al. [1986]; Rardin and Choe [1979]; and Williams [1974]). It should also be noted that the extra number of constraints (4.6) and (4.7) by comparison with constraints (4.10) and (4.11), does not offer any difficulty whatever when Benders decomposition is used, since at each iteration the continuous sub-problem is separable by origin and destination cities.

In addition, Benders decomposition method utilizes information obtained from the dual of the linear programming relaxation. The disaggregate formulation (4.1) - (4.9) will produce much more effective duals. Because the disaggregate formulation has more constraints, it has a richer collection of linear programming dual variables and, therefore, provides more flexibility in algorithmic development.

4.2 Algorithms based on Benders decomposition

The typical size of real-life applications of the PDP is too large to be solved economically by existing general mixed integer linear programming codes, such as, Marsten's M31PX branch and bound code (Singhal [1982]). However, as has already been noted, a special property of the PDP enables it to be decomposed in such a way that the transportation problem becomes simpler to solve. When the binary variables are temporarily held fixed for selected y's and z's, the remaining problem for the variable x reduces to a collection of transportation problems comprising independent and separable pairs of origin-destination cities. This property decreases the size of the total problem to manageable dimensions, and suggests an application of Benders decomposition method (Benders [1962]).

4.2.1 Benders decomposition method

Benders decomposition assumes a particularly simple form when applied to PDP defined by (4.1) - (4.9) in the standard fashion.

Step 0 Initialize $h = 0$, UB = 0, LB = $-\infty$.

Let $\varepsilon \geq 0$ be a convergence tolerance parameter.

Step 1 Increment h by 1.

Solve the current relaxed master problem.

$$\text{Minimize } Z_M = \sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij}\, y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N} f_{kl}\, z_{kl} + M_0 \tag{4.12}$$

$$\text{subject to} \quad \sum_{j=1}^{M} y_{ij} = 1 \qquad i = 1, \ldots, N \tag{4.13}$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il} y_{ij} \leq Q_j^c \qquad j = 1, \ldots, M \tag{4.14}$$

$$\sum_{k=1}^{M} z_{kl} = 1 \qquad l = 1, \ldots, N \qquad (4.15)$$

$$\sum_{i=1}^{N} \sum_{l=1}^{N} q_{il}\, z_{kl} \le Q_k^d \qquad k = 1, \ldots, M \qquad (4.16)$$

$$\sum_{i=1}^{N} \sum_{j=1}^{M} \sum_{l=1}^{N} \alpha_{ijl}^{h} q_{il} y_{ij} + \sum_{i=1}^{N} \sum_{k=1}^{M} \sum_{l=1}^{N} \beta_{ikl}^{h} q_{il} z_{kl} \le M_0 \qquad (4.17)$$

$$h = 1, \ldots, H$$

$$y_{ij}, z_{kl} \in \{0, 1\} \qquad \text{for all } i, j, k, l \qquad (4.18)$$

$$M_0 \qquad \text{unbounded} \qquad (4.19)$$

Let $(y_{ij}^{h+1}, z_{kl}^{h+1}, M_0^{h+1})$ be the optimal solution and Z_M^* be the optimal value. Z_M^* is the new lower bound, i.e. put $\text{LB} = Z_M^*$. If UB - LB $\le \varepsilon$, stop with the increment accepted as the ε-optimal solution.

Step 2 Using $(y_{ij} = y_{ij}^{h+1}, z_{kl} = z_{kl}^{h+1})$, solve the transportation subproblem.

$$\text{Minimize } Z_S = \sum_{i=1}^{N} \sum_{j=1}^{M} \sum_{k=1}^{M} \sum_{l=1}^{N} c_{ijkl}\, x_{ijkl} \qquad (4.20)$$

$$\text{Subject to} \quad \sum_{k=1}^{M} x_{ijkl} = q_{il} y_{ij} \qquad \begin{matrix} i = 1, \ldots, N \\ j = 1, \ldots, M \\ l = 1, \ldots, N \end{matrix} \qquad (4.21)$$

$$\sum_{j=1}^{M} x_{ijkl} = q_{il} z_{kl} \qquad \begin{matrix} i = 1, \ldots, N \\ k = 1, \ldots, M \\ l = 1, \ldots, N \end{matrix} \qquad (4.22)$$

$$x_{ijkl} \ge 0 \qquad \text{for all } i, j, k, l \qquad (4.23)$$

Determine an optimal solution x_{ijkl}^{h+1} and its objective value Z_S^{h+1}. If,

$$Z_T = Z_S^{h+1} + \left[\sum_{i=1}^{N} \sum_{j=1}^{M} f_{ij} y_{ij} + \sum_{k=1}^{M} \sum_{l=1}^{N} f_{kl}\, z_{kl} \right] \leq \text{UB}, \text{ put UB} = Z_T,$$

and update the incumbent solution, i.e. put,

$$\{ y_{ij}^{*}, z_{kl}^{*}, x_{ijkl}^{*} \} = \{ y_{ij}^{h+1}, z_{kl}^{h+1}, x_{ijkl}^{h+1} \}.$$

If UB - LB $\leq \varepsilon$ stop; $\{ y_{ij}^{*}, z_{kl}^{*}, x_{ijkl}^{*} \}$ is a ε-optimal solution. If not, determine an optimal dual solution for the transportation sub-problem. Denote it α_{ijl}^{h+1}, β_{ikl}^{h+1}, corresponding respectively to constraints (4.21) and (4.22); go to step 1.

The standard procedure for the Benders algorithm is initiated by the user specifying initial values of y and z. An advantage of standard BDM is the fact that the set of N x N disjoint sub-problem and the relaxed master problem may be solved by any appropriate algorithm. A successful implementation of BDM depends quite heavily on the ability to solve both the master problem and the transportation sub-problem with a high level of efficiency. In the following sections, the efficient solution procedures to tackle the mentioned difficulties will be discussed.

4.3 Solution of the master problem

A critical problem posed by the BDM is the time consumed in solving the integer master problem. A general purpose mixed integer linear code such as Marsten's M31PX branch and bound code, fails to capitalize on the underlying structure of PDP. The branch and bound algorithm contributes to reducing the execution time. In any branch and bound algorithm, the effectiveness is entirely dependent on the quality of the bounds used to limit the tree search (Christofides [1985]). An outline of the calculation of such bounds will be discussed.

4.3.1 LP relaxation approach

The primary approach to obtain lower bounds for the master problem Z_M would be the usual approach of relaxing the integrality conditions and solving the resulting problem by LP codes. It is hoped that the disaggregate formulation used the linear program to provide a sharper lower bound on the value of the integer programming formulation, that is, that it more closely approximates the integer program. When solving the current master problem, it is often possible to use the LP solution obtained at a previous iteration of Benders as an initial basis for current LP relaxation of the master problem.

The described relaxations are an easy way of calculating lower bounding values for the master problem. However, a better choice of relaxation, especially as a supplement to the usual LP relaxation is the lagrangean relaxation.

4.3.2 Lagrangean relaxation approach

The master problem Z_M has two generalized assignment problems with additional constraints generated by Benders cuts. If it were not for the additional Benders cuts the master problem could have been solved as two independent generalized assignment problems. One way to retrieve that structure is to relax Benders cuts with the help of lagrange multipliers (Geoffrion [1974]). The resulting relaxed problem can then be separated into two independent generalized assignment problems. Those problems could be further relaxed by dualizing constraints (4.13) and (4.15). Then the relaxed problem is solved by an approach similar to that of Martello and Toth [1981]. As mentioned earlier, it is also possible to take advantage of the solution obtained for the master problem on a previous iteration. For example, it should be possible to use multiplier values from previous master problems to initialize the current multipliers (Fisher and Jaikumar [1978]).

It should be noted that it is not necessary to solve the master problem to optimality

on every iteration (Geaffrion [1974]; Mc Daniel and Devine [1977]; and Magnanti and Wong [1981]). Either a primal feasible non-optimal solution or a dual feasible Lagrangean solution can be fed to the sub-problem to generate a Benders cut. After a few iterations one would need to introduce a branching process in the master, in order to overcome the inherent limitations of lagrangean relaxation techniques which can not on their own bridge the duality gap existing between the partial dual and master problem.

4.4 Solution of the transportation sub-problem

At iteration h with $y_{ij} = y_{ij}^{h+1}$ and $z_{kl} = z_{kl}^{h+1}$, it is required to solve the following transportation sub-problem.

$$\text{Minimize } Z_S = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{M}\sum_{l=1}^{N} c_{ijkl}\, x_{ijkl} \tag{4.24}$$

$$\text{Subject to} \quad \sum_{k=1}^{M} x_{ijkl} = q_{il} y_{ij}^{h+1} \qquad \begin{matrix} i = 1, \ldots, N \\ j = 1, \ldots, M \\ l = 1, \ldots, N \end{matrix} \tag{4.25}$$

$$\sum_{j=1}^{M} x_{ijkl} = q_{il}\, z_{kl}^{h+1} \qquad \begin{matrix} i = 1, \ldots, N \\ k = 1, \ldots, M \\ l = 1, \ldots, N \end{matrix} \tag{4.26}$$

$$x_{ijkl} \geq 0 \qquad \text{for all } i, j, k, l \tag{4.27}$$

The above problem, as indicated previously, can be separated as N x N independent sub-problems. Since it has the following block structure;

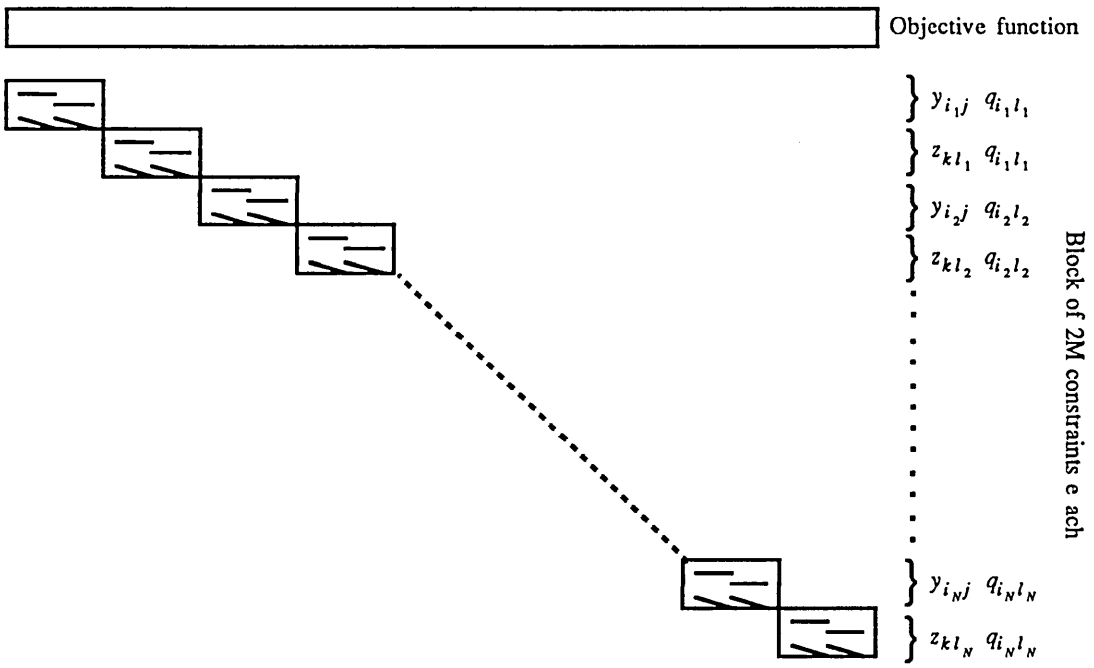


Fig. 4.3 Block structure of coefficients

Thus, for a given origin and destination city (let $i = i'$ and $l = l'$), the problem could be separated as the following LP problem;

$$\text{Minimize } Z_{il}^{\cdot} = \sum_{j=1}^{M} \sum_{k=1}^{M} c_{i'jkl'} \, x_{i'jkl'} \tag{4.28}$$

$$\text{Subject to} \quad \sum_{k=1}^{M} x_{i'jkl'} = q_{i'l'} \, y_{i'j}^{h+1} \qquad j = 1, \ldots, M \tag{4.29}$$

$$\sum_{j=1}^{M} x_{i'jkl'} = q_{i'l'} \, z_{kl'}^{h+1} \qquad k = 1, \ldots, M \tag{4.30}$$

$$x_{i'jkl'} \geq 0 \qquad \text{for all } j, k \tag{4.31}$$

Since y_{ij}^{h+1} must satisfy (4.2), then it implies that $y_{i'j}^{h+1} = 1$ for the j^{th} index. This index is represented by $j(i')$.

From constraint (4.29), it implies that $x_{i'jkl'} = 0$ for all i', j, k, l' with $j(i')$. Therefore, constraint (4.29) reduces to

$$\sum_{k=1}^{M} x_{i'\, j(i')\, k\, l'} = q_{i'l'} \tag{4.32}$$

Similarly, since $z_{kl'}^{h+1}$ must satisfy (4.4), then it implies that $z_{kl'}^{h+1} = 1$ for the k^{th} index. This index is represented by $k(l')$.

Thus, it could be deduced that constraint (4.30) will reduce to

$$\sum_{j=1}^{M} x_{i'\, j\, k(l')\, l'} = q_{i'l'} \tag{4.33}$$

The problem Z'_{il} could therefore be written as

$$\text{Minimize } Z'_{il} = \sum_{j=1}^{M}\sum_{k=1}^{M} c_{i'\, j(i')\, k(l')\, l'}\, x_{i'\, j(i')\, k(l')\, l'} \tag{4.34}$$

$$\sum_{k=1}^{M} x_{i'\, j(i')\, k\, l'} = q_{i'l'} \tag{4.35}$$

$$\sum_{j=1}^{M} x_{i'\, j\, k(l')\, l'} = q_{i'l'} \tag{4.36}$$

$$x_{i'jkl'} \geq 0 \tag{4.37}$$

It can be simply observed that the only solution for problem Z'_{il} is $x_{i'\, j(i')\, k(l')\, l'} =$

$q_{i'l'}$. The solution procedure will be repeated for each origin and destination city. The total optimal solution to the sub-problem Z_S is equal to the following summation;

$$Z_S = \sum_{i=1}^{N} \sum_{l=1}^{N} Z_{il}$$

4.5 Dual solution of the transportation sub-problem

In order to satisfy the requirements of the BDM, the optimal dual variables must also be computed. The required optimal dual variables (α_{ijl}^{h+1}, β_{ikl}^{h+1}) to the problem Z_S with y and z fixed at y^{h+1} and z^{h+1} respectively, must be synthesized from available dual variables to problems Z'_{il}, since problem Z_S is solved via a set of N x N independent problems Z'_{il}.

Since the mechanics of Benders decomposition are the same at each iteration, while synthesizing the optimal dual variables, the superscript h+1 will be dropped for notational simplicity.

Let the optimal dual variables to Z'_{il} for a given origin i and destination city l to be:

μ_j^{il} (corresponding to constraints 4.29), and γ_k^{il} (corresponding to constraints 4.30). It will be shown that the following relation exists between the optimal dual variables of Z'_{il} and Z_S.

$$\mu_j^{il} = \alpha_{ijl} \qquad \forall\, i,j,l \qquad (4.38)$$

$$\upsilon_k^{il} = \beta_{ikl} \qquad \forall\, i,k,l \qquad (4.39)$$

In order to deduce the above relations, the duals of problem Z'_{il} and Z_S must be compared. The duals of Z'_{il} for each origin i and destination city l, which may be combined into a single LP since there are no common variables. That is, (μ, υ) is an optimal solution of

$$\text{Maximize} \sum_{i=1}^{N}\sum_{l=1}^{N}\left[\sum_{j=1}^{M}\mu_j^{il}(-q_{il}y_{ij}) + \sum_{k=1}^{M}\upsilon_k^{il}(-q_{il}z_{kl})\right]$$

$$\text{subject to} \quad \mu_j^{il} + \upsilon_k^{il} \geq -c_{ijkl} \qquad \forall\, i,j,k,l \qquad (4.40)$$

The dual of problem Z_S is now considered

$$\text{Maximize} \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{l=1}^{N}\alpha_{ijl}(-q_{il}y_{ij}) + \sum_{i=1}^{N}\sum_{k=1}^{M}\sum_{l=1}^{N}\beta_{ikl}(-q_{il}z_{kl})$$

$$\text{subject to} \quad \alpha_{ijl} + \beta_{ikl} \geq -c_{ijkl} \qquad \forall\, i,j,k,l \qquad (4.41)$$

Note that for any fixed i and l, the dual constraints are disjoint. Therefore it is clear that (4.40) and (4.41) are identical optimization problems. Hence relations (4.38) and (4.39) provide required duals of the problem Z_S which are used in order to generate Benders cuts in the BDM.

4.6 Strengthening Benders Cuts

The standard Benders cut (4.17) is but one of many possible lower bounds on the cost of PDP. As has already been noted, this cut is derived from dual variables to the linear programming solution of transportation sub-problems. It has also been noted that a considerable number of these constraints may simply be dropped from problem

(Z'_{il}), since they may always be satisfied without any effect on the value of the objective function. Thus, these problems are usually degenerate, and their dual problems typically have alternative optimal solutions each defining a Benders cut. In this section, it will be discussed how better cuts might be generated.

It is shown in section (4.5) that the solution to problem Z'_{il}, after preliminary reductions, may be determined by a simple inspection. Hence it is not required to solve i x l linear programming problems for a given origin and destination pair. Thus, in order to calculate the preferred optimal dual variables, the following formulae may be derived from the Duality theorem.

$$\sum_{j=1}^{M}\sum_{k=1}^{M} C_{i'jkl'}\, x_{i'jkl'} = \sum_{j=1}^{M} \mu_j^{i'l'}(-q_{i'l'}\, y_{i'j'}) + \sum_{k=1}^{M} \upsilon_k^{i'l'}(-q_{i'l'}\, z_{kl'}) \qquad (4.42)$$

Notice that $(-\, q_{i'l'}\, y_{i'j}) = 0$, when $j \neq j(i')$, and similarly also $(-q_{i'l'}\, z_{kl'}) = 0$, when $k \neq k(l')$. It can therefore be deduced that:

$$\mu_{j\,(i')}^{i'l'} = C_{i'\,j\,(i')\,k\,(l')\,l'}$$

$$\upsilon_{k\,(l')}^{i'l'} = 0$$

The corresponding duals for the remaining constraints may take a range of values without affecting optimality. The best choice for $\mu_j^{i'l'}$ when $j \neq j\,(i')$ is considerd to be:

$$\mu_j^{i'l'} = \underset{j}{\text{Min}}\, \{C_{i'jkl'}\}$$

For any fixed $\mu_j^{i'l'}$, the optimal choice of $\upsilon_k^{i'l'}$ is obvious since there are no joint constraints with $\upsilon_k^{i'l'}$

$$\upsilon_k^{i'l'} = \underset{j}{\text{Min}}\, \{C_{i'jkl'} - \mu_j^{i'l'}\}$$

The lower bound obtained from a Benders cut which is derived from the above

choice of dual variables is generally better. This also illustrates the fundamental role of duality (in generating the Benders cut) for developing objective function bounds.

4.7 Numerical example

The PDP studied in the example involved 3 origin cities and 3 destination cities. The delivery operation is carried out via two intermediate depots, namely, collection and delivery depots. Estimates of these demands are shown in Table 4.1.

Table 4.1
Estimated delivery demand

i \ l	1	2	3
1	20	12	0
2	0	24	0
3	4	0	16

The fixed prices charged for each origin city and collection depot pair are shown in Table 4.2. Similarly, the fixed prices charged for each delivery depot and destination city pair are given in Table 4.3.

Table 4.2
fixed collection charges

i \ j	1	2
1	60	30
2	40	60
3	70	20

Table 4.3
fixed delivery charges

k \ l	1	2	3
1	20	60	60
2	40	30	10

There are also unit operating costs charged for local collection and local delivery, and a mass unit transport cost between pairs of collection and delivery depots. These are shown in Figures 4.4, 4.5, and 4.6 respectively. The large circle represents the depot and the small circle the city. Numbers above the arrows are unit costs.

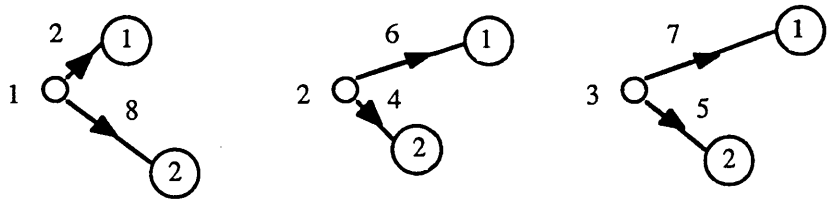


Fig. 4.4 Per unit collection costs

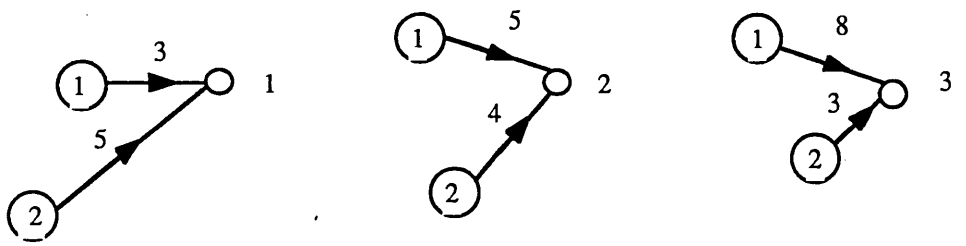


Fig. 4.5 Per unit delivery costs

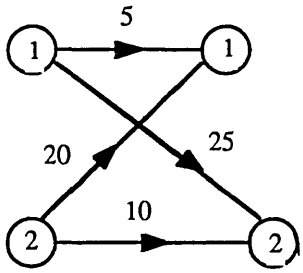


Fig. 4.6 Per unit transportation costs

Table 4.4 shows the per unit cost for each pair of origin and destination cities via intermediate depots. These are calculated from the above figures.

Table 4.4
Origin - Destination Unit Cost Matrix

i \ l		1		2		3	
	j \ k	1	2	1	2	1	2
1	1	10	32	12	31	15	30
	2	31	23	33	22	36	21
2	1	14	36	16	35	19	34
	2	27	19	29	18	32	17
3	1	15	37	17	36	20	35
	2	28	20	30	19	33	18

It is also assumed that each depot has total throughput capacity of 60 units.

The following describes a step by step solution using BDM,

Step 0 Initialization

The iteration step set to 0, i.e. $h = 0$.

The upper bound (UB) set to ∞.

The lower bound (LB) set to $-\infty$.

The convergence tolerance parameter set to 0, i.e. $\varepsilon = 0$.

Step 1 Increment step by 1, i.e. $h = 1$.

Initially the master problem could be solved as two generalized assignment problems, since there is no additional Benders cut to couple the two sub-problems. Thus, two GAPs are solved to obtain,

$$y_{12}^1 = y_{21}^1 = y_{32}^1 = z_{11}^1 = z_{22}^1 = z_{23}^1 = 1$$

and all other variables are equal to zero. The lower bound obtained is equal to the summation of optimal solutions of the two GAPs, i.e. LB = 90 + 60 = 150.

Step 2 Fixing the y's and z's obtained at step 1, yields a sub-problem which can be separated as 9 individual sub-problems. These problems are solved using results of section (4.4), and the following optimal solution is obtained;

$x_{1211} = 20$, $x_{1222} = 12$, $x_{2122} = 24$, $x_{3211} = 4$, and $x_{3233} = 16$.

The total optimal variable cost (TVC) is equal to the summation of the optimal solution to 9 individual problems, i.e. TVC = 2124. It was shown, in section (4.5), that dual solutions of the sub-problem are equal to the dual solutions of the 9 individual problems. Thus, the following dual solutions are derived from them,

$$\alpha_{111}^1 = 32,\ \alpha_{121}^1 = 31,\ \alpha_{112}^1 = 12,\ \alpha_{122}^1 = 22,\ \alpha_{212}^1 = 35,\ \alpha_{222}^1 = 29,$$

$$\alpha_{311}^1 = 37,\ \alpha_{321}^1 = 28,\ \alpha_{313}^1 = 35,\ \alpha_{323}^1 = 18.$$

$$\beta^1_{111} = -22,\ \beta^1_{121} = -8,\ \beta^1_{212} = -19,\ \beta^1_{222} = -11,\ \beta^1_{311} = -22,$$

$$\beta^1_{321} = -8,\ \beta^1_{313} = -15.$$

Thus, the first Benders cut derived is as follows,

$$784y_{11} + 884y_{12} + 840y_{21} + 696y_{22} + 708y_{31} + 400y_{32} - 528z_{11} - 192z_{21} - 456z_{12} - 264z_{22} - 240z_{13} - M_0 \leq 0$$

The upper bound is now updated to be UB = TFC + TVC = 2274.

Step 1 Increment step by 1, i.e. $h = 2$.

Add the first Benders cut to the master problem, using the Lagrangean relaxation of the master problem. A feasible lower bound is obtained of value 698. The feasible solution is,

$$y^2_{11} = y^2_{22} = y^2_{32} = z^2_{11} = z^2_{22} = z^2_{13} = 1$$

Step 2 Using the current fixed variables, the transportation problem is solved, and the following result is obtained,

$x_{1111} = 20$, $x_{1122} = 12$, $x_{2222} = 24$, $x_{3211} = 4$, and $x_{3213} = 16$.

The TVC = 1644. The updated upper bound is equal to UB = 1894. The new dual variables are obtained from the 9 individual problems, and the second Benders cut is derived to be,

$$344y_{11} + 1016y_{12} + 384y_{21} + 696y_{22} + 620y_{31} + 400y_{32} - 192z_{21} - 396z_{22} - 240z_{13} - M_0 \leq 0$$

Step 1 Increment step by 1, i.e. $h = 3$.

The second cut is augmented to the master problem, which is then solved using a tree search to obtain a feasible solution,

$$y^3_{12} = y^3_{22} = y^3_{31} = z^3_{11} = z^3_{12} = z^3_{23} = 1.$$

The new lower bound is LB = 1126.

Step 2 The current fixed variables would result in the following solution of the transportation problem,

x_{1211}= 20, x_{1212}= 12, x_{2212}= 24, x_{3111}= 4, and x_{3123}= 16.

The upper bound remained the same. The 3rd cut was generated to be,

$$784y_{11}+ 1016y_{12}+ 384y_{21}+ 696y_{22}+ 620y_{31}+ 368y_{32}- 440z_{11}- 160z_{21} - 396z_{22}- 240z_{13}- M_0 \leq 0$$

Step 1 Increment step by 1, i.e. $h = 4$.

The third cut is augmented to the previous master problem, which is then solved using a tree search to optimality. An optimal feasible solution is obtained to be,

$$y_{11}^4 = y_{21}^4 = y_{32}^4 = z_{11}^4 = z_{12}^4 = z_{23}^4 = 1.$$

The lower bound is LB = 1338.

Step 2 Using the fixed variables obtained at the third cut, the transportation problem is solved and the following results are obtained,

x_{1111}= 20, x_{1112}= 12, x_{2112}= 24, x_{3211}= 4, and x_{3223}= 16.

The updated upper bound is UB = 1338. At this iteration UB = LB = 1338. Therefore, a feasible optimal solution to the PDP example is calculated to be 1338.

4.8 Conclusion

It is common practice to economize on the number of constraints in large scale problems. The major conclusion arising from this Chapter is contrary to this practice. It is illustrated that a seemingly less efficient and substantially larger disaggregate formulation of PDP can yield more efficient algorithms.

It was also demonstrated how in some cases a large transportation problem could be separated into smaller size problems whose solutions can be determined by inspection. Therefore, the large number of constraints in comparison with the model presented in Chapter 3, does not offer any difficulty.

Another conclusion reached was that a disaggregate model generated much sharper Benders cuts, since it utilizes more information from the dual of the transportation problem. Hence, the solution to the problem could be obtained in a smaller number of iterations.

Also in this Chapter, special problem structures of the master were exploited in order to develop efficient solution techniques. This will be extended in the next Chapter, since it is noted that in all the implementations of Benders decomposition, a large proportion of the solution time is used by the master problem.

CHAPTER 5

An Algorithm for the Formulation in Model (2)

It has been pointed out in Chapter 4 that the performance of the BDM rested partially on the efficiency with which the master problem could be solved. The master problem becomes a mixed integer problem with a single continuous variable when the Benders cuts are added. Because of the necessity to solve a sequence of such problems, the performance of BDM can be computationally prohibitive if an efficient procedure is not available. The development of such a procedure is based on branch and bound, using bounds obtained from LP and lagrangean relaxation presented in this Chapter. These relaxations provide lower bounds on the master problem and at the same time good lower bounds on the optimal value of the PDP.

5.1 The generalized assignment master problem

The master problem given in the Benders decomposition technique of section 4.2.1 was as follows :

$$\text{Minimize } Z_M = \sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij}\, y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N} f_{kl}\, z_{kl} + M_0 \tag{5.1}$$

$$\text{subject to} \quad \sum_{j=1}^{M} y_{ij} = 1 \qquad i = 1, \ldots, N \tag{5.2}$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il} y_{ij} \le Q_j^c \qquad j = 1, \ldots, M \tag{5.3}$$

$$\sum_{k=1}^{M} z_{kl} = 1 \qquad l = 1, \ldots, N \tag{5.4}$$

$$\sum_{i=1}^{N}\sum_{l=1}^{N} q_{il}\, z_{kl} \le Q_k^d \qquad k = 1, \ldots, M \tag{5.5}$$

$$\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{l=1}^{N} \alpha_{ijl}^{h} q_{il} y_{ij} + \sum_{i=1}^{N}\sum_{k=1}^{M}\sum_{l=1}^{N} \beta_{ikl}^{h} q_{il} z_{kl} \le M_0 \tag{5.6}$$

$$h = 1, \ldots, H$$

$$y_{ij},\, z_{kl} \in \{0,\, 1\} \qquad \text{for all } i, j, k, l \tag{5.7}$$

$$M_0 \ge 0 \qquad \text{unbounded} \tag{5.8}$$

where,

H Number of Benders cuts added so far.

α_{ijl}^{h} , β_{ikl}^{h} Coefficients for the h^{th} cut calculated from the dual solution to transportation sub-problem.

M_0 Slack variable added for modelling convenience.

The other variables were described in section (1.3).

The above master problem is a mixed integer programming problem with a single

continuous variable M_0. The solution of mixed integer programming problems has already received much attention (Geoffrion and Marsten [1972]; Geoffrion [1976]). The different solution approaches have been divided into two categories: exact methods, where convergence towards the optimal solution can be guaranteed, and heuristic methods where no such guarantee can be given. For the purpose of this Chapter only exact methods will be considered.

5.2 The solution technique selected for the master problem

The master problem defined by (5.1) - (5.8) consists of two generalized assignment problems with additional Benders cuts (constraints). Many algorithms were developed for solving generalized assignment problems (Ross and Soland [1975]; Martello and Toth [1981]; and Fisher, Jaikumar and Van Wassenhove [1986]). These algorithms could be easily adapted to deal with the above problem. However a basic weakness with the above approach is that the master problem must be solved at every iteration of the Benders decomposition method. Hence, even in moderate size problems solving the master problem becomes a formidable task, as additional cuts are generated at each iteration .

Branch and bound algorithms are currently the most widely applied optimization techniques for such a problem. As is the case with all branch and bound algorithms, their effectiveness is entirely dependent on the quality of the bounds used to limit the tree search. Therefore, the derivation of such bounds will be discussed first.

5.3 Bounds from LP relaxation

One of the basic questions faced in the design of any branch and bound algorithm for the master problem is which relaxation to use for (lower) bounding. The chosen relaxation must be a suitable compromise between ease of computation on one hand and tightness of the resulting bound on the other. Generally a relaxation which yields a very

tight lower bound on the optimal value of the master will be very costly to compute, whereas an easily computed relaxation is likely to result in relatively poor bounds and consequently a large tree search. So there is a trade off between the quality of bounds and the time needed to compute them. The most obvious choice, the one that would be made by default, is the usual LP relaxation of the master problem obtained by relaxing constraint (5.7) to

$$0 \le y_{ij} \le 1 \qquad \text{for all } i, j \qquad (5.9a)$$

$$0 \le z_{kl} \le 1 \qquad \text{for all } k,l \qquad (5.9b)$$

This relaxation exploits the efficiency of the new LP codes, for example Marsten's [1981] XMP code, which is particularly difficult in the usual situation where the generalized upper bound (GUB) constraints (5.2) and (5.4) constitute a large fraction of all constraints.

Another important issue that should favor this relaxation is the fact that since the dissagreggate formulation of PDP was considered, then its relaxed primal problem has a smaller feasible region (Magnanti and Wong [1981]). Since the BDM utilizes the dual of this relaxation, then the Benders cut generated from these duals yields to "tight" constraints in the master problem. Hence LP relaxation provides a sharper lower bound on the value of the master problem.

5.4 Bounds from lagrangean relaxation

The relaxation to be developed is based on the fact that the master problem defined by (5.1) - (5.8) could be solved as two independent generalized assignment problems, if it were not for the Benders cuts. One way to handle these cuts is to introduce lagrange multipliers $\lambda_h \ge 0$, $(h = 1, \ldots, H)$ and then to obtain the lagrangean dual program by bringing them into the objective function. The result is

problem L_λ:

$$\text{Minimize } L_\lambda = \sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij}y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N} f_{kl}z_{kl} + M_0 + \sum_{h=1}^{H} \lambda_h \Big(\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{l=1}^{N} \alpha_{ijl}^{h} q_{il} y_{ij} + \sum_{i=1}^{N}\sum_{k=1}^{M}\sum_{l=1}^{N} \beta_{ikl}^{h} q_{il} z_{kl} - M_0 \Big) \tag{5.10}$$

subject to (5.2) - (5.5), (5.7), and (5.8)

If the lagrange multipliers λ_h are chosen so that

$$\sum_{h=1}^{H} \lambda_h = 1 \tag{5.11}$$

the variable M_0 will vanish and a new relaxed master problem will result, namely to minimize :

$$L_\lambda = \sum_{i=1}^{N}\sum_{j=1}^{M} \Big(f_{ij} + \sum_{h=1}^{H}\sum_{l=1}^{N} \lambda_h \alpha_{ijl}^{h} \Big) y_{ij} + \sum_{k=1}^{M}\sum_{l=1}^{N} \Big(f_{kl} + \sum_{h=1}^{H}\sum_{i=1}^{N} \lambda_h \beta_{ikl}^{h} \Big) z_{kl} \tag{5.12}$$

subject to (5.2) - (5.5), and (5.7)

The problem L_λ can now be separated as two generalized assignment problems to be designated as L_λ^y and L_λ^z, one for each GAP in y and z, respectively. Thus, the following formula for the optimal value of L_λ could be written :

$$V(L_\lambda) = V(L_\lambda^y) + V(L_\lambda^z) \tag{5.13}$$

These two problems can be further relaxed by defining lagrange multipliers $\mu_i \geq 0$, $(i = 1, \ldots, N)$ for constraints (5.2), and similarly $\pi_l \geq 0$, $(l = 1, \ldots, N)$ for

constraints (5.4). Therefore L_λ separated as the following problems :

$$\text{Minimize } L^{y}_{\lambda,\mu} = \sum_{i=1}^{N}\sum_{j=1}^{M} (f_{ij} + \mu_i + \sum_{h=1}^{H}\sum_{l=1}^{N} \lambda_h \alpha^{h}_{ijl}) \, y_{ij} - \sum_{i=1}^{N} \mu_l \qquad (5.14)$$

$$\text{subject to} \quad \sum_{i=1}^{N}\sum_{l=1}^{N} q_{il} y_{ij} \le Q^{c}_{j} \qquad j = 1, \ldots, M$$

$$y_{ij} \in \{0, 1\} \qquad \text{for all } i,j$$

and similarly,

$$\text{Minimize } L^{z}_{\lambda,\pi} = \sum_{k=1}^{M}\sum_{l=1}^{N} (f_{kl} + \pi_l + \sum_{h=1}^{H}\sum_{l=1}^{N} \lambda_h \beta^{h}_{ikl}) \, z_{kl} - \sum_{l=1}^{N} \pi_l \qquad (5.15)$$

$$\text{subject to} \quad \sum_{i=1}^{N}\sum_{l=1}^{N} q_{il} z_{kl} \le Q^{d}_{k} \qquad k = 1, \ldots, M$$

$$z_{kl} \in \{0, 1\} \qquad \text{for all } k, l$$

Therefore (5.13) could be modified as:

$$V(L_{\lambda,\mu,\pi}) = V(L^{y}_{\lambda,\mu}) + V(L^{z}_{\lambda,\pi}) \qquad (5.16)$$

where $L_{\lambda,\mu,\pi}$ represents the lagrangean relaxation of the master problem. The term "relaxation" is applied to $L_{\lambda,\mu,\pi}$ because its feasible region is larger than that of the original master problem, and its objective value is less than or equal to that of the original master problem on the true feasible region (5.2) - (5.8). These properties are sufficient to ensure that $L_{\lambda,\mu,\pi}$ provides a lower bound on the optimal value of the

master problem. Geoffrion [1974] has demonstrated the fact that a lagrangean relaxation is equivalent to a partial dual, so that many of the results of duality theory are available. In particular

$$Z_D \leq Z_M \tag{5.17}$$

where
$$Z_D = \underset{\substack{\lambda \in S \\ \mu,\pi \geq 0}}{\text{Max}} \; (L_{\lambda,\mu,\pi}) \tag{5.18}$$

$$S = \{\lambda \geq 0 \mid \sum \lambda_h = 1\}$$

The aim in introducing $L^y_{\lambda,\mu}$ and $L^z_{\lambda,\pi}$ is to generate a relaxed version of Z_M which will be easier to solve. Both the above problems could be solved as M 0-1 knapsack problems, once λ, μ, and π are fixed, using the procedure based on Martello and Toth's algorithm [1977] which was pointed out in section 2.4. Problem (5.18) on the other hand could be used to select values of λ, μ, and π to provide a lower bound on the value of Z_M. There remains to determine whether a solution obtained from $L^y_{\lambda,\mu}$, $L^z_{\lambda,\pi}$, and (5.18) is optimal in Z_M and to develop a method to solve (5.18).

5.5 Projection method for lagrange multipliers

In the previous section it is assumed that the lagrange multipliers λ_h can be chosen so that

$$\sum_{h=1}^{H} \lambda_h = 1 \tag{5.19}$$

However, this assumption has not been discussed yet. This is taken into account by projecting the multipliers λ_h onto the set S defined in (5.18), using the inspection

procedure which is a specialization of work described in Held, Wolfe and Crowder [1974].

Given a positive vector $\bar{\lambda}_h$ ($h = 1, \ldots, H$), a vector of multipliers $\lambda \in S$ will be sought where

$$S = \{ \lambda \geq 0 \mid \sum_{h=1}^{H} \lambda_h = 1 \} \tag{5.20}$$

The problem can be written more formally as

$$\text{Min} \{ \sum_{h=1}^{H} 1/2 \, (\lambda_h - \bar{\lambda}_h)^2 \quad \text{s.t. } \lambda \in S \} \tag{5.21}$$

By the Kuhn-Tuker theorem (Bazaraa and Shetty [1979]), sufficient conditions that $\lambda \in S$ solves (5.21) are the existence of multipliers η, ω_h such that

$$\lambda_h - \bar{\lambda}_h + \eta - \omega_h = 0 \qquad h = 1, \ldots, H \tag{5.22}$$

$$\lambda_h \, \omega_h = 0 \qquad h = 1, \ldots, H \tag{5.23}$$

$$\omega_h \geq 0$$

Such a vector λ is easily constructed. Suppose that the components of $\bar{\lambda}$ are ordered so that

$$\bar{\lambda}_1 \geq \bar{\lambda}_2 \ldots \geq \bar{\lambda}_H \tag{5.24}$$

then define

$$\lambda_h = \text{Max}\ (\bar{\lambda}_h - \eta\ ,\ 0) \tag{5.25}$$

$$\omega_h = \text{Max}\ (\eta\ - \bar{\lambda}_h\ ,\ 0) \tag{5.26}$$

Constraints (5.22) and (5.23) together with the positivity of λ are satisfied. If the value of η is fixed, then constraint (5.19) is also satisfied. Given (5.24) and (5.25) it is easy to check that there will exist an index H_0 such that

$$\lambda_h > 0 \qquad \text{if } h \le H_0 \tag{5.27}$$

$$\lambda_h = 0 \qquad \text{otherwise} \tag{5.28}$$

Hence

$$\sum_{h=1}^{H} \lambda_h = \sum_{h=1}^{H_0} \lambda_h = \sum_{h=1}^{H_0} (\bar{\lambda}_h - \eta) = \sum_{h=1}^{H_0} \bar{\lambda}_h - H_0\,\eta = 1 \tag{5.29}$$

$$\eta = \Big[\sum_{h=1}^{H_0} \bar{\lambda}_h - 1\Big] / H_0 \tag{5.30}$$

The value for H_0 is obtained by calculating (5.30) iteratively starting with $H_0 = 1$. The procedure is stopped when a value of λ is found such that $\Sigma\, \lambda_h = 1$. It can be easily verified that

$$H_0 = \text{Min}\ \Big\{H / (\sum_{h=1}^{H} \bar{\lambda}_h - 1) \mid H > \bar{\lambda}_{h+1}\Big\} \tag{5.31}$$

5.6 Duality gaps

A duality gap is said to exist between the master and its lagrangean relaxation if a strict in equality holds in (5.17). Insight as to why a duality gap exists is gained by

observing that problem (5.17) is equivalent to the LP dual of the LP relaxation of Z_M. This was first pointed out by Nemhauser and Ullman [1968]. When no such gap exists, an optimal solution $(\lambda^*, \mu^*, \pi^*, y^*, z^*)$ of (5.18) is also optimal in Z_M provided :

$$\text{(i)}\quad L_{\lambda^*,\mu^*,\pi^*} = \sum_{i=1}^{N}\sum_{j=1}^{M}(f_{ij} + \mu_i^* + \sum_{h=1}^{H}\sum_{l=1}^{N}\lambda_h^*\alpha_{ijl}^h)\, y_{ij}^* + \sum_{k=1}^{M}\sum_{l=1}^{N}(f_{kl} + \pi_l^* + \sum_{h=1}^{H}\sum_{l=1}^{N}\lambda_h^*\beta_{ikl}^h)\, z_{kl}^* - \sum_{i=1}^{N}\mu_i^* - \sum_{l=1}^{N}\pi_l^*$$

$$\mu_i^*(\sum_{j=1}^{M} y_{ij}^* - 1) = 0 \qquad i = 1, \ldots, N$$

$$\text{(ii)}\quad \pi_l^*(\sum_{k=1}^{M} z_{kl}^* - 1) = 0 \qquad l = 1, \ldots, N$$

$$\lambda_h^*(\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{l=1}^{N}\alpha_{ijl}^h q_{il} y_{ij}^* + \sum_{i=1}^{N}\sum_{k=1}^{M}\sum_{l=1}^{N}\beta_{ikl}^h q_{il} z_{kl}^* - M_0) = 0 \qquad h = 1, \ldots, H \tag{5.32}$$

$$\sum_{j=1}^{M} y_{ij}^* \le 1 \qquad i = 1, \ldots, N$$

$$\text{(iii)}\quad \sum_{k=1}^{M} z_{kl}^* \le 1 \qquad l = 1, \ldots, N$$

$$\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{l=1}^{N}\alpha_{ijl}^h q_{il} y_{ij}^* + \sum_{i=1}^{N}\sum_{k=1}^{M}\sum_{l=1}^{N}\beta_{ikl}^h q_{il} z_{kl}^* \le M_0 \qquad h = 1, \ldots, H$$

If on the other hand there exists a duality gap, then (5.18) fails to yield a solution $(\lambda^*, \mu^*, \pi^*, y^*, z^*)$ satisfying the optimality conditions (5.32). It follows that it

is not possible to establish directly whether a solution (y^*, z^*) obtained from (5.18) is also optimal in Z_M. Instead one has to try to overcome this duality gap by some other artificial means. This problem has been investigated by various authors, including Fisher et al. [1975]. They indicated that branch and bound could be viewed as a method which guarantees to bridge this duality gap by a systematic search, and also illustrated that bounds derived from the LP relaxation of the master could be naturally replaced by dual problems for selecting lagrange multipliers. This method requires an efficient technique to solve the lagrangean relaxation at each node.

5.7 Subgradient optimization for solving the lagrangean relaxed master problem

Subgradient optimization is a technique for progressively updating the lagrange multipliers (λ, μ; π) in a systematic way in an attempt to maximize the value of the lagrangean dual problem (5.18). This procedure is one of the more successful techniques for obtaining good lagrange multiplier values, as detailed in the paper by Held, Wolfe, and Crowder [1974]. The subgradient procedure works in the following way :

(1) Determine an initial value for Z_{UB} - the upper bound on the master. This can be done using any heuristic for the generalized assignment problem (e.g. Martello and Toth [1981]). alternatively the solution found so far for PDP in the BDM could be selected as an upper bound.

(2) Choose an initial value for the lagrange multipliers. Two simple methods are available to select (λ^0, μ^0, π^0). The integrality condition in the master Z_M could be relaxed, and it is solved by an LP code. The dual variables corresponding to constraints (5.6), (5.2), and (5.4) offer a candidate for initial values of (λ^0, μ^0, π^0).

Another method is based on the fact that (5.18) is a relaxed version of the master

Z_M and that at each Benders iteration this master is modified by addition of a single constraint. Hence in Benders iteration h, the last multipliers (λ, μ, π) obtained from subgradient optimization could be used in iteration $h + 1$ to form the initial vector of multipliers of the following form :

$$(\lambda^0, \mu^0, \pi^0) = \begin{bmatrix} \lambda \\ \mu \\ \pi \end{bmatrix} \tag{5.33}$$

This last method is prefered because it was found that the (λ^0, μ^0, π^0) obtained were very good at locating a feasible solution to the master, and secondly because it required no extra computational effort.

(3) For the current set of lagrange multipliers (λ, μ, π) solve two M 0-1 knapsack problems, $L^{y}_{\lambda, \mu}$ and $L^{z}_{\lambda, \pi}$, using Martello and Toth's algorithm [1977].

The solution to these problems are (y, z) and their objective function value is $V(L_{\lambda, \mu, \pi})$.

(4) Determine the cost of the feasible solution to Z_M associated with (y, z). If this is better than Z_{UB} then update Z_{UB} accordingly. Similarly set

$$Z_L^* = \text{Max}\,(Z_L^*, Z_D^*) \tag{5.34}$$

(5) If any of the termination criteria is fulfilled, then stop.

(6) Define the subgradient vector

$$v_i^p = \sum_{j=1}^{M} y_{ij}^* - 1 \qquad i = 1, \ldots, N$$

$$v_l^p = \sum_{k=1}^{M} z_{kl}^* - 1 \qquad l = 1, \ldots, N \tag{5.35}$$

$$v_h^p = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{l=1}^{N} \alpha_{ijl}^h q_{il} y_{ij}^* + \sum_{i=1}^{N}\sum_{k=1}^{M}\sum_{l=1}^{N} \beta_{ikl}^h q_{il} z_{kl}^* - M_0 \qquad h = 1, \ldots, H$$

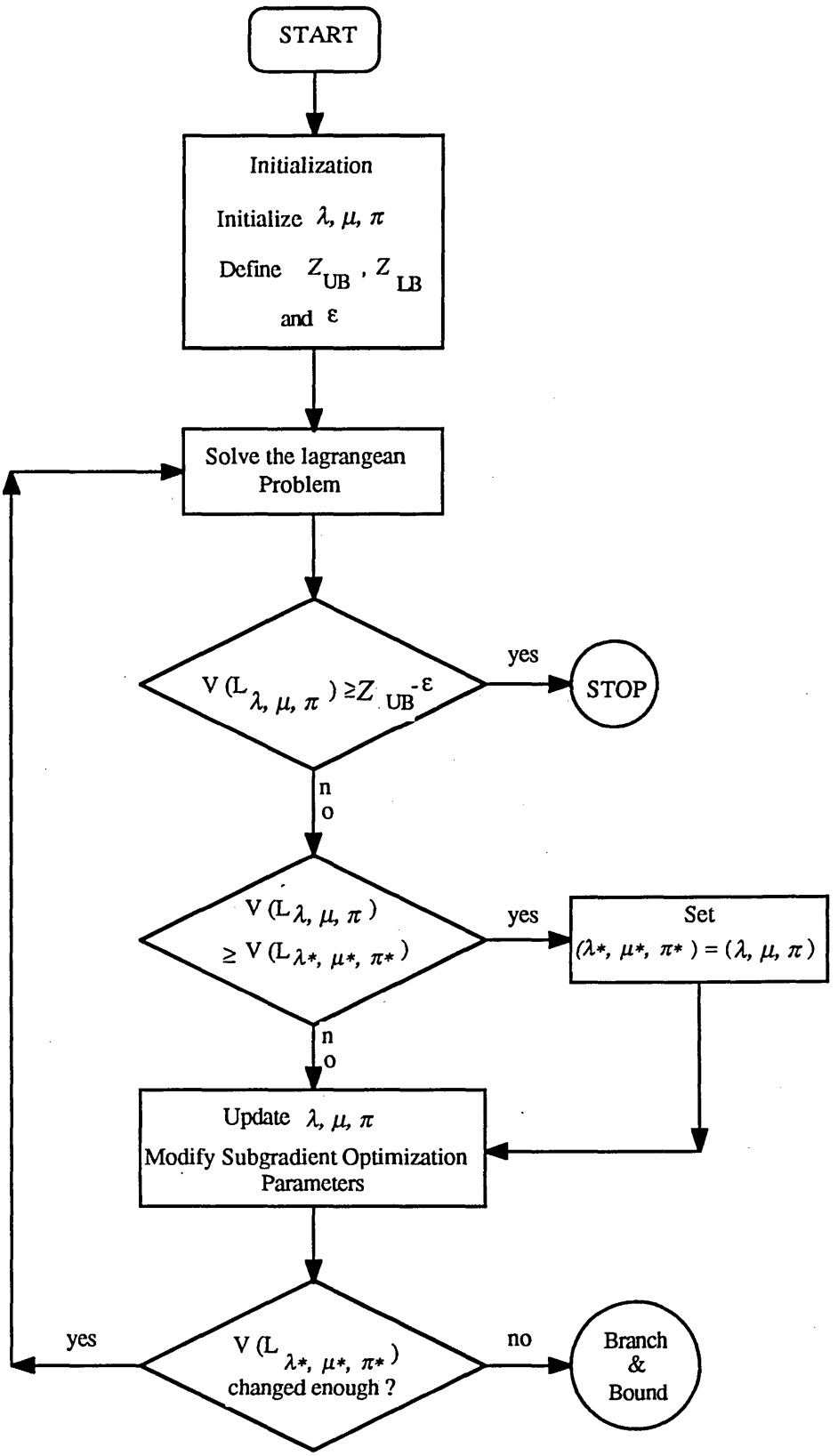


Fig. 5.1 Flow chart of bounding the master problem

Then $\upsilon = [\ \upsilon_i^p,\ \upsilon_l^p,\ \upsilon_h^p\ ]$ is the subgradient of the lagrangean dual at (λ, μ, π).

(7) Stop, if $\upsilon_i^p = 0, \forall i$, $\upsilon_l^p = 0, \forall l$, and $\upsilon_h^p = 0, \forall h$ when an optimal solution (y^*, z^*) has been obtained.

(8) The new set of multipliers are determined as:

$$\mu_i^{p+1} = \mu_i^p + t_p\, \upsilon_i^p$$

$$\pi_l^{p+1} = \pi_l^p + t_p\, \upsilon_l^p \qquad (5.36)$$

$$\lambda_h^{p+1} = \lambda_h^p + t_p\, \upsilon_h^p$$

$$\text{where} \quad t_p = \delta^p \frac{z_{UB} - z_D^*}{\| \upsilon^p \|^2} \qquad (5.37)$$

δ^p are appropriate scalars with $0 < \delta^p \leq 2$. δ^p were chosen with starting value of 2, and this value is halved every 10 successive iterations. $\| \upsilon^p \|$ denotes the Euclidean norm of the subgradient vector. The following norm was used.

$$\left(\sum_{i=1}^{N} \gamma_i^p + \sum_{l=1}^{N} \gamma_l^p + \sum_{h=1}^{H} \gamma_h^p \right)^{1/2}$$

(9) Go to (3) and resolve the lagrangean dual program with the new set of multipliers, unless some termination condition is satisfied in which case stop.

The stopping criteria are derived with reference not only to (5.18) but to the BDM as a whole.

(i) When the subgradient optimization method reaches a solution $V(L_{\lambda,\mu,\pi}) \geq Z - \varepsilon_1$, where Z is the cost of the best solution found so far in the BDM, then the method

has established that the solution which yielded z is ε_1-optimal and BDM is stopped.

(ii) When for a given vector (λ, μ, π) a solution (y, z) is reached , and the cost of this solution when evaluated in Z_M is less than $Z - \varepsilon_2$, then the subgradient optimization method is stopped and the transportation sub-problem is solved using these (y, z). This condition corresponds to the situation of step 1, section 3.4.2, when a feasible solution has been found.

(iii) If the number of subgradient iterations exceeded twenty, which implied slow convergence or the existence of a duality gap, then the BDM is stopped.

5.8 The branch and bound procedure

In the event that the subgradient procedure does not optimally solve the master, a usual depth-first tree search will be utilized (Figure 5.2). The procedure has the following features. At any level of the branch and bound tree, a separation variable is selected. A set of sub-problems are generated; and for each sub-problem, twenty subgradient iterations are performed, seeking to improve the value of the lower bound $V(L_{\lambda, \mu, \pi})$ for the restricted master. The sequence of iterations is halted if one of the following situations occurs:

(i) fathoming is achieved,

(ii) there is little improvement in $V(L_{\lambda, \mu, \pi})$,

(iii) the number of iterations exceeds a preset limit.

The next node selected for branching is the one with the lowest lower bound value among the sub-problems. Whenever a node is fathomed, backtracking is performed (in a depth-first procedure) to the lowest level node along the path to the fathomed node. The best feasible solution found at the end of the branch and bound procedure is the optimal solution to the master.

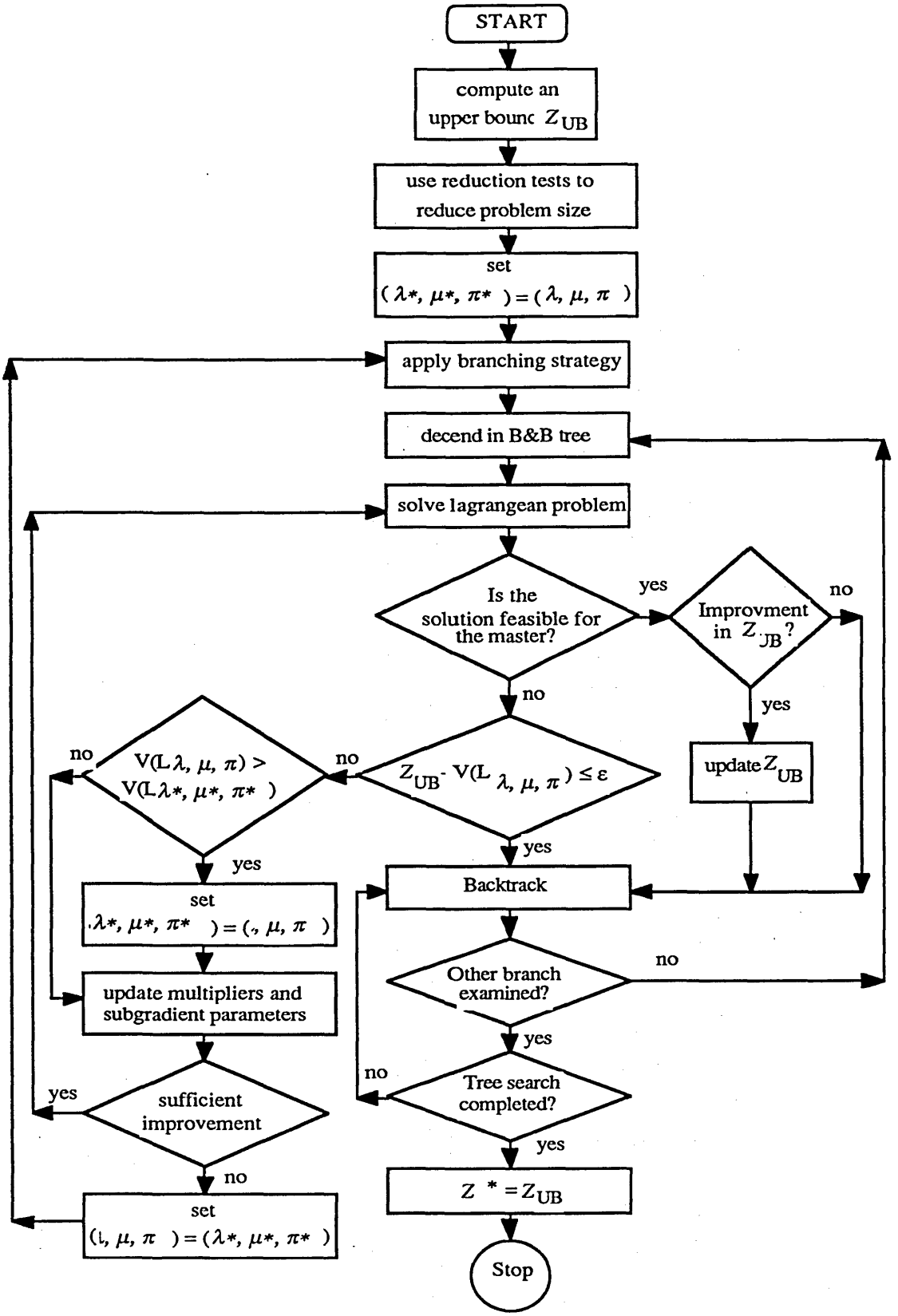


Fig. 5.2 Flow Chart of the branch and bound procedure

5.9 Computational Results

In this section computational results are given for 80 test problems, all problems being randomly generated using the procedure described in section 3.6. These 80 problems comprise two sets of 40 problems namely Data Sets A and B, which differ in their capacity constraints as also defined in section 3.6. The problems solved in this section are however substantially larger than those solved in the previous chapters, the largest having 250 zero-one variables and 15,625 other variables, compared to 80 zero-one variables and 1,600 other variables in chapter 3.

The algorithm described in this chapter was implemented in Fortran IV and run on a CDC 7600 using FTN5 compiler. The master problems were solved using two different methods, Linear programming based on a branch and bound algorithm and lagrangean relaxation (LR). As in Chapter 3, Marsten's XMP code was used to solve the LP problems, and the generalized assignment problems (GAP) were solved using Martello and Toth's code.

Table 5.1 gives the computational results for 80 test problems of which 78 were solved to optimality (allowing for the usual convergence tolerance) using LP-bounds obtained from the Linear relaxation of the Master problem. (The smallest problems solved are almost equal to the largest problems solved in Chapter 3.)

Two problems of type B were terminated, one for exceeding the iteration limit (of 30 iterations), the other for exceeding the CPU time limit (of 300 seconds).

The average execution times for problems of Data Set B were at least 7% greater than for Set A, and at worst 61% greater (for the 20x5x5x20 problems). This is due to the tighter capacity constraints used for problems of Data Set B. By contrast, the average number of iterations for problems in Set B is little more than the average for Set A. This is because each iteration in Set B takes longer due to the tighter constraints.

It is notable that although the same relaxation method is used for the master problem as in the results in Table 3.3, the average number of iterations and average execution

Table 5.1
Representative Runs Using LP Bounds in the Master.

Data Set	Problem Size				No. of 0-1 Variables	No. of Other Variables	No. of problem attempted	Major Iterations			Total execution time (Sec.)		
	i	*j*	*k*	*l*				Ave.	Min.	Max.	Ave.	Min.	Max.
A	10	3	3	10	60	900	10	12	8	13	57.645	35.595	69.346
A	10	5	5	10	100	2500	10	14	9	16	60.277	40.080	76.044
A	20	3	3	20	120	3600	5	14	12	16	65.120	52.028	83.920
A	20	5	5	20	200	10000	5	19	16	22	92.067	78.528	115.340
A	25	3	3	25	150	5625	5	13	10	15	68.470	54.129	88.095
A	25	5	5	25	250	15625	5	26	24	30	190.202	167.449	254.260
B	10	3	3	10	60	900	10	12	9	13	69.255	48.726	89.323
B	10	5	5	10	100	2500	10	15	14	17	64.769	43.278	86.866
B	20	3	3	20	120	3600	5	14	12	18	71.678	55.763	97.763
B	20	5	5	20	200	10000	5	20	18	21	148.243	112.361	151.341
B	25	3	3	25	150	5625	5	15	13	17	86.215	72.892	98.102
B	25	5	5	25	250	15625	5	26[a]	22	>30[b]	272.824[a]	241.028	300.00[c]

a Average given for 3 problems only.
b One problem exceeded the iteration limit. Thus, program terminated.
c One problem exceeded the time limit. Thus, program terminated.

times for the 10x3x3x10 problems are some 33% lower in Table 5.1 due to the improved formulation of the problems. (This would be true for problems of all sizes.)

Table 5.2 shows the results for the identical sets of problems as Table 5.1, using lagrangean relaxation to solve the master problem. All 80 problems were solved to optimality, the maximum number of iterations being 22. The average execution time for problems of Set B was at least 11% greater than for Set A, and at worst 52% greater (for the 25x5x5x25 problems). These figures are comparable to the corresponding figures of Table 5.1.

As in Table 5.1, the average number of iterations for problems of Set B is little more than for Set A.

The average solution times for problems in both sets are more dependent on the number of iterations than on the problem size (as measured by the number of variables). So, for instance, the average time of 63.4 seconds for 25x3x3x25 problems in Set A corresponds to 10 iterations rather than the number of variables.

The comparison of Tables 5.1 and 5.2 illustrates several features of the relative performance of the LR and LP methods. Firstly, the average number of iterations required to solve the problems is less for the LR method for every problem size; (the maximum number is less than or equal to the number required by the LP method in every case.) At worst the number of iterations required by LR is 92% of the LP case (10x3x3x10 problems in Set B), and at best 65% (25x5x5x25 problems in Set B). This is due to the fact that LR produces tighter bounds at each iteration, and hence fewer iterations are required.

The average time taken to solve the problems is also less for LR than for LP in most cases (8 out of 12), but the improvement is proportionally less than the reduction in number of iterations, ranging from 99% to 76%. As pointed out before, this is due to the fact that each iteration of the LR method takes longer.

Table 5.2
Representative Runs Using Lagrangean Relaxation Bounds in the Master.

Data Set	Problem Size				No. of 0-1 Variables	No. of Other Variables	No. of problem attempted	Major Iterations			Total execution time (Sec.)		
	i	*j*	*k*	*l*				Ave.	Min.	Max.	Ave.	Min.	Max.
A	10	3	3	10	60	900	10	9	7	12	54.097	43.210	72.011
A	10	5	5	10	100	2500	10	11	8	13	66.093	49.308	84.560
A	20	3	3	20	120	3600	5	12	10	13	64.144	57.158	91.346
A	20	5	5	20	200	10000	5	15	11	17	82.185	65.282	119.615
A	25	3	3	25	150	5625	5	10	10	11	63.408	59.834	67.631
A	25	5	5	25	250	15625	5	19	16	22	144.168	112.764	185.014
B	10	3	3	10	60	900	10	11	9	12	68.332	52.009	79.097
B	10	5	5	10	100	2500	10	12	10	15	73.102	65.631	95.436
B	20	3	3	20	120	3600	5	12	9	16	76.647	64.340	104.024
B	20	5	5	20	200	10000	5	15	12	16	105.760	81.742	122.282
B	25	3	3	25	150	5625	5	11	10	14	87.153	72.406	95.121
B	25	5	5	25	250	15625	5	17	13	20	218.94	144.067	288.945

Perhaps most important, the superiority of LR over LP, in both the number of iterations and the execution time, increases as the size of the problem increases. Thus, for example, in Set B, for problems of 120 zero-one variables or less, the average number of iterations required for the LR method is 86% of those required for the LP method, while for problems with 150 or more zero-one variables, the average number of iterations is 71% of the number required by the LP method. Likewise the corresponding average execution times are 106% and 84%. This is consistent also with the fact that LR method solved even the largest problems to optimality.

It also confirms the fact that LR is more likely to perform better as the problem sizes increase.

Figures 5.3 and 5.4 illustrate the convergence of the BDM applying two different relaxation methods to the master problem. The problem solved in both figures is a 20x5x5x20 problem from Data Set B. It is clear that using the lagrangean relaxation method, the convergence between the upper and lower bounds is much quicker. This is due to the fact that solution of the master problem using LR produces tighter Benders cuts, and hence rapid convergence.

It is notable that in both cases the upper bound converges towards the solution quickly: In Figure 5.3, it is within 5% of the optimum after 5 iterations, and in Figure 5.4, within 3% after 5 iterations. It is common that using a powerful heuristic will permit reaching within a few percent of the optimum quite quickly.

By contrast, the corresponding lower bounds after 5 iterations are 17% and 5% below the optimal.

Tables 5.3 and 5.4 give detailed run times for a 25x5x5x25 problem from both Sets A and B using the two different relaxation methods for the master problem.

As seen in Tables 5.1 and 5.2 the problem from Set B requires 15% fewer iterations, but takes 58% longer time to solve, due to tighter constraints in the Set B

Fig. 5.3 Convergence of the BDM (LP Bounds)

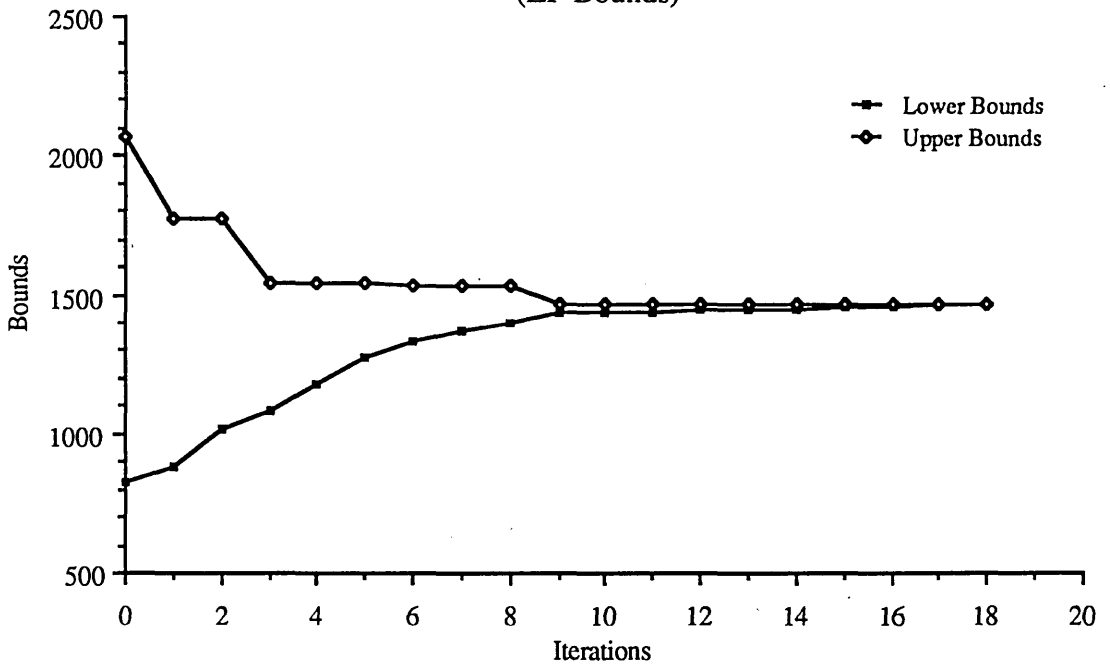
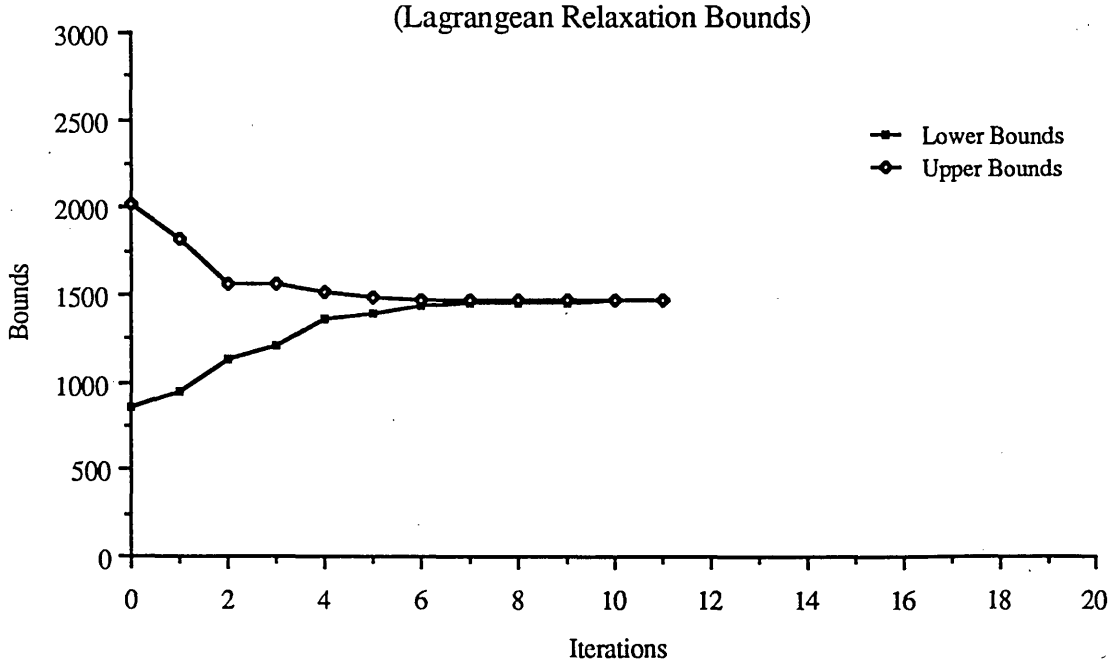


Fig. 5.4 Convergence of the BDM (Lagrangean Relaxation Bounds)



problems. (i.e. reducing the region of feasible solutions).

For both problem types, the great majority (> 80%) of the solution time is spent solving the master problem, and as the number of iterations increases, and so the master problem becomes more complex, the time taken to solve the master problem at each iteration increases.

Table 5.3
Detailed run time for a 25x5x5x25 problem
(using LP bounds in the master)

	DATA SET A CPU Time*			DATA SET B CPU Time		
Major Iteration	Master	Sub-problem	Total	Master	Sub-problem	Total
1	3.46	1.62	5.08	6.90	1.19	8.09
2	3.75	0.90	4.65	6.86	1.25	8.11
3	3.88	1.11	4.99	6.92	1.24	8.16
4	4.13	1.54	5.67	6.98	1.30	8.28
5	4.20	1.13	5.33	7.13	1.34	8.47
6	4.29	1.21	5.50	7.14	1.28	8.42
7	4.44	1.45	5.89	7.35	1.22	8.57
8	4.46	1.58	6.04	8.45	1.26	9.71
9	4.57	1.22	5.79	8.67	1.30	9.97
10	5.07	1.36	6.43	8.73	1.33	10.06
11	5.16	1.02	6.18	8.82	1.47	10.29
12	5.20	1.19	6.39	9.23	1.41	10.64
13	5.36	1.17	6.53	9.55	1.45	11.00
14	5.74	0.96	6.70	9.89	1.48	11.37
15	6.07	1.40	7.47	9.93	1.42	11.35
16	6.25	1.26	7.51	10.21	1.40	11.61
17	6.43	1.33	7.76	10.43	1.43	11.86
18	6.78	1.23	8.01	11.62	1.46	13.08
19	6.96	1.37	8.33	12.24	1.49	13.73
20	7.14	1.18	8.32	13.64	1.57	15.21
21	7.27	1.05	8.32	14.76	1.51	16.37
22	7.71	1.17	8.88	15.17	1.61	16.68
23	8.50	1.53	10.03			
24	8.66	1.72	10.38			
25	8.83	1.33	10.16			
26	9.11	1.48	10.59			
Total Time	153.42	33.51	186.93	210.62	30.41	241.03

*Computations (in Seconds) on a CDC 7600

However, the solution of the sub-problem in either case remains almost constant for each iteration.

These comments apply also to Table 5.4, with one important difference. The LR of the master problem takes fewer iterations to converge to the optimal solution. However, at each iteration, a greater percentage of time is spent solving the master problem. For example, at iteration 17 89% of time is spent on the master problem using LR for data Set A, whereas only 82% is spent using LP relaxation. Likewise for iteration 13 (in Data Set B) LR spends 92% of the time for solving the master problem, whereas LP spends only 87%.

Table 5.4
Detailed run time for a 25x5x5x25 problem
(using Lagrangean Relaxation bounds in the master)

	DATA SET A CPU Time*			DATA SET B CPU Time		
Major Iteration	Master	Sub-problem	Total	Master	Sub-problem	Total
1	4.64	1.12	5.76	8.95	1.06	10.01
2	4.31	0.74	5.05	8.18	0.87	9.05
3	4.47	0.97	5.44	8.62	1.11	9.73
4	5.14	1.08	6.22	9.46	1.08	10.54
5	5.29	1.02	6.31	9.64	0.94	10.58
6	5.35	1.07	6.42	9.71	1.12	10.83
7	5.43	0.81	6.24	9.89	0.89	10.78
8	5.47	1.02	6.49	10.03	1.11	11.14
9	5.69	0.84	6.53	10.12	0.83	10.05
10	5.98	0.93	6.91	10.30	0.99	11.29
11	6.07	1.02	7.09	11.53	0.92	12.54
12	7.05	0.99	8.04	12.05	0.90	12.95
13	7.26	1.04	8.30	12.63	1.14	13.77
14	7.34	0.86	8.20			
15	8.28	0.82	9.10			
16	8.68	0.98	9.66			
17	8.74	1.06	9.80			
Total Time	105.19	16.37	121.56	131.11	12.96	144.07

*Computations (in Seconds) on a CDC 7600

Finally, Table 5.5 shows that the starting bounds at the root node are significantly grater for LR than for LP for all iterations. As the number of Benders cuts increases, the quality of the relaxed master problem declines faster for LP bounds, and therefore requires more iterations. It is also notable that the minimum, maximum, and average bounds all follow this pattern.

Table 5.5
Bounds at root node (%) for a set of 20x5x5x20 problems

	DATA SET B							
Major Iteration	No. of Problems	LP bound at root node (%)			No. of Problems	LR bound at root node (%)		
		Ave.	Min.	Max.		Ave.	Min.	Max.
1	5	94.29	93.81	94.86	5	97.76	97.56	97.95
2	5	93.41	93.20	93.61	5	97.33	96.78	97.63
3	5	92.91	92.51	93.17	5	97.06	96.50	97.54
4	5	92.28	91.55	92.77	5	96.80	96.45	97.30
5	5	91.75	90.85	92.61	5	96.09	95.67	96.54
6	5	91.60	90.58	92.44	5	95.60	95.29	95.85
7	5	91.32	90.40	92.29	5	95.27	95.06	95.48
8	5	90.94	90.27	92.06	5	94.97	94.50	95.34
9	5	90.60	90.07	91.89	5	94.74	94.40	95.08
10	5	90.20	89.70	91.35	5	94.18	93.53	94.70
11	5	89.77	89.19	90.38	5	93.58	93.09	93.80
12	5	89.31	88.24	90.20	5	93.13	92.69	93.66
13	5	89.07	87.84	89.77	4	92.81	92.42	93.32
14	5	88.55	87.49	89.33	4	92.41	92.00	92.88
15	5	88.03	86.82	88.59	4	92.05	91.66	92.53
16	5	87.57	86.81	87.93	2	91.78	91.17	92.18
17	5	87.19	86.73	87.50				
18	5	86.67	86.32	87.29				
19	4	86.25	85.70	86.75				
20	3	85.95	85.52	86.20				
21	2	85.77	85.37	86.10				

5.10 Conclusion

Following the theoretical analysis of Chapter 4, this Chapter has described the implementation of a disaggregated formulation of the PDP problem. The relative performance of two different algorithms, using different solution methods, namely LR

and LP has been compared. Sets of problems were generated of two different types following the method described in chapter 3. However, whereas in Chapter 3 the largest problem had 80 0-1 variables, in this Chapter problems with as many as 250 0-1 variables are solved. The results obtained show that the process of disaggregation was effective in the case of these larger problems. In addition the stronger formulation developed in Chapter 4 permits the algorithm to produce sharper Benders cuts in practical problems, hence reducing the number of iterations required to solve the problems.

The results also demonstrate clearly the superiority of LR over LP in producing better bounds for the master problem thereby reducing the number of iterations required to solve the problems to optimality, and reducing solution time (though the improvement in the latter is proportionately less due to the longer time taken for each iteration).

CHAPTER 6

Conclusion

The 3-stage parcel distribution problem is an NP-hard problem, and is significantly more complex than multi-commodity capacitated plant location problems. Hitherto only 2-stage problems have been solved, and consequently the PDP problem is an interesting combinatorial optimisation problem.

In this thesis I have formulated the PDP problem as a mixed integer programming problem, and initially solved it with Lagrangian Relaxation (LR), using sensitivity analysis in order to improve the lagrangean bounds. The results of this procedure are shown in chapter 2.

From the structure of the problem a new formulation was derived which lent itself to decomposition by Bender's method, a technique that has been used quite successfully for solving the multi-commodity transportation problem. In the rest of my thesis I have shown that the combination of Bender's decomposition and Lagrangian Relaxation is capable of solving PDP problems of reasonable (25x5x5x25) size, and

therefore appears promising for further development.

It is notable that the solution of the Master Problem absorbed most of the computation time. Because of this I developed the method of disaggregating the problem, thereby producing better Benders cuts, and enabling me to reduce the solution time significantly. An alternative procedure which would offer further improvements, would be to pre-process the Master Problem so as to produce pareto optimal cuts.

A number of other directions for further work are possible. For example, further constraints could be imposed so that depots would act as collection and delivery points. This would have the effect of enabling larger problems to be solved.

Another approach would be to alter the practical details of the formulation in order to extend the range of application of the problem - such as siting of relay systems and selection of the optimal routes through high-speed data transmission and computer networks.

From this it can be seen that further refinements of my work can substantially enhance both the performance of the algorithm, and the range of practical applications.

REFERENCES

Aho, A.V., J.E. Hopcroft, and J.D. Ullman [1974], *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, Massachusetts.

Ahrens, J.H., and G. Finke [1975], "Merging and Sorting Applied to the Zero-One Knapsack Problem", *Opns. Res.*, **23**, 1099-1109.

Aking, U., and B.M. Khumawala [1977], "An Efficient Branch and Bound Algorithm for the Capacitated Warehouse Location Problem", *Mgmt. Sci.*, **23**, 585-594.

Balas, E., and C. Bergthaller [1977], "Benders's Method Revisited", *Mgmt. Sci. Res. Report, 401, Carnegie-mellon University, Pittsburgh, USA.*

Balinski, M.L. [1965], "Integer Programming: Methods, Uses, Computation", *Mgmt. Sci.*, **12**, 253-313.

Balinski, M.L., and K. Spielberg [1969], "Methods for integer programming; algebraic, combinatorial, and enumerative, in: J.S. Aronofsky, Ed. *Progress in Opns. Res. iii*, 195-292, Wiley, New York.

Balinski, M.L., and P. Wolfe [1963], "On Benders Decomposition and a Plant Location Problem", *Working Paper ARO-27, Mathematica.*

Barr, R.S., and G.T. Ross [1975], "A Linked List Data Structure for a Binary Knapsack Algorithm", *Research Report CCS 232, Center for Cybernetic Studies,*

University of Texas.

Bartakke, M.N., J.V. Bloomquist, J.K. Korah, and J.P. Popino [1971], "Optimization of a Multi-National Physical Distribution System", *Sperry Rand Corporation, Blue Bell, Pa., presented at the 40th National ORSA Meeting, Anaheim, California.*

Bazaraa, M.S., and C.M. Shetty [1979], *Nonlinear Programming: Theory and Applications*, Wiley, New York.

Beale, E.M.L., and J.A. Tomlin [1972], "An Integer Programming Approach to a Class of Combinatorial Problems", *Math. Program.*, **3**, 339-344.

Benders, J.F. [1962], "Partitioning Procedures for Solving Mixed-Variables Programming Problems", *Numerische Mathematick*, 238-252.

Benders, J.F., and J.A.E.E. Van Nunen [1983], "A Property of Assignment Type Mixed Integer Linear Programming Problems", *Opns. Res. Letters*, **2**, 47-52.

Bilde, O., and Krarup, J. [1967], "Bestemmelse of Optimal Beliggenhed af Produktionssteder", *Research Report, IMSOR, Technical University of Denmark.*

Bilde, O., and Krarup, J. [1977], "Sharp Lower Bounds for the Simple Location Problem", *Annals of Discrete Mathematics*, **1**, 79-97.

Bloom, J.A., M. Caramanis, and L. Charny [1984], "Long-Range Generation Planning Using Generalized Benders' Decomposition : Implementation and

Experience", *Opns. Res.*, **32,** 290-313.

Boffey, T.B. [1978], "On finding P-Medians", *Research Report University of Liverpool.*

Brooks, L., and A. Geoffrion [1966], "Finding Everett's Lagrange Multipliers by Linear Programming", *Opns. Res.*, **14,** 1149-1153.

Burkard, R.E., H.W. Hamacher, and J. Tind [1985], "On General Decomposition Schemes in Mathematical Programming", *Math. Program. Study*, **24,** 238-252.

Carrareoi, P., and G. Gallo [1982], "Optimal Location of Files and Programs in Computer Networks", *Math. Program. Study*, **20,** 39-53.

Christofides, N. [1975], *Graph Theory - An Algorithmic Approach*, Academic Press, London.

Christofides, N., and J.E. Beasley [1982], "A Tree Search Algorithm for the P-Median Problem", *Eur. J. Opns. Res.*, **10,** 196-204.

Christofides, N., and J.E. Beasley [1983], "Extensions to a Lagrangean Relaxation Approach for the Capacitated Warehouse Location Problem", *Eur. J. Opns. Res.*, **12,** 19-28.

Clasen, R.J. [1984], "The solution of the Chemical Equilibrium Programming Problem With Generalized Benders Decomposition", *Opns. Res.*, **32,** 70-79.

Cook, S.A. [1971], "The Complexity of Theorem-Proving Procedures", *3rd Annual ACM Symp. on the Theory of Computing*, 151-158.

Cornuéjols, G. [1978], "Analysis of Algorithms for a Class of Location Problems", *Technical Report No. 382, SORIE, Cornell University.*

Cornuéjols, G., M.L. Fisher, and G.L. Nemhauser [1977], "Location of Bank Accounts to Optimize Float: An Analytic Study of Exact And Approximate Algorithms", *Mgmt. Sci.*, **23,** 789-810.

Côté, G., And M.A. Laughton [1982], "Stochastic Production Costing in Generation Planing : A Large-Scale Mixed Integer Model", *Math. Progam. Study*, **20,** 54-71.

Côté, G., and M.A. Laughton [1984], "Large-Scale Mixed integer Programming : Benders-Type Heuristics", *Eur. J. Opns. Res.*, **16,** 327-333.

Crowder, H.P. [1976], "Computational Improvement for Subgradient Optimization", *Symp. Math.*, **19,** 357-372.

Crowder, H., and M.W. Padberg [1980], "Solving Large-Scale Symmetric Traveling Salesman Problem to Optimality", *Mgmt. Sci.*, **26,** 495-509.

Crowder, H., L. Johnson, and M. Padberg [1983], "Solving Large-Scale Zero-One Linear Programming Problems", *Opns. Res.*, **31,** 803-834.

Cuninghame-Greene, R.A., and G. Harries [1988], "Nearest-Neighbour Rules for Emergency Services", *Zeitschrift Fur Opns. Res.* , **32,** 299-306.

Davis, P.S., and T.L. Ray [1969], "A Branch and Bound Algorithm for Capacitated Facilities Location Problem", *Naval Res. Logist. Quart.*, **16,** 331-343.

Drucker, P. [1962], "The Economy's dark Continent", *Fortune*, **72,** 103-104.

Duffy, H., Financial Times [1982], "Distribution Management", 1 November.

Efroymson, M.A., and T.L. Ray [1966], "A Branch and Bound Algorithm for Plant Location", *Opns. Res.*, **14,** 361-368.

Eilon, S., C.D.T. Watson-Gandy, and N. Christofides [1971], *Distribution Management: Mathematical Modelling and Practical Analysis*, Griffin, London.

Ellwein, L.B., and P. Gray [1971], "Solving Fixed Charge Location-Allocation Problems with Capacity and Configuration Constraints", *AIIE Transactions, III,* **4,** 290-298.

Elshafei, A.N., and K.B. Haley [1974], "Facilities Location: Some Formulations, Methods of Solution, Applications and Computational Experience", *OR Report no. 90, University of Birmingham.*

Elson, D.G. [1972], "Site Location via Mixed-Integer Programming", *Opns. Res. Quart.*, **23,** 31-43.

Erlenkotter, D. [1978], "A Dual-Based Procedure for Uncapacitated Facility Location", *Opns. Res.*, **26,** 992-1009.

Everett, H., III [1963], "Generalized Lagrange Multiplier Method for solving Problems of Optimum Allocation of Resources", *Opns. Res.*, **11,** 399-417.

Fayard, D., and G. Plateau [1975], "Resolution of the 0-1 Knapsack Problem: Comparison of Methods", *Math. Program.*, **8,** 272-307.

Federgruen, A., and P. Zipkin [1984], "A Combined Vehicle Routing and Inventory Allocation Problem", *Opns. Res.*, **32,** 1019-1037.

Fisher, M.L. [1981], "The Lagrangian Relaxation Method for Solving Integer Programming Problems", *Mgmt. Sci.*, **27,** 1-18.

Fisher, M.L., and R. Jaikumar [1978], "A Decomposition Algorithm for Large-Scale Vehicle Routing", *Working Paper University of Pennsylvania.*

Fisher, M.L., and R. Jaikumar [1981], " A Generalized Assignment Heuristic for Vehicle Routing", *Networks*, **11,** 181-124

Fisher, M.L., R. Jaikumar, and L.N. Van Wassenhove, [1986], "A Multiplier Adjustment Method for the Generalized Assignment Problem", *Mgmt. Sci.*, **32,** 1095-1103.

Fisher, M.L., and D.S. Hochbaum [1980], "Probabilistic Analysis of the Plannar K-Median Problem", *Math. of Opns. Res.*, **5,** 27-34.

Florian, M., G. Bushell, J. Ferland, G. Guérin, and L. Nastansky [1976], "The Engine Scheduling Problem in a Railway Network", *J. INFOR*, **14,** 121-138.

França, P.M., and H.P.L. Luna [1982], "Solving Stochastic Transportation-Location Problems by Generalized Benders Decomposition", *Tans. Sci*, **16**, 113-126.

Francis, R.L., and J.M. Goldstein [1974], "Location Theory: A Selective Bibliography", *Opns. Res.*, **22**, 400-410.

Frieze, A.M. [1976], "Shortest Path Algorithms for Knapsack Type Problems", *Math. Program.*, **11**, 150-157.

Fulkerson, D.R. [1961], "An Out-of-Kilter Method for Minimal Cost Flow Problems", *J. SIAM*, **9**, 19-27.

Galvao, R.D. [1980], "A Dual-Bounded Algorithm for the P-Median Problem", *Opns. Res.*, **28**, 1112-1121.

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

Garfinkel, R.S., A.W. Neebe, and M.R. Rao [1974], "An Algorithm for the M-Median Plant Location Problem", *Trans.Sci.*, 8, 217-236.

Gavish, B. [1982], "Topological Design of Centralized Computer Networks - Formulations and Algorithms", *Networks*, **12**, 355-377.

Gavish, B., and H. Pirkul [1985], "Efficient Algorithms for Solving Multi-constraint Zero-One Knapsack Problems to Optimality", *Math. Program.*, **31**, 78-105.

Geoffrion, A.M. [1972], "Generalized Benders Decomposition", *J. Optimization Theory and Applications*, **10.**

Geoffrion, A.M. [1974], "Lagrangean Relaxation for Integer Programming", *Math. Program. Study*, **2,** 82-114.

Geoffrion, A.M. [1976], "Better Distribution Planning with Computer Models", *Harvard Business Review*,

Geoffrion, A.M., and G.W. Graves [1974], "Multi-commodity Distribution System Design by Benders Decomposition", *Mgmt. Sci.*, **20,** 822-844.

Geoffrion, A.M., G.W. Graves, and S. Lee [1978], "Strategic Distribution System Planning: A Status Report", Chapter 7 in A. Hax (Ed.), *Studies in Operations Management*, North-Holland/American Elsevier, Amsterdam/New York.

Geoffrion, A.M., and R. Marsten [1972], "Integer Programming Algorithms: A Framework and State-of-the-Art Survey", *Mgmt. Sci.*, **18,** 465-491.

Geoffrion, A.M., and R. McBride [1978], "Lagrangean Relaxation Applied to Capacitated Facility Problems", *AIIE Transact.*, **10,** 40-47.

Glover, F. [1968], "Surrogate Constraints", *Opns. Res.*, **16,** 741-749.

Greenberg, H., and R.L. Hegerich [1970], "A Branch Search Algorithm for the Knapsack Problem", *Magmt. Sci.*, **16,** 327-332.

Guignard, M., and K. Spielberg [1977], "Algorithm for Exploiting the Structure of the Simple Plant Location Problem", *Ann. Discrete Math.*, **1,** 247-271.

Guignard, M., and K. Spielberg [1979], "A Direct Dual Method for the Mixed Plant Location Problem with Some Side Constraints", *Math. Program.*, **17,** 198-228.

Hanjoul, P., and D. Peeters [1985], "A Comparison of Two Dual-Based Procedures for Solving the P-Median Problem", *Eur. J. Opns. Res.*, **20,** 387-396.

Hansen, P. [1972], "Two Algorithms for the Simple Plant Location Problem Using Additive Penalties", Presented at the European Congress of the Econometric Society, Budapest.

Held, M., P. Wolfe, and H.P. Crowder [1974], "Validation of Subgradient Optimization", *Math. Program.*, **6,** 62-88.

Horowitz, E., and S. Sahni [1974], "Computing Partitions with Applications to the Knapsack Problem", *J. of ACM*, **21,** 277-292.

Hu, T.C. [1969], *Integer Programming and Network Flows*, Addison Wesley.

Johnson, D.S. [1987], "The NP-Completeness Column: An ongoing guide", *J. Algorithms,* **8,** 285-303, and other issues.

Jornsten, K.O. [ ], "A Maximum Entropy Combined Distribution and Assignment Model Solved by Benders Decomposition"

Karkazis, J., and T.B. Boffey [1981], "The Multi-Commodity Facilities Location Problem", *J. Opns. Res. Society*, **32,** 803-814.

Karp, R.M. [1972], "Reducibility Among Combinatorial Problems", In: R.E. Miller and J.W. Thatcher (Eds.), *Complexity of Computer Computations,* Plenum Press, New York, 85-103.

Karp, R.M. [1975], "On the Computational Complexity of Combinatorial Problems", *Networks*, **5,** 45-68.

Khumawala, B.M. [1972], "An Efficient Branch-and-Bound Algorithm for the Warehouse Location Problem", *Mgmt. Sci.*, **18,** 718-731.

Kolesar, P.J. [1967], "A Branch and Bound Algorithm for the Knapsack Problem", *Mgmt. Sci.*, **13,** 723-735.

Krarup, J., and P.M. Pruzan [1983], "The Simple Plant Location Problem: Survey and Synthesis", *Eur. J. Opns. Res.*, **12,** 36-81.

Kuehn, A.A., and M.J. Hamburger [1963], "A Heuristic Program for locating Warehouses", *Mgmt. Sci.*, **9,** 643-666.

Lasdon, L.S.[1970], *Optimization Theory for Large System.* Mac Millan Co., New York.

Laundy, R.S. [1985], "A Tree Search Algorithm for the Multi-Commodity Location

Problem", *Eur. J. Opns. Res.*, **20,** 344-351.

Lea, A.C. [1973], "Location-allocation Systems: an annotated bibliography", *Discussion Paper No. 13, Dept. of Geography, University of Toronto.*

Love, R.R. Jr. [1981], "Traffic Scheduling via Benders Decomposition", *Math. Program. Study*, **15,** 102-124.

Lucena Filho, A.P. [1986], "Exact Solution Approaches for the Vehicle Routing Problem", *Ph.D. Thesis, Dept. Mgmt. Sci., Imperial College.*

Maffioli, F. [1978], "The Complexity of Combinatorial Optimization Algorithms and the Challenge of Heuristics", In: N. Christofides et al. (Eds.), *Combinatorial Optimization*, John Wiley & Sons.

Magnanti, T.L., and R.T. Wong [1981], "Accelerating Benders Decomposition : Algorithmic Enhancement and Model Selection Criteria", *Opns. Res.*, **29,** 464-484.

Magnanti, T.L., P. Mireault, and R.T. Wong [1986], "Tailoring Benders Decomposition for Uncapacitated Network Design", *Math. Program. Study*, **26,** 112-154.

Marks, D.H., J.C. Liebman, and M. Bellmore [1970], "Optimal Location of Intermediate Facilities in a Trans-shipment Network", *Paper R-TP3.5 presented at the 37th National ORSA Meeting, Washington, D.C.*

Marsten, R.E. [1972], "An Algorithm for Finding Almost All the Medians of a

Network", *Discussion Paper No. 23, The Center for Mathematical Studies in Econometrics and Management Science, Northwestern University.*

Martello, S., and P. Toth [1977], "An Upper Bound for the Zero-One Knapsack Problem and a Branch and Bound Algorithm", *Eur. J. Opns. Res.*, **1,** 169-175.

Martello, S., and P. Toth [1981], "An Algorithm for the Generalized Assignment Problem," in J.P. Brans (Ed.), *Operational Research 81*, North-Holland, Amsterdam, 589-603.

Mc Daniel, D., and M. Devine [1977], "A modified Benders' Partitioning Algorithm for Mixed Integer Programming", *Mgmt. Sci.*, **24,** 312-319.

Mulvey, J.M., and H.P. Crowder [1979], "Cluster Analysis: An Application of Lagrangean Relaxation", *Mgmt. Sci.*, **25,** 329-340.

Narula, S.C., U.I. Ogbu, and H.M. Samuelson [1977], "An Algorithm for the P-Median Problem", *Opns. Res.*, **25,** 709-713.

Nauss, R.M. [1976], "An Efficient Algorithm for the 0-1 Knapsack Problem", *Mgmt. Sci.*, **23,** 27-31.

Nauss, R.M. [1978], "An Improved Algorithm for the Capacitated Facility Location Problem", *J. Opns. Res.*, **29,** 1195-1202.

Neebe, A.W., and B.M. Khumawala [1981], "An Improved Algorithm for the Multi-Commodity Location Problem", *J. Opns. Res. Society*, **32,** 143-169.

Nemhauser, G.L., and Z. Ullmann [1968], "A Note on the Generalized Lagrange Multiplier Solution to an Integer Programming Problem", *Opns. Res.*, **16,** 450-453.

Nemhauser, G.L., L.A. Wolsey, and M.L. Fisher [1978], "An Analysis of Approximations for Maximizing Submodular Set Functions I", *Math. Program.*, **14,** 265-294.

Noonan, F., and R.J. Giglio [1977], "Planning Electric Power Generation : A Nonlinear Mixed Integer Model Employing Benders Decomposition", *Mgmt. Sci.*, **23,** 946-956.

Parker, D.D. [1962], "Improving Efficiency and Reduced Cost in Marketing", *J. Marketing*, **26,** 15-21.

Peeters, D. [1980], "Contribution Aux Modéles de Localisation de Services Publics", *Ph. D. Thesis, Université Catholique de Louvain.*

Rardin, R.L., and V.E. Unger [1976a], "Surrogate Constraints and Strength of Bounds Derived from 0-1 Benders' Partitioning Procedures", *Opns. Res.*, **24,** 1169-1175.

Rardin, R.L., and V.E. Unger [1976b], "Some Computationally Relevant Group Theoretic Structures of Fixed Charge Problems", *Math. Program.*, **10,** 379-400.

ReVelle, C.S., D. Marks, and J.C. Liebman [1970], "An Analysis of Private and Public Sector Location Models", *Mgmt. Sci.*, **16,** 692-707.

ReVelle, C.S., and R.S. Swain [1970], "Central Facilities Location", *Geographical Anal.*, **2,** 30-42.

Richardson, R. [1976], "An Optimization Approach to Routing Aircraft", *Trans. Sci.*, **10,** 52-71.

Rosing, K.E., E.L. Hillsman, and H. Rosing-Vogelaar [1979], "A Note Comparing Optimal and Heuristic Solutions to the P-Median Problem", *Geographical Analysis*, **11,** 86-93.

Rosing, K.E., and C.S. ReVelle [1978], "A Method for Optimal Solutions to Large-Scale P-Median Problems", *Economic Geography Institute, Erasmus Universiteit, Rotterdam.*

Ross, G.T., and R.M. Soland [1975], "A Branch and Bound Algorithm for the Generalized Assignment Problem", *Math. Program.*, **8,** 92-103.

Ross, G.T., and R.M. Soland [1977], "Modeling Facility Location Problems as Generalized Assignment Problems", *Mgmt.Sci.*, **24,** 345-357.

Rouhani, R., L. Lasdon, W. Lebow, and A.D. Waren [1985], "A Generalized Benders Decomposition Approach to Reactive Source Planning in Power Systems", *Math. Program. Study*, **25,** 62-75.

Sa, G. [1969], "Branch-and-Bound and Approximate Solutions to the Capacitated Plant-Location Problem", *Opns. Res.* **17,** 1005-1016.

Salkin, H.M. [1975], *Integer Programming*, Addison-Wesley.

Schrage, L. [1975], "Implicit Representation of Variable Upper Bounds in Linear Programming", *Math. Program. Study* **4,** 118-132.

Shapiro, G.F. [1968], "Dynamic Programming Algorithms for the Integer Programming Problem I; The Integer Programming Problem Viewed as a Knapsack Type Problem", *Opns. Res.*, **16,** 103-121.

Shapiro, J.F. [1971], "Generalized Lagrange Multipliers in Integer Programming", *Opns. Res.*, **19,** 68-76.

Shapiro, J.F. [1984], "A Note on Node Aggregation and Benders' Decomposition", *Math. Program.*, **29,** 113-119.

Shapiro, J.F., and H.M. Wagner [1966], "A Finite Renewal Algorithm for the Knapsack and Turnpike Models", *Opns. Res.*, **14,** 319-341.

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

Spielbrg, K. [1969], "Algorithms for the Simple Plant-Location Problem with Some Side Conditions", *Opns. Res.*, **17,** 85-111.

Stollsteimer, J.F. [1963], "A Working Model for Plant Numbers and Locations", *J. Farm Econom.*, **45,** 631-645.

Van Roy, T.J. [1983], "Cross Decomposition for Mixed Integer Programming", *Math. Program.*, **25**, 46-63.

Warszawski, A., and S. Peer [1973], "Optimizing the Location of Facilities on a Building Site", *Opns. Res. Quart.*, **24**, 35-44.

Williams, H.P. [1974], "Experiments in the Formulation of Integer Programming Problems", *Math. Program. Study*, **2**, 180-197.

Zionts, S. [1974], *Linear and Integer Programming*, Prentice-Hall, Inc.