Journal of Global Optimization **26:** 43–78, 2003.
© 2003 *Kluwer Academic Publishers. Printed in the Netherlands.*

Scheduling Deliveries in Vehicles with Multiple Compartments

EMMANUEL D. CHAJAKIS[1] and MONIQUE GUIGNARD[2]
[1]*Merrill Lynch & Co., Plainsboro, Plainsboro, NJ 08542, USA;* [2]*Operations and Information Management Department, The Wharton School, University of Pennsylvania, Philadelphia, PA 19104, USA*

Abstract. Vehicles with multiple compartments are used, among others, for distribution to convenience stores. Based on the convenience stores paradigm we propose optimization models for two possible cargo space layouts and explore their characteristics through computational experiments with randomly generated data sets. In a small real data set an optimal solution of one of the models requires fewer vehicles because compartment capacities are utilized more tightly. We develop and test approximation schemes based on Lagrangean Relaxation that generate good feasible solutions in reasonable time. The good quality of the solutions is guaranteed by the gap between their value and the Lagrangean Relaxation bound. These schemes could be valuable for large real applications.

1. Introduction

Distribution with motor vehicles is one of a number of modes of transportation used by firms to deliver goods to their customers. It requires scheduling of some of the firm’s resources, i.e., vehicles and personnel, on a routine basis. The principal considerations when scheduling deliveries of goods are customer satisfaction and costs of distribution. Improving the efficiency of distribution can have a big impact for the firm and the economy as a whole, since physical transportation of goods is currently equivalent to about 10% of the U.S. Gross Domestic Product [13]. (This is only demonstrative of relative magnitude since GDP, a net input-output figure, is not directly comparable to transportation expenditure, a gross amount [14].) Optimization or other methods that have been developed by academic researchers and private practitioners for improving the efficiency of distribution with motor vehicles are described by the term *Vehicle Routing*. Christophides [10], Golden and Assad [31] and more recently Fisher [23] have provided comprehensive surveys on vehicle routing.

Optimization models for vehicle routing must encompass the real-world need for accuracy, implementability and speed of computation, without losing sight of the value of model simplicity and theoretical soundness. Bell et al. [4] describe an on-line system for vehicle scheduling and routing for distribution of industrial gases; the system is based on a mixed integer programming model and Lagrangean Relaxation is used to provide near-optimal delivery schedules.

The level of complexity of vehicle routing problems treated by optimization methods has been ever increasing. Among others, Desrosiers et al. [18], Desrosiers et al. [17], Desrochers et al. [16], Desrochers et al. [15], Kolen et al. [38] have been studying vehicle routing problems where deliveries to customers must be done within specified time windows. Dror et al. [19] and Laporte et al. [40] consider problems in which customer demands and travel/service times, respectively, are treated as stochastic. Laporte et al. [39] study the problem of simultaneously locating a depot, determining fleet size and designing a set of routes through a set of customers whose demands are random.

1.1. VEHICLES WITH MULTIPLE COMPARTMENTS

Vehicles with multiple compartments is a variation of the standard vehicle routing problem that has been mentioned [10, 23] but, to our knowledge, never directly addressed in optimization models in the past. The rationale for using vehicles with multiple compartments is to be able to carry products of different temperature or composition in the same vehicle. There are at least two applications in which vehicles with multiple compartments are used, i.e., transportation of petroleum products and deliveries of food and grocery items to convenience stores. In this study we will concentrate on the latter, while we will only briefly describe the former in the following paragraph.

Marine vessels that transport petroleum products from refineries to customers have multiple compartments that can carry more than five different products together in a single trip. Trucks that carry liquid fuels have divided tanks that can hold fuelds of different types, for example, gasoline of different grades. Since accidental mixing of fuel types can be hazardous, compartments are fixed and very well separated from each other, preferably with double bulkheads. Miller [43] considers scheduling of a tanker fleet that deliver gasoline antiknock compounds to distribution and customer terminals around the world. Tankers have eight to ten compartments and typically carry six or seven products. The problem can be modeled as a space-time network, but cannot be solved by optimization methods, due to the large number of nodes and arcs involved.

The rest of the paper is organized as follows: In Section 2 we describe the operation of distribution to convenience stores. In Section 3 we propose two models for scheduling deliveries to convenience stores and we show how the formulation of the models can be tightened by appending certain logical constraints. A critique of the models, and preliminary experimentation is presented in Section 4. In Section 5 we present Lagrangean Relaxations, a Lagrangean Substitution and a Lagrangean heuristic based on the first model. In Section 6, we present computational experience with them. Conclusions are discussed in Section 7. The procedure for generating random test problems is presented in the Appendix.

2. Distribution to convenience stores

Convenience stores represent the fastest growing segment of the food retail industry.

Oil companies are the most recent entrants in the field, that has traditionally been the domain of nationwide chains like Southland's 7–11 stores and more regional ones, like for example Wawa stores in the Mid-Atlantic states. Oil companies have been gradually transforming their gas stations from strictly fuel dispensing units to small or medium sized supermarkets or even food courts by subletting space to companies like Subway or Burger King. Some try to create a brand image for their stores like Sunoco's A-plus. Others enter the field through alliances, like Citgo's with Southland. The economic motivation behind this is clear: convenience store retailing carries much higher margins than fuel.

Convenience stores are of relatively small size and have extended hours of operation. Their purpose is to offer mainly grocery and food items to consumers outside regular business hours or to motorists on highways and isolated locations. Unlike in a supermarket that typically has large storage spaces in the back, a convenience store carries most of its inventory in the front of the store. These space limitations impose the need for very tight control of inventories (Weinstein [47]) and this is the main reason why convenience store orders must be small, and served by a single distributor that can deliver dry, refrigerated and frozen items together in the same truck. In contrast, a supermarket places large orders, often to more than one distributor that can deliver items at various temperatures separately in different trucks.

Another concern that makes delivery of mixed orders in a single truck to convenience stores preferable is that of practicality. Supermarkets typically have special doors and facilities in the back, so that trucks can be unloading without interfering with consumers or other functions of the business. Most convenience stores do not have doors for unloading in the back, therefore all orders have to be delivered through the front door. Clearly, when only one truck delivers instead of two or three, the inconvenience is minimized.

Vehicles for deliveries to convenience stores have a cooling unit (e.g., Thermo King, Frigidaire) attached to the top front end of the cargo space behind the driver's cabin. In addition to the main back door, vehicles often have one or more side access doors. They may or may not have separate spaces or compartments to hold items at different temperatures. A compartment can often be as simple as a box filled with dry ice. In larger trucks, compartments are formed by separating spaces using bulkheads that can be fixed or movable. Movable bulkheads slide along special rails in the inside walls of the truck to allow the volume capacity of compartments to vary according to the cubic volume of the items they must hold during a trip. The movement of bulkheads is often limited by the position of side doors.

Our experience shows that convenience store distributors either deliver mixed orders in undivided trucks or divide the truck cargo using boxes and bulkheads, fixed or movable, in a variety of ways and in many creative combinations. The four most common layouts seem to be the following:

- No boxes or bulkheads: Air from the cooling unit keeps the front of the cargo

space at a freezing temperature. It dissipates and becomes warmer in the moddle while leaving the back at ambient temperature. Therefore, frozen, refrigerated and dry items are loaded in sequence from front to back.
- Two boxes: Filled with dry ice, they hold frozen and refrigerated items while the cooling unit is switched off. Most common when the average non-dry volume is small.
- Bulkhead and box: A frozen compartment is separated out in front and a small volume of refrigerated items is carried in a box with dry ice. For access to the frozen compartment, a side door is placed as near as possible to the front end of the cargo space to allow maximum flexibility in sizing the compartment.
- Two bulkheads: They divide the cargo space in a frozen, a refrigerated and an ambient temperature compartment. Cool air that enters from the frozen compartment through special vents on the front bulkhead keep the refrigerated compartment at the appropriate temperature. This layout has the disadvantage of requiring two side doors, thus limiting the flexibility in varying compartment sizes.

Belardo et al. [3] describe a decision support system that helps schedulers create routes for trucks that deliver to convenience stores; however this system does not use any optimization model.

3. Models for scheduling deliveries to convenience stores

The basic vehicle routing problems involves two types of decisions: (i) Assigning customers to routes or trips to be served by trucks and (ii) determining the sequence in which customers assigned to the same route will be visited by the truck that serves the route. Due to its complexity, the first approaches for the vehicle routing problem were simple heuristics, the more representative being the ones by Clarke and Wright [12], Christophides and Eilon [11], Gillet and Miller [30] and their variations. Other heuristics find fairly good feasible solutions for the vehicle routing problem by solving to optimality a mathematical program that is a suitable approximation to it. Heuristics of this type are the generalized assignment heuristic by Fisher and Jaikumar [24], the heuristic based on solving a set partitioning problem, proposed by Balinski and Quandt [2] (see also Foster and Ryan [25]), and asymptotically optimal heuristics based on probabilistic analysis results, by Haimovich and Rinnooy Kan [35] and Bramel et al. [6]. Algorithms that find optimal solutions to the vehicle routing problem were proposed by Laporte et al. [41] and Fisher [22], among others.

Our approach is similar to the generalized assignment heuristic for vehicle routing [24]. Our models for scheduling deliveries to convenience stores will be addressing only the first type of decisions involved in the vehicle routing problem, i.e., assigning customers to routes to be served by trucks, and therefore can be the first stage of heuristics for the vehicle routing problem with multiple compartments.

Once an assignment of customers to vehicles is found by solving these models to optimality or near-optimality, the sequencing of customers in a route can be determined by any traveling salesman algorithm.

We propose two integer programming models, described in the following subsections and show how they can be tightened by appending certain logical constraints. The first model assumes that the whole fleet will be using the two boxes layout, while the second assumes that the one bulkhead and one box layout will be used throughout the entire fleet. In a real application, many different layouts could coexist in the same fleet and same-layout subfleets can be identified. Lagrangean schemes presented in this article will be based on the two boxes layout model.

3.1. A 0–1 PROGRAMMING MODEL FOR THE TWO BOXES LAYOUT

The notation defined here will be used in the rest of the paper. Let $\mathscr{P}$ be an optimization problem. Then $OV(\mathscr{P})$, $OS(\mathscr{P})$ and $FS(\mathscr{P})$ denote the optimal value, optimal solution set and feasible solution set of $\mathscr{P}$ respectively. If $\mathscr{P}(\pi)$ is an optimization problem depending on parameter π, then $V(OV(\mathscr{P}(\tilde{\pi})))$ denotes the value of the optimal value of $\mathscr{P}(\pi)$ for $\pi = \tilde{\pi}$. $\|B\|$ denotes the Euclidean norm of a vector B. If $\mathscr{P}$ is an integer programming minimization problem and $\overline{\mathscr{P}}$ its linear programming relaxation then we define the integrality gap to be:

$$\text{Integrality Gap } (\%) := 100 * (OV(\mathscr{P}) - OV(\overline{\mathscr{P}}))/OV(\mathscr{P})$$

A fleet of vehicles operating from a single depot delivers food and groceries to a set of customers. Customer orders consist of three types of items: frozen, refrigerated and dry. The two boxes layout is used in all vehicles of the fleet and boxes will be referred to as compartments. We make the following assumptions:

1. Customer orders may consist of one, two or all three types of items.
2. Weight and volume capacity of each vehicle must not be exceeded. Volume capacity of each compartment must not be exceeded.
3. Visits by multiple vehicles to each customer are not allowed; the order of each customer must be delivered by exactly one vehicle. It follows that all types of items in a customer's order must be loaded in the appropriate compartment of the same vehicle.
4. Each vehicle can deliver to one or more customers during a single trip from the depot.
5. The weight of each customer's order is assumed to be small compared to the weight capacity of each vehicle. Also the volume of each part of a customer's order is assumed to be small compared to the volume capacity of the space on which it must be loaded, i.e. the compartments and the remaining space of the vehicle.
6. The design of the groceries packages is such that integral numbers of items can be loaded onto any space of the vehicle, utilizing the full capacity of the space or,

at worst, with an insignificant loss in capacity. Similarly, the refrigerated and frozen compartments are designed to fit onto a truck without creating inconveniently narrow spaces that cannot be used.

We distinguish between two types of costs in the delivery operation:

Delivery costs: They depend on the distance traveled by each vehicle along its assigned route. They are assumed to consist primarily of fuel expenditure, tolls, personnel compensation, vehicle maintenance and depreciation. Delivery costs are minimized when the total distance traveled by the whole fleet is minimal.

Cooling costs: They are the costs for keeping non-ambient temperature items at the appropriate temperatures during delivery. They are assumed to consist of maintenance and depreciation costs of refrigerating and freezing equipment as well as cost of energy for maintaining low temperatures. Cooling costs are minimized when the minimal required number of compartments for non-ambient temperature items is used.

The problem is to find an assignment of customers to vehicles such that all customer demands are satisfied and the sum of delivery and cooling costs is minimal.

Delivery costs are a function of the sum of the distances traveled directly between two customers. Therefore, we do not know the minimal delivery costs before we find the optimal solution to the vehicle routing problem, part of which is the problem we are considering here. In order to find an assignment of customers to vehicles that is close to being optimal we need to have a good aproximation of the delivery costs in terms of costs of assigning a customer to a vehicle. Fisher and Jaikumar [24] have developed heuristic methods for approximating the optimal delivery costs. We adapt their method for approximating delivery costs.

We formulate the delivery scheduling problem with the two boxes layout as a 0-1 programming problem. Our notation is summarized below. A symbol referring to a set denotes the set itself as well as its cardinality.

I:	set of available vehicles
K:	set of item types, $K = \{A, R, F\}$
K_{-A}:	set of non-ambient temperature items, $K_{-A} = \{R, F\}$
J:	set of customers
K_j:	set of item types in the order of customer j, $\forall j \in J$
J_k:	set of customers who have ordered item type k, $\forall k \in K$
g_j:	total weight of the order of customer j, $\forall j \in J$
w_{jk}:	total weight of items of type k, $\forall k \in K_j$ in the order of customer j, $\forall j \in J$
t_{jk}:	total volume of items of type k, $\forall k \in K_j$ in the order of customer j, $\forall j \in J$
c_{ij}:	delivery cost; cost of assigning customer j, $\forall j \in J$, in the route to be served by vehicle i, $\forall i \in I$

p_k:	weight of compartment for items of type k, $k \in K_{-A}$
r_k:	volume of compartment for items of type k, $k \in K_{-A}$
q_k:	volume capacity of compartment for items of type k, $k \in K_{-A}$
e_k:	cooling cost; cost of compartment for items of type k, $k \in K_{-A}$
b_i:	weight capacity of vehicle i, $\forall i \in I$
d_i:	volume capacity of vehicle i, $\forall i \in I$

Very often all vehicles in the fleet have the same weight and volume capacities ($b_i = b$, $d_i = d$), and we will treat them as such in our computational experiments.

The decision variables of the optimization model are 0-1 variables defined as follows:

$$z_{ij} = \begin{cases} 1\,, & \text{if customer } j,\ \forall j \in J \text{ is assigned to vehicle } i,\ \forall i \in I \\ 0\,, & \text{otherwise}\,. \end{cases}$$

$$y_{ik} = \begin{cases} 1\,, & \text{if a compartment for items of type } k,\ k \in K_{-A} \text{ is added} \\ & \text{on vehicle } i,\ \forall i \in I \\ 0\,, & \text{otherwise.} \end{cases}$$

The formulation of the delivery scheduling problem with the two boxes layout as a 0-1 integer programming problem is the following:

DSP1

$$\text{Minimize} \quad \sum_{i \in I} \sum_{k \in K_{-A}} e_k y_{ik} + \sum_{i \in I} \sum_{j \in J} c_{ij} z_{ij}$$

subject to

$$(S) \qquad \sum_{i \in I} z_{ij} = 1\,, \quad \forall j \in J$$

$$(W) \qquad \sum_{k \in K_{-A}} p_k y_{ik} + \sum_{j \in J} g_j z_{ij} \leqslant b_i\,, \quad \forall i \in I$$

$$(CV) \qquad \sum_{j \in J_k} t_{jk} z_{ij} \leqslant q_k y_{ik}\,, \quad \forall i \in I,\ \forall k \in K_{-A}$$

$$(VV) \qquad \sum_{k \in K_{-A}} r_k y_{ik} + \sum_{j \in J_A} t_{jA} z_{ij} \leqslant d_i\,, \quad \forall i \in I$$

$$(I1) \qquad z_{ij} \in \{0, 1\}\,, \quad \forall i \in I,\ j \in J$$

$$(I2) \qquad y_{ik} \in \{0, 1\}\,, \quad \forall i \in I,\ k \in K_{-A}$$

By constraints (S), each customer must be assigned to exactly one route, or the whole order of each customer has to be delivered by exactly one vehicle. By constraints (W), the weight capacity of each vehicle cannot be exceeded. Constraints (CV) state that the volume capacity of each compartment cannot be exceeded given that such a compartment has been added onto a vehicle. Constraints (VV) state that the volume capacity of the vehicle cannot be exceeded.

3.2. A MIXED INTEGER PROGRAMMING MODEL FOR THE ONE PARTITION AND ONE BOX LAYOUT

The same assumptions hold here as in the previous subsection except that the one partition and one box layout is used. We will use the notation defined in the previous subsection except that we also need to define some additional notation:

$r_F^{\max}, r_F^{\min}$:	maximum and minimum volume of the frozen compartment.
$q_F^{\max}, q_F^{\min}$:	maximum and minimum volume capacity of the frozen compartment
r_B:	volume of the bulkhead of the frozen compartment.

Also in the present subsection, p_F is essentially the weight of the bulkhead. It is easy to see that $r_F^{\min} = q_F^{\min} + r_B$ and $r_F^{\max} = q_F^{\max} + r_B$. The decision variables used in the previous model will also be used here. However, since volume capacity of the frozen compartment is not fixed, we introduce a new decision variable:

v_i: volume capacity of the frozen compartment.

Another approximation we have implicitly made here is that e_F, that is the cost of the frozen compartment, is constant for each vehicle, while a principal determinant of these costs, that is, the volume of frozen merchandise, may vary across vehicles. This is only a minor deviation from reality, since in the real application, cooling costs represent only a small portion of delivery costs.

The formulation of the delivery scheduling problem with the one partition and one box layout as a mixed integer programming problem is the following:

DSP2

$$\text{Minimize} \quad \sum_{i\in I}\sum_{k\in K_{-A}} e_k y_{ik} + \sum_{i\in I}\sum_{j\in J} c_{ij} z_{ij}$$

subject to

$$(S) \qquad \sum_{i\in I} z_{ij} = 1\,, \quad \forall j \in J$$

$$(W) \qquad \sum_{k\in K_{-A}} p_k y_{ik} + \sum_{j\in J} g_j z_{ij} \leqslant b_i\,, \quad \forall i \in I$$

$$(RV) \qquad \sum_{j\in J_R} t_{jR} z_{ij} \leqslant q_R y_{iR}\,, \quad \forall i \in I$$

$$(FV) \qquad \sum_{j\in J_F} t_{jF} z_{ij} \leqslant q_F^{\max} y_{iF}\,, \quad \forall i \in I$$

$$(V1) \qquad r_R y_{iR} + r_B y_{iF} + v_i + \sum_{j\in J_A} t_{jA} z_{ij} \leqslant d_i\,, \quad \forall i \in I$$

$$(V2) \qquad \sum_{j\in J_F} t_{jF} z_{ij} \leqslant v_i\,, \quad \forall i \in I$$

$$(V3) \qquad q_F^{\min} y_{iF} \leqslant v_i\,, \quad \forall i \in I$$

$$(V4) \qquad v_i \leqslant q_F^{\max} y_{iF}\,, \quad \forall i \in I$$

$$(I1) \qquad z_{ij} \in \{0, 1\}\,, \quad \forall i \in I,\ j \in J$$

$$(I2) \qquad y_{ik} \in \{0, 1\}\,, \quad \forall i \in I,\ k \in K_{-A}$$

$$(C) \qquad v_i \geqslant 0\,, \quad \forall i \in I$$

Constraints (C) and (W) have the same meaning as in model (DSP1). Constraints (RV) state that the volume capacity of the refrigerated compartment should not be exceeded. Constraints (FV) state that the volume of frozen merchandise cannot exceed the maximum volume capacity of the frozen compartment. Constraints $(V1)$ state that the total volume capacity of the truck cannot be exceeded. Constraints $(V2)$ state that the actual volume of frozen merchandise can be at most as large as the (variable) volume capacity of the frozen compartment. By constraints $(V3)$ and $(V4)$, the volume capacity of the frozen compartment, if one is added in a vehicle, must be between the minimum and the maximum volume capacity of the frozen compartment of the truck.

3.3. MODEL TIGHTENING

Certain logical constraints can be appended to either (DSP1) or (DSP2) resulting in a tighter formulation. The following constraints $(L1)$ are derived as implications of constraints (CV), $(I1)$ and $(I2)$. They state that a customer j whose order includes items of type $k \in K_{-A}$, cannot be assigned to vehicle i, unless a compartment for items of type k is added to the vehicle:

$$(L1) \quad z_{ij} \leqslant y_{ik}\,, \quad \forall i \in I,\ j \in J_k,\ k \in K_{-A}$$

Constraints $(L1)$ are not necessary in the description of the model; they are implications of other constraints already present in the formulation and are implicitly satisfied by all feasible solutions of the problem. However, writing and enforcing them explicitly, could result in tighter linear programming bounds and consequently a reduction in the number of nodes of branch-and-bound algorithms for solving the problem to optimality. Furthermore, these constraints may result in improved lower bounds and feasible solutions if they are used in Lagrangean bounding/heuristic schemes.

3.4. A NOTE ON THE DERIVATION OF DELIVERY COSTS c_{ij}

It was mentioned earlier that the delivery cost c_{ij} of assigning customer j to a route traveled by vehicle i is estimated as an approximation of the optimal cost of the associated vehicle routing problem. The method we use to estimate the delivery costs is an adaptation of one of the methods proposed by Fisher and Jaikumar [24]. A brief description of the method we are using follows.

Let (ρ_j, θ_j) be the polar coordinates of customer j on the plane, whose origin is occupied by the depot. We assume that customer indices in J are sorted by order of increasing angle θ_j. The plane is partitioned into J customer cones. Customer cones are formed such that the infinite half ray forming the boundary between customer cones j and $j+1$ bisects the angle $(\theta_{j+1} - \theta_j)$, formed by the half rays through customers j and $j+1$. We associate the total weight g_j of the order of customer j with customer cone j. We assume that the weight capacity of all vehicles is the same $b_i = b$, $\forall i \in I$. We define

$$\gamma = \frac{\sum_{j \in J} g_j}{I \times b}$$

that is, the fraction of the weight capacity of each vehicle that would be used under the fictitious assumption that orders are allocaed to vehicles in such a way that all vehicles are loaded up to exactly the same weight. Vehicle cones are then formed from a contiguous group of customer cones or fractions of customer cones such that the weight within each vehicle cone equals $\gamma \times b$. We select a seed point with polar coordinates (τ_i, δ_i) in vehicle cone i. Seed point i lies on the infinite half ray that bisects customer cone i. Its distance τ_i from the origin is chosen such that the total weight included inside the vehicle cone through τ_i is $0.75 \times \gamma \times b$. We assume that the total weight within customer cone i is evenly distributed within the sector of radius $\rho_{\max}$, that is the distance of the customer who is the remotest from the depot among those who are located in customer cone i. Under this simplifying assumption we can determine τ_i as

$$\tau_i^2/\rho_{\max}^2 = 0.75 \Rightarrow \tau_i \approx 0.866 \times \rho_{\max} .$$

Delivery cost c_{ij} then is computed as

$$c_{ij} = \rho_j + d_{ij} - \tau_i$$

where

$$d_{ij} = \sqrt{\rho_j^2 + \tau_i^2 - 2\rho_j \tau_i \cos(\theta_j - \delta_i)} .$$

We describe the procedure for generating random test problems in the Appendix.

After delivery costs c_{ij} are determined, cooling costs e_k are set appropriately following a principle that the ratio of cooling to delivery cost per vehicle must equal a specified value.

4. Critique of the (DSP1) model and preliminary experimentation

In this section we discuss features of the (DSP1) and (DSP2) models that make them attractive both from a computational and a modeling point of view. Branch-and-bound algorithms for either (DSP1) or (DSP2) can be speeded up by exploiting the powerful concept of *double contraction*. In Subsection 4.1 we discuss double contraction for (DSP1); the analysis and computational results would be similar for

(DSP2). In Subsection 4.2 we discuss modeling issues addressed by (DSP1) and (DSP2). Subsections 4.3, 4.4 and 4.5 contain experimental results with randomly generated problems, real data, and a large real-like instance, respectively. In Subsection 4.6 we investigate the effect of appending constraints ($L1$) in strengthening the linear programming bound and speeding up branch-and-bound algorithms for solving the problem to optimality.

4.1. DOUBLE CONTRACTION AND THE (DSP1) MODEL

Double contraction is described in Spielberg [46] and Guignard and Spielberg [33] as a measure of merit of a 0-1 variable with regard to branching. A 0-1 variable is said to be *double contracting* if fixing it to either 0 or 1 reduces the collective feasibility range of the remaining unfixed variables of an integer programming problem, without necessarily forcing other variables to be fixed to some value. It becomes evident that giving higher branching priorities to double contracting variables results to a considerable reduction of nodes in a branch-and-bound tree. Guignard and Spielberg [34] showed how the number of nodes of the branch-and-bound tree as well as the computation times can be dramatically reduced by giving high branching priorities to double contracting variables when solving a mixed integer problem for harvest scheduling and transportation planning in forest management.

Variables y_{ik} of (DSP1) are double contracting. Indeed, constraints (W) of (DSP1) imply the following cover inequalities:

$$(C) \quad \sum_{k \in K_{-A}} y_{ik} + \sum_{j \in J} z_{ij} \leqslant n\,, \quad \forall i \in I$$

where n is an integer such that $n_m^i \leqslant n \leqslant |K_{-A}| + |J|$, and n_m^i is the cardinality of the minimal cover of (W) for each $i \in I$. Also constraints (CV) imply:

$$(L) \quad \sum_{j \in J_k} z_{ij} \leqslant |J_k| y_{ik}\,, \quad \forall i \in I,\ k \in K_{-A}\,.$$

If for some $k* \in K_{-A}$ and $i* \in I$, y_{i*k*} is fixed to 0, by (L) this results to fixing of z_{i*j} to 0 for all $j \in J_{k*}$. If on the other hand y_{i*k*} is fixed to 1, by (C) we obtain the following 'tighter' inequality on the remaining unfixed variables:

$$(C') \quad \sum_{k \in K_{-A}/\{k*\}} y_{i*k} + \sum_{j \in J} z_{i*j} \leqslant n - 1\,.$$

In subsections 4.3 and 4.4, and the tables referenced within, we show how the size of the branch-and-bound tree and CPU times are dramatically reduced by giving higher branching priorities to y_{ik} variables.

4.2. MODELING ISSUES ADDRESSED BY (DSP1)

The generalized assignment heuristic [24] considers only a single resource (e.g., weight or volume) whose availability is limited on the vehicles. In delivery with multiple compartments we need to consider both weight and volume limitations on the vehicles and volume limitations in each separate compartment. In this respect (DSP1) captures both weight and volume limitations in a rather efficient way without the need to introduce too many binary variables.

An interesting feature of the model is that by penalizing the use of compartments one not only helps determine a feasible delivery schedule using the smallest possible number of compartments but one that can possibly require fewer vehicles than are available. In contrast, the delivery cost structure of the generalized assignment heuristic incents the model to use all available vehicles.

The presence of multiple resource constraints in (DSP1) gives the model a multi-resource generalized assignment problem structure. The latter problem has been studied by Gavish and Pirkul [28], who have proposed non-Lagrangean based as well as Lagrangean heuristics and a branch and bound algorithm that have solved medium sized instances of the problem in reasonable times.

(DSP2) has a more complex structure but offers more flexibility in the allocation of capacities, that would allow better assignments of customers to trucks. We compare the solutions that (DSP1) and (DSP2) give on the same random and real data in the following subsections.

4.3. EXPERIMENTATION WITH RANDOMLY GENERATED DATA

We test the model, solving a set of small randomly generated test problems. The common characteristics of the test problems are shown in the upper part of Table 1.

Table 1. Values of the random test problem parameters

Parameter	Two boxes	One partition-one box
	Value	
J	25	
b_i (100 lbs)	300	
d_i (10 cuft)	450	
p_R (100 lbs)	10	
p_F (100 lbs)	15	
r_R	$d_i/3$	$d_i/6 \div d_i/2$
r_F	$d_i/3$	$d_i/3$
q_R	$0.90 \times r_R$	
q_F	$0.85 \times r_F$	
$\rho_{\min}$ (miles)	1	
$\rho_{\max}$ (miles)	40	
I	3, 4, 5, 6, 7	
π	0.30, 0.40, 0.50, 0.60, 0.70, 1.00	
$\rho_{P/D} \approx$	0.05, 0.10, 0.20, 0.30, 0.40, 0.50	

The variable charactristics of the problem are: (1) number of vehicles I, (2) percentage π of customers who have ordered all three item types, (3) ratio $\rho_{P/D}$ of cooling cost per vehicle to delivery costs per vehicle. Their ranges are shown in the lower part of Table 1.

The problems were generated to have volume capacity utilization (load factor) comparable to that of the real problems described in the next subsection. We define load factor as the ratio of the total volume of customer orders to the total volume capacity of the fleet. Load factors for the random problems range from 30.2% (problem 9) to 44.3% (problem 11).

Picking different combinations of values for the problem parameters from Table 1 allows the generation of hundreds of random problem instances. We experimented with a good portion of all the possible instances; in Tables 2–4 we show results of solving selected non-trivial problems using (DSP1). Test problems were generated by a portable random problem generator written in FORTRAN. The random problem generator is described in Appendix A. We used the GAMS modeling system [7] in our experiments. The random problem generator produces as output a GAMS model that can be solved by a variety of available MIP solvers. We solved all random problems to optimality using GAMS/OSL on an IBM RISC6000 workstation.

These experiments have revealed some key characteristics of the (DSP1) model. Problem size can increase rapidly as the number of customers, and the number of vehicles to serve them, becomes larger. Even for these small problem instances, the integrality gap is large. A method that can give us better approximations that the linear programming relaxation is Lagrangean Relaxation (see Fisher [20, 21], Geoffrion [29], Held and Karp [36, 37], and Shapiro [45] among others).

Another interesting characteristic of the model is that it can naturally find, i.e., without a need to force it through constraints, the smallest possible number of vehicles for a feasible assignment, I^*, that is shown in the third column of Tables 2–4. Since delivery costs are computed based on the assumption that exactly I vehicles are used, if the optimal assignment has $I^* < I$, delivery costs need to be recomputed with I set to I^* and the problem must be solved again using the new costs. We chose some random problems of which the optimal (DSP1) solution requires $I^* < I$ vehicles. We recomputed the delivery costs with I^* vehicles and solved the problems again. Table 5 shows the results of these experiments. The optimal (DSP1) costs in all cases are smaller when delivery costs are computed with I^* seed points. This was expected, since when I^* seed points are used, the delivery costs are better approximations of the actual optimal vehicle routing costs. Figures 1 and 2 show the optimal DSP1 assignments of customers to vehicles, and the optimal routes with I and I^* seed points, respectively, for problem 4 shown in Tables 2 and 5. The better (DSP1) optimal solution in the latter corresponds to a better feasible solution for the vehicle routing problem.

Finally, the last four columns of Tables 2–4 show how the number of nodes and CPU times are dramatically reduced when higher priorities are given by the user to

Table 2. Experimental results for varying I

$\pi = 0.50,\ \rho_{P/D} = 0.20$

Prob. no.	I	I*	LP bound	(DSP1) IP optimum	Int. gap (%)	Variables	Constraints	Nodes		RS6000 CPU sec		(DSP2) IP optimum
								Priorities		Priorities		
								On	Off	On	Off	
1	3	3	289.450	424.257	31.8	82	38	20	20	0.620	0.620	424.257
2	4	4	287.146	421.765	31.9	109	42	55	143	1.470	2.450	419.198
3	5	4	334.030	468.950	28.8	136	46	122	>6700	2.460	>143.000	448.570
4	6	4	348.701	480.852	27.5	163	50	356	>17000	8.510	>554.000	472.906
5	7	5	421.830	551.929	23.6	190	54	538	>40000	13.320	>1200.000	496.971

Table 3. Experimental results for varying π

$I = 5,\ \rho_{P/D} = 0.20$

Prob. no.	π	I*	LP bound	IP optimum	Int. gap (%)	Variables	Constraints	Nodes		RS6000 CPU sec		(DSP2) IP optimum
								Priorities		Priorities		
								On	Off	On	Off	
6	0.30	4	353.854	475.023	25.56	136	46	94	6819	1.440	109.220	432.710
7	0.40	4	353.439	466.214	24.19	136	46	115	12510	2.840	227.720	429.458
8	0.50	4	334.024	468.950	28.77	136	46	122	>28128	2.430	>909.070	448.570
9	0.60	4	309.240	467.073	33.79	136	46	130	>11060	3.140	>208.990	423.726
10	0.70	4	343.647	483.666	28.90	136	46	91	>3260	2.150	>51.660	428.408
11	1.00	4	405.343	548.379	26.08	136	46	93	>19140	2.210	>604.000	492.974

Table 4. Experimental results for varying $\rho_{P/D}$

$I = 5,\ \pi = 0.50$

Prob. no.	$\rho_{P/D}$	I^*	LP bound	IP optimum	Int. gap (%)	Variables	Constraints	Nodes		RS6000 CPU sec		(DSP2) IP optimum
								Priorities		Priorities		
								On	Off	On	Off	
12	0.05	5	177.858	243.766	27.03	136	46	167	663	3.460	10.020	241.794
13	0.10	4	229.913	318.950	27.91	136	46	102	4238	2.480	65.800	318.950
14	0.20	4	334.024	468.950	28.77	136	46	122	>17000	2.490	>300.000	448.570
15	0.30	5	438.134	616.141	28.99	136	46	122	>20000	2.820	>350.000	548.570
16	0.40	5	542.245	746.141	27.32	136	46	129	>20000	2.910	>350.000	648.570
17	0.50	5	646.355	876.141	26.22	136	46	128	>20000	3.050	>350.000	748.570

variables y_{ik}, compared to solving the model having the solver set priorities internally.

We obtained optimal solutions for the randomly generated data sets using (DSP2). In almost all problems (DSP2) gave solutions of lower cost than (DSP1), because it allows more flexibility in allocating capacities. The optimal costs of the

Table 5. Results when delivery costs are computed with *I* and *I** seed points

Prob. no.	*I*	*I**	Seed points used for delivery cost computation					
			I			*I**		
			LP bound	IP optimum	Integrality gap (%)	LP bound	IP optimum	Integrality gap (%)
3	5	4	334.030	468.950	28.8	330.756	452.582	26.9
4	6	4	348.701	480.852	27.5	374.645	460.391	18.6
5	7	5	421.830	551.929	23.6	423.231	549.582	23.0
6	5	4	353.854	475.023	25.6	326.040	428.017	23.8
13	5	4	229.913	318.950	27.9	226.645	302.582	25.1

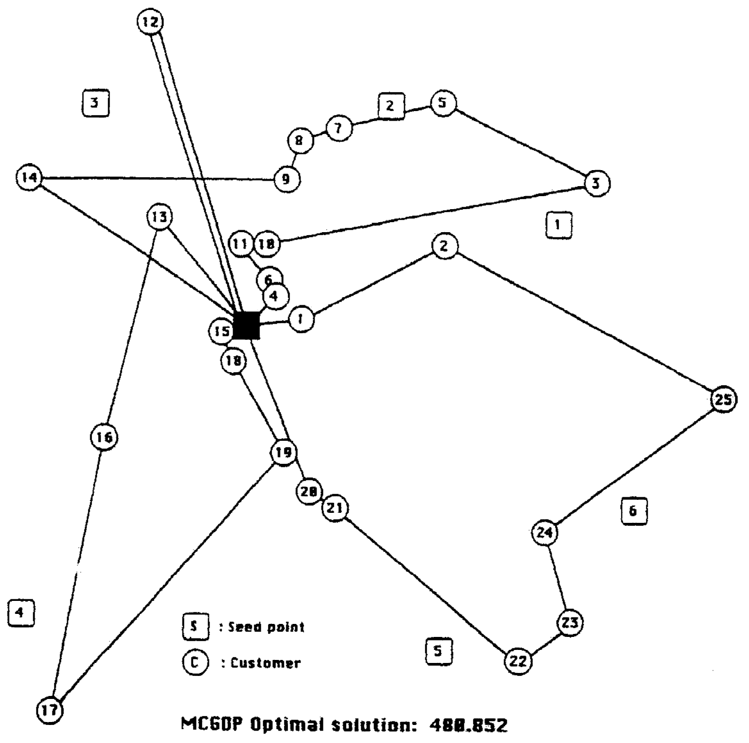

Figure 1. Optimal DSP1 assignment of dustomers to vehicles when *I* seed points are used.

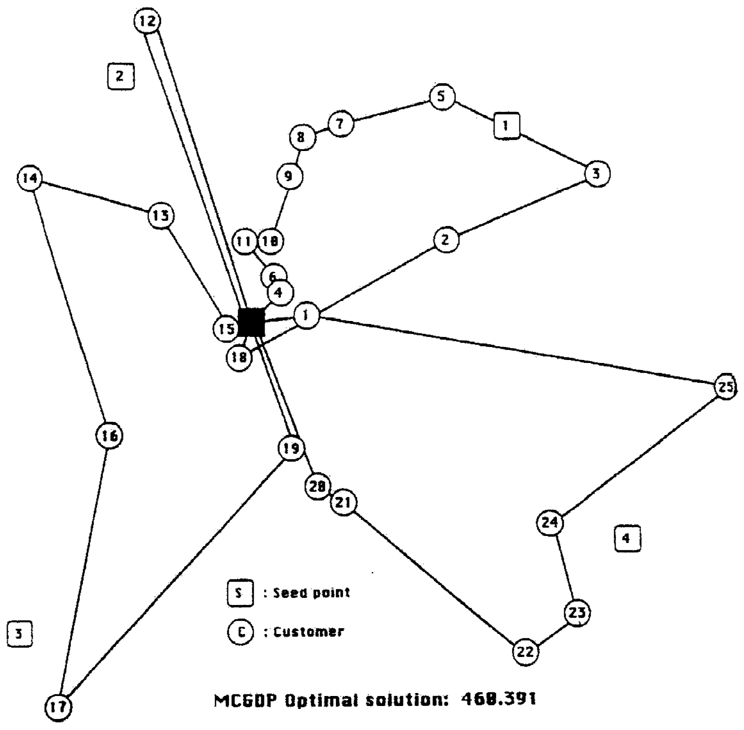

Figure 2. Optimal DSP1 assignment of customers to vehicles when I^* seed points are used.

randomly generated problems with (DSP2) can be seen in the last column of Tables 2–4.

4.4. EXPERIMENTS WITH REAL DATA

Real data were obtained from a company that specializes in distribution to convenience stores. The company operates a warehouse in Lancaster, Pa. On an average day, their fleet of 45 vehicles serves about 240 customers. We were given data for five routes prepared by a dispatcher, including locations of customers, and weights and volumes of the frozen and dry parts of their orders. The number of customers in these routes is 44, that is one-fifth to one-sixth of the total number of customers served in a typical day. We were also given the weight and volume capacities of the box for frozen items and that of the remaining space for dry items, assuming that the box has a fixed capacity. The main difference between these and the randomly generated data is that in the former, the customers are clustered while in the latter, they are uniformly distributed on the plane. Because of this peculiarity of the data, automatic seed setting gives poor locations for seeds. Therefore, we choose seeds manually, such that they coincide with customer locations within clusters. The load factor of the real delivery schedule is a low 29.9%, i.e., capacities of the vehicles are not very tightly utilized.

A brief description of the spatial distribution of customers and their allocation to routes follows. There are 10 customers in Connecticut, nine of which form a cluster in the Hartford area and one is in Bridgeport. There is one customer in Dayton, New Jersey, seven in east Maryland, and 26 in Pennsylvania, of which 10 form a cluster west of Harrisburg, three are in West Chester, two in Media, and 11 form a cluster in the Greater Philadelphia area. The routes prepared by the dispatcher are as follows:

Route 1: Serves all 10 customers in Connecticut and the customer in New Jersey.
Route 2: Serves all seven customers in east Maryland.
Route 3: Serves all 10 customers west of Harrisburg.
Route 4: Serves eight customers in Philadelphia.
Route 5: Serves the three customers in West Chester, the two customers in Media and the remaining three customers in the Philadelphia area.

We prepared the data for the (DSP1) model to be solved using GAMS with LAMPS or OSL as MIP solvers. We experimented with the number of trucks assuming that there are five, four, three or two trucks in the fleet. Delivery costs were computed using five, four, three or two customers, respectively, as seed points. (DSP1) was solved to optimality. When two trucks are available no feasible solution was found. Table 6 summarizes the computational results with real data. It is worth mentioning that integrality gaps of the real problem are even larger than those of the randomly generated ones, ranging from 53 to 67%. The dramatic reduction of nodes and CPU times when higher branching priorities are given to variables y_{ik} is also evident here. The last column of Table 6 gives the optimal solutions with (DSP2). In all three problems the minimal costs with (DSP2) are better than with (DSP1), as was also observed with the randomly generated data.

A description of the routes obtained follows.

4.4.1. *Five available vehicles*

The load factor using five vehicles is 29.9%. The furthermost customers in the Hartford, Harrisburg and east Maryland clusters, one customer in West Chester and one in Philadelphia were chosen as seeds. The optimal (DSP1) solution assigned customers to only four routes as follows:

Route 1: Serves all 10 customers in Connecticut and the customer in New Jersey.
Route 2: Serves all seven customers in east Maryland.
Route 3: Serves all 10 customers west of Harrisburg, the three customers in West Chester, the two customers in Media and one customer in the Philadelphia area.
Route 4: Serves 10 customers in the Philadelphia area.

Table 6. Experimental results for the real distribution problem with varying number of available vehicles

Avail./ used vehicles	(DSP1) IP opt.	Int. gap (%)	Variables		Nodes		RS6000 CPU sec		(DSP2) IP opt.
					Priorities		Priorities		
					On	Off	On	Off	
5/4	164.444	67.2	225	59	12	>12500	0.910	>500.000	161.352
4/4	164.444	67.1	180	56	11	41313	0.700	1093.770	161.352
3/3	295.378	52.6	135	53	1816	>31000	30.910	>800.000	219.395

Table 7. Experimental results for the large real-like distribution problem

Avail./ used vehicles	(DSP1) IP sol.	Int. gap (%)	Variables	Constraints	Nodes		J5000 CPU sec		(DSP2) IP sol.
					Priorities		Priorities		
					On	Off	On	Off	
32/32	1222.783	59.9	11297	449	>50100	>129000	100 K +	100 K+	1050.937

4.4.2. *Four available vehicles*

When four vehicles are used, the load factor is 37.3%. The same seed points as above except for the West Chester one were chosen. The optimal (DSP1) solution assigned customers to the same four routes as above.

4.4.3. *Three available vehicles*

When three vehicles are used the load factor is 49.8%. The furthermost customers in the Hartford, Harrisburg and east Maryland clusters were chosen as seeds. The optimal (DSP1) solution assigned customers to three routes as follows:

Route 1: Serves all 10 customers in Connecticut, the customer in New Jersey, one customer from the Harrisburg cluster and four customers in Philadelphia.
Route 2: Serves all seven customers in east Maryland, two customers in Philadelphia and one customer west of Harrisburg.
Route 3: Serves eight customers west of Harrisburg, the three customers in West Chester, the two customers in Media and six customers in the Philadelphia area.

When three trucks are available the total distance traveled by the fleet is larger than in the four- and five-vehicle cases, because of poor routes found. Forcing the model to use three vehicles by making only that many available, as opposed to letting it naturally find an optimal three-vehicle solution when more are available, produced a solution of poor quality.

4.5. EXPERIMENTS WITH A LARGE REAL-LIKE INSTANCE

We sought to test the models on an instance in the order of magnitude of the distribution company's daily operation of serving 240 customers. Instead of generating large random instances, in which customer locations would be uniformly distributed on the plane rather than clustered as in reality, we created one from the 44-customer real instance. For each customer in that instance, we created seven more customers whose orders' weight and volume by temperature were normally distributed around those of the original customer with 10% of those as standard deviation. One duplicate customer was located 5% of the maximum radius to the southeast of the original; two were located 1.25 and 6.25% of the maximum radius southeast of the symmetrical image of the original customer with respect to the east–west axis; two were located 2.50 and 7.50% of the maximal radius southeast of the symmetrical image of the original customer with respect to the north–south axis; and two were located 3.75 and 8.75% of the maximum radius southeast of the symmetrical image of the original axis with respect to the depot. Thus we created a 352-customer instance, almost 50% larger than the distribution company's daily problem. Since we found good 4-vehicle solutions to the 44-customer instance, we sought to serve the large instance with 32 vehicles and computed delivery costs using 32 customer locations as seed points. We used Microsoft Excel to generate the large instance, which has a load factor of 37.0%.

We solved the problem with CPLEX 7.0 through GAMS on a Hewlett-Packard J5000; cliques and covers are generated at each branch-and-bound node. Because of the large size of the problem, we did not expect to solve it to optimality. We show on Table 7 the solution found after 100 000 CPU seconds.

4.6. MODEL TIGHTENING

As experiments with randomly generated and real problems show, (DSP1) has relatively large integrality gaps. Appending constraints ($L1$) to (DSP1) and solving the augmented model, has resulted in stronger linear programming bounds (smaller integrality gaps) and consequently fewer branch-and-bound nodes for solving problems to optimality, or finding a good feasible solution to the large real-like instance. Comparative results with the original and augmented (DSP1) are shown in Tables 8 and 9. Randomly generated problems are numbered as in Tables 2–4. Real problems 1, 2 and 3 consist of the set of 44 customers and five, four and three available vehicles, respectively. Large is the 352-customer, 32-vehicle instance.

When solving the problems to optimality priorities to double contracting variables are on. Results show that the augmented (DSP1) gives much stronger lower bounds (smaller integrality gaps) than the original model. As a result of that, the number of nodes needed to find the optimal solution is drastically smaller except for Large, whose constraint matrix has about as many thousands of rows as columns when the logical constraints are appended. However, the augmented model is computationally more expensive. It needs considerably higher CPU times to compute LP bounds and the time spent at each node of the branch-and-bound tree is considerably larger; for random problems 5, 11 and 17 and real problems 2 and 3 the total OSL CPU time with the augmented model is larger than with the original one despite the considerably smaller number of nodes in all of them but real problem 3.

5. Lagrangean approximations for (DSP1)

We will use Lagrangean Relaxation, Substitution and Decomposition to develop approximations for (DSP1). All approximations presented here are based on the augmented (DSP1) model. In Subsection 1, we discuss Lagrangean Relaxations $LR1$ through $LR4$ and we outline a subgradient optimization algorithm for $LR3$, the one that our computational experiments have shown to give the best tradeoff between ease of computation and tightness of the obtainable bounds. In Subsection 2, we discuss Lagrangean Decomposition LD and Lagrangean Substitution LS together with the corresponding subgradient algorithms. In Subsection 3, we present a simple Lagrangean heuristic.

The subgradient algorithms we use are of the traditional sort. We are aware of

Table 8. Comparative results with the original (w/o ($L1$)) and augmented (with ($L1$)) (DSP1) model for the random and real problems

Prob. no.	IP opt.	LP bound		RS6000 CPU sec		Int. gap (%)		OSL nodes		RS6000 CPU sec	
		w/o ($L1$)	with ($L1$)	w/o ($L1$)	with ($L1$)	w/o ($L1$)	with ($L1$)	w/o ($L1$)	with ($L1$)	w/o ($L1$)	with ($L1$)
1	424.257	289.459	424.257	0.14	0.30	31.77	0.00	20	1	0.50	0.16
2	421.765	287.146	421.064	0.18	0.43	31.91	0.17	55	1	1.23	0.36
3	468.950	334.023	447.297	0.18	0.72	28.77	4.62	122	4	2.33	1.06
4	480.852	348.323	464.615	0.23	0.74	27.56	3.38	356	108	8.19	4.88
5	551.929	421.839	494.965	0.25	1.00	23.57	10.32	538	456	12.96	31.37
6	475.023	353.584	440.933	0.19	0.52	25.57	7.17	94	24	1.95	1.57
7	466.214	353.439	456.785	0.21	0.72	24.19	2.02	115	18	2.71	1.10
9	467.073	309.240	434.880	0.20	0.62	33.79	6.89	130	9	2.97	0.66
10	483.666	343.647	448.061	0.21	0.64	28.95	7.36	91	4	1.98	0.84
11	548.379	405.343	501.602	0.20	0.71	26.08	8.53	93	50	2.00	3.83
12	243.766	177.858	242.013	0.19	0.54	27.04	0.72	167	4	3.44	0.97
13	318.950	229.913	318.950	0.20	0.56	27.92	0.00	102	1	2.28	0.23
15	616.141	438.134	561.069	0.21	0.71	28.89	8.93	122	31	2.72	2.10
16	746.141	542.245	672.122	0.20	0.59	27.33	9.92	129	31	2.83	2.42
17	876.141	646.355	780.654	0.19	0.60	26.22	10.90	128	29	3.01	3.08
Real 1	164.444	53.898	161.729	0.27	0.68	67.22	1.65	12	4	0.75	0.88
Real 2	164.444	54.102	161.729	0.23	0.56	67.10	1.65	11	4	0.55	0.77
Real 3	295.378	139.988	247.924	0.20	0.19	52.61	16.06	1816	2599	29.78	62.07

Table 9. Comparative results with the original (w/o ($L1$)) and augmented (with ($L1$)) (DSP1) model for the large real-like instance

Prob. no.	IP opt.	LP bound		J5000 CPU sec		Int. gap (%)		CPLEX nodes		J5000 CPU sec	
		w/o ($L1$)	with ($L1$)	w/o ($L1$)	with ($L1$)	w/o ($L1$)	with ($L1$)	w/o ($L1$)	with ($L1$)	w/o ($L1$)	with ($L1$)
Large	1190.578	490.263	997.485	0.98*	3.68*	60.36	16.22	50100*	326797*	100K*	100K*

more sophisticated subgradient search algorithms (see for example Sen and Sherali [44]). Nevertheless, we decided to keep the focus of this work on efficient solution of subproblems and effective heuristics.

Lagrangean Relaxations ignore the assignment aspect of the model and induce decompositions across vehicles. The Lagrangean Relaxation problem decomposes into as many subproblems as there are vehicles in the fleet. Each subproblem is constrained by the weight capacity of the corresponding vehicle and the volume capacities of its compartments. Keeping all capacity constraints, as in $LR4$, yields subproblems that can be hard to solve. In $LR1$, $LR2$ and $LR3$, some capacity constraints are relaxed thus yielding easier subproblems.

In Lagrangean Decomposition LD and Substitution LS, the assignment aspect of the model is not ignored but rather separated from the vehicle capacity aspect. Both schemes decompose the problem into: (i) A single subproblem that addresses only assignments of customers to vehicles ignoring vehicle weight and volume capacities. (ii) One subproblem per vehicle that addresses only capacities on that vehicle. LD and LS require $I \times (K + J - 1)$ and I multipliers, respectively. We do not expect LD or LS to yield much stronger lower bounds than the Lagrangean Relaxations because the assignment subproblems usually has naturally integer solutions. However, they could accommodate two Lagrangean heuristics as opposed to one for Lagrangean Relaxations, therefore increasing the probability of yielding better feasible solutions.

Our Lagrangean heuristic attempts to find feasible solutions at each subgradient iteration. It consists of fixing variables y_{ik} to their optimal Lagrangean values, solving the reduced problem to optimality, and if a feasible solution is found, attempting to improve it. The heuristic can be used in any of our approximation schemes.

5.1. LAGRANGEAN RELAXATIONS $LR1$, $LR2$, $LR3$ AND $LR4$

The first Lagrangean Relaxation $LR1$ is obtained if we relax constraints (S) with multipliers u_j, $\forall j \in J$, constraints (CV) with multipliers $\lambda_{ik} \leqslant 0$, $\forall i \in I$, $k \in K_{-A}$ and constraints (VV) with multipliers $\mu_i \leqslant 0$, $\forall i \in I$. The resulting Lagrangean subproblem decomposes into I independent 0-1 knapsack problems with logical constraints, similar to those in the $LD1\beta$ decomposition for the PUMS problem, presented in Guignard [32] (see also Lee and Guignard [42]). These subproblems can be strengthened to Setup Knapsack Problems, described in Chajakis and Guignard [9], by appending constraints $(L2)$:

$$(L2) \quad \sum_{j \in J_k} z_{ij} \geqslant y_{ik}\,, \quad \forall i \in I,\ k \in K_{-A}\,.$$

Constraints $(L2)$ are redundant in the augmented $(DSP1)$ model but not in the Lagrangean subproblems. The Setup Knapsack Problem can be solved with a number of efficient algorithms, presented in Chajakis and Guignard [9].

The second Lagrangean Relaxation $LR2$ is obtained if we relax constraints (S)

with multipliers u_j, $\forall j \in J$, and constraints (CV) with multipliers $\lambda_{ik} \leqslant 0$, $\forall i \in I$, $k \in K_{-A}$. The resulting Lagrangean subproblem decomposes into I independent two-dimensional multi-knapsack problems, with logical constraints $(L1)$, and can be further strengthened with constraints $(L2)$, thus becoming *Setup Multi-Knapsack Problems*. Although efficient exact algorithms for multi-knapsack problems have been developed by Gavish and Pirkul [27], and even more efficient algorithms for the special case of two-dimensional knapsacks have been proposed by Freville and Plateau [26], to our knowledge, no efficient algorithm exists for the Setup Multiknapsack Problem.

The third Lagrangean Relaxation $LR3$ is obtained if we relax constraints (S) with multipliers u_j, $\forall j \in J$, and constraints (VV) with multipliers $\mu_i \leqslant 0$, $\forall i \in I$. We will describe this scheme in more detail because our computational experiments have shown that it gives the best tradeoff between ease of computation and tightness of the obtainable bounds, among all four schemes we have examined. The resulting Lagrangean relaxation $LR3(u, \mu)$ of (DSP1) is:

$LR3(u, \mu)$

$$\begin{aligned} \text{Minimize} \quad & \sum_{i \in I} \sum_{k \in K_{-A}} (-\mu_i r_k + e_k) y_{ik} + \\ & \sum_{i \in I} \sum_{j \in J} (c_{ij} - u_j - \mu_i t_{jA}) z_{ij} + \sum_{j \in J} u_j + \sum_{i \in I} \mu_i d_i \\ \text{subject to} \quad & (W),\ (CV),\ (L1),\ (I1) \text{ and } (I2). \end{aligned}$$

$LR3(u, \mu)$ decomposes into I independent 0-1 multi-knapsack subproblems with logical constraints, that can be strengthened to Setup Multi-knapsacks by appending constraints $(L2)$:

$$V(LR3(u, \mu)) = -\sum_{i \in I} V(LR3_i(u, \mu)) + \sum_{j \in J} u_j + \sum_{i \in I} \mu_i d_i$$

where $LR3_i(u, \mu)$ is:

$LR3_i(u, \mu)$

$$\begin{aligned} \text{Maximize} \quad & \sum_{k \in K_{-A}} (\mu_i r_k - e_k) y_{ik} + \sum_{j \in J} (u_j + \mu_i t_{jA} - c_{ij}) z_{ij} \\ \text{subject to} \quad & \\ (W_i) \quad & \sum_{k \in K_{-A}} p_k y_{ik} + \sum_{j \in J} g_j z_{ij} \leqslant b_i \\ (CV_i) \quad & \sum_{j \in J_k} t_{jk} z_{ij} \leqslant q_k y_{ik}, \quad \forall k \in K_{-A} \\ (L1_i) \quad & z_{ij} \leqslant y_{ik}, \quad \forall j \in J_k,\ k \in K_{-A} \\ (L2_i) \quad & \sum_{j \in J_k} z_{ij} \geqslant y_{ik}, \quad \forall k \in K_{-A} \\ (I1_i) \quad & z_{ij} \in \{0, 1\}, \quad \forall j \in J \end{aligned}$$

$$(I2_i) \qquad y_{ik} \in \{0, 1\}\,, \quad \forall k \in K_{-A}$$

The Lagrangean dual is a maximization over multipliers u and μ:

$$LR3 = \max\{V(LR3(u, \mu)) | (u), \mu \geqslant 0\}$$

The optimal value of Lagrangean dual $LR3$ and the corresponding multipliers are computed iteratively with a subgradient-like method that is outlined below:

Initialization: Set multipliers u_j^0, μ_i^0 either to zero or to values that give a good $LR3$ at the first iteration. We use as initial multipliers the optimal dual variables corresponding to the dualized constraints of the LP relaxation of (DSP1).

Iterative procedure: Execute subgradient iterations until certain termination criteria are satisfied. The l-th subgradient consists of the following steps:

- Solve subproblems $LR3_i(u, \mu)$ for each $i \in I$. Let $(\bar{y}_{ik}, \bar{z}_{ij})$ be the optimal solution of $LR3(u, \mu)$.
- Compute modified subgradient vectors $U^l = \{U_j^l\}_{\forall j \in J}$, $M^l = \{M_i^l\}_{\forall i \in I}$ of Camerini et al. [8] as follows:

  $$U_j^l = (U_j^l)' + \beta_u^l U_j^{l-1}\,, \quad j \in J\,, \qquad M_i^l = (M_i^l)' + \beta_\mu^l M_i^{l-1}\,, \quad i \in I$$

  where

  $$(U_j^l)' = 1 - \sum_{i \in I} \bar{z}_{ij}\,, \ \forall j \in J\,, \qquad (M_i^l)' = d_i - \sum_{j \in J_A} t_{jA}\bar{z}_{ij} - \sum_{k \in K_{-A}} r_k \bar{y}_{ik}\,, \ \forall i \in I$$

  and β_u^l, for subgradient vector U is:

  $$\beta_u^l = \begin{cases} -1.5 \dfrac{U^{l-1}V^l}{\|U^{l-1}\|^2} & \text{if } U^{l-1}U^l < 0 \\ 0 & \text{otherwise} \end{cases}$$

  β_μ^l for subgradient vector M is defined in a similar manner.
- Multipliers u are unrestricted in sign and multipliers μ must be nonnegative. They are adjusted as follows:

  $$u_j^l = u_j^{l-1} + \theta_u^l U_j^l\,, \qquad \mu_i^l = \max\{0, (\mu_i^{l-1} + \theta_\mu^l M_i)\}$$

  Separate stepsizes θ_u^l and θ_μ^l are used for the u and μ multipliers, respectively, because the magnitudes of U and M subgradients are much different. Using separate stepsizes promises faster convergence of the algorithm, as computational evidence suggests (see Afentakis and Gavish [1]). Stepsizes θ_u^l and θ_μ^l are:

$$\theta_u^l = \sigma^l \frac{T - V(LR3(u^{l-1}, \mu^{l-1}))}{\|U\|^2}, \qquad \theta_\mu^l = \sigma^l \frac{T - V(LR3(u^{l-1}, \mu^{l-1}))}{\|M\|^2}$$

where $0 < \sigma^k < 2$ and T is a target computed as the average of the current lower bound and the value of the best known feasible solution of (DSP1).

The fourth Lagrangean Relaxation $LR4$ is obtained if we relax constraints (S) with multipliers u_j, $\forall j \in J$. The resulting Lagrangean subproblem decomposes into I independent 0-1 multi-knapsack subproblems with logical constraints that are strengthened to Setup Multiknapsacks by appending constraints $(L2)$. These are harder to solve than the ones of $LR3$, because they have one extra constraint.

5.2. LAGRANGEAN SUBSTITUTION LS

We introduce copies z'_{ij}, y'_{ik} of variables z_{ij}, y_{ik}, and append the following aggregate copy constraints to the augmented $(DSP1)$ model:

$$(ACO) \qquad \sum_{k \in K_{-A}} p_k y'_{ik} + \sum_{j \in J} g_j z'_{ij} = \sum_{k \in K_{-A}} p_k y_{ik} + \sum_{j \in J} g_j z_{ij}$$

We also duplicate constraints $(L1)$ as $(L1')$ and write constraints (W), (CV) and (VV) as (W'), (CV') and (VV') in the copy variables. We dualize constraints (ACO) with multipliers u_i, thus obtaining Lagrangean Substitution $LS(u)$ that decomposes as follows:

$$V(LS(u)) = V(LS^1(u)) - \sum_{i \in I} V(LS_i^2(u))$$

where $LS^1(u)$ is a minimization problem in the original z_{ij}, y_{ik} variables:

$LS^1(u)$

$$\text{Minimize} \quad \sum_{i \in I} \sum_{k \in K_{-A}} (e_k + u_i p_k) y_{ik} + \sum_{i \in I} \sum_{j \in J} (c_{ij} + u_i g_j) z_{ij}$$

subject to

$$(S) \qquad \sum_{i \in I} z_{ij} = 1, \quad \forall j \in J$$

$$(L1) \qquad z_{ij} \leqslant y_{ik}, \quad \forall i \in I,\ k \in K_{-A},\ j \in J$$

$$(I1) \qquad z_{ij} \in \{0, 1\}, \quad \forall i \in I,\ j \in J$$

$$(I2) \qquad y_{ik} \in \{0, 1\}, \quad \forall i \in I,\ k \in K_{-A}$$

and $LS_i^2(u)$ is a maximization problem for each vehicle $i \in I$ in the copy z'_{ij}, y'_{ik} variables, that has a Setup Multiknapsack structure. After appending constraints $(L2)$ to it, $LS_i^2(u)$ is:

$LS_i^2(u)$

Maximize $\sum_{k\in K_{-A}} u_i p_k y'_{ik} + \sum_{j\in J} u_i g_j z'_{ij}$

subject to

(W'_i) $\sum_{k\in K_{-A}} p_k y'_{ik} + \sum_{j\in J} g_j z'_{ij} \leqslant b_i$

(CV'_i) $\sum_{j\in J_k} t_{jk} z'_{ij} \leqslant q_k y'_{ik}\,, \quad \forall k\in K_{-A}$

$(L1'_i)$ $z'_{ij} \leqslant y'_{ik}\,, \quad \forall j\in J_k,\ k\in K_{-A}$

$(L2'_i)$ $\sum_{j\in J_k} z'_{ij} \geqslant y'_{ik}\,, \quad \forall k\in K_{-A}$

(VV'_i) $\sum_{k\in K_{-A}} r_k y'_{ik} + \sum_{j\in J_A} t_{jA} z'_{ij} \leqslant d_i$

$(I1'_i)$ $z'_{ij} \in \{0,1\}\,, \quad \forall j\in J$

$(I2'_i)$ $y'_{ik} \in \{0,1\}\,, \quad \forall k\in K_{-A}$

The Lagrangean dual is a maximization over multipliers u:

$$LS = \max\{V(LS(u)) | (u)\}$$

and is optimized with a subgradient procedure adapted from the one for solving $LR3$ as follows:

- LS multipliers u_i are unrestricted in sign and get their initial values from the optimal values of the dual variables of constraints (ACO) in the LP relaxation of the expanded (DSP1) model.
- Lagrangean subproblems are $LS^1(u)$ and $LS_i^2(u)$ for each $i\in I$.
- Modified subgradient vectors $U^l = \{U_i^l\}_{\forall i\in I}$ of Camerini et al. [8] are based on constraints (ACO).

5.3. LAGRANGEAN DECOMPOSITION LD

We introduce copies z'_{ij}, y'_{ik} of variables z_{ik}, y_{ik}, and append the following copy constraints to the augmented $(DSP1)$ model:

(CZ) $z'_{ij} = z_{ij}$

(CY) $y'_{ik} = y_{ik}$

We also duplicate constraints $(L1)$ as $(L1')$ and write constraints (W), (CV) and (VV) as (W'), (CV') and (VV') in the copy variables. We dualize constraints (CZ) with multipliers v_{ij} and constraints (CY) with multipliers u_{ik} thus obtaining Lagrangean Decomposition $LD(u,v)$ that decomposes as follows:

$$V(LD(u,v)) = V(LD^1(u,v)) - \sum_{i\in I} V(LD_i^2(u,v))$$

where $LD^1(u,v)$ is a minimization problem in the original z_{ij}, y_{ik} variables:

$LD^1(u,v)$

$$\text{Minimize} \quad \sum_{i\in I}\sum_{k\in K_{-A}} (e_k + u_{ik})y_{ik} + \sum_{i\in I}\sum_{j\in J}(c_{ij}+v_{ij})z_{ij}$$

subject to

$$(S) \qquad \sum_{i\in I} z_{ij} = 1\,, \quad \forall j\in J$$

$$(L1) \qquad z_{ij} \leqslant y_{ik}\,, \quad \forall i\in I,\ k\in K_{-A},\ j\in J$$

$$(I1) \qquad z_{ij}\in\{0,1\}\,, \quad \forall i\in I,\ j\in J$$

$$(I2) \qquad y_{ik}\in\{0,1\}\,, \quad \forall i\in I,\ k\in K_{-A}$$

and $LD_i^2(u,v)$ is a maximization problem for each vehicle $i\in I$ in the copy z'_{ij}, y'_{ik} variables, that has a Setup Multiknapsack structure. After appending constraints $(L2)$ to it, $LD_i^2(u,v)$ is:

$LD_i^2(u,v)$

$$\text{Maximize} \quad \sum_{k\in K_{-A}} u_{ik}y'_{ik} + \sum_{j\in J} v_{ij}z'_{ij}$$

subject to

$$(W'_i) \qquad \sum_{k\in K_{-A}} p_k y'_{ik} + \sum_{j\in J} g_j z'_{ij} \leqslant b_i$$

$$(CV'_i) \qquad \sum_{j\in J_k} t_{jk}z'_{ij} \leqslant q_k y'_{ik}\,, \quad \forall k\in K_{-A}$$

$$(L1'_i) \qquad z'_{ij}\leqslant y'_{ik}\,, \quad \forall j\in J_k,\ k\in K_{-A}$$

$$(L2'_i) \qquad \sum_{j\in J_k} z'_{ij} \geqslant y'_{ik}\,, \quad \forall k\in K_{-A}$$

$$(VV'_i) \qquad \sum_{k\in K_{-A}} r_k y'_{ik} + \sum_{j\in J_A} t_{jA} z'_{ij} \leqslant d_i$$

$$(I1'_i) \qquad z'_{ij}\in\{0,1\}\,, \quad \forall j\in J$$

$$(I2'_i) \qquad y'_{ik}\in\{0,1\}\,, \quad \forall k\in K_{-A}$$

The Lagrangean dual is a maximization over multipliers u:

$$LD = \max\{V(LD(u,v))|(u),(v)\}$$

and is optimized with a subgradient procedure adapted from the one for solving $LR3$ as follows:

- LD multipliers u_{ik} and v_{ij} are unrestricted in sign and get as their initial values the optimal dual variables of constraints (CY) and (CZ) in the LP relaxation of the expanded (DSP1) model.

- Lagrangean subproblems are $LD^1(u, v)$ and $LD_i^2(u, v)$ for each $i \in I$.
- Modified subgradient vectors $U^l = \{U_{ik}^l\}_{\forall i \in I, k \in K_{-A}}$ and $V^l = \{V_{ij}^l\}_{\forall i \in I, j \in J}$ of Camerini et al. [8] are based on constraints (CV) and (CZ).

5.4. LAGRANGEAN HEURISTIC

We developed a simple Lagrangean heuristic that finds feasible solutions from the possibly infeasible Lagrangean ones. We set y_{ik} to optimal Lagrangean values, and solve a reduced (DSP1), in the remaining free z_{ij} vaiables only, to optimality. If for all $i \in I$ and $k \in K_{-A}$ y_{ik} is 0, then, unless no customer order has refrigerated or frozen items, no feasible solution can be found. In this case, y_{ik} for all i and k are set to 1. The heuristic is applied at each subgradient iteration. However, it is not guaranteed to find a feasible solution at each subgradient iteration. If a solution is found, boxes for non-ambient temperature items are possibly added on a vehicle while no customers having ordered the corresponding item types are assigned to the vehicle. In this case, the feasible solution is improved by removing these boxes from the vehicle and subtracting their cooling costs from the feasible solution value.

Lagrangean Relaxations $LR1$ through $LR4$ provide only one set of y_{ik} optimal Lagrangean values, therefore only one feasible solution can be obtained with the above heuristic based on these schemes. LS and LD provide two sets of y_{ik} and y'_{ik} optimal Lagrangean values, therefore repeated application of the heuristic for each set can yield two feasible solutions at each subgradient iteration.

6. Computational experience with the Lagrangean approximations of (DSP1)

Subgradient procedures for all Lagrangean schemes based on (DSP1) were implemented in GAMS and tested on a Hewlett-Packard J2240 workstation with CPLEX 6.5. We used an initial subgradient step size factor $\sigma^0 = 2$ that was halved every 25 iterations, if the best lower bound found failed to improve in each one of them. For all problems except for Large we could obtain the optimal solution, therefore we used it as the value of the best known feasible solution in the calculation of the upper bound; for Large, we used the best solution found. As initial Lagrangean multipliers we used the optimal dual prices of the appropriate constraints of the linear programming relaxation of the model.

Table 10 shows, for the random problems and the small real ones, the lower bounds obtained after 10 subgradient iterations with each of the Lagrangean bounding schemes, the feasible solutions obtained with the corresponding heuristic and the CPU times of each run. For Large, results are shown after 100 subgradient iterations. Numbers of random problems are the same as earlier in the article. CPU

Table 10. Comparison of lower bounds, feasible solutions and computation times

Random prob.	Lower bounds						
	LP	*LR*1	*LR*2	*LR*3	*LR*4	*LS*	*LD*
1	424.257	424.257	424.257	424.257	424.257	424.257	424.257
2	421.064	421.064	421.064	421.101	421.468	419.198	421.064
3	447.297	447.297	447.297	450.088	447.963	426.176	447.297
4	464.615	464.615	464.615	466.061	474.753	421.755	465.552
5	494.965	494.965	494.654	514.890	515.600	409.016	495.502
6	440.933	440.933	440.933	444.492	440.933	429.856	440.933
7	456.785	456.785	456.785	457.957	456.785	425.413	456.785
9	434.880	434.880	434.880	438.577	437.118	423.726	434.879
10	448.061	448.061	448.061	450.999	448.061	422.748	448.061
11	501.602	501.602	501.602	508.925	502.561	459.272	501.816
12	242.013	242.013	242.013	242.388	243.308	241.794	242.013
13	318.950	318.950	318.950	318.950	318.950	318.045	318.045
15	561.069	561.069	561.069	566.238	561.069	492.663	561.069
16	672.122	672.121	672.121	677.918	672.122	548.237	672.122
17	780.654	780.654	780.654	787.410	780.740	591.409	780.654
Real 1	161.729	161.729	161.729	161.729	161.729	152.293	161.729
Real 2	161.729	161.729	161.729	161.729	161.729	157.717	161.729
Real 3	247.934	249.064	257.262	247.934	267.424	219.455	247.934
Large	997.485	1047.770	1048.522	1035.730	1118.463	1093.200	904.960
	Opt.	Feasible solutions with Lagrangean heuristic					
1	424.257	468.481	468.481	468.481	468.481	424.257	468.481
2	421.765	421.765	421.765	421.765	421.765	421.765	421.765
3	468.950	487.466	605.263	468.950	468.950	487.466	477.746
4	480.852	480.852	615.119	697.136	550.133	494.339	494.339
5	551.929	551.929	609.413	825.776	618.322	601.493	569.134
6	475.023	475.023	559.185	475.023	475.023	519.126	503.085
7	466.214	481.357	624.018	466.214	466.214	511.356	466.214
9	467.073	539.389	467.073	539.389	467.073	538.690	467.073
10	483.666	486.206	519.794	486.206	486.206	578.626	483.666
11	548.379	548.379	548.379	3066.76	3066.76	568.88	548.379
12	243.766	243.766	243.766	250.803	243.766	243.766	243.766
13	318.950	2767.53	2767.53	2767.53	2767.53	318.950	2767.53
15	616.141	637.466	616.141	3267.53	749.635	637.466	618.950
16	746.141	943.766	897.472	3517.53	808.248	808.248	767.167
17	876.141	937.466	1077.47	3767.53	1143.77	937.466	3767.53
Real 1	164.444	164.444	164.444	164.444	164.444	164.444	164.444
Real 2	164.444	164.444	164.444	164.444	164.444	164.444	164.444
Real 3	295.378	1034.90	1034.90	1034.90	1034.90	1034.90	1034.90
Large	1190.578	N/A	N/A	N/A	1218.956	N/A	N/A

times of Lagrangean schemes are often greater than that required to solve a problem to optimality, except for Large.

*LR*1 and *LR*2 give lower bounds that are as good as the linear programming bound, although their subproblems do not have the integrality property. *LR*3 and

Table 10. Continued

HP J2240 CPU seconds							
1	0.03	0.55	0.80	0.80	0.80	0.12	1.25
2	0.04	1.09	1.11	1.32	1.30	2.28	2.11
3	0.27	1.22	1.30	1.48	1.50	4.51	2.62
4	1.06	1.25	1.75	2.00	1.74	4.17	4.00
5	1.92	0.96	2.98	0.42	5.50	5.93	3.96
6	0.32	0.84	1.14	1.66	1.44	2.33	1.18
7	0.30	0.99	1.21	0.76	1.60	3.80	2.33
9	0.22	1.32	1.49	1.72	1.55	2.94	2.35
10	0.38	1.37	1.35	1.63	1.68	4.88	2.48
11	0.59	1.38	1.79	1.80	2.00	7.48	1.60
12	0.07	1.10	1.37	1.36	1.35	5.87	2.17
13	0.05	1.30	1.30	1.31	1.30	11.39	1.91
15	0.40	1.17	1.67	1.70	0.85	6.02	2.59
16	0.41	1.23	1.70	0.59	2.26	5.58	1.92
17	0.46	1.37	1.73	1.80	1.46	6.42	5.59
Real 1	0.09	2.11	2.85	1.96	2.66	6.02	4.99
Real 2	0.08	2.44	2.76	1.66	2.04	4.88	4.17
Real 3	0.96	3.67	7.11	2.86	5.79	9.78	10.35
Large	$>$100 K	775.75	3846.19	826.15	2046.51	4992.08	18790.35

*LR*4 yield improved bounds but their CPU times are higher too. *LS* yields the worst bounds of all schemes, primarily because the optimal dual LP variables do not give a good starting bound. *LD* gives the linear programming bound in most cases, but this is expected given that the number of subgradient iterations is small and that the subgradient algorithm for *LD* converges slowly due to the large number of multipliers. The heuristic gives very good solutions for the random and small real problems; in most cases it finds the optimal solution at the first 10 subgradient iterations. Large is a little harder to solve; only coupled with *LR*4 did the heuristic give a feasible solution for this instance, albeit a good one and within a reasonable CPU time.

7. Conclusions

In this article we present two models for scheduling deliveries in vehicles with multiple compartments. Unlike the standard vehicle routing problem that can be solved by generalized assignment heuristics, in deliveries with multiple compartments both weight and volume limitations should be considered, the former for the whole vehicle and the latter for both the vehicle and each compartment. A useful characteristic of the models is that they find the minimal number of vehicles required for a feasible delivery schedule. Very interesting computational and modeling features of the models were explored with experiments using randomly generated and real data.

Preliminary experiments with several Lagrangean Relaxations, a Substitution and a Decomposition of the first model have revealed tradeoffs between tightness of obtained bounds and computation times. We have seen that keeping and not relaxing the compartment volume capacity constraints improves the lower bounds by much, at least in our data sets. Also, a heuristic based on these schemes can give very good feasible solutions.

The models proposed here are far from capturing all complexities of a real application. In the case of deliveries to convenience stores such complexities can be the following: (i) By law, a driver cannot work more than a certain number of hours each day, thus there is a limit to the duration of trips. (ii) Travel times between stops are not known a priori, but depend on traffic, weather conditions, etc. Unloading times at each customer's location are also not known a priori. (iii) Customers require that deliveries are done within certain time windows. Addressing these complexities could be the subject of a future study.

Appendix A. Generating random test problems

The procedure we used in generating random test problems was adapted from Fisher and Jaikumar [24] to incorporate different item types and compartments. It consists of the following steps:

- Set random problem parameters: Specify random number generator seed, number of customers, number of vehicles, their (equal) weight and volume capacities, number of compartments, their weight and volume, as well as volume capacities, maximum and minimum distance of customers from the depot, customers whose order consists of items of all three types as a fraction of the total number of customers.
- Randomly generate locations for customers in polar coordinates, setting the depot at the origin.
- Randomly generate an order for each customer, indicating the type of items (frozen, refrigerated, ambient temperature) that it consists of, as well as the aggregate weight and volume of each item type.
- Set seed points and compute delivery costs based on the locations of customers and the depot.

Acknowledgements

We are greatly indebted to John Borst and Michael Sadusky [5] for devoting a significant amount of their time explaining the distribution operation and correcting a lot of our misconceptions. Research was partially supported by NSF grants DDM-9014901 and DMI-9900183, and a grant from the Research Foundation of the University of Pennsylvania and GAMS Dev. Corp.

References

[1] Afentakis, P. and Gavish, B. (1986), Optimal lot-sizing algorithms for complex product structures, *Operations Research* 34(2): 237–249.

[2] Balinski, M.L. and Quandt, R.E. (1964), On an integer program for a delivery problem, *Operations Research* 12: 300–304.

[3] Belardo, S., Duchessi, P. and Seagle, J.P. (1985), Microcomputer graphics in support of vehicle fleet routing, *Interface* 15(6): 84–92.

[4] Bell, W.J., Dalberto, L.M., Fisher, M.L., Greenfield, A.J., Jaikumar, R., Kedia, P., Mack, R.G. and Prutzman, P.J. (1983), Improving the distribution of industrial gases with an online computerized routing and scheduling optimizer, *Interface* 13(6): 4–23.

[5] Borst, J. and Sadusky, M. (1992), Private communication.

[6] Bramel, J., Coffman Jr., E.G., Shor, P.W. and Simchi-Levi, D. (1992), Probabilistic analysis of the capacitated vehicle routing problem with unsplit demands, *Operations Research* 40(6): 1095–1106.

[7] Brooke, A., Kendrick, D. and Meeraus, A. (1992), *GAMS: A User's Guide, Release 2.25*.

[8] Camerini, P.M., Fratta, L. and Maffioli, F. (1975), On improving relaxation methods by modified gradient techniques, *Mathematical Programming Study* 3: 26–34.

[9] Chajakis, E.D. and Guignard, M. (1994), Exact algorithms for the setup knapsack problem. *INFOR (Information Systems and Operational Research)* 32(3): 124–142. (Special issue on Knapsack, Packing and Cutting. Guest editor: Silvano Martello).

[10] Christophides, N. (1985), Vehicle routing. In: Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G., Shmoys, D.B. (1985), *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, John Wiley and Sons Ltd. pp. 431–465.

[11] Christophides, N. and Eilon, S. (1969), An algorithm for the vehicle dispaching problems, *Operations Research Quarterly* 20.

[12] Clarke, G. and Wright, J., (1964), Scheduling of vehicles from a central depot to a number of delivery points, *Operations Research* 12(4): 568–581.

[13] Delaney, R.V. (2001), Annual State of Logistics Report, 11th Edition. 6-month update. Cass Information Systems, http://www.cassinfo.com.

[14] Delaney, R.V. (2001), Private communication.

[15] Desrochers, M., Desrosiers, J. and Solomon, M. (1989), A new optimization algorithm for the vehicle routing problem with time windows. Technical report, GÉRAD, Montreal, Canada.

[16] Desrochers, M., Lenstra, J.K., Savelsbergh, M.W.P. and Soumis, F. (1988), Vehicle routing with time windows: Optimization and approximation. In: Golden, B.L. and Assad, A.A. (eds.), *Vehicle Routing: Methods and Studies*, Elsevier Science Publishers, Amsterdam, pages 65–84.

[17] Desrosiers, J., Sauvé, M. and Soumis, F. (1988), Lagrangian Relaxation methods for solving the minimum fleet size multiple traveling salesman problem with time windows, *Management Science* 34(8): 1005–1022.

[18] Desrosiers, J., Soumis, F. and Desrochers, M. (1984), Routing with time windows by column generation, *Networks* 14: 545–565.

[19] Dror, M., Laporte, G. and Trudeau, P. (1989), Vehicle routing with stochastic demands, *Transportation Science* 23: 166–176.

[20] Fisher, M.L. (1981), The Lagrangian relaxation method for solving integer programming problems, *Management Science* 27(1): 1–17.

[21] Fisher, M.L. (1985), An applications oriented guide to Lagrangian relaxation, *Interfaces* 15(2): 10–21.

[22] Fisher, M.L. (1994), Optimal solution of vehicle routing problems using minimum k-trees, *Operations Research* 42(4): 626–642.

[23] Fisher, M.L. (1995), Vehicle routing. In: *Handbooks in Operations Research and Management Science*, Chapter 1 in Vol. 8 on *Network Routing*, Elsevier Science, Amsterdam, pp. 1–33.

[24] Fisher, M.L. and Jaikumar, R. (1981), A generalized assignment heuristic for vehicle routing, *Networks* 11: 109–124.

[25] Foster, B.A. and Ryan, D.M. (1976), An integer programming approach to the vehicle scheduling problem, *Operations Research* 27: 367–384.

[26] Freville, A. and Plateau, G. (1993), An exact search for the solution of the surrogate dual for the 0-1 bidimensional knapsack problem, *European Journal of Operational Research* 68: 413–421.

[27] Gavish, B. and Pirkul, H. (1985), Efficient algorithms for solving multiconstraint zero-one knapsack problems to optimality, *Mathematical Programming* 31(1): 78–105.

[28] Gavish, B. and Pirkul, H. (1991), Algorithms for the multi-resource generalized assignment problem, *Management Science* 37(6): 695–713.

[29] Geoffrion, A.M. (1974), Lagrangean Relaxation for integer programming, *Mathematical Programming Study* 2: 82–114.

[30] Gillet, B. and Miller, L. (1974), A heuristic algorithm for the vehicle dispatch problem, *Operations Research* 22: 240–349.

[31] Golden, B.L. and Assad, A.A. (1986), Perspectives on vehicle routing: Exciting new developments, *Operations Research* 14(5): 803–810.

[32] Guignard, M. (1993), Solving Makespan minimization problems with Lagrangean decomposition, *Discrete Applied Mathematics* 42: 17–29.

[33] Guignard, M. and Spielberg, K. (1981), Logical reduction methods in zero-one programming: Minimal preferred variables, *Operations Research* 29(1): 49–74.

[34] Guignard, M. and Spielberg, K. (1992), Double contraction and branch-and-bound for the IRPM forestry problem. Working paper 92-11-04, Decision Sciences Department, The Wharton School, University of Pennsylvania, Philadelphia, PA 19104.

[35] Haimovich, M. and Rinnooy Kan, A.H.G. (1985), Bounds and heuristics for capacitated routing problems, *Mathematics of Operations Research* 10(4): 527–542.

[36] Held, M. and Karp, R.M. (1970), The traveling salesman problem and minimum spanning trees, *Operations Research* 18: 1138–1162.

[37] Held, M. and Karp, R.M. (1971), The traveling salesman problem and minimum spanning trees: Part ii, *Mathematical Programming* 1: 6–25.

[38] Kolen, A.W.J., Rinnooy Kan, A.H.G. and Trienekens, H.W.J.M. (1987), Vehicle routing with time windows, *Operations Research* 35(2): 266–273.

[39] Laporte, G., Louveaux, F. and Mercure, H. (1989), Models and exact solutions for a class of stochastic location-routing problems, *European Journal of Operational Research* 39: 71–78.

[40] Laporte, G., Louveaux F. and Mercure, H. (1992), The vehicle routing problem with stochastic travel times, *Transportation Science* 26: 161–170.

[41] Laporte, G., Nobert, Y. and Desrochers, M. (1985), Optimal routing under capacity and distance restrictions, *Operations Research* 33: 1050–1073.

[42] Lee, H. and Guignard, M. (1996), A hybrid bounding procedure for the workload allocation problem on parallel unrelated machines with setups, *Journal of the Operational Research Society* 47: 1247–1261.

[43] Miller, D.M. (1987), An interactive computer-aided ship scheduling system, *European Journal of Operational Research* 32(3): 363–379.

[44] Sen, S. and Sherali, H.D. (1986), A class of convergent primal-dual subgradient algorithms for decomposable convex programs, *Mathematical Programming* 35: 279–297.

[45] Shapiro, J.F. (1979), A survey of Lagrangean techniques for discrete optimization, *Annals of Discrete Mathematics* 5: 113–138.

[46] Spielberg, K. (1971), Minimal preferred variable reduction for zero-one programming. Rep. 320-3013, IBM Philadelphia Scientific Center, Philadelphia, PA.

[47] Weinstein, S. (1991), How retailers can maximize space, *Progressive Grocer (Part 1)* 70(4): 101–104.