An Exact Algorithm for the Vehicle Routing Problem with Backhauls

PAOLO TOTH AND DANIELE VIGO

D.E.I.S., Università di Bologna, Viale Risorgimento 2, 40136 Bologna, Italy

The Vehicle Routing Problem with Backhauls is an extension of the capacitated Vehicle Routing Problem where the customers' set is partitioned into two subsets. The first is the set of Linehaul, or Delivery, customers, while the second is the set of Backhaul, or Pickup, customers. The problem is known to be NP-hard in the strong sense and finds many practical applications in distribution planning. In this paper we consider, in a unified framework, both the symmetric and the asymmetric versions of the vehicle routing problem with backhauls, for which we present a new integer linear programming model and a Lagrangian lower bound which is strengthened in a cutting plane fashion. The Lagrangian lower bound is then combined, according to the additive approach, with a lower bound obtained by dropping the capacity constraints, thus obtaining an effective overall bounding procedure. A branch-and-bound algorithm, reduction procedures and dominance criteria are also described. Computational tests on symmetric and asymmetric instances from the literature, involving up to 100 customers, are given, showing the effectiveness of the proposed approach.

We consider the Vehicle Routing Problem with Backhauls (VRPB), also known as the Linehaul–Backhaul Problem, an extension of the capacitated Vehicle Routing Problem (VRP) where the customers' set is partitioned into two subsets. The first subset contains the *linehaul customers*, each requiring a given quantity of product to be delivered. The second subset contains the *backhaul customers*, where a given quantity of inbound product must be picked up.

This customer partition is extremely frequent in practical situations. A common example is that of the grocery industry, where supermarkets and shops are the linehaul customers, and grocery suppliers are the backhaul customers. In recent years it has been widely recognized that in this mixed distribution/collection context a significant saving in terms of transportation costs can be achieved by visiting backhaul customers in distribution routes (see, e.g., GOLDEN et al., 1985).

The problem calls for the determination of a set of routes for a given set of vehicles visiting all customers and such that: i) each vehicle performs exactly one route; ii) for each route the total load associated with linehaul and backhaul customers does not exceed, separately, the vehicle capacity; iii) in each route the backhaul customers, if any, are visited after all linehaul customers; iv) routes containing only backhaul customers are not allowed; v) the total distance traveled is minimized. The precedence constraint (iii) is motivated by the fact that in many practical applications linehaul customers have a higher priority. Moreover, vehicles are often rear-loaded, hence the on-board load rearrangement required by a "mixed" service is difficult, or even impossible, to carry out at customer locations.

In this paper we consider the two basic versions of the problem: symmetric and asymmetric. The symmetric version (which, for short, is hereafter indicated as VRPB) arises when the distance between each pair of locations is the same in both directions, whereas we have the asymmetric version (indicated as AVRPB) when this assumption does not hold. All previous works from the literature have considered only the symmetric version of VRPB; hence, for the sake of brevity, throughout this paper we normally refer to the symmetric version only. However, since in our approach we model the VRPB by reformulating it as an asymmetric problem, all the lower bounds and the solution procedures described hereafter, unless explicitly stated, are valid for the AVRPB as well.

Transportation Science
Vol. 31, No. 4, November 1997

0041-1655/97/3104-0372 $01.25

The VRPB (resp. AVRPB) can be formulated as the following graph theory problem. Let $G' = (V'_0, A')$ be a complete undirected (resp. directed) graph, with vertex set $V'_0 := \{0\} \cup \{1, \ldots, n\} \cup \{n+1, \ldots, n+m\}$. Subsets $L = \{1, \ldots, n\}$ and $B = \{n+1, \ldots, n+m\}$ correspond to linehaul and backhaul vertex sets, respectively. Each vertex is associated with a customer location, whereas vertex $\{0\}$ corresponds to the *depot*. A nonnegative *demand* d_j, to be delivered or collected is associated with each vertex j in $V' = V'_0 \setminus \{0\}$, and the depot is associated with a fictitious demand $d_0 = 0$. In the depot, we have K identical vehicles with a given *capacity*, D. Let c'_{ij} be the nonnegative *distance*, or *cost*, associated with arc$(i, j) \in A'$, $c_{ii} = +\infty$ for each $i \in V'_0$. The VRPB (or AVRPB) then consists of finding a minimum-cost collection of K simple *circuits* (vehicle routes) such that:

i. each vehicle performs exactly one circuit;
ii. each circuit visits vertex 0;
iii. each vertex $j \in V'$ is visited by exactly one circuit;
iv. the total demands of the linehaul and backhaul vertices visited by a circuit do not exceed, separately, the vehicle capacity D;
v. in each circuit all the linehaul vertices precede the backhaul vertices, if any.

The objective is to minimize the total routing cost, defined as the sum of the costs of the arcs belonging to the circuits.

As usually assumed in the literature, circuits containing only backhaul customers are not allowed. However, the solution approach we propose may be extended to consider the case when this assumption is not present. Moreover, observe that precedence constraint (v) introduces an implicit orientation of the "mixed" vehicle routes, i.e., the routes which visit both linehaul and backhaul vertices.

Let K_L and K_B denote the minimum number of vehicles needed to serve all the linehaul and backhaul customers, respectively. These values can be obtained by solving the *Bin Packing Problem* (BPP) instances associated with the corresponding customer subsets. In these instances the bin capacity is equal to D and each item j has a weight equal to the demand d_j of the corresponding customer. Note that, in spite of the fact that BPP is NP-hard in the strong sense, instances with hundreds of items can be optimally solved very effectively (see, e.g., MARTELLO and TOTH (1990)). In order to ensure feasibility, we assume that K (number of available vehicles) is not smaller than the minimum number of vehicles needed to serve all the customers, i.e., $K \geq \max\{K_L, K_B\}$.

VRPB is NP-hard in the strong sense, since it generalizes the well-known capacitated VRP, arising when $B = \varnothing$ (see, for example, FISCHETTI, TOTH and VIGO (1994)).

Several heuristic algorithms for the solution of VRPB have been presented in the literature so far. DEIF and BODIN (1984) proposed an extension of the well-known Clarke–Wright VRP heuristic, where the saving of the arcs connecting linehaul to backhaul customers is modified to delay the formation of "mixed" routes. Different extensions of the Clarke–Wright algorithm have been presented by CASCO, GOLDEN and WASIL (1988), and Golden, et al. (1985) (in this latter paper the precedence constraint between linehaul and backhaul customers is not present). GOETSCHALCKX and JACOBS-BLECHA (1989) proposed an algorithm that builds initial routes with customers of the same type by using a *spacefilling curves* heuristic; these routes are then merged to form the final set of vehicle routes. It should be noted that the solutions obtained with both the Deif–Bodin and the spacefilling heuristics could use more than K vehicles (hence leading to infeasible solutions). More recently, GOETSCHALCKX and JACOBS-BLECHA (1993) described an extension to VRPB of the Fisher–Jaikumar VRP heuristic. TOTH and VIGO (1996) proposed a different cluster-first–route-second heuristic for VRPB, which starts from a (possibly infeasible) solution, and tries to improve it through a local search procedure based on inter-route and intra-route arc exchanges.

YANO et al. (1987) proposed an exact, set-covering-based algorithm for the special case of VRPB in which the number of customers of each type in a circuit is not greater than 4. The variant of VRPB where time window constraints are present has also been considered in the literature (see, e.g., KONTOVRADIS and BARD (1995), DUHAMEL, POTVIN and ROUSSEAU (1996), and GÉLINAS et al. (1995)).

In this paper we present the first exact approach for the solution of the VRPB with both symmetric and asymmetric cost matrices. We first give a new integer linear programming model where the VRPB is reformulated as an asymmetric problem. The model is used to derive a Lagrangian lower bound, based on projection of the solution space, which requires the determination of the shortest spanning arborescences with fixed degree at the depot vertex, and the optimal solution of min-cost flow problems. The Lagrangian lower bound is strengthened in a cutting-plane fashion by separating valid inequalities, and is combined, according to the *additive approach* proposed by FISCHETTI and TOTH (1989), with a lower bound obtained by dropping the capacity constraints. A branch-and-bound algorithm,

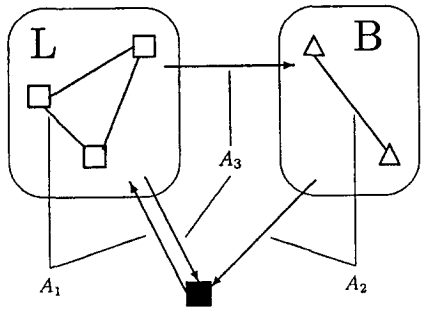

Fig. 1. The arcs set of graph G.

which makes use of reduction procedures, dominance criteria, feasibility checks, and a heuristic algorithm, are also presented. The computational tests performed on several symmetric and asymmetric instances from the literature, involving up to 100 customers, show the effectiveness of the proposed approaches.

The paper is organized as follows. Section 1 gives the new integer linear programming model for VRPB. Section 2 presents different relaxations of VRPB, the Lagrangian lower bound and the overall additive bounding procedure. The implemented branch-and-bound algorithm is described in Section 3, while computational results are given in Section 4.

1. INTEGER LINEAR PROGRAMMING MODEL

IN THE FOLLOWING we present a new integer linear programming model for VRPB. The model is based on the reformulation of VRPB as an asymmetric problem, thus it is valid for AVRPB as well.

Let us define $L_0 := L \cup \{0\}$ and $B_0 := B \cup \{0\}$. Let $G = (V_0, A)$ be a directed graph obtained from G' by defining $V_0 = V'_0$, $V = V_0 \setminus \{0\}$, and $A = A_1 \cup A_2 \cup A_3$, where

$$A_1 := \{(i,j) \in A' : i \in L_0, j \in L\},$$

$$A_2 := \{(i,j) \in A' : i \in B, j \in B_0\},$$

$$A_3 := \{(i,j) \in A' : i \in L, j \in B_0\}.$$

In other words, as illustrated in Figure 1, the arc set A can be partitioned into three disjoint subsets. The first subset, A_1, contains all the arcs from the depot and linehaul vertices to linehaul vertices. The second subset, A_2, contains all the arcs from backhaul vertices to backhaul vertices and the depot. The third subset, A_3, contains the so-called "interface arcs", connecting linehaul vertices to backhaul vertices or the depot. Note that A does not contain arcs that cannot belong to a feasible solution. In fact, no arc from a backhaul to a linehaul vertex, or from the depot to a backhaul vertex can belong to a feasible solution to VRPB, either because of the precedence constraint or because routes with only backhaul vertices are not allowed. Moreover, a cost $c_{ij} = c'_{ij}$ is associated with each arc $(i,j) \in A$.

Let $\mathcal{L}$ (resp. $\mathcal{B}$) be the family of all subsets of vertices in L (resp. B), and let $\mathcal{F} = \mathcal{L} \cup \mathcal{B}$. For each $S \in \mathcal{F}$, let $\sigma(S)$ be the minimum number of vehicles needed to serve all the customers in S. This value may be computed as the optimal solution value of the BPP with item set S and bin capacity equal to D. For each $i \in V_0$ let us define $\Gamma_i^+ = \{j : (i,j) \in A\}$ (i.e., the *forward star* of i) and $\Gamma_i^- = \{j : (j,i) \in A\}$ (i.e., the *backward star* of i). An integer linear programming formulation of VRPB and AVRPB is then:

$$\text{(P)} \qquad v(P) = \min \sum_{(i,j) \in A} c_{ij} x_{ij} \tag{1}$$

subject to

$$\sum_{i \in \Gamma_j^-} x_{ij} = 1 \quad \text{for each } j \in V \tag{2}$$

$$\sum_{j \in \Gamma_i^+} x_{ij} = 1 \quad \text{for each } i \in V \tag{3}$$

$$\sum_{i \in \Gamma_0^-} x_{i0} = K \tag{4}$$

$$\sum_{j \in \Gamma_0^+} x_{0j} = K \tag{5}$$

$$\sum_{j \in S} \sum_{i \in \Gamma_j^- \setminus S} x_{ij} \geqslant \sigma(S) \quad \text{for each } S \in \mathcal{F} \tag{6}$$

$$\sum_{i \in S} \sum_{j \in \Gamma_i^+ \setminus S} x_{ij} \geqslant \sigma(S) \quad \text{for each } S \in \mathcal{F} \tag{7}$$

$$x_{ij} \in \{0, 1\} \quad \text{for each } i, j \in V_0 \tag{8}$$

where $x_{ij} = 1$ if and only if arc $(i,j) \in A$ is in the optimal solution. Equations (2), (4) and (3), (5) impose indegree and outdegree constraints for the customer and the depot vertices, respectively. The so-called *capacity-cut constraints*, (6) and (7), impose both the connectivity and the capacity constraints. Note that because of degree constraints (2)–(5), for any given S the left-hand sides of (6) and (7) are equal (i.e., the number of arcs entering S is equal to the number of arcs leaving it). Hence, if constraints (6) are imposed, then constraints (7) are redundant and vice-versa. Alternatively, an equivalent model for the VRPB is obtained by imposing (6) only for each $S \in \mathcal{L}$ and (7) only for each $S \in \mathcal{B}$. Note also that in (6) and (7), the value $\sigma(S)$ can be replaced by

any lower bound on the optimal solution of the associated BPP, e.g., $\lceil \Sigma_{j \in S} d_j/D \rceil$.

For the symmetric version of the problem the cost matrix associated with graph G is asymmetric due to the removal from G of the arcs connecting backhaul to linehaul vertices and the depot to backhaul vertices. However, the two submatrices corresponding to arcs with both endpoints in L_0 and in B, respectively, are symmetric.

2. RELAXATIONS

IN THIS SECTION three different relaxations for the VRPB are presented. We first describe a relaxation obtained by dropping the capacity-cut constraints, leading to the solution of a Transportation Problem (TP). Then we introduce a relaxation of the problem based on the projection of the feasible solution space, which requires the determination of shortest spanning trees (SST) or arborescences (SSA) and the optimal solution of min-cost flow problems. A Lagrangian lower bound is finally derived by including the relaxed degree constraints in the objective function. The lower bound is then strengthened in a cutting-plane fashion by adding some of the previously relaxed capacity-cut constraints. The separation of valid inequalities, and a subgradient optimization procedure for the determination of good Lagrangian multipliers are also discussed.

2.1 Relaxation Obtained by Dropping the Capacity-Cut Constraints

As proposed by LAPORTE, MERCURE, and NOBERT (1986) and by Fischetti, Toth, and Vigo (1994) for the VRP, we relax problem (1)–(8) by dropping capacity-cut constraints (6) and (7). We thus obtain a TP requiring the determination of a min-cost collection of circuits of G covering all the vertices in V once, and visiting the depot K times. This solution can be infeasible for VRPB because the total linehaul or backhaul customer demands of a circuit can exceed the vehicle capacity, and/or because of the possible existence of circuits not visiting the depot. Note that the precedence constraint between linehaul and backhaul vertices is satisfied because of the absence in G of arcs from backhaul to linehaul vertices or from the depot to backhaul vertices.

The solution of TP requires $O((n + m)^3)$ time through a transportation algorithm. However, in practice it is more effective to transform the problem into an *Assignment Problem* (AP) defined on the extended digraph $\bar{G}$ obtained by adding $K - 1$ copies of the depot to G (see Laporte, Mercure, and Nobert (1986), and Fischetti, Toth and Vigo (1994) for further details).

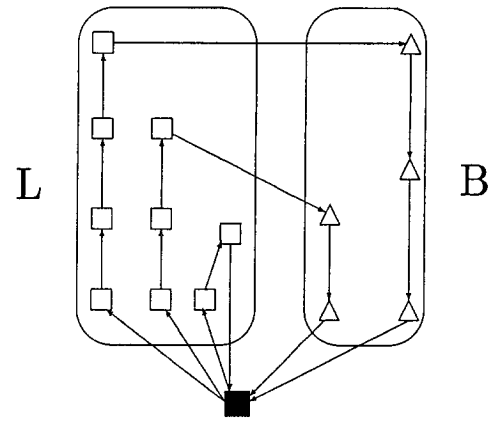

Fig. 2. Structure of a feasible solution to VRPB.

As expected, our computational experience showed that the quality of the lower bound provided by the TP relaxation is generally poor when symmetric instances are considered. However, this bound may be combined in an additive fashion, as shown in Section 2.6, with the other bounds proposed in the following, leading to an overall effective bound.

2.2 Relaxation Based on Projection

Let us consider the structure of the feasible solutions to VRPB. As illustrated in Figure 2, the arcs of any feasible solution can be divided into three groups, belonging to the previously defined subsets A_1, A_2, and A_3, respectively. Problem P can be relaxed by dividing it into three different subproblems, each relative to a different arc subset A_h, $h = 1, 2, 3$. A valid lower bound on $v(P)$ can be obtained by adding the three corresponding optimal solution values. This relaxation is used as the basis of the Lagrangian relaxation of VRPB described in the next section.

We now briefly describe the three resulting subproblems and discuss their mathematical model and solution. The first subproblem, corresponding to the arc set A_1, is that of determining a min-cost collection of K capacitated disjoint simple paths, starting from the depot and spanning all the linehaul vertices. This problem is NP-hard in the strong sense, and remains such even if we relax it by requiring the determination of a capacitated tree with fixed outdegree K at the depot, and spanning all the linehaul vertices (see PAPADIMITRIOU (1978), and MALIK and YU (1993)). To obtain a polynomially solvable problem, we further relax the subproblem by dropping the capacity constraints. This leads to the determination of an SST with fixed degree K at the depot vertex (K-SST). Indeed, this problem (called P_1) can be efficiently solved, in $O(n^2)$ time, e.g., by using the

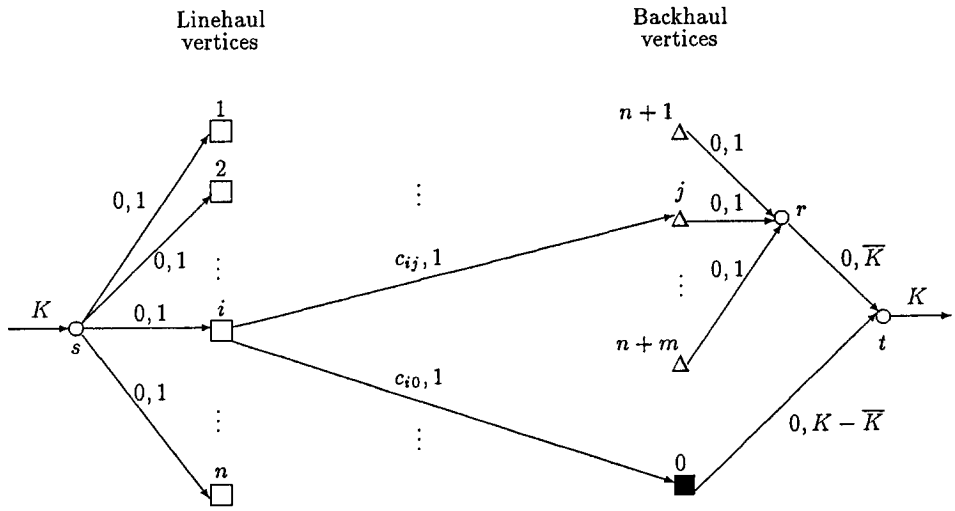

Fig. 3. The auxiliary layered network for the solution of the third subproblem.

algorithm proposed by GLOVER and KLINGMAN (1974). If we consider the asymmetric version of VRPB, the only difference is that instead of determining trees we must determine arborescences. Also in this case the capacitated SSA is an NP-hard problem (see, e.g., TOTH and VIGO (1995)). By dropping the capacity constraints, we obtain the problem of determining an SSA with fixed outdegree K at the depot vertex (K-SSA), which can be solved in $O(n^2)$ time, e.g., by using the algorithm by GABOW and TARJAN (1984). The cost matrix of the Lagrangian relaxation described in the next section is asymmetric also for VRPB. Therefore, in the following we give only the models of the subproblems corresponding to AVRPB.

To obtain the K-SSA on the linehaul vertices, we impose the outdegree constraint (5) for the depot, the indegree constraints (2) for each $j \in L$, and we weaken the capacity-cut constraints (6) into *connectivity* constraints, by replacing the right-hand side with 1. Thus we have (5) and:

$$\sum_{i \in \Gamma_j^-} x_{ij} = 1 \quad \text{for each} \quad j \in L \tag{9}$$

$$\sum_{j \in S} \sum_{i \in \Gamma_j^- \setminus S} x_{ij} \geq 1 \quad \text{for each} \quad S \in \mathcal{L}. \tag{10}$$

Analogously, the second subproblem requires the determination of a min-cost collection of $\bar{K}$, with $K_B \leq \bar{K} \leq K_M = \min\{K, m\}$, disjoint capacitated paths entering the depot and spanning all the backhaul vertices. For any given $\bar{K}$, the subproblem can be relaxed in the same way as the previous one, thus requiring the determination of an SST with fixed degree $\bar{K}$ at the depot vertex. In the asymmetric case we must determine a shortest spanning anti-arborescence (SSAA) with fixed indegree $\bar{K}$ at the depot vertex ($\bar{K}$-SSAA). This problem (called $P_2(\bar{K})$) can be efficiently solved through the Gabow and Tarjan algorithm, by transposing the corresponding cost matrix.

To obtain the $\bar{K}$-SSAA on the backhaul vertices (with $K_B \leq \bar{K} \leq K_M$), we impose:

$$\sum_{i \in \Gamma_0^- \cap B} x_{i0} = \bar{K} \tag{11}$$

$$\sum_{j \in \Gamma_i^+} x_{ij} = 1 \quad \text{for each} \quad i \in B \tag{12}$$

$$\sum_{i \in S} \sum_{j \in \Gamma_i^+ \setminus S} x_{ij} \geq 1 \quad \text{for each} \quad S \in \mathcal{B}. \tag{13}$$

The third subproblem requires the determination of the so-called interface arcs, i.e., a min-cost collection of K arcs, $\bar{K}$ of which connect distinct linehaul vertices to distinct backhaul vertices (with $K_B \leq \bar{K} \leq K_M$), while the remaining $K - \bar{K}$ ones connect distinct linehaul vertices to the depot. For any given $\bar{K}$, this problem (called $P_3(\bar{K})$) can be efficiently solved by transforming it into an equivalent min-cost flow problem on an auxiliary layered network (see Figure 3 for an illustration). The network contains $|V_0| + 3$ nodes: one for each $i \in V_0$, plus a source node, s, a sink node, t, and an intermediate node, r. The arcs of the network, with the associated cost and capacity, are:

—for each $(i, j) \in A_3$: an arc (i, j) with cost c_{ij} and capacity 1;

—for each $i \in L$: an arc (s, i) with cost 0 and capacity 1;
—for each $j \in B$: an arc (j, r) with cost 0 and capacity 1;
—an arc (r, t) with cost 0 and capacity $\bar{K}$;
—an arc $(0, t)$ with cost 0 and capacity $K - \bar{K}$.

It is easy to verify that the determination of the min-cost flow from s to t, with value K, solves the original problem. Indeed, because of the unit capacity of arcs (s, i), in any feasible flow no more than one arc leaving a linehaul node can be used. Analogously, no more than one arc entering a backhaul node, and $K - \bar{K}$ arcs entering the depot node, can be used. In addition, because of flow conservation both arcs entering the sink node t must be saturated; hence the number of arcs entering the depot node is exactly $K - \bar{K}$, and the total number of arcs entering the backhaul nodes is exactly $\bar{K}$. The solution of the min-cost flow problem can be computed in $O(K(n + m)^2)$ time by using, for example, shortest augmenting path algorithms, like those described in AHUJA, MAGNANTI and ORLIN (1993).

The constraints associated with this subproblem are:

$$\sum_{i \in L} \sum_{j \in \Gamma_i^+ \cap B_0} x_{ij} = K \tag{14}$$

$$\sum_{j \in \Gamma_i^+ \setminus L} x_{ij} \leq 1 \quad \text{for each} \quad i \in L \tag{15}$$

$$\sum_{i \in \Gamma_j^- \setminus B_0} x_{ij} \leq 1 \quad \text{for each} \quad j \in B \tag{16}$$

$$\sum_{j \in B} \sum_{i \in \Gamma_j^- \setminus B_0} x_{ij} = \bar{K}. \tag{17}$$

Constraint (14) is obtained from (2), (3), (5), and (7) when $S \equiv L$. Constraints (15) and (16) are derived from (3) and (2) and impose that each linehaul and backhaul vertex is connected with at most one vertex in B_0 and L, respectively. Constraint (17) imposes that exactly $\bar{K}$ arcs connect linehaul with backhaul vertices.

A valid lower bound on $v(P)$ is then

$$\text{LB} = v(P_1) + \min_{K_B \leq \bar{K} \leq K_M} \{v(P_2(\bar{K})) + v(P_3(\bar{K}))\}. \tag{18}$$

Note that lower bound LB can be obtained from the mathematical model (1)–(8) by adding the redundant constraints (10), (13), (14), (15), (16), and by removing constraints (6), (7), (3) for $j \in L$, and (2) for $j \in B$. The additional constraints (11) and (17) can be imposed because, in (18), the minimum value of $v(P_2(\bar{K})) + v(P_3(\bar{K}))$ with respect to $\bar{K}$ is considered.

By using parametric techniques, the computation of $v(P_2(\bar{K}))$ and $v(P_3(\bar{K}))$ for all values of $\bar{K}$ can be performed in $O(m^2)$ and in $O((2K - K_B)(n + m)^2)$ time, respectively. Hence, the overall time complexity of the computation of LB is $O((2K - K_B)(n + m)^2)$.

2.3 Lagrangian Relaxation

The LB of the previous section can be strengthened by considering some of the removed constraints and by introducing them in a Lagrangian fashion into the objective function. In particular, we use as Lagrangian problem the relaxation of Section 2.2, calling for the determination of a K-SSA on the linehaul vertices, as well as a sequence of $\bar{K}$-SSAAs on the backhaul vertices with the associated min-cost flow problem for the interface arcs. To this end we consider the model where capacity-cut constraints (6) and (7) are imposed only for each $S \in \mathcal{L}$ and $S \in \mathcal{B}$, respectively.

Let us introduce into the objective function in a Lagrangian fashion the indegree constraints (2) only for the backhaul vertices ($j \in B$), and the outdegree constraints (3) only for the linehaul vertices ($i \in L$). Let λ be the vector of the corresponding $n + m$ Lagrangian multipliers.

In addition, let $\mathcal{F}_1 \subset \mathcal{L}$, and $\mathcal{F}_2 \subset \mathcal{B}$, be two given families of the exponentially many subsets contained in $\mathcal{F}$. We impose capacity-cut constraints (6) and (7) only for the subsets of $\mathcal{F}_1$ and $\mathcal{F}_2$, respectively, thus obtaining:

$$\sum_{j \in S_h} \sum_{i \in \Gamma_j^- \setminus S_h} x_{ij} \geq \sigma(S_h) \quad \text{for each} \quad S_h \in \mathcal{F}_1 \tag{19}$$

$$\sum_{i \in T_h} \sum_{j \in \Gamma_i^+ \setminus T_h} x_{ij} \geq \sigma(T_h) \quad \text{for each} \quad T_h \in \mathcal{F}_2. \tag{20}$$

Families $\mathcal{F}_1$ and $\mathcal{F}_2$ are determined, in a branch-and-cut fashion, by considering vertex subsets for which the associated capacity-cut constraint is violated by previous relaxations. Further details are given in the next subsections. The families of capacity-cut constraints (19) and (20) are introduced into the objective function with nonnegative Lagrangian multiplier vectors π and ρ, respectively. To simplify the notation in the following, unless strictly necessary, we not explicitly include the Lagrangian multipliers within the name of the Lagrangian problem and that of the derived subproblems using, e.g., LR rather than LR (λ, μ, ρ). The resulting Lagrangian

relaxation LR is then:

$$v(\mathrm{LR}) = \min\Bigg\{ \sum_{(i,j)\in A} c_{ij}x_{ij} - \sum_{j\in B}\lambda_j\Bigg(1 - \sum_{i\in\Gamma_j^-} x_{ij}\Bigg) - \sum_{i\in L}\lambda_i\Bigg(1 - \sum_{j\in\Gamma_i^+} x_{ij}\Bigg) + \sum_{S_h\in\mathcal{F}_1}\pi_h\Bigg(\sigma(S_h) - \sum_{j\in S_h}\sum_{i\in\Gamma_j^-\setminus S_h} x_{ij}\Bigg) + \sum_{T_h\in\mathcal{F}_2}\rho_h\Bigg(\sigma(T_h) - \sum_{i\in T_h}\sum_{j\in\Gamma_i^+\setminus T_h} x_{ij}\Bigg)\Bigg\}$$

$$= -\sum_{j\in V}\lambda_j + \sum_{S_h\in\mathcal{F}_1}\pi_h\sigma(S_h) + \sum_{T_h\in\mathcal{F}_2}\rho_h\sigma(T_h) + \min\Bigg\{\sum_{(i,j)\in A}\bar{c}_{ij}x_{ij}\Bigg\}$$

subject to (5), (8), (9), (10), (11), (12), (13), (14), (15), (16), and (17).

The modified cost matrix $\bar{c}$ is defined as:

$$\bar{c}_{ij} := \begin{cases} c_{ij} - \sum_{S_h\in\mathcal{F}_1: j\in S_h}\pi_h & \text{for } i = 0, \text{ and for each } j \in L, \\ c_{ij} + \lambda_i & \text{for each } i \in L, \text{ and for } j = 0, \\ c_{ij} - \sum_{T_h\in\mathcal{F}_2: i\in T_h}\rho_h & \text{for each } i \in B, \text{ and for } j = 0, \\ c_{ij} + \lambda_i - \sum_{S_h\in\mathcal{F}_1: j\in S_h, i\notin S_h}\pi_h & \text{for each } i \in L, \ j \in L, \\ c_{ij} + \lambda_i + \lambda_j & \text{for each } i \in L, \ j \in B, \\ c_{ij} + \lambda_j - \sum_{T_h\in\mathcal{F}_2: j\in T_h, i\notin T_h}\rho_h & \text{for each } i \in B, \ j \in B. \end{cases}$$

It should be noted that given arbitrary multipliers, the modified cost matrix could be completely asymmetric even if the original matrix contained symmetric submatrices, as happens for VRPB. Hence, in the following we consider only the asymmetric version of the problem.

As previously mentioned, we solve problem LR by dividing it into the three subproblems previously described. The first problem, LR_1, is defined as follows:

$$v(\mathrm{LR}_1) = \min \sum_{(i,j)\in A_1}\bar{c}_{ij}x_{ij} \qquad (21)$$

subject to (5), (9), (10), and $x_{ij} \in \{0, 1\}$ for each $(i, j) \in A_1$, and requires the determination of the K-SSA over L_0.

The second problem, $\mathrm{LR}_2(\bar{K})$, for any given $\bar{K}$, with $K_B \leq \bar{K} \leq K_M$, is defined as:

$$v(\mathrm{LR}_2(\bar{K})) = \min \sum_{(i,j)\in A_2}\bar{c}_{ij}x_{ij} \qquad (22)$$

subject to (11), (12), (13), and $x_{ij} \in \{0, 1\}$ for each $(i, j) \in A_2$, and requires the determination of the $\bar{K}$-SSAA over B_0.

The last problem, $\mathrm{LR}_3(\bar{K})$, for any given $\bar{K}$, is defined as:

$$v(\mathrm{LR}_3(\bar{K})) = \min \sum_{(i,j)\in A_3}\bar{c}_{ij}x_{ij} \qquad (23)$$

subject to (14), (15), (16), (17), and $x_{ij} \in \{0, 1\}$ for each $(i, j) \in A_3$, and calls for the determination of a min cost collection of K interface arcs, $\bar{K}$ of which connect distinct linehaul to distinct backhaul vertices, and $K - \bar{K}$ of which connect distinct linehaul vertices to the depot.

Separately solving, as illustrated in Section 2.2, the three subproblems, we obtain the quantity:

$$v(\mathrm{LR}(\lambda, \pi, \rho)) = \sum_{S_h\in\mathcal{F}_1}\pi_h\sigma(S_h) + \sum_{T_h\in\mathcal{F}_2}\rho_h\sigma(T_h) - \sum_{j\in V}\lambda_j + v(\mathrm{LR}_1) + \min_{K_B\leq\bar{K}\leq K_M}\{v(\mathrm{LR}_2(\bar{K})) + v(\mathrm{LR}_3(\bar{K}))\}$$

which is a valid lower bound on the value of $v(P)$.

2.4 Subgradient Optimization Procedure

In this section we describe an overall lower bounding procedure in which the Lagrangian relaxation is iteratively strengthened by separating new valid inequalities associated with the capacity-cut constraints and adding them to families $\mathcal{F}_1$ and $\mathcal{F}_2$ (i.e., to the Lagrangian problem). This approach contains some analogies with the cutting-plane approach and has been applied by FISHER (1994) to the capacitated VRP, by Malik and Yu (1993) to the capacitated SST, and by Toth and Vigo (1995) to the capacitated SSA. The way in which violated capacity-cut constraints are separated is discussed in Section 2.5.

Given the families $\mathcal{F}_1$ and $\mathcal{F}_2$, the best Lagrangian relaxation value $v(\mathrm{LR}(\lambda, \pi, \rho))$ can be obtained by solving the so-called *Lagrangian dual problem*, i.e., by determining multiplier vectors λ^*, π^*, and ρ^* such that

$$v(\mathrm{LR}(\lambda^*, \pi^*, \rho^*)) = \max_{\lambda,\pi,\rho}\{v(\mathrm{LR}(\lambda, \pi, \rho))\}. \qquad (24)$$

For problems where the degree and the capacity constraints have been relaxed in a Lagrangian fashion (see, e.g., Fisher (1994) for the Vehicle Routing

Problem) the *subgradient optimization* technique proved to be an effective approach for the solution of problem (24). We have implemented a standard subgradient procedure following the scheme proposed by HELD and KARP (1971) for the Traveling Salesman Problem (see also HELD, WOLFE and CROWDER (1974) for further details). We separately optimize the Lagrangian multipliers λ, associated with the degree constraints, and multipliers π and ρ, associated with the capacity-cut constraints of families $\mathcal{F}_1$ and $\mathcal{F}_2$. In other words, the procedure is organized into two levels. In the outer level (called the "capacity-cut subgradient") multipliers π and ρ are updated, whereas in the inner level (called the "degree subgradient") only multipliers λ are optimized, with π and ρ fixed. According to our computational experience, this approach proved to give better results than the simultaneous optimization of all the multipliers.

We next give a pseudo-Pascal description of the overall procedure LB, ($\bar{x}$ denotes the solution vector associated with the current Lagrangian problem LR).

Procedure LB:

begin
0 **let** $\mathcal{F}_1 := \emptyset$; $\mathcal{F}_2 := \emptyset$; $\lambda_i := 0$, for each $i \in V$;
1 Optimize λ multipliers by performing I_1 degree subgradient iterations;
repeat
2 Find, if any, a collection of *infeasible subsets*, i.e., subsets such that $\bar{x}$ violates the associated capacity-cut constraints (separation);
3 Add the infeasible subsets to families $\mathcal{F}_1$ and $\mathcal{F}_2$ and initialize the Lagrangian multipliers π and ρ associated with the new subsets;
4 Remove from families $\mathcal{F}_1$ and $\mathcal{F}_2$ the subsets whose multiplier has value 0 since I_2 separation iterations;
repeat
5 Optimize only λ multipliers by performing I_3 degree subgradient iterations with π and ρ fixed;
6 Update the Lagrangian multipliers π and ρ
7 **until** (I_4 capacity-cut subgradient iterations have been performed)
8 **until** (no violated constraint is found) **or** (I_5 separation iterations have been performed)
end.

Our computational experience has shown that the initialization Step 1 substantially improves the overall performance of Procedure LB. Moreover, the number I_1 of performed subgradient optimization iterations is much greater than I_3, since at Step 5 good multipliers can normally be determined in a few iterations. Obviously the procedure stops as soon as the value of the current lower bound is greater than or equal to that of the best feasible solution found so far, or when the solution of the current Lagrangian problem is feasible.

2.5 Separation of Violated Capacity-Cut Constraints

In the following we describe a polynomial time exact procedure to determine infeasible vertex subsets, if any, such that the current Lagrangian problem solution, $\bar{x}$, violates the associated capacity-cut constraints (6) or (7).

In order to separate violated capacity-cut constraints of type (6) for $S \in \mathcal{L}$, we consider the arborescence corresponding to arc subset $W_L = \{(i,j) \in A_1 : \bar{x}_{ij} = 1\}$. For each linehaul vertex i, let X_i be the subset containing all the linehaul vertices belonging to the subarborescence in W_L rooted at i (with $i \in X_i$), and let w_i denote the total demand of subset X_i. If $w_i > D$, then constraint (6) for $S = X_i$ is violated, because $\sigma(X_i)$ is greater than 1 and only one arc in W_L enters subset X_i. Hence, we add subset X_i to family $\mathcal{F}_1$. The computation of w_i, for all $i \in L$, can be performed in $O(n)$ time. Note that although we consider only vertex sets associated with subarborescences whose arcs are in W_L, it is easy to see that if a violated constraint of type (6) exists, then it is detected by the above procedure. Generally $\sigma(X_i)$ is replaced by $\lceil \sum_{j \in X_i} d_j / D \rceil$. Stronger inequalities may be detected by using better estimates of $\sigma(X_i)$, e.g., by computing tighter BPP lower bounds or even by exactly solving the associated BPP.

Analogously, we can separate violated capacity-cut constraints of type (7) by considering only the antiarborescence corresponding to arcs subset $W_B = \{(i,j) \in A_2 : \bar{x}_{ij} = 1\}$, thus determining vertex subsets to be added to family $\mathcal{F}_2$.

2.6 An Overall Additive Lower Bound

The lower bounds obtained through the Lagrangian and the TP relaxations of VRPB, described in the previous section, are now combined according to the *additive approach* proposed by Fischetti and Toth (1989). This approach allows for the combination of different lower bounding procedures, each exploiting different substructures of the problem. When applied to VRPB each procedure returns a lower bound δ and a *residual cost matrix*, $\tilde{c}$, such that:

$$\tilde{c} \geq 0 \quad \delta + \tilde{c}x \leq cx$$

for all x satisfying (1)–(8).

The entries of $\tilde{c}$ represent lower bounds on the increment of the optimal solution value if the corresponding arc is imposed in the solution. The bounding procedures are applied in sequence and each of them uses the residual cost matrix given by the previous procedure (the first procedure starts with the original cost matrix). The additive lower bound is the sum of the lower bounds obtained by each procedure. For further details, see also FISCHETTI and TOTH (1992), and Fischetti, Toth and Vigo (1994).

It can be easily shown (see Fischetti and Toth (1989) and Fischetti, Toth and Vigo (1994)) that the Lagrangian modified costs, as well as the reduced costs of the AP relaxation of Section 2.1, are valid residual costs. At the end of the computation of the Lagrangian lower bound, the modified cost matrix $\tilde{c}$ is extended by adding $K - 1$ rows and columns corresponding to the copies of the depot, and the resulting AP relaxation is solved, returning the lower bound increment. The final AP reduced costs can be used for reduction purposes, as described in Section 3.3.

Our computational experience showed that for symmetric instances the overall contribution of the AP relaxation in the additive bound is generally marginal. However, the additional computing time required is practically negligible with respect to that needed for the Lagrangian bound. Moreover, for asymmetric instances the improvement is often more relevant.

3. BRANCH-AND-BOUND ALGORITHM

WE NEXT DESCRIBE a lowest-first branch-and-bound algorithm for finding the optimal solution to VRPB. The performance of this algorithm is analyzed through computational experiments in Section 4.

3.1 Branching

The implemented branching rule is an extension of the well-known *subtour elimination* scheme of BELLMORE and MALONE (1971) for the asymmetric traveling salesman problem.

Let ν be the current node of the branch-decision tree and let I_ν and F_ν denote the set of arcs imposed and excluded in the current solution, respectively. By construction, I_ν induces a collection of feasible routes and feasible paths, some of which may correspond to isolated vertices. We construct a restricted VRPB instance associated with I_ν and F_ν by means of the following preprocessing. We first remove from the graph all the vertices covered in I_ν by complete routes, and update accordingly the number $\tilde{K}$ of available vehicles. We then shrink into a single vertex each path induced by the arcs of I_ν not incident in vertex 0, if any. The demand associated with the new vertex is the sum of the demands of the vertices belonging to the shrunken path. We thus obtain a new restricted cost matrix and new demands for the shrunken vertices. The arcs $(0, j) \in I_\nu$ (resp. $(j, 0) \in I_\nu$) are imposed by preventing all the other arcs from entering (resp. leaving) vertex j. We finally exclude all the arcs in F_ν by setting the cost of the corresponding arc in the restricted matrix to $+\infty$.

After the preprocessing we apply the overall bounding procedure and store the best solution $\bar{x}$ of the Lagrangian Problem (reconstructed for the original graph). If $\bar{x}$ is feasible for VRPB we possibly update the current incumbent solution; if, in addition, all the constraints (19) and (20) are satisfied by $\bar{x}$ at equality, then $\bar{x}$ is the optimal solution for the current problem, and we backtrack. Otherwise, we determine a sequence $Q := \{(v_1, v_2), (v_2, v_3), \ldots, (v_h, v_{h+1})\}$ to branch with, chosen as the subset of $\{(i, j) \in A: \bar{x}_{ij} = 1\}$ having the minimum number of nonimposed arcs, which defines either a path which is infeasible (i.e., it is either a cycle disconnected from the depot or a path in which the total demand of linehaul and backhaul vertices exceed, separately, the vehicle capacity), or, if $\bar{x}$ is feasible, a circuit through vertex 0 (in this case, $v_1 = v_{h+1} = 0$). We then generate h descendant nodes ν_i, $i = 1, \ldots, h$, defined by:

$$I_{\nu_i} := I_\nu \cup \{(v_1, v_2), \ldots, (v_{i-1}, v_i)\}$$

$$F_{\nu_i} := F_\nu \cup \{(v_i, v_{i+1})\}$$

(with $I_{\nu_1} := I_\nu$). When Q is a feasible circuit, we generate an additional son node, ν_{h+1}, with $I_{\nu_{h+1}} := I_\nu \cup Q$, and $F_{\nu_{h+1}} := F_\nu$. Clearly, nodes ν_i such that $I_{\nu_i} \cap F_{\nu_i} \neq \varnothing$ are not generated.

3.2 Lower Bounding

At the root node of the branch-decision tree we perform a large number of iterations of the subgradient optimization procedure (e.g., $I_1 = 200$, $I_2 = 50$, $I_3 = 10$, $I_4 = 5$, $I_5 = 1000$) to obtain the best possible lower bound value. At the other nodes of the branch-decision tree a much smaller number of iterations are performed. At each node of the branch-decision tree we initialize the families $\mathcal{F}_1$ and $\mathcal{F}_2$, the associated multipliers μ and ρ and the multipliers λ with the sets and multiplier values corresponding to the best Lagrangian lower bound obtained at the father node. The initialization of the multipliers takes suitably into account the vertex shrinking induced by the imposed arcs described in the previous section. This initialization proved to be computationally very effective, greatly reducing the overall

computing time. Indeed, at the descendant nodes there is no need to execute the λ initialization step (i.e., Step 1 of Procedure LB) and the overall number of subgradient iterations executed is much smaller than that of the root node (e.g., $I_1 = 0$, $I_2 = 10$, $I_3 = 5$, $I_4 = 1$, $I_5 = 50$).

3.3 Reduction and Dominance Procedures

The performance of the branch-and-bound algorithm is enhanced by means of variables reduction and feasibility checks. The first reduction is carried out before the start of the branch-and-bound procedure and consists of setting $c_{ij} = +\infty$ for all $i, j \in L$ or $i, j \in B$ such that $d_i + d_j > D$, because those customers cannot be consecutive in a feasible route. Moreover, the following reduction and feasibility checks are applied at each branching node v (right after preprocessing).

We forbid, inside the bounding procedure, the arcs whose current residual cost is greater or equal to the gap between the incumbent solution value and the current lower bound.

Whenever only one arc in the forward (or backward) star of a vertex is not forbidden, we impose it. If the new imposed or forbidden arcs lead to an infeasible instance (i.e., an infeasible path or circuit is imposed), we backtrack. Otherwise, we reapply preprocessing, and repeat.

After the preprocessing, and before the lower bound computation, we check whether the current number of vehicles, say $\tilde{K}$, suffices for all the demands, i.e., whether $\tilde{K}$ bins of capacity D are enough to load all the demands associated with unrouted linehaul and backhaul customers, separately. To this end, we use the Martello and Toth (1990) branch-and-bound code MTP for the BPP, modified to stop whenever a solution with no more than $\tilde{K}$ bins has been found, or after a prefixed number NB of backtrackings (we set $NB = 500$).

We have finally implemented the following dominance criterion, according to the general framework proposed in FISCHETTI and TOTH (1988). Let $P := \{(v_1, v_2), \ldots, (v_{h-1}, v_h)\}$ be any imposed path. If a different path, $\tilde{P}$, exists from v_1 to v_h, visiting the same intermediate nodes and with a lower cost, then the current branching node is clearly dominated, and hence fathomed. The existence of $\tilde{P}$ is detected in a heuristic way by applying 3-optimality exchanges within P.

3.4 Upper Bounding

At the root node the best incumbent solution is initialized by using the heuristic algorithm HTV, described in Toth and Vigo (1996). The heuristic is applied to the Lagrangian solution $\bar{x}$ obtained at the end of each iteration of the subgradient procedure. As shown in the next section, this multistart technique provided almost always optimal or nearly optimal initial solution for the branch-and-bound algorithm.

Moreover, at each node of the branch-decision tree, if the node is not fathomed by the lower bound or by the dominance rules, we apply the algorithm HTV, for possible updating of the incumbent solution. To reduce the overall computing time needed to examine each branching node, the heuristic algorithm is run only at the first 200 nodes of the branch-decision tree and is applied only to the best solution obtained by the Lagrangian lower bound.

Algorithm HTV is a cluster-first–route-second heuristic. The clusters are obtained in three steps. Let $W := \{(i, j): \bar{x}_{ij} = 1\}$, first we remove from W the $2K$ arcs incident into the depot and the interface arcs from linehaul to backhaul vertices, then we determine groups containing only either linehaul or backhaul vertices, by considering the connected components induced by the remaining arcs in W. Finally, K clusters are obtained by matching each group of linehaul vertices with either a group of backhaul vertices or the depot. The initial K routes, possibly infeasible, are built by sequencing the vertices belonging to each cluster by means of a modified traveling salesman problem farthest insertion heuristic. The final set of routes, generally feasible, are determined by using intraroute and interroute exchange procedures.

4. COMPUTATIONAL RESULTS

THE HEURISTIC ALGORITHM HTV and the branch-and-bound algorithm, called TV in the sequel, have been implemented in FORTRAN, and run on a Pentium 60 MHz personal computer (5.3 Mflops according to DONGARRA (1996)) on several problem instances from the literature. For the solution of the assignment problems, we used the FORTRAN code APC given in CARPANETO, MARTELLO, and TOTH (1988).

The first class examined contains the randomly generated Euclidean instances proposed by Goetschalckx and Jacobs-Blecha (1989), where customers' coordinates are uniformly distributed in interval [0, 24000] for the x values, and in interval [0, 32000] for the y values. The depot is located centrally in (12000, 16000). The cost c_{ij} of arc (i, j) is defined as the Euclidean distance between vertices i and j, multiplied by 10 and rounded to the nearest integer. The final lower bound, upper bound, and optimal solution values reported in the following are those obtained using the integer matrix c, divided by

TABLE I

Test Problems Proposed by Goetschalckx and Jacobs-Blecha

Name	$n+m$	n	m	K	K_L	K_B	LB_0 %	LB_0^D %	UB_0 %	z^*	time	nodes
A1	25	20	5	8	7	2	98.3	91.2	100.0	229886	901.5	273
A2	25	20	5	5	4	1	98.1	94.5	100.0	180119	209.0	60
A3	25	20	5	4	3	1	100.0	98.5	100.0	163405	3.4	1
A4	25	20	5	3	3	1	100.0	98.4	100.0	155796	2.9	1
B1	30	20	10	7	7	4	96.0	86.8	100.0	239080	147.7	11
B2	30	20	10	5	4	3	97.4	91.9	100.0	198048	151.4	53
B3	30	20	10	3	3	2	100.0	100.0	100.0	169372	0.9	1
C1	40	20	20	7	6	6	95.7	86.3	100.7	249448	226.6	7
C2	40	20	20	5	4	4	96.5	90.4	100.0	215020	321.9	59
C3	40	20	20	5	3	3	99.8	98.6	100.0	199346	84.1	8
C4	40	20	20	4	3	3	100.0	98.5	100.0	195366	5.1	1
D1	38	30	8	12	10	3	97.0	91.3	100.3	322530	289.0	28
D2	38	30	8	11	10	3	94.5	88.0	100.6	316709	491.2	71
D3	38	30	8	7	6	2	95.9	91.3	100.0	*239479	—	2315
D4	38	30	8	5	5	2	95.4	93.1	100.0	*205832	—	1803
E1	45	30	15	7	6	3	95.1	90.7	100.0	238880	475.7	46
E2	45	30	15	4	4	2	97.9	94.9	100.9	212263	787.6	142
E3	45	30	15	4	3	2	98.2	96.5	100.0	206659	481.6	166
F1	60	30	30	6	5	6	96.6	93.5	100.0	263173	756.0	77
F2	60	30	30	7	5	6	98.3	95.1	100.3	265213	891.0	50
F3	60	30	30	5	4	4	98.0	96.7	100.0	241120	468.3	17
F4	60	30	30	4	3	3	97.3	96.8	100.0	233861	3522.6	920
G1	57	45	12	10	9	3	91.1	82.0	100.0	*307274	—	258
G2	57	45	12	6	6	2	93.3	88.9	100.0	*245441	—	366
G3	57	45	12	5	5	2	96.2	92.9	100.7	229507	4225.3	683
G4	57	45	12	6	5	2	96.5	93.6	100.1	*233184	—	793
G5	57	45	12	5	4	1	97.9	96.1	100.1	221730	3433.2	714
G6	57	45	12	4	3	1	96.6	96.1	100.0	213457	840.3	189
H1	68	45	23	6	6	3	96.6	93.6	100.1	268933	1344.2	214
H2	68	45	23	5	5	3	99.4	98.2	100.0	253365	5019.6	1127
H3	68	45	23	4	4	2	99.2	99.0	100.2	247449	1464.5	130
H4	68	45	23	5	4	2	99.7	99.5	100.0	250221	1287.0	134
H5	68	45	23	4	3	2	99.3	98.8	102.3	246121	405.9	7
H6	68	45	23	5	3	2	99.4	98.0	100.0	249135	416.1	10

*Best incumbent solution value.

Computing times in Pentium 60 MHz seconds. Time limit of 6000 seconds.

10 and rounded to the nearest integer. This definition of the costs is motivated by the fact that in Goetschalckx and Jacobs-Blecha (1989) real values were used for the cost matrix, whereas we use integer values, and we want to obtain a better precision. Customer demands are generated from a normal distribution with mean value 500 and standard deviation 200. Fourteen values for the total number of customers, $n + m$, (from 25 to 150) have been considered, with linehaul percentage equal to 50, 66, and 80%. For each value of $n + m$, the vehicle capacity has been chosen so that approximately 3 to 12 vehicles are needed to serve all the demands.

Table I reports the results obtained by algorithm TV on instances of this class with $n + m$ between 25 and 68, within an imposed time limit of 6000 CPU seconds. For each problem we give the problem name, the problem size (namely the values of n and m), the available number of vehicles K, and the minimum number of vehicles needed to serve the linehaul and the backhaul vertices, K_L and K_B, respectively. The values of K_L and K_B are determined by solving the associated BPP using the code MTP by Martello and Toth (1990). The table also reports:

—the percentage error of the best additive lower bound, LB_0, computed at the root node.

—the percentage error of the lower bound, LB_0^D, obtained at the end of Step 1 of procedure LB, where only the degree constraints have been relaxed in a Lagrangian fashion.

—the percentage error of the best upper bound com-

TABLE II

Problems Obtained from Symmetric VRP Instances

Name	$n+m$	n	m	K_L	K_B	LB_0 %	LB_0^D %	UB_0 %	z^*	time	nodes
eil22_50	21	11	10	3	2	100.0	98.9	100.0	371	2.6	1
eil22_66	21	14	7	3	1	100.0	95.6	100.0	366	5.7	1
eil22_80	21	17	4	3	1	98.9	90.7	100.0	375	55.3	26
eil23_50	22	11	11	2	1	100.0	94.3	100.0	682	1.8	1
eil23_66	22	15	7	2	1	98.8	93.5	100.0	649	65.3	19
eil23_80	22	18	4	2	2	98.1	98.1	100.0	623	36.1	23
eil30_50	29	15	14	2	2	100.0	92.8	100.0	501	2.7	1
eil30_66	29	20	9	3	1	98.5	86.0	100.0	537	119.3	12
eil30_80	29	24	5	3	1	100.0	90.3	100.0	514	12.9	1
eil33_50	32	16	16	3	2	98.4	93.9	100.4	738	292.4	80
eil33_66	32	22	10	3	1	94.8	93.1	100.4	750	1337.6	791
eil33_80	32	26	6	3	1	93.9	89.5	100.0	736	1655.0	1054
eil51_50	50	25	25	3	3	99.3	98.4	100.5	559	441.4	42
eil51_66	50	34	16	4	2	97.8	96.2	100.5	548	2754.3	347
eil51_80	50	40	10	4	1	98.0	94.9	101.6	565	4436.1	1976
eilA76_50	75	37	38	6	5	98.2	95.1	102.7	739	15931.2	2804
eilA76_66	75	50	25	7	4	95.4	93.4	100.5	768	13463.5	1106
eilA76_80	75	60	15	8	2	90.5	88.1	111.4	*781	—	1160
eilB76_50	75	37	38	8	7	97.6	92.3	103.9	801	16344.7	2892
eilB76_66	75	50	25	10	5	91.2	86.6	104.1	873	12990.1	796
eilB76_80	75	60	15	12	3	85.2	82.2	103.0	919	10413.5	499
eilC76_50	75	37	38	5	4	98.9	96.8	100.8	713	10344.3	2752
eilC76_66	75	50	25	6	3	97.6	96.3	101.2	*734	—	2312
eilC76_80	75	60	15	7	2	93.7	92.8	102.7	*733	—	1248
eilD76_50	75	37	38	4	3	99.7	98.4	101.7	690	401.0	21
eilD76_66	75	50	25	5	2	98.5	97.8	100.6	*715	—	2558
eilD76_80	75	60	15	6	2	95.6	94.9	100.9	*703	—	1520
eila101_50	100	50	50	4	4	96.3	95.0	100.9	*843	—	1359
eila101_66	100	67	33	6	3	99.2	98.2	101.1	846	10912.8	539
eila101_80	100	80	20	6	2	89.5	88.5	101.2	*916	—	499

*Best incumbent solution value.
Times in Pentium 60 MHz seconds. Time limit of 18000 seconds.

puted at the root node by using procedure HTV, UB_0.

—the value of the optimal solution, z^*. If the time limit has been reached, z^* indicates the best incumbent solution value and the instance is marked with an asterisk.

—the number of nodes explored by the branch-and-bound and the overall computing time expressed in Pentium 60 MHz CPU seconds.

Percentage errors are computed as the ratio of the lower or upper bound divided by z^* (i.e., the optimal or the best incumbent solution) and multiplied by 100.

By observing Table I we note that the value of the overall additive lower bound is tight and substantially improves the value of the Lagrangian lower bound LB_0^D which takes into account only the degree constraints. The exact algorithm has been able to solve, within the imposed time limit of 6000 seconds, 29 out of the 34 instances. However, by allowing a longer computing time we have proved that the solution obtained for instances D3, D4, and G2 are indeed optimal.

We examined a second class of VRPB instances obtained from Euclidean VRP instances proposed in the literature, with a total number of customers between 21 and 100 (see Table II). For each VRP instance we generated three VRPB instances each corresponding to a linehaul percentage of 50, 66, and 80%, respectively. The customer set is partitioned by defining as backhaul the first customer every two, three, and five, depending on the desired linehaul percentage. The cost c_{ij} of arc (i, j) is defined as the Euclidean distance between vertex i and j, rounded to the nearest integer. Customer demands and vehicle capacity are set equal to the original VRP ones.

TABLE III

Problems Obtained from Asymmetric VRP Instances

Name	$n+m$	n	m	K_L	K_B	LB_0 %	LB_0^D %	UB_0 %	z^*	time	nodes
FTV33_50	33	17	16	2	1	100.0	100.0	100.0	1841	2.7	1
FTV33_66	33	22	11	2	1	99.5	98.9	100.0	1899	123.2	84
FTV33_80	33	27	6	2	1	99.9	99.5	100.0	1704	37.4	3
FTV35_50	35	18	17	2	2	98.7	97.7	103.6	2077	366.9	272
FTV35_66	35	24	11	2	1	98.0	97.4	101.1	2150	652.8	603
FTV35_80	35	28	7	2	1	98.7	97.9	100.3	1996	412.0	585
FTV38_50	38	19	19	2	2	100.0	99.6	100.0	2162	56.0	4
FTV38_66	38	26	12	2	2	98.3	97.0	101.5	2132	1382.4	1425
FTV38_80	38	31	7	3	1	98.8	96.9	100.4	1982	1303.0	519
FTV44_50	44	22	22	2	2	98.6	98.4	100.6	2348	1557.2	716
FTV44_66	44	30	14	2	1	97.2	96.4	100.6	2225	4035.2	3377
FTV44_80	44	36	8	3	1	98.0	97.2	102.7	2184	3439.3	3370
FTV47_50	47	24	23	2	2	99.6	98.8	100.8	2343	229.6	61
FTV47_66	47	32	15	2	1	99.7	99.4	102.3	2427	288.2	34
FTV47_80	47	38	9	2	1	99.3	99.1	102.4	2312	1949.7	560
FTV55_50	55	28	27	2	2	99.2	98.2	101.4	2425	2450.2	356
FTV55_66	55	37	18	2	1	98.6	97.9	102.9	2246	5045.4	3086
FTV55_80	55	44	11	2	1	98.6	97.5	102.8	2264	5090.7	2578
FTV64_50	64	32	32	2	2	98.7	97.9	102.2	2728	4634.8	2968
FTV64_66	64	43	21	2	1	98.5	98.3	100.2	2673	5797.3	2058
FTV64_80	64	52	12	3	1	98.0	97.5	100.4	*2679	—	3652
FTV70_50	70	35	35	2	2	98.3	97.9	103.0	*2940	—	2446
FTV70_66	70	47	23	2	1	99.6	99.3	101.6	2808	4950.3	1071
FTV70_80	70	56	14	2	1	99.3	98.4	102.7	2688	5049.3	1797

*Best incumbent solution value.

Times in Pentium 60 MHz seconds. Time limit of 6000 seconds.

The values of K_L and K_B are determined by solving the associated BPP instances using the code MTP by Martello and Toth (1990). For instances where $K_L < K_B$, linehaul and backhaul customer sets are swapped. The number K of available vehicles is set equal to K_L. The time limit for this class was set to 18,000 seconds.

As for the previous class, the overall additive bound substantially improves LB_0^D. Moreover, it can be noted that the problem tends to be more difficult when the linehaul percentage increases. The exact algorithm was able to solve instances with up to 100 vertices.

We finally examined a class of asymmetric VRPB instances obtained from the real-world asymmetric VRP (AVRP) instances described in Fischetti, Toth, and Vigo (1994) (see Table III). As in the previous class, for each AVRP instance we generated three AVRPB instances (corresponding to a linehaul percentages of 50, 66, and 80%, respectively), and the customer set is partitioned by defining as backhaul the first customer every two, three, and five. Customer demands, vehicle capacity, and cost matrix c are equal to the original AVRP ones. The values of K_L, K_B and K are defined as for the second class. Also in this case, the exact algorithm was able to solve instances with up to 70 nodes within the imposed time limit of 6000 seconds. Moreover, the Lagrangian lower bound is on average tighter for these instances than for the symmetric instances. For this class the difficulty of the problem seems to be less correlated with the linehaul percentage.

5. CONCLUSIONS

IN THIS PAPER we have described the first exact algorithm for the solution of the Vehicle Routing Problem with Backhauls. The algorithm uses a new Lagrangian lower bound strengthened by adding valid inequalities in a cutting-plane fashion.

The proposed approach proved to be effective, being able to solve problems proposed in the literature with up to 100 customers. The overall additive lower bound is considerably tight and substantially improves the value of the other examined relaxations. The Lagrangian heuristic used during the search of the branch-decision tree, in many cases, detected

the optimal solution value very soon, thus considerably reducing the overall computing time.

Future research can extend the separation algorithm, allowing for a further improvement of the bound quality and hence of the branch-and-bound algorithm.

ACKNOWLEDGEMENTS

THIS WORK HAS BEEN supported by Ministero dell'Università e della Ricerca Scientifica e Tecnologica and by Consiglio Nazionale delle Ricerche, Italy. We thank Marc Goetschalckx and Charlotte Jacobs-Blecha for providing us the problem data for the first class. We also thank the anonymous referees for their helpful comments.

REFERENCES

R. K. AHUJA, T. L. MAGNANTI AND J. B. ORLIN, *Network Flows*, Prentice Hall, Englewood Cliffs, NJ, 1993.

M. BELLMORE AND J. C. MALONE, "Pathology of Travelling Salesman Subtour-Elimination Algorithms," *Oper. Res.* **19,** 278–307 (1971).

D. O. CASCO, B. L. GOLDEN AND E. A. WASIL, "Vehicle Routing with Backhauls: Models, Algorithms, and Case Studies," in *Vehicle Routing: Methods and Studies*, B. L. Golden, A. A. Assad (eds.), North-Holland, Amsterdam, 127–147, 1988.

G. CARPANETO, S. MARTELLO, AND P. TOTH, "Algorithms and Codes for the Assignment Problem," *Ann. Oper. Res.* **13,** 193–223 (1988).

I. DEIF AND L. BODIN, "Extension of the Clarke and Wright Algorithm for Solving the Vehicle Routing Problem with Backhauling," in *Proceedings of the Babson Conference on Software Uses in Transportation and Logistic Management*, A. E. Kidder (ed.), Babson Park, MA, 75–96, 1984.

J. J. DONGARRA, Performance of Various Computers using Standard Linear Equations Software, Technical Report CS-89-85, University of Tennessee, Knoxville, 1996.

C. DUHAMEL, J.-Y. POTVIN AND J.-M. ROUSSEAU, "A Tabu Search Algorithm for the Vehicle Routing Problem with Backhauls and Time Windows," *Transp. Sci.* **31,** 49–59 (1997).

M. FISCHETTI AND P. TOTH, "A New Dominance Procedure for Combinatorial Optimization," *Oper. Res. Lett.* **7,** 181–187 (1988).

M. FISCHETTI AND P. TOTH, "An Additive Bounding Procedure for Combinatorial Optimization Problems," *Oper. Res.* **37,** 319–328 (1989).

M. FISCHETTI AND P. TOTH, "An Additive Bounding Procedure for the Asymmetric Travelling Salesman Problem," *Math. Program.* **53,** 173–197 (1992).

M. FISCHETTI, P. TOTH AND D. VIGO, "A Branch and Bound Algorithm for the Capacitated Vehicle Routing Problem on Directed Graphs," *Oper. Res.* **42,** 846–859 (1994).

M. L. FISHER, "Optimal Solution of Vehicle Routing Problem using Minimum k-Trees," *Oper. Res.* **42,** 626–642 (1994).

H. N. GABOW AND R. E. TARJAN, "Efficient Algorithms for a Family of Matroid Intersection Problems," *J. Algorithms* **5,** 80–131 (1984).

S. GÉLINAS, M. DESROCHERS, J. DESROSIERS AND M. M. SOLOMON, "A New Branching Strategy for the Time Constrained Routing Problem with Application to Backhauling," *Ann. Oper. Res.* **61,** 91–110 (1995).

F. GLOVER AND D. KLINGMAN, "Finding Minimum Spanning Tree with a Fixed Number of Links at a Node," in *Colloq. Math. Soc. János Bolyai*, 425–439 (1974).

M. GOETSCHALCKX AND C. JACOBS-BLECHA, "The Vehicle Routing Problem with Backhauls," *Eur. J. Oper. Res.* **42,** 39–51 (1989).

M. GOETSCHALCKX AND C. JACOBS-BLECHA, The Vehicle Routing Problem with Backhauls: Properties and Solution Algorithms, Technical Report MHRC-TR-88-13, Georgia Institute of Technology, 1993.

B. L. GOLDEN, E. BAKER, J. ALFARO AND J. SCHAFFER, "The Vehicle Routing Problem with Backhauling: Two Approaches," in *Proceedings of the XXI Annual Meeting of S.E. TIMS*, R. D. Hammesfahr (ed.), Myrtle Beach, SC, 90–92, 1985.

M. HELD AND R. M. KARP, "The Travelling Salesman Problem and Minimum Spanning Trees: Part II," *Math. Program.* **1,** 6–25 (1971).

M. HELD, P. WOLF AND H. P. CROWDER, "Validation of the Subgradient Optimization," *Math. Program.* **6,** 62–88 (1974).

G. KONTOVRADIS AND J. BARD, "A GRASP for the Vehicle Routing Problem with Time Windows," *ORSA J. Comput.* **7,** 10–23 (1995).

G. LAPORTE, H. MERCURE AND Y. NOBERT, "An Exact Algorithm for the Asymmetrical Capacitated Vehicle Routing Problem," *Networks* **16,** 33–46 (1986).

K. MALIK AND G. YU, "A Branch and Bound Algorithm for the Capacitated Minimum Spanning Tree Problem," *Networks* **23,** 525–532 (1993).

S. MARTELLO AND P. TOTH, *Knapsack Problems: Algorithms and Computer Implementations*, John Wiley & Sons, Chichester, 1990.

C. H. PAPADIMITRIOU, "The Complexity of the Capacitated Spanning Tree Problem," *Networks* **8,** 217–230 (1978).

P. TOTH AND D. VIGO, "An Exact Algorithm for the Capacitated Shortest Spanning Arborescence," *Ann. Oper. Res.* **61,** 121–142 (1995).

P. TOTH AND D. VIGO, "A Heuristic Algorithm for the Vehicle Routing Problem with Backhauls," in *Advanced Methods in Transportation Analysis*, L. Bianco and P. Toth (eds.), Springer Verlag, Berlin, 585–608, 1996.

C. YANO, T. CHAN, L. RICHTER, T. CUTLER, K. MURTY AND D. MCGETTIGAN, "Vehicle Routing at Quality Stores," *Interfaces* **17(2),** 52–63 (1987).

(Received: July 1996; revisions received: March 1997; accepted: March 1997)