Chapter 2
Branch-and-Bound Algorithms for the Capacitated VRP

Paolo Toth
Daniele Vigo

2.1 Introduction

The branch-and-bound method has been used extensively in recent decades to solve the CVRP and its main variants. In many cases, as for the *Asymmetric CVRP* (ACVRP) and the *Distance-Constrained CVRP* (DCVRP), these algorithms still represent the state of the art with respect to the exact solution methods. In their extensive survey devoted to exact methods, Laporte and Nobert [23] gave a complete and detailed analysis of the branch-and-bound algorithms proposed up until the late 1980s.

In this chapter, we concentrate on the most recent branch-and-bound algorithms, proposed during the last few years for the exact solution of CVRP, for both symmetric and asymmetric cost matrices. When the explicit distinction between SCVRP and ACVRP is not needed, we simply use CVRP. Although no new result has been presented for the DCVRP, we briefly review the known algorithms for this problem, too.

As mentioned in the introduction, the CVRP is an extension of the well-known Traveling Salesman Problem (TSP), calling for the determination of a Hamiltonian circuit with minimum cost visiting a given set of points exactly once. Therefore, many exact approaches for the CVRP were inherited from the extensive and successful work done for the exact solution of the TSP. Until the late 1980s, the most effective exact approaches for the CVRP were mainly branch-and-bound algorithms, which used basic combinatorial relaxations, such as the *Assignment Problem* (AP), the degree-constrained *Shortest Spanning Tree* (SST), and the state space relaxation. Recently, more sophisticated bounds were proposed, like those based on Lagrangian relaxations or on the additive approach, which substantially increased the size of the problems that can be solved to optimality by branch-and-bound.

When presenting the basic relaxations used to compute lower bounds, we treat separately problems with asymmetric and symmetric cost matrices. In fact, although the symmetric problems are special cases of the asymmetric ones, the latter were much less studied in the literature and the exact methods developed for them have in general a poor performance when applied to symmetric instances. Analogously, not all the approaches proposed for symmetric problems can be easily adapted to solve asymmetric problems.

In section 2.2 we consider the basic combinatorial relaxations proposed for ACVRP and SCVRP. We next present, in section 2.3, the more effective relaxations based on Lagrangian and additive approaches. In section 2.4 the main features and the relative performance of the branch-and-bound algorithms are discussed. Section 2.5 examines the relaxations proposed for the DCVRP, and in the last section we draw some conclusions and outline possible future directions of research.

We remind the reader that throughout this chapter, the graphs, directed or undirected, are assumed to be complete. Information on the performance of the computers used for testing the algorithms presented, expressed in Mflops, is taken (when available) from Dongarra [10]. In this chapter we extensively refer to the basic notation and to the models presented in Chapter 1.

2.2 Basic Relaxations

In this section we describe the basic combinatorial relaxations for ACVRP and SCVRP that were used within the early branch-and-bound algorithms.

The first type of relaxation may be obtained from the integer linear programming (ILP) formulations of ACVRP and SCVRP (see section 1.3) by dropping the constraints used to impose the connectivity and the capacity requirements, such as the *Capacity-Cut Constraints* (CCCs) or the *Generalized Subtour Elimination Constraints* (GSECs). The resulting problem amounts to an AP or to a b-matching problem for the asymmetric and symmetric case, respectively.

The second type of relaxation leads instead to the solution of cardinality-constrained shortest spanning arborescences and trees for the asymmetric and symmetric case, respectively. These relaxations are obtained by weakening the CCCs or GSECs so as to impose only the connectivity of the solution and by ignoring part of the degree requirements of the vertices.

As we will see at the end of this section, the quality of the lower bounds obtained with these relaxations is generally poor and substantial efforts are needed to improve them.

2.2.1 Bounds Based on Assignment and Matching

Laporte, Mercure, and Nobert [22] proposed the first branch-and-bound algorithm for ACVRP. The algorithm is based on the relaxation obtained from model VRP1 of section 1.3.1 by dropping the CCCs (1.8). The resulting problem is a *Transportation Problem* (TP), calling for a min-cost collection of circuits of G visiting once all the vertices in $V\setminus\{0\}$, and K times vertex 0. This solution can be infeasible for ACVRP since

(i) the total customer demand on a circuit may exceed the vehicle capacity, and

(ii) there may exist "isolated" circuits, i.e., circuits not visiting the depot (vertex 0).

It is well known that determining the optimal TP solution requires $O(n^3)$ time. In practice, it is more effective to transform the problem into an AP defined on the extended complete directed graph $G' = (V', A')$, obtained by adding $K - 1$ copies of the depot vertex as described in section 1.3.2, where the extended cost matrix, c', is defined by (1.55). The resulting relaxation is thus

$$\text{(2.1)} \qquad \text{(AP)} \quad L_{AP} = \min \sum_{i \in V'} \sum_{j \in V'} c'_{ij} x_{ij}$$

subject to

$$\text{(2.2)} \qquad \sum_{i \in V} x_{ij} = 1 \qquad \forall\, j \in V',$$

$$\text{(2.3)} \qquad \sum_{j \in V} x_{ij} = 1 \qquad \forall\, i \in V',$$

$$\text{(2.4)} \qquad x_{ij} \geq 0 \qquad \forall\, i, j \in V'.$$

Several efficient public domain codes for the AP are available; see, e.g., Dell'Amico and Toth [8].

The counterpart, for the symmetric case, of the AP relaxation is the so-called b-matching relaxation, which may be obtained by considering model VRP3 of section 1.3.1 and by removing the CCCs (1.24). The resulting relaxed problem requires the determination of a min-cost collection of cycles covering all the vertices and such that the degree of each vertex i is equal to b_i, where $b_i = 2$ for all the customer vertices, and $b_0 = 2K$ for the depot vertex.

$$\text{(2.5)} \qquad (b\text{-matching}) \qquad L_{bM} = \min \sum_{e \in E} c_e x_e$$

subject to

$$\text{(2.6)} \qquad \sum_{e \in \delta(i)} x_e = b_i \qquad \forall\, i \in V,$$

$$\text{(2.7)} \qquad x_e \in \{0, 1\} \qquad \forall\, e \notin \delta(0),$$

$$\text{(2.8)} \qquad x_e \in \{0, 1, 2\} \qquad \forall\, e \in \delta(0).$$

This relaxation was used by Miller [25], after the development of efficient algorithms for the b-matching problem (see, e.g., Miller and Pekny [27]), which can solve it in time $O(|V|^2|E|)$. Similar to what may happen to the AP relaxation for the ACVRP, the b-matching solution may be infeasible for SCVRP since

(i) the demand associated with a cycle may exceed the vehicle capacity, and

(ii) some cycle may be isolated, i.e., disconnected from the depot.

Also, in this case it is possible to obtain an equivalent 2-matching relaxation by adding $K - 1$ copies of the depot.

2.2.2 Bounds Based on Arborescences and Trees

An alternative combinatorial relaxation for the ACVRP is based on the solution of degree-constrained spanning arborescences. This relaxation may be obtained from model VRP1 by (i) removing the outdegree constraints (1.5) for all the customer vertices and (ii) weakening the CCCs (1.8) so as to impose only the connectivity of the solution, i.e., by replacing the right-hand side with 1. The resulting relaxed problem, called the *K-Shortest Spanning Arborescence* problem (KSSA), is

$$\text{(KSSA)} \qquad L_{KSSA} = \min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} \tag{2.9}$$

subject to

$$\sum_{i \in V} x_{ij} = 1 \qquad \forall\, j \in V \setminus \{0\}, \tag{2.10}$$

$$\sum_{i \in V} x_{i0} = K, \tag{2.11}$$

$$\sum_{j \in V} x_{0j} = K, \tag{2.12}$$

$$\sum_{i \notin S} \sum_{j \in S} x_{ij} \geq 1 \qquad \forall\, S \subseteq V \setminus \{0\},\ S \neq \emptyset, \tag{2.13}$$

$$x_{ij} \in \{0, 1\} \qquad \forall\, i, j \in V. \tag{2.14}$$

The KSSA can be effectively solved by considering two separate subproblems:

(i) the determination of a min-cost spanning arborescence with outdegree K at the depot vertex, defined by (2.9), (2.10), (2.12), (2.13), and the continuous relaxation of (2.14), with variables x_{ij} for $i \in V$, $j \in V \setminus \{0\}$, and

(ii) the determination of a set of K min-cost arcs entering the depot, defined by (2.9), (2.11), and the continuous relaxation of (2.14), with variables x_{i0} for $i \in V$.

Therefore, L_{KSSA} can be determined in $O(n^2)$ since the first subproblem can be solved in $O(n^2)$ time (see Gabow and Tarjan [16] and Toth and Vigo [28]), while the second subproblem clearly requires $O(n)$ time.

A similar lower bound may be obtained by considering the antiarborescence rooted at the depot (KSSAA), in which the branches are union of paths starting from the customers and directed toward the depot, whereas in the KSSA the paths are oriented in the opposite way. It is easy to see that the L_{KSSAA} bound may be obtained by computing the KSSA on the transpose of the original cost matrix. In the following, we use the best of these two bounds, defined as

$$L'_{KSSA} = \max\{L_{KSSA}, L_{KSSAA}\}. \tag{2.15}$$

The above-described lower bound was never used within branch-and-bound algorithms, and the preliminary computational results discussed in the next section show that its quality is generally poor and inferior to that of the lower bound L_{AP}. However, it should be mentioned that for a problem closely related to the SCVRP and ACVRP, such as the symmetric and asymmetric VRP with backhauls (see Chapter 8), Toth and Vigo [29] successfully used a Lagrangian relaxation based on the solution of KSSAs, solving to optimality problems with up to 100 customers.

Several relaxations based on spanning trees were proposed for SCVRP by extending the well-known 1-tree relaxation proposed by Held and Karp [19] for the TSP. The earliest branch-and-bound algorithm based on such relaxations, which proved to be able to solve small size instances, was proposed by Christofides, Mingozzi, and Toth [7]. More recently, Fisher [14] presented another tree-based relaxation that requires the determination of a so-called K-tree, defined as a min-cost set of $n + K$ edges spanning the graph. The approach used by Fisher is based on formulation VRP3 of section 1.3.1 with the additional assumption that single-customer routes are not allowed. This is imposed by defining as binary all the variables associated with edges incident into the depot. However, as Fisher observed, in many cases this assumption is not constraining (see section 1.3.2 for a discussion).

Fisher modeled the SCVRP as the problem of determining a K-tree with degree equal to $2K$ at the depot vertex, and with additional constraints imposing the vehicle capacity requirements and the degree of each customer vertex, which must be equal to 2.

The determination of a K-tree with degree $2K$ at the depot requires $O(n^3)$ time (see Fisher [15]). This degree-constrained K-tree relaxation may easily be obtained by considering formulation VRP3 and by removing the degree constraints (1.22) for the customer vertices and weakening the CCCs (1.24) into connectivity constraints by replacing the right-hand side with 1. The resulting relaxed problem is

$$(K\text{-tree}) \qquad \min \sum_{e \in E} c_e x_e \tag{2.16}$$

subject to

$$\sum_{e \in \delta(0)} x_e = 2K, \tag{2.17}$$

$$\sum_{e \in \delta(S)} x_e \geq 1 \qquad S \subseteq V \setminus \{0\}, S \neq \emptyset, \tag{2.18}$$

$$x_e \in \{0, 1\} \qquad \forall e \in E. \tag{2.19}$$

It can easily be seen that the K-tree solution may be infeasible for SCVRP because some vertices may have degree different from 2. Moreover, the demand associated with the branches leaving the depot may exceed the vehicle capacity.

2.2.3 Comparison of the Basic Relaxations

The basic relaxations of ACVRP and SCVRP presented in the previous sections have in general a poor quality, as shown by the results presented in this section, obtained by considering widely used test instances from the literature.

Table 2.1 reports the percentage ratios of the different lower bound values for ACVRP with respect to the optimal solution value, when applied to the ACVRP real-world instances of pharmaceutical and herbalist's product delivery in downtown Bologna, described by Fischetti, Toth, and Vigo [13]. In particular, the table contains the ratios corresponding to L_{AP}, L'_{KSSA}, and the overall additive bound L_{ADD}, which is described in section 2.3.1. The average gap, over the eight instances, of the lower bound with respect to the optimal

Table 2.1. *Percentage ratios of different ACVRP lower bounds with respect to the optimal solution value on real-world instances.*

Problem	n	K	$\%L_{AP}$	$\%L'_{KSSA}$	$\%L_{ADD}$
A034-02v	33	2	85.8	78.7	90.1
A036-03v	35	3	90.9	75.2	93.2
A039-03v	38	3	93.8	77.6	96.1
A045-03v	44	3	93.4	75.6	95.7
A048-03v	47	3	93.6	79.0	97.2
A056-03v	55	3	88.5	75.4	94.3
A065-03v	64	3	92.6	75.6	95.5
A071-03v	70	3	91.7	79.3	94.6
			91.3	77.1	94.6

solution value is about 8.7% for L_{AP} and 22.9% for L'_{KSSA}. As a consequence, none of these instances were solved by a branch-and-bound based on such basic relaxations, whereas they were solved by adopting the L_{ADD} bound, whose average ratio is 5.4%. Moreover, the computational experiments described by Fischetti, Toth, and Vigo [13] show that on randomly generated instances the gap was normally much smaller, being equal to 2% to 5% for L_{AP} and to 1% to 2% for L_{ADD}.

Table 2.2 reports the average percentage ratios of the basic lower bounds L_{KT} and L_{bM} with respect to the optimal or the best-known solution value, for a set of widely used Euclidean CVRP instances from the literature. The table also reports the ratios of L_{AP}, L'_{KSSA} and of the overall additive lower bound L_{ADD} by Fischetti, Toth, and Vigo [13], which are clearly valid lower bounds for SCVRP as well.

The L_{KT} values are those reported by Fisher [14], who used real-valued cost matrices. The best-known solution values used to compute the ratios are those reported by Toth and Vigo [30], which were obtained by using real-valued cost matrices. The L_{bM} values were computed with the CPLEX 6.0 ILP solver. All the remaining lower bound values were

Table 2.2. *Percentage ratios of different basic SCVRP lower bounds with respect to the best known solution value of Euclidean instances.*

Problem	n	K	$\%L_{bM}$	$\%L^1_{KT}$	$\%L'_{KSSA}$	$\%L_{AP}$	$\%L_{ADD}$
E045-04f	44	4	71.4	62.6 *	62.2	57.4	70.3
E051-05e	50	5	87.9	84.9	79.4	80.9	87.5
E072-04f	71	4	80.9	77.7	72.0	69.8	77.9
E076-10e	75	10	76.7	76.2	69.2	71.0	76.1
E101-08e	100	8	86.4	81.5	77.5	80.7	86.1
E101-10c	100	10	70.3	77.6 *	72.2	66.5	69.6
E135-07f	134	7	63.4	59.2	57.5	47.5	60.3
E151-12c	150	12	80.5	78.4 *	73.6	68.6	77.6
E200-16c	199	16	72.4	74.1	66.4	64.6	72.2
			76.7	74.7	70.0	67.4	75.5

^{1}Single-customer routes not allowed.

*May include single-customer routes.

computed by using integer cost matrices, where the arc cost is defined as the real cost multiplied by 10,000 and rounded to the nearest integer. The final value is then scaled down by dividing it by 10,000. It should be recalled that the problem considered by Fisher in [14] was slightly different from what we defined as CVRP, since the single-customer routes were not allowed. In particular, among the instances reported in Table 2.2, those marked with an asterisk may include single-customer routes. As a consequence, the L_{KT} values computed by Fisher for these instances may by slightly larger than those that could be obtained in the case where single-customer routes are allowed.

By observing Table 2.2, it can be noted that none of the basic relaxations reaches a quality sufficient to solve moderate-size problems. As an example, we used the Fischetti, Toth, and Vigo code FTV, proposed for the ACVRP and based on the additive bound L_{ADD}: the largest SCVRP instance it could solve included 47 customers (i.e., problem E048-04y not included in the table), and some problems with 25 to 30 customers were not solved to optimality.

2.3 Better Relaxations

As discussed in the previous section, the basic combinatorial relaxations available for both ACVRP and SCVRP have a poor quality, and, when used within branch-and-bound approaches, they allow for the optimal solution of small instances only. Therefore, different improved bounding techniques were proposed, which considerably increased the size of the instances solvable by branch-and-bound algorithms. In particular, for the ACVRP we examine the additive bounding procedures proposed by Fischetti, Toth, and Vigo [13], whereas for the SCVRP we describe the bounding procedures based on Lagrangian relaxation proposed by Fisher [14] and Miller [25]. We also describe the bound based on the set partitioning formulation proposed by Hadjiconstantinou, Christofides, and Mingozzi [18].

2.3.1 Additive Bounds for ACVRP

The following two relaxations were introduced by Fischetti, Toth, and Vigo [13], who embedded them into overall additive bounding procedures. The additive approach was proposed by Fischetti and Toth [12] and allows for the combination of different lower bounding procedures, each exploiting different substructures of the considered problem. When applied to a minimization problem of the form $\min\{cx : x \in F\}$, each procedure returns a lower bound, ρ, and a *residual cost matrix*, $\tilde{c}$, such that

$$\begin{aligned} &\tilde{c} \geq 0, \\ &\rho + \tilde{c}x \leq cx, \qquad x \in F. \end{aligned}$$

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 different bounding procedures are applied in sequence, and each of them uses as costs the residual cost matrix returned by the previous procedure (obviously, the first procedure starts with the original cost matrix). The overall additive lower bound is given by the sum of the lower bounds obtained by the different procedures. It can easily be shown that if the lower bounding procedures are based on linear programming relaxations, as those described for ACVRP (i.e., AP and KSSA), the linear programming reduced costs are valid residual costs. For further details see Fischetti, Toth, and Vigo [13] and Fischetti and Toth [12].

2.3.1.1 Disjunctive Lower Bound

The first relaxation described by Fischetti, Toth, and Vigo [13] is based on a disjunction on infeasible arc subsets. A given arc subset $B \subset A$ is called *infeasible* if no feasible solution to ACVRP can use all its arcs, i.e., when

$$\sum_{(a,b)\in B} x_{ab} \leq |B| - 1 \tag{2.20}$$

is a valid inequality for ACVRP. For any given (minimal) infeasible arc subset $B \subset A$, the following logical disjunction holds for each $x \in F$, where F is the set of all the feasible ACVRP solutions:

$$\bigvee_{(a,b)\in B} \left(x \in Q^{ab} = \left\{ x \in \Re^A : x_{ab} = 0 \right\} \right). \tag{2.21}$$

Then, $|B|$ restricted problems can be defined, each denoted as RP^{ab}, by including the additional condition $x_{ab} = 0$, imposed for a different arc $(a, b) \in B$. For each RP^{ab}, a valid lower bound, ϑ^{ab}, is computed through the AP relaxation described in the previous section, with $c_{ab} = M \equiv +\infty$ to impose $x_{ab} = 0$. The disjunctive bound

$$L_D = \min \left\{ \vartheta^{ab} : (a, b) \in B \right\} \tag{2.22}$$

clearly dominates the lower bound L_{AP} based on AP since $\vartheta^{ab} \geq L_{AP}$ for all $(a, b) \in B$.

A possible way to determine infeasible arc subsets B, used in [13], is the following. First solve the AP relaxation with no additional constraints, and store the corresponding optimal solution $(x^*_{ij} : i, j \in V')$. If x^* is feasible for ACVRP, then clearly L_{AP} cannot be improved; otherwise, a suitable infeasible arc subset B is chosen to possibly improve it. Note that imposing $x_{ab} = 0$ for any $(a, b) \in A$ such that $x^*_{ab} = 0$ would produce $\vartheta^{ab} = L_{AP}$, hence a disjunctive bound $L_D = L_{AP}$. Therefore, B is chosen as a subset of $A^* = \left\{ (i, j) \in A' : x^*_{ij} = 1 \right\}$, if any, corresponding to one of the following cases:

(i) a circuit disconnected from the depot vertex,

(ii) a sequence of customer vertices whose total demand exceeds C,

(iii) a feasible circuit that leaves uncovered a set of customers, S, whose total demand cannot be served by the remaining $K - 1$ vehicles, i.e., such that $r(S) > K - 1$, where $r(S)$ represents the minimum number of vehicles needed to serve all the customers in S.

Different choices of the infeasible arc subset B lead to different lower bounds. Therefore, Fischetti, Toth, and Vigo [13] used an overall additive bounding procedure, called ADD_DISJ, which considers, in sequence, different infeasible arc subsets so as to produce a possibly better overall lower bound.

Procedure ADD_DISJ starts by solving the AP relaxation with no additional constraints and defines the initial lower bound as L_{AP} and the arc set A^* as the arcs used in the optimal AP solution. Then, iteratively, an infeasible subset B, if any, is chosen from A^* and used for the computation of the disjunctive lower bound returning a lower bound L_D and the corresponding residual cost matrix. The current additive lower bound is increased by L_D and the set A^* is updated by removing from it all the arcs whose corresponding variables are not equal to 1 in the current optimal solution of the disjunctive bound. The process is iterated until A^* does not contain further infeasible arc subsets. Procedure ADD_DISJ can be implemented, through parametric techniques, to have an overall time complexity equal to $O(n^4)$.

2.3.1.2 Lower Bound Based on Min-Cost Flow

The second lower bound described by Fischetti, Toth, and Vigo [13] is a projective bound based on a min-cost flow relaxation of ACVRP. Let $\{S_0, \dots, S_m\}$ be a given partition of V with $0 \in S_0$, and define

$$A_1 = \bigcup_{h=0}^{m} E(S_h), \qquad A_2 = A \setminus A_1,$$

where $E(S_h)$ is the set of arcs internal to set S_h. In other words, A is partitioned into $\{A_1, A_2\}$, where A_1 contains the arcs internal to the subsets S_h, and A_2 contains those connecting vertices belonging to different S_h's.

In the following, a lower bound L_P based on projection is described. The bound is given by $L_P = \vartheta_1 + \vartheta_2$, where ϑ_t, $t = 1, 2$, is a lower bound on $\sum(c_{ij} : (i, j) \in A^* \cap A_t)$ for every (optimal) ACVRP solution $A^* \subset A$.

The contribution to L_P of the arcs in A_1 (internal to the given subsets S_h) is initially neglected, i.e., ϑ_1 is set equal to 0. The rationale for this choice is clarified later. As to ϑ_2, this is computed by solving the following linear programming relaxation, called R1, obtained from model VRP1 by weakening degree equations (1.4)–(1.7) into inequalities, to take into account the removal of the arcs in A_1, and imposing the CCCs (1.8) and (1.11) only for the subsets $S_0, S_1, \dots, S_m$. The model of R1 is

$$\text{(R1)} \qquad \vartheta_2 = \min \sum_{(i,j) \in A_2} c_{ij} x_{ij} \tag{2.23}$$

subject to

$$\sum_{i \in V : (i,j) \in A_2} x_{ij} \le \begin{cases} 1 & \forall\, j \in V \setminus \{0\}, \\ K, & j = 0, \end{cases} \tag{2.24}$$

$$\sum_{j \in V : (i,j) \in A_2} x_{ij} \le \begin{cases} 1 & \forall\, i \in V \setminus \{0\}, \\ K, & i = 0, \end{cases} \tag{2.25}$$

$$\sum_{i \notin S_h} \sum_{j \in S_h} x_{ij} = \sum_{i \in S_h} \sum_{j \notin S_h} x_{ij} \ge \begin{cases} r(V \setminus S_h), & h = 0, \\ r(S_h) & \forall\, h = 1, \dots, m, \end{cases} \tag{2.26}$$

$$x_{ij} \in \{0, 1\} \qquad \forall\, (i, j) \in A_2. \tag{2.27}$$

This model can be solved efficiently, since it can be viewed as an instance of a min-cost flow problem on an auxiliary layered network, as illustrated in Figure 2.1. The network contains $2(n + m + 2)$ vertices, namely,

- two vertices, say i^+ and i^-, for all $i \in V$;
- two vertices, say a_h and b_h, for all $h = 1, \dots, m$;
- a source vertex, s, and a sink vertex, t.

The arcs in the network, and the associated capacities and costs, are

- for all $(i, j) \in A_2$: arc (i^+, j^-) with cost c_{ij} and capacity $+\infty$;
- for all $h = 0, \dots, m$: arcs (a_h, i^+) and (i^-, b_h) for all $i \in S_h$, with cost 0 and capacity 1 (if $i \neq 0$) or K (if $i = 0$);
- for all $h = 0, \dots, m$: arc (a_h, b_h) with cost 0 and capacity $|S_h| - r(S_h)$ (if $h \neq 1$) or $|S_0| + K - r(V \setminus S_0)$ (if $h = 0$);
- for all $h = 0, \dots, m$: arcs (s, a_h) and (b_h, t), both with cost 0 and capacity $|S_h|$ (if $h \neq 1$) or $|S_0| + K$ (if $h = 0$).

It can easily be seen that finding the min-cost s-t flow of value $n + K$ on this network actually solves relaxation R1. The worst-case time complexity for the computation of ϑ_2, and of the corresponding residual costs, is $O(n^3)$ by using a specialized algorithm based on successive shortest path computations.

Different choices of the vertex partition $\{S_0, \dots, S_m\}$ lead to different lower bounds. Note that choosing $S_h = \{h\}$ for all $h \in V$ produces a relaxation R1 that coincides with the AP relaxation of section 2.2.1. When, on the other hand, nonsingleton S_h's are present, relaxation R1 can take into account the associated CCCs (that are, instead, neglected by AP), while losing a possible contribution to the lower bound of the arcs inside S_h (which belong to A_1) and weakening the degree constraints of the vertices in S_h. Fischetti, Toth, and Vigo [13] used, in sequence, different partitions obtaining an overall additive procedure, called ADD_FLOW.

The procedure is initialized with the partition $S_h = \{h\}$ for all $h \in V$ (i.e., with the AP relaxation). At each iteration of the additive scheme, relaxation R1 is solved, the current lower bound is increased, and the current costs are reduced accordingly. Then a convenient collection of subsets $S_{h_1}, \dots, S_{h_r}$ (with $r \geq 2$) belonging to the current partition is selected and the subsets are replaced with their union, say, S^*. The choice of this collection is made to produce an infeasible set S^*, i.e., a vertex set whose associated CCC is violated by the solution of the current relaxation R1. This, hopefully, produces an increase of the additive lower bound in the next iteration. The additive scheme ends when either $m = 1$ or no infeasible S^* is detected.

Procedure ADD_FLOW takes $O(n^4)$ time, and the resulting additive lower bound clearly dominates bound L_{AP}, which is used to initialize it. On the other hand, no dominance relation exists between ADD_FLOW and procedure ADD_DISJ. Therefore, Fischetti, Toth, and Vigo proposed to apply procedures ADD_DISJ and ADD_FLOW in sequence, again in an additive fashion. To reduce the average overall computing time, procedure ADD_FLOW

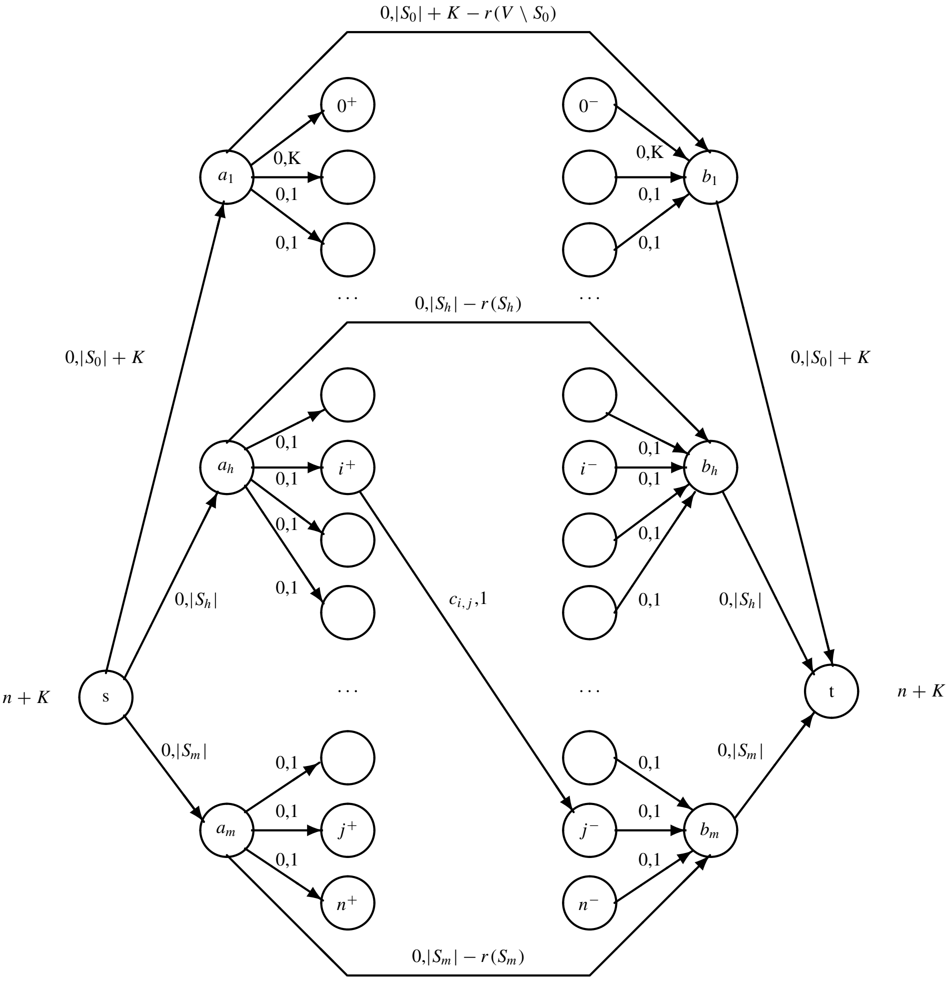

Figure 2.1. *The auxiliary layered network for relaxation* R1.

was stopped when no increase of the current additive lower bound L_{ADD} was observed for five consecutive iterations.

Table 2.1 reports the percentage ratios of the overall additive bounding procedure when applied to the ACVRP real-world instances of pharmaceutical and herbalist's product delivery in downtown Bologna. It can be noted that the additive procedures considerably improve the L_{AP} lower bound.

2.3.2 Further Lower Bounds for ACVRP

Other bounds for the ACVRP may be derived by generalizing the methods proposed for the symmetric case. For example, Fisher [14] proposed a way to extend to ACVRP the

Lagrangian bound based on a K-tree derived for the SCVRP (described in section 2.3.3). In this extension the Lagrangian problem calls for the determination of an undirected K-tree on the undirected graph obtained by replacing each pair of directed arcs (i, j) and (j, i) with a single edge (i, j) with cost $c'_{ij} = \min\{c_{ij}, c_{ji}\}$. No computational testing for this bound was presented by Fisher [14].

Possibly better bounds may be obtained by explicitly considering the asymmetry of the problem, i.e., by using K-arborescences rather than K-trees and by strengthening the bound in a Lagrangian fashion as proposed by Toth and Vigo [28, 29] for the capacitated shortest spanning arborescence problem and the VRPB, respectively.

2.3.3 Lagrangian Lower Bounds for SCVRP

Fisher [14] and Miller [25] proposed to strengthen the basic SCVRP relaxations by dualizing, in a Lagrangian fashion, some of the relaxed constraints. In particular, Fisher included in the objective function the degree constraints (1.22) and some of the CCCs (1.24), whereas Miller included some of the GSECs (1.27). Note that Fisher did not allow single-customer routes. As in related problems, good values for the Lagrangian multipliers associated with the relaxed constraints are determined by using a standard subgradient optimization procedure (see, e.g., Held and Karp [19] and Held, Wolfe, and Crowder [20]).

The main difficulty associated with these relaxations is represented by the exponential cardinality of the set of relaxed constraints (i.e., the CCCs and the GSECs) which does not allow for the explicit inclusion of all of them into the objective function. To this end, both Fisher and Miller proposed to include only a limited family $\mathcal{F}$ of CCCs or GSECs and to iteratively add to the Lagrangian relaxation the constraints violated by the current solution of the Lagrangian problem. In particular, at each iteration of the subgradient optimization procedure, the arcs incident to the depot in the current Lagrangian solution are removed. Violated constraints (i.e., CCCs or GSECs, depending on the approach), if any, are *separated* (i.e., detected) by examining the connected components obtained in this way. This separation routine is exact, i.e., if a constraint associated with, say, vertex set S is violated by the current Lagrangian solution, then there is a connected component of that solution spanning all the vertices in S and violating the constraint. The new constraints are added to the Lagrangian problem, i.e., to $\mathcal{F}$, with an associated multiplier, and the process is iterated until no violated constraint is detected (hence the Lagrangian solution is feasible) or a prefixed number of subgradient iterations has been executed. Slack constraints are periodically *purged* (i.e., removed) from $\mathcal{F}$.

Fisher [14] initialized $\mathcal{F}$ with an explicit set of constraints containing the customer subsets nested around $K + 3$ *seed customers*. The seeds were chosen as the K customers farthest from the depot in the routes corresponding to an initial feasible solution, whereas the last three were the customers maximally distant from the depot and the other seeds. For each seed, 60 sets were generated by including customers according to increasing distances from the seed. After 50 subgradient iterations, new sets were added to $\mathcal{F}$ by identifying violated CCCs in the current Lagrangian solution as previously explained. The step size used in the subgradient optimization method was initially set to 2 and was reduced by a factor of 0.75 if the lower bound was not improved in the last 30 iterations. The number of iterations of the subgradient optimization procedure performed at the root node of the branch-and-bound algorithm ranged between 2000 and 3000. The overall Lagrangian bound, LAG_{KT}, considerably improved the basic K-tree relaxation and was, on average, larger than 99% of the optimal solution value for the three Euclidean instances with $n \leq 100$ solved to optimality in Fisher [14] (see section 2.3.5).

Miller [25] initialized $\mathcal{F}$ as the empty set, and at each iteration of the subgradient procedure detected violated GSECs and additional constraints belonging to the following two classes. The first type of constraint is given by additional GSECs which were added when the current Lagrangian solution $\bar{x}$ contains k (with $k \geq 2$) overloaded routes. The customer set of these new GSECs is the union of the sets $S_1, \ldots, S_k$ associated with the GSECs violated by $\bar{x}$. This increases the probability that arcs connecting customers belonging to the overloaded routes to those in sets $S_1, \ldots, S_k$ are selected by the b-matching solution. The second type of constraint was added when $\bar{x}$ contained routes that were underloaded, i.e., whose associated load was smaller than the minimum vehicle load $C_{\min}$ defined by (1.61). In this case for each such set S, with $0 \in S$, a constraint of the form

$$\sum_{e \in E(S)} x_e \leq |S| - 1, \tag{2.28}$$

which breaks the current underloaded route in $\bar{x}$, was added to $\mathcal{F}$. The procedure was iterated until no improvement was obtained over 50 subgradient iterations. The step size is modified in an adaptive way every five subgradient iterations to produce a slight oscillation in lower bound values during the progress of the subgradient procedure. If the lower bound is monotonically increasing, the step size is increased by 50%; if the oscillation of the lower bound value is greater than 2%, the step size is reduced by 20%, and when the oscillation is smaller than 0.5% it is increased by 10%. The final Lagrangian bound LAG_{bM} of Miller is considerably tight, being on average 98% of the optimal solution value for the eight problems with $n \leq 50$ solved in Miller [25] (see section 2.3.5).

2.3.4 Lower Bounds from a Set-Partitioning Formulation

Hadjiconstantinou, Christofides, and Mingozzi [18] proposed a branch-and-bound algorithm where the lower bound is computed by heuristically solving the dual of the linear programming relaxation of the *Set-Partitioning* (SP) formulation of the SCVRP.

As described in section 1.3.4, the SP formulation of the VRP was originally proposed by Balinski and Quandt [2] and uses a possibly exponential number of binary variables, each associated with a different feasible circuit of G.

Model VRP8 of section 1.3.4 is a very general one and may easily take into account several constraints (as, for example, time windows), since route feasibility is implicitly considered in the definition of set $\mathcal{H}$. Agarwal, Mathur, and Salkin [1] proposed an exact algorithm for SCVRP based on the SP approach, whereas several successful applications of this technique to tightly constrained VRPs were reported by Desrosiers et al. [9]. (See also Chapters 4 and 7 of the present volume.) Moreover, the linear programming relaxation of this formulation typically is very tight.

Hadjiconstantinou, Christofides, and Mingozzi [18] proposed to obtain a valid lower bound for SCVRP by considering the dual of the linear relaxation of model VRP8:

$$\text{(DVRP8)} \quad \max \; K\pi_0 + \sum_{i=1}^{n} \pi_i \tag{2.29}$$

subject to

$$\pi_0 + \sum_{i \in H_j} \pi_i \le c_j \qquad \forall\, j = 1, \dots, M, \tag{2.30}$$

$$\pi_i \quad \text{unrestricted} \qquad \forall\, i = 0, \dots, n. \tag{2.31}$$

where $\pi_i, i = 1, \dots, n$, are the dual variables associated with the partitioning constraints (1.72) and π_0 is that associated with constraint (1.73). It is clear that any feasible solution to problem DVRP8 provides a valid lower bound for SCVRP. Hadjiconstantinou, Christofides, and Mingozzi [18] determined the heuristic dual solutions by combining in an additive way two relaxations of the original problem: the q-path relaxation proposed by Christofides, Mingozzi, and Toth [7], and the K-shortest path relaxation proposed by Christofides and Mingozzi [6]. The proposed approach was able to solve randomly generated Euclidean instances with up to 30 vertices and instances proposed in the literature with up to 50 vertices, within a time limit of 12 hours on a Silicon Graphics Indigo R4000 (12 Mflops). The percentage ratios of the overall bound L_{SP} computed in [18] on some test instances from the literature are reported in Table 2.3.

2.3.5 Comparison of the Improved Lower Bounds

The lower bounds described in this section are considerably better than those corresponding to the basic relaxations on which they are based, and they allow for the solution of quite larger problems.

We presented in Table 2.1 the percentage ratios of the lower bound obtained by the additive bounding procedure for ACVRP described in section 2.3.1. As to the symmetric case, a direct computational comparison of the effectiveness of the bounds presented in this chapter is not possible. In fact, as illustrated in Table 2.3, each author either considered a slightly different problem (e.g., in Fisher [14] single-customer routes were not allowed, whereas Miller [25] allowed them) or solved a completely different set of instances. The only instance that has been tackled by almost all the authors is the 50-customers Euclidean problem described by Christofides and Eilon [5], indicated as E051-05e. However, also in this case all the authors defined the cost matrix in a different way. In particular, Table 2.3 includes the Lagrangian bounds by Fisher and Miller described in section 2.3.3, compared with the corresponding basic relaxations, the bound L_{SP} based on the SP formulation by Hadjiconstantinou, Christofides, and Mingozzi described in section 2.3.4, and the overall additive bound L_{ADD} of section 2.3.1. In Table 2.3 an asterisk denotes the instances that were solved to optimality by the corresponding branch-and-bound code.

We included in the table the L_{KT} and the Lagrangian bound LAG_{KT} values computed by Fisher [14] by using real-valued cost matrices, and we compared the bounds with

Table 2.3. *Comparison of the percentage ratios of the basic and improved lower bounds for SCVRP with respect to different test instances.*

Problem	n	K	$\%L^1_{KT}$	$\%LAG^1_{KT}$	$\%L^2_{bM}$	$\%LAG^2_{bM}$	$\%L^3_{SP}$
`S007-02a`	6	2				100.0 *	
`S013-04d`	12	4				96.8 *	
`E016-05m`	15	5					97.6 *
`E021-04m`	20	4					100.0 *
`E022-04g`	21	4			90.1	99.7 *	
`E023-03g`	22	3			96.5	100.0 *	
`E026-08m`	25	8					100.0 *
`E030-03g`	29	3			71.7	95.3 *	
`S031-07w`	30	7				96.0 *	
`E031-09h`	30	9					97.9 *
`E033-03n`	32	3			86.5	98.9 *	
`E036-11h`	35	11					99.5 *
`E041-14h`	40	14					98.9 *
`E045-04f`	44	4	62.6	99.6 *			
`E051-05e`	50	5	84.9	96.7	92.9	96.9 *	98.5 *
`E072-04f`	71	4	77.7	98.3 *			
`E076-10e`	75	10	76.2	90.5			97.6
`E101-08e`	100	8	81.5	95.1			95.9
`E101-10c`	100	10	77.6	99.8 *			
`E135-07f`	134	7	59.2	97.4			
`E151-12c`	150	12	78.4	90.7			97.2
`E200-16c`	199	16	74.1	84.7			

[1]Real-valued costs and single-customer routes not allowed.

[2]Rounded integer costs.

[3]Real costs multiplied by 10,000 and rounded to the nearest integer.

*Solved to optimality.

respect to the optimal or the best-known-solution values determined by using real-valued cost matrices and reported by Toth and Vigo [30]. Over the nine instances considered by Fisher, the average ratio of L_{KT} is 74.7% while that of LAG_{KT} is 94.8%.

The results reported in Table 2.3 relative to the Lagrangian bound LAG_{bM} are those obtained by Miller [25] by using integer rounded cost matrices and whose overall ratio is about 98%. The table also includes some values of the pure b-matching relaxation computed by Miller [26].

Finally, the L_{SP} values are computed by using integer costs for the arcs, defined as the Euclidean distance between the endpoints multiplied by 10^4 and then rounded to the nearest integer. The ratios for these bounds are obtained by comparing the scaled-down value of the lower bound with the optimal or the best-known-solution value determined by using real-valued cost matrices.

2.4 Structure of the Branch-and-Bound Algorithms for CVRP

We now briefly describe the main ingredients of the branch-and-bound algorithms used for the exact solution of ACVRP and SCVRP, recently proposed in the literature.

2.4.1 Branching Schemes and Search Strategies

The two algorithms proposed for ACVRP by Laporte, Mercure, and Nobert [22] and by Fischetti, Toth, and Vigo [13] have the same basic structure, derived from that of the algorithm for the asymmetric TSP described by Carpaneto and Toth [4] and originally proposed by Bellmore and Malone [3]: the first one uses as lower bound the AP relaxation (see section 2.2.1), whereas the second uses the additive bounding procedure described in section 2.3.1.

The branching rules used by both algorithms are related to the *subtour elimination scheme* used for the asymmetric TSP, and they handle the relaxed constraints by imposing the connectivity and the capacity requirements of the feasible ACVRP solutions.

At a node ν of the branch-decision tree, let I_ν and F_ν contain the arcs imposed and forbidden in the current solution, respectively (with $I_\nu = \emptyset$ and $F_\nu = \emptyset$ if ν is the root node). Given the set A^* of arcs corresponding to the optimal solution of the current relaxation, a nonimposed arc subset $B := \{(a_1, b_1), (a_2, b_2), \ldots, (a_h, b_h)\} \subset A^*$ on which to branch is chosen.

Fischetti, Toth, and Vigo defined B by considering the subset of A^* with the minimum number of nonimposed arcs among those defining a path or a circuit that is infeasible according to conditions (i), (ii), and (iii) in section 2.3.1. Note that since the additive bounding procedure modifies the objective function of the problem, an optimal solution of the relaxed problem that is feasible for ACVRP is not necessarily optimal for it. Therefore, if A^* defines a feasible ACVRP solution whose cost is greater than the current lower bound value, set B is chosen as the feasible circuit through vertex 0 with the minimum number of nonimposed arcs. Then $h = |B|$ descendant nodes are generated. The subproblem associated with node $\nu_i, i = 1, \ldots, h$, is defined by excluding the ith arc of B and by imposing the arcs up to $i - 1$:

$$I_{\nu_i} := I_\nu \cup \{(a_1, b_1), \ldots, (a_{i-1}, b_{i-1})\}, \tag{2.32}$$

$$F_{\nu_i} := F_\nu \cup \{(a_i, b_i)\}, \tag{2.33}$$

where $I_{\nu_1} := I_\nu$.

Laporte, Mercure, and Nobert defined B as an infeasible subtour according to conditions (i) and (ii) of section 2.2.1 and used a more complex branching rule in which, at each descendant node, at most r arcs of B are simultaneously excluded, where $r := \lceil d(S)/C \rceil$, S is the set of vertices spanned by B, and $d(S)$ represents the sum of the demands of the vertices in S. In this case, since at most $\binom{|B|}{r}$ descendant nodes may be generated, the set B is chosen as the one minimizing $\binom{|B|}{r}$.

These algorithms adopt a *best-bound-first* search strategy, i.e., branching is always executed on the pending node of the branch-decision tree with the smallest lower bound value. This rule allows for the minimization of the number of subproblems solved at the expense of larger memory usage, and it is computationally proved to be more effective than the *depth-first* strategy, where the branching node is selected according to a last-in-first-out rule.

Many branching schemes were used for SCVRP, and in this case almost all are extensions of those used for the TSP. The first scheme we consider, proposed by Christofides, Mingozzi, and Toth [7], is known as *branching on arcs*, and it proceeds by extending partial paths, starting from the depot and finishing at a given vertex. At each node of the branch-decision tree, an arc (i, j) is selected to extend the current partial path, and two descendant nodes are generated: the first node is associated with the inclusion of the selected arc in the solution (i.e., $x_{ij} = 1$), while in the second node the arc is excluded (i.e., $x_{ij} = 0$).

Miller [25] used the same branching scheme, where the arc selected for branching is determined by examining the solution obtained by the Lagrangian relaxation based on b-matching described in section 2.3.3. When a partial path is present in the current subproblem ending, say, with vertex v, the arc (v, h) belonging to the current Lagrangian solution is selected. If the current subproblem does not contain a partially fixed path, e.g., at the root node or when a route has been closed by the last imposed arc, the arc connecting the depot with the unrouted customer j with the largest demand is selected for branching. In this case a third descendant node is also created, by imposing $x_{0j} = 2$, i.e., by considering, if feasible, the route containing only customer j.

Fisher [14] used a mixed scheme where branching on arcs is used whenever no partial path is present in the current subproblem. In this case the currently unserved customer i with the largest demand is chosen and the arc (i, j) is used for branching, where j is the unserved customer closest to i. At the node where arc (i, j) is excluded from the solution, branching on arcs is again used, whereas at the second node the scheme known as *branching on customers* is used. One of the two ending customers, say, v, of the currently imposed sequence of customers is chosen, and branching is performed by enumerating the customers that may be appended to that end of the sequence. A subset T of currently unserved customers is selected (for example, that including the unserved customers closest to v) and $|T| + 1$ nodes are generated. Each of the first $|T|$ nodes corresponds to the inclusion in the solution of a different arc (v, j), $j \in T$, while in the last node all the arcs (v, j), $j \in T$ are excluded.

The mixed branching scheme was used by Fisher to attempt the solution of Euclidean CVRP instances with real distances and about 100 customers, but this proved unsuccessful. In fact, Fisher observed that in instances where many small clusters of close customers exist (as is the case of several instances from the literature), any solutions in which these customers are served contiguously in the same route have almost the same cost. Thus, when the sequence of these customers has to be determined through branching, unless an extremely tight bound is used, it would be very difficult to fathom many of the resulting nodes. Therefore, in Fisher [14] an alternative branching scheme was proposed, aimed at exploiting macro properties of the optimal solution whose violation would have a large impact on the cost, thus allowing the fathoming of the corresponding nodes. To this end, a subset T of currently unserved customers is selected and two descendant nodes are created: at the first node the additional constraint $\sum_{e\in\delta(T)} x_e = 2\lceil d(T)/C\rceil$ is added to the current problem, while at the second node the constraint $\sum_{e\in\delta(T)} x_e \geq 2\lceil d(T)/C\rceil + 2$ is imposed. Some ways to identify suitable subsets, as well as additional dominance rules, were described by Fisher [14].

2.4.2 Reduction, Dominance Rules, and Other Features

Several rules may be used to possibly remove some arcs that cannot belong to an optimal solution, by forbidding their use in the computation of bounds and allowing for the early detection of infeasibilities and dominance relations, thus speeding up the solution of CVRP. Many of these rules are inspired by the work done on the TSP. In the following we refer, for short, to the more general case of the ACVRP and we explicitly remove arcs from A. An often-used alternative way to remove arcs from A, which preserves the completeness of graph G and simplifies the notation, is obtained by setting the cost of the arcs to be removed equal to a very large positive value, say, M, practically equivalent to $+\infty$.

The reduction rules may be applied either to the original problem or to a subproblem associated with a node of the branch-decision tree, where arcs of a given subset I are imposed in the solution, as happens in branch-and-bound and branch-and-cut algorithms. In this case the arcs of I define complete routes and paths, some of which may enter or leave the depot. For reduction purposes, all the customers belonging to the ρ complete routes (with $\rho \geq 0$) induced by I are removed from V. Let $\tilde{G} = (\tilde{V}, \tilde{A})$ be the subgraph of G induced by vertex set $\tilde{V}$ obtained from V by removing all the customers belonging to complete routes in I, and let $\tilde{K} = K - \rho$. Moreover, let $\mathcal{P} = \{P_1, \dots, P_r\}$ be the set of paths induced by I, each defined as an ordered set of vertices, and let h_j and t_j denote the first and the last vertex of path P_j, $j = 1, \dots, r$. To simplify the notation, each customer vertex i covered by no arc in T is represented by a degenerate path $P_j \in \mathcal{P}$ made up by a singleton vertex, where $h_j = t_j = i$. Note that when $I = \emptyset$, then $r = |\mathcal{P}| = n$ and each path is degenerate.

The first type of reduction rule tries to remove from $\tilde{A}$ all the arcs that, if used, would produce infeasible ACVRP solutions:

1. For each arc $(i, j) \in I$, remove from $\tilde{A}$ all the arcs (i, p), $p \in \tilde{V}$ if $i \neq 0$, and (p, j), $p \in \tilde{V}$ if $j \neq 0$.

2. For each nondegenerate path P_i such that $h_i, t_i \neq 0$, remove all the arcs which would form a subtour disconnected from the depot. If $h_i = 0$ (resp., $t_i = 0$), we may remove arc $(t_i, 0)$, (resp., $(0, h_i)$) when

$$d(P_i) < C_{\min} = d(\tilde{V}) - (\tilde{K} - 1)C,$$

i.e., when, on the remaining $\tilde{K} - 1$ vehicles, there is not enough space to load the demand of the other customers.

3. For each pair of paths $P_i, P_j \in \mathcal{P}$ such that $d(P_i) + d(P_j) > C$, remove arc (t_i, h_j) from $\tilde{A}$ if $t_i, h_j \neq 0$.

The second type of reduction rule tries to remove for $\tilde{A}$ the arcs that, if used, would not improve the currently best known solution. For example, let L and U be a lower and an upper bound on the optimal ACVRP solution value, respectively. For each $(i, j) \in \tilde{A}$ let $\overline{c}_{ij}$ be the reduced cost of arc (i, j) associated with the lower bound L. It is well known that the reduced cost of an arc represents a lower bound on the increase of the optimal solution value if this arc is used. Therefore, for each $(i, j) \in \tilde{A}$ if $L + \overline{c}_{ij} \geq U$ we may remove (i, j) from $\tilde{A}$.

Whenever a customer has only one entering or leaving arc belonging to $\tilde{A}$, we may impose this arc (by adding it to I), redefine the set of complete routes and paths in I, and again execute steps 1–3 above.

The performance of the branching schemes may be enhanced by means of a dominance test proposed by Fischetti and Toth [11]. A node of the branch-decision tree where a partial sequence of customers $v, \dots, w$ is fixed can be fathomed if there exists a lower cost ordering of the customers in the sequence starting with v and ending with w. The improved ordering may be heuristically determined, e.g., by means of insertion and exchange procedures.

In addition, several branch-and-bound algorithms include the use of heuristic algorithms that exploit the information associated with the relaxed problems to obtain feasible solutions that may improve the current incumbent solution (see, e.g., Fisher [14] and Fischetti, Toth, and Vigo [13]).

2.4.3 Performance of the Branch-and-Bound Algorithms

Laporte, Mercure, and Nobert [22] used their algorithm LMN to solve, on a VAX 11/780 computer (0.14 Mflops), ACVRP test instances where demands d_j and costs c_{ij} were randomly generated from a uniform distribution in [0, 100] and rounded to the nearest integer. The vehicle capacity was defined as

$$C := (1-\alpha)\max_{j\in V}\{d_j\} + \alpha\, d(V),$$

where α is a real parameter chosen in [0, 1]. The number of available vehicles was defined as $K = K_{\min}$ and was computed by using the trivial BPP lower bound. Note that larger values of α produce larger C and hence smaller values of K. (When $\alpha = 1$, ACVRP reduces to the asymmetric TSP, since $K = 1$.) No monotone correlation between α and the average percentage load of a vehicle, defined as $100\, d(V)/(K\ C)$, can instead be inferred. Laporte, Mercure, and Nobert considered $\alpha = 0.25, 0.50, 0.75$, and 1.0, producing $K = 4, 2, 2$, and 1, respectively.

For each pair (n,α), five instances were generated and algorithm LMN was run by imposing a limit on the total available memory. The LMN algorithm was able to solve instances with up to 90 vertices if $\alpha \geq 0.50$ (i.e., with $K \leq 2$). For the larger values of n, only half or fewer of the instances were actually solved, while with $\alpha = 0.25$ only the instances with 10 vertices and one of those with 20 vertices were solved. The computing times for the most difficult instances solved were above 5000 seconds, whereas no statistics were reported for the nonsolved instances. The algorithm was also tested on instances of the same type but with $K = K_{\min} + 2$ or $K = K_{\min} + 4$. These problems proved much easier than the previous ones.

Fischetti, Toth, and Vigo [13] tested their algorithm FTV on the same class of randomly generated instances used for LMN, with $K = K_{\min}$. Algorithm FTV was able to solve all the instances with up to 300 vertices and up to four vehicles within 1000 CPU seconds on a DECstation 5000/240 (5.3 Mflops). On these instances the additive lower bound considerably improved the AP value. Algorithm FTV was also tested on a class of more realistic problems where the cost matrices were obtained from those of the previous class by triangularizing the costs, i.e., by replacing each c_{ij} with the cost of the shortest path from i to j. The number of vehicles K and the average percentage vehicle load, say, r, were fixed, and the vehicle capacity was defined as $C := \lceil 100 d(V)/(rK) \rceil$. Instances of this type with up to 300 vertices and eight vehicles and with r equal to 80% and 90% were solved, those with $n \geq 150$ being easier than the smaller ones. Algorithm FTV was applied to eight real-world instances with up to 70 vertices and three vehicles, coming from pharmaceutical and herbalist's product delivery in the center of an urban area with several one-way restrictions imposed on the roads. These instances proved more difficult than the randomly generated ones: the computing time and the number of nodes were higher than those required for analogous random instances. Moreover, the average gap, over the eight instances, of the additive bound with respect to the optimal solution value was about 5.5% (that of AP being 8.9%), whereas on random instances the gap was normally much smaller (1% to 2% for the additive bound and 2% to 5% for the AP bound).

The results of branch-and-bound algorithms for symmetric problems were discussed in section 2.3.5. The branch-and-bound algorithm by Miller [25] was applied to Euclidean SCVRP instances from the literature, where the edge costs are computed as the Euclidean distances between the customers and rounded to the nearest integer. The algorithm was able to solve problems with up to 50 customers within 15,000 seconds on a Sun Sparc 2 (4 Mflops). The branch-and-bound algorithm by Fisher [14] was successfully applied to some Euclidean CVRP instances with real-valued cost matrices and with no single customer route allowed. The largest solved instance included 100 customers and was solved within less than 60,000 seconds on an Apollo Domain 3000 computer (0.071 Mflops).

2.5 Distance-Constrained VRP

Exact methods for the Distance-Constrained VRP and CVRP (DVRP and DCVRP, respectively) received relatively little attention in the literature. Moreover, since the seminal articles by Laporte, Nobert, and Desrochers [21, 24], no new exact algorithm specifically designed to handle these problems has been presented. In the following we briefly describe the algorithm presented in [24] for the symmetric version of the more general DCVRP case.

Laporte, Nobert, and Desrochers [24] assumed, as usual, that the travel time and arc cost matrices coincide and are symmetric, i.e., $t_{ij} = c_{ij}$ for each $i, j \in A$, $i < j$, and that no service time is present, i.e., $s_i = 0$ for each $i \in V$. The algorithm is based on an adaptation of formulation VRP2 for SCVRP described in section 1.3. The model is

$$\text{(DCVRP)} \quad \min \sum_{i \in V \setminus \{n\}} \sum_{j > i} c_{ij} x_{ij} \tag{2.34}$$

subject to

$$\sum_{h<i} x_{hi} + \sum_{j>i} x_{ij} = 2 \qquad \forall\, i \in V \setminus \{0\}, \tag{2.35}$$

$$\sum_{j \in V \setminus \{0\}} x_{0j} = 2K, \tag{2.36}$$

$$\sum_{i \in S} \sum_{\substack{j > i \\ j \in S}} x_{ij} \leq |S| - r'(S) \qquad \forall\, S \subseteq V \setminus \{0\}, S \neq \emptyset, \tag{2.37}$$

$$x_{ij} \in \{0, 1\} \qquad \forall\, i, j \in V \setminus \{0\}, i < j, \tag{2.38}$$

$$x_{0j} \in \{0, 1, 2\} \qquad \forall\, j \in V \setminus \{0\}. \tag{2.39}$$

The degree constraints (2.35) and (2.36) impose that exactly two edges are incident into each vertex associated with a customer and that $2K$ edges are incident into the depot vertex, respectively. The GSECs (2.37) impose the connectivity of the solution, the vehicle capacity and the maximum route length requirements, by forcing that a sufficient number of edges leaves each subset of vertices. Given a subset S of customer vertices, the quantity $r'(S)$ represents the minimum number of vehicles needed to serve all the customers in S. This quantity is given by the maximum between $r(S)$, which takes into account the capacity constraints, and the smallest value v satisfying

$$v = \lceil H_v(S)/L \rceil, \qquad v = r(S), \ldots, \min\{K, |S|\}, \tag{2.40}$$

where $H_v(S)$ is the optimal cost of a multiple TSP visiting all customers in S and using exactly v tours passing through the depot. Since the multiple TSP is an NP-hard problem, an approximation from below of the above value may be obtained by using any lower bound on the value of $H_v(S)$.

The lower bound used in [21, 24] is based on the continuous relaxation of model DCVRP in which the GSECs (2.37) are initially removed. The approach adopted by Laporte and others may be seen as a forerunner of the branch-and-cut algorithms, since the initial continuous relaxation is iteratively strengthened by adding violated GSECs and, at the root node of the branch-decision tree, Gomory cuts [17].

The branch-and-bound algorithm described by Laporte, Nobert, and Desrochers [24] was able to consistently solve randomly generated Euclidean DCVRP instances with up to 20 customers and different numbers of vehicles within 500 CPU seconds on a Cyber 173 computer (about 1.5 Mflops). Some larger problems, involving up to 45 customers, were also solved when both the capacity and the maximum distance constraints were loose and few vehicles were available. Non-Euclidean randomly generated problems were also considered in [24]. Laporte, Desrochers, and Nobert [21] specialized the algorithm to the case in which the capacity constraint is not present (DVRP), and they obtained analogous results.

Acknowledgments

This work was supported by Ministero dell'Università e della Ricerca Scientifica e Tecnologica and by Consiglio Nazionale delle Ricerche, Italy. We thank the Laboratory of Operations Research at Dipartimento di Elettronica, Informatica e Sistemistica (DEIS), University of Bologna, for the assistance in the computational experiments.

Bibliography

[1] Y. Agarwal, K. Mathur, and H.M. Salkin. A set-partitioning-based exact algorithm for the vehicle routing problem. *Networks*, 19:731–749, 1989.

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

[3] M. Bellmore and J.C. Malone. Pathology of travelling salesman subtour-elimination algorithms. *Operations Research*, 19:278–307, 1971.

[4] G. Carpaneto and P. Toth. Some new branching and bounding criteria for the asymmetric traveling salesman problem. *Management Science*, 26:736–743, 1980.

[5] N. Christofides and S. Eilon. An algorithm for the vehicle dispatching problem. *Operational Research Quarterly*, 20:309–318, 1969.

[6] N. Christofides and A. Mingozzi. Vehicle routing: Practical and algorithmic aspects. In C.F.H. van Rijn, editor, *Logistics: Where Ends Have to Meet*, Oxford, UK, Pergamon Press, 1989, pp. 30–48.

[7] N. Christofides, A. Mingozzi, and P. Toth. Exact algorithms for the vehicle routing problem based on the spanning tree and shortest path relaxations. *Mathematical Programming*, 20:255–282, 1981.

[8] M. Dell'Amico and P. Toth. Algorithms and codes for dense assignment problems: The state of the art. *Discrete Applied Mathematics*, 100:17–48, 2000.

[9] J. Desrosiers, Y. Dumas, M.M. Solomon, and F. Soumis. Time constrained routing and scheduling. In M.O. Ball, T.L. Magnanti, C.L. Monma, and G.L. Nemhauser, editors, *Network Routing*, *Handbooks in Operations Research and Management Science* 8, North-Holland, Amsterdam, 1995, pp. 35–139.

[10] J.J. Dongarra. Performance of various computers using standard linear equations software. Technical Report CS–89–85, University of Tennessee, Knoxville, 1998.

[11] M. Fischetti and P. Toth. A new dominance procedure for combinatorial optimization. *Operations Research Letters*, 7:181–187, 1988.

[12] M. Fischetti and P. Toth. An additive bounding procedure for combinatorial optimization problems. *Operations Research*, 37:319–328, 1989.

[13] M. Fischetti, P. Toth, and D. Vigo. A branch-and-bound algorithm for the capacitated vehicle routing problem on directed graphs. *Operations Research*, 42:846–859, 1994.

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

[15] M.L. Fisher. A polynomial algorithm for the degree constrained k-tree problem. *Operations Research*, 42:776–780, 1994.

[16] H.N. Gabow and R.E. Tarjan. Efficient algorithms for a family of matroid intersection problems. *Journal of Algorithms*, 5:80–131, 1984.

[17] R.E. Gomory. An algorithm for integer solution to linear programs. In *Recent Advances in Mathematical Programming*, McGraw-Hill, New York, 1963, pp. 269–302.

[18] E. Hadjiconstantinou, N. Christofides, and A. Mingozzi. A new exact algorithm for the vehicle routing problem based on q-paths and k-shortest paths relaxations. *Annals of Operations Research*, 61:21–43, 1995.

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

[20] M. Held, P. Wolfe, and M.P. Crowder. Validation of the subgradient optimization. *Mathematical Programming*, 6:62–88, 1974.

[21] G. Laporte, M. Desrochers, and Y. Nobert. Two exact algorithms for the distance-constrained vehicle routing problem. *Networks*, 14:161–172, 1984.

[22] G. Laporte, H. Mercure, and Y. Nobert. An exact algorithm for the asymmetrical capacitated vehicle routing problem. *Networks*, 16:33–46, 1986.

[23] G. Laporte and Y. Nobert. Exact algorithms for the vehicle routing problem. *Annals of Discrete Mathematics*, 31:147–184, 1987.

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

[25] D.L. Miller. A matching based exact algorithm for capacitated vehicle routing problems. *ORSA Journal on Computing*, 7:1–9, 1995.

[26] D.L. Miller. *Personal communication*, 1997.

[27] D.L. Miller and J.F. Pekny. A staged primal-dual algorithm for perfect b-matching with edge capacities. *ORSA Journal on Computing*, 7:298–320, 1995.

[28] P. Toth and D. Vigo. An exact algorithm for the capacitated shortest spanning arborescence. *Annals of Operations Research*, 61:121–142, 1995.

[29] P. Toth and D. Vigo. An exact algorithm for the vehicle routing problem with backhauls. *Transportation Science*, 31:372–385, 1997.

[30] P. Toth and D. Vigo. The granular tabu search and its application to the vehicle routing problem. *INFORMS Journal on Computing*, 15:333–346, 2003.