ISTITUTO DI ANALISI DEI SISTEMI ED INFORMATICA
CONSIGLIO NAZIONALE DELLE RICERCHE

D. Naddef, G. Rinaldi

BRANCH AND CUT ALGORITHMS FOR THE VEHICLE ROUTING PROBLEM

R. 515 Aprile 1999

Denis Naddef – LID-IMAG, Institut National Polytechnique de Grenoble - ENSIMAG-Antenne de Montbonnot ZIRST - 51 Avenue Kuntzmann, 38330 Monbonnot St. Martin, France, (denis.naddef@imag.fr).

Giovanni Rinaldi – Istituto di Analisi dei Sistemi ed Informatica "Antonio Ruberti" del CNR, viale Manzoni 30, 00185 Roma, Italy, (rinaldi@iasi.cnr.it).

ISSN: 1128–3378

Collana dei Rapporti dell'Istituto di Analisi dei Sistemi ed Informatica "Antonio Ruberti", CNR

viale Manzoni 30, 00185 ROMA, Italy

tel. ++39-06-77161
fax ++39-06-7716461
email: iasi@iasi.rm.cnr.it
URL: http://www.iasi.rm.cnr.it

Abstract

This article will be chapter 3 of the volume "The Vehicle Routing Problem" edited by Daniele Vigo and Paolo Toth for the *SIAM Monographs on Discrete Mathematics and Applications*.

1. Introduction and Two-Index Flow Model

In this chapter we are concerned with solving the basic symmetric CVRP to optimality by a method known as Branch-and-Cut. This method has been extremely successful in finding optimal solutions of large instances of a closely related problem, the Symmetric Traveling Salesman Problem (STSP). However, the amount of research effort spent to solve CVRP by this method is not comparable with what has been dedicated to the STSP; the reader should not expect that we will report such spectacular results.

The amount of research carried out on Branch-and-Cut applied to the CVRP is still quite limited and most of it is not published yet. The main aim of this chapter is to try to bring the results obtained in the last decade to the public attention. At the time of writing this chapter several research groups were still working on the subject, but not all their results were yet available to us. We will mention them at the end, so the interested reader can stay alert for forthcoming results.

We mainly concentrate on what in Chapter 1 has been called the *two index flow model*. Whether or not this model is the most suitable for the Branch-and-Cut approach it is not obvious at all, but the success of the corresponding model for the STSP certainly pleads for it. However we will also mention another formulation which seems to give promising preliminary computational results, the two-commodity network flow model studied by Baldacci, Mingozzi and Hadjiconstantinou [7].

We start by recalling and integrating some of the definitions and the notation given in Chapter 1 and by stating a formal definition of the problem.

We are given an undirected complete graph $G(V, E)$ with node set V containing $n + 1$ nodes numbered $0, 1, \ldots, n$. The distinguished node 0 corresponds to the depot and the other nodes correspond to the n clients. We denote by V_0 the set of clients, i.e., $V_0 = V \setminus \{0\}$. For each client i we are given a positive demand d_i. With each edge $e \in E$, we associate a positive cost value c_e which corresponds to the cost of traveling along it. Let K be the fixed number of vehicles available at the depot. All vehicles are assumed to have the same capacity C.

For a subset F of E, $G(F)$ denotes the subgraph $(V(F), F)$ induced by F, where $V(F)$ is the set of nodes incident to at least one edge of F.

A *route* is defined as a nonempty subset $R \subset E$ of edges for which the induced subgraph $G(R)$ is a simple cycle containing the depot 0 (i.e., $0 \in V(R)$, $G(R)$ is connected, the degree of each node of $V(R)$ in $G(R)$ is 2) and such that the total demand of the nodes in $V(R) \setminus \{0\}$ does not exceed the vehicle capacity C. Such a route represents the trip of one vehicle leaving the depot, delivering the demand of the nodes in $V(R)$ (traveling along the edges in R), and going back to the depot. The cost of a route is the sum of the edge costs c_e over all edges $e \in R$. Note that we

allow the degenerate case $|V(R)| = 2$, in which a route R is the set consisting of two identical copies of an edge of E incident with node 0. In all the other cases, an edge appears only once in a route.

A *K-route* is the union of K routes $R_1, R_2, \ldots, R_K$ such that each node $i \in V_0$ belongs to exactly one set $V(R_j)$, $1 \leq j \leq K$. The cost of a K-route is the sum of the costs of the K different routes defining it. Each K-route defines a feasible solution to CVRP and the optimization problem consists of finding a minimum length K-route. A K-route induces a partition $P = \{S_1, S_2, \ldots, S_K\}$ of V_0 into K subsets such that $d(S_i) \leq C$ for $i = 1, 2, \ldots, K$. Such a partition is called a *feasible K-partition* and represents a feasible assignment of the clients to the vehicles. Each feasible K-partition may correspond to several K-routes.

In the two index flow model, we associate to each K-route R a vector $x^R \in \mathbb{R}^E$, i.e., a real vector indexed by the elements of E, such that the value of a component x_e^R associated with edge e is number of times e appears in the K-route R. Such a vector is called the *representative vector* of R (the terminology, *characteristic* or *incidence* vectors is usually used for vectors whose components are 0 or 1, which is not the case here since $x_e^R \in \{0, 1, 2\}$).

For a subset F of the edge set E and for a vector $y \in \mathbb{R}^E$, we denote by $y(F)$ the sum $\sum_{e \in F} y_e$. For $S \subset V, T \subset V, S \cap T = \emptyset$, we denote by $(S : T)$, $\delta(S)$, and $E(S)$ the sets of edges with one endpoint in S and the other in T, with one endpoint in S and the other in $V \setminus S$, and with both the endpoints in S, respectively.

We can now formulate the problem of finding a minimum c-cost feasible K-route as an integer linear program. It is easy to check that the set of representative vectors of all the K-routes of G coincides with the feasible set of the following integer linear program:

$$
\begin{aligned}
\text{(IP(CVRP))} \quad & \min \sum_{e \in E} c_e x_e && && (1)\\
& \text{subject to} \\
& \quad x(\delta(i)) = 2 && \text{for} \quad i \in V_0 && (2)\\
& \quad x(\delta(0)) = 2K && && (3)\\
& \quad x(\delta(S)) \geq 2 \left\lceil \tfrac{d(S)}{C} \right\rceil && \text{for} \quad \emptyset \neq S \subseteq V_0 && (4)\\
& \quad 0 \leq x_e \leq 1 && \text{for} \quad e \in E \setminus \delta(0) && (5)\\
& \quad 0 \leq x_e \leq 2 && \text{for} \quad e \in \delta(0) && (6)\\
& \quad x_e \text{ integer} && \text{for} \quad e \in E && (7)
\end{aligned}
$$

The constraints (2)–(3) and (4) are commonly called the *degree equations* and the *capacity inequalities*, respectively. In the following we will see that there are several types of capacity inequalities and (4) will be referred to as the *rounded capacity inequalities*. Due to (4), the integer linear program IP(CVRP) has an exponential number of constraints. However, as we will see in the next sections, the difficulty in solving IP(CVRP) to optimality does not arise solely from this fact.

In Section 2 we outline the Branch-and-Cut approach and in the following two sections 3 and 4 we survey some of the theoretical and computational work that has been done on the CVRP Polytope in order to provide a Branch-and-Cut algorithm with suitable tools for solving some nontrivial CVRP instances to optimality. Finally in Section 6 we give some numerical results based on a Branch-and-Cut implementation of Augerat et al. [6] and we mention some very recent results of Ralphs, Kopman, Pulleyblank, and Trotter [40] and Blasum and Hochstättler [8]. We end with conclusions and perspectives for further research.

2. Branch-and-Cut

In this section we give a short overview of the Branch-and-Cut method. For an extensive and comprehensive description of this method and areas of successful application, the reader is referred to Padberg and Rinaldi [37], Jünger, Reinelt, and Rinaldi [20], Jünger, Reinelt, and Thienel [21], Thienel [42], Jünger and Thienel [22], and Caprara and Fischetti [9].

The *linear relaxation* of an integer linear program IP is the linear program obtained from IP by dropping the condition that all variables have to be integer. For example, the linear relaxation of IP(CVRP) is obtained by dropping the constraints (7). Therefore, the optimal value z_{LP} of the relaxation (in the minimization case) is a lower bound to the optimal value z_{IP} of the integer linear program, i.e., $z_{LP} \leq z_{IP}$.

If the number of constraints of an integer linear program is small enough so that its linear relaxation can be fed into an LP solver, a classical method to solve it is Branch-and-Bound with linear programming bounds. That is, we solve first the linear relaxation. If the optimal solution $\bar{x}$ is integral we are done, else we choose a variable $\bar{x}_e$ with a fractional value and build two new linear programs. In the first we add an upper bound to x_e equal to $\lfloor x_e \rfloor$, while in the second we set a lower bound on x_e equal to $\lceil x_e \rceil$ (by $\lfloor x_e \rfloor$ and $\lceil x_e \rceil$ we denote the largest integer smaller than x_e and the smallest integer greater than x_e, respectively). From there on we proceed by classical Branch-and-Bound, in which the bounds are given by the optimal solution values of the linear programs associated with the nodes of the search tree.

When the number of linear constraints of IP is large or when the linear relaxation is strengthen by adding some families of valid inequalities, which typically have exponential size, then the constraint system cannot be fed into an LP solver and a *cutting plane* technique has to be used to solve the linear program.

Let IP be an integer program and LP(∞) be its linear relaxation, possibly enriched by additional valid inequalities, having a very large number of constraints and let us assume that we want to minimize the objective function. The cutting plane algorithm works as follows:

For $h \geq 0$, let LP(h) be a linear program consisting of a subset of reasonable size of the constraints in LP(∞). Solve LP(h), which yields an optimal solution $\bar{x}^h$. If this solution is feasible for IP, it is an optimal solution, else assume we have a "black box algorithm" that gives us at least one constraint of LP(∞) violated by $\bar{x}^h$, if one exists, or else tells us that all such constraints are satisfied. If some violated constraints are returned, LP($h+1$) is obtained from LP(h) by adding them to LP(h). Note that for every $h \geq 0$, if $z_{LP(h)}$ is the optimal value of LP(h), we have $z_{LP(h)} \leq z_{LP(h+1)} \leq z_{LP(\infty)} \leq z_{IP}$.

The black box algorithm is called the *separation algorithm*. Therefore we normally end with an optimal solution either to the integer program IP or to its linear relaxation LP(∞). In practice we may have a separation algorithm which is not exact, that is, it may return no violated inequality even though there are some. What follows remains true also in this case, since the value of the last linear program is still, as we just noted, a lower bound on z_{IP}.

If we have not terminated with an optimal solution to IP, we decompose the problem into two new problems, for example by adding upper and lower bounds to a variable whose current value is fractional, like it is done in Branch-and-Bound. This process is called *branching*. Then we solve each new problem recursively, that is by this very same method, and the optimal solution to the original problem will be the best of these two solutions (an infeasible LP($\cdot$) returns a value of $+\infty$). Such an integration of enumeration with cutting plane is the kernel of the method called Branch-and-Cut.

Note that in Branch-and-Cut enumeration and cutting plane benefit from each other: on the

one hand the bound produced at each node of the enumeration tree is in general better than in Branch-and-Bound, because new inequalities are added to the formulation of the corresponding subproblem; on the other hand the separation algorithm takes advantage from the branching process as it produces a perturbation on the fractional solution that could not be cut away by an inequality of LP(∞). Such a cross fertilization of different techniques is typical also for other components of Branch-and-Cut (see the cited references) and is the basic philosophy underlying the whole method.

In the case of IP(CVRP) we can let LP(0) be:

$$
\begin{aligned}
\text{(LP(0))} \qquad & \min \sum_{e \in E} c_e \; x_e \\
& \text{subject to} \\
& \qquad x(\delta(\{0\})) = 2K \\
& \qquad x(\delta(\{i\})) = 2 \qquad \text{for} \quad i \in V \setminus \{0\} \\
& \qquad 0 \le x_e \le 1 \qquad \text{for} \quad e \in E \setminus \delta(\{0\}) \\
& \qquad 0 \le x_e \le 2 \qquad \text{for} \quad e \in \delta(\{0\})
\end{aligned}
$$

and LP(∞) be the linear program consisting of the objective function and constraints (1) through (6). Therefore the cutting plane procedure consists of finding rounded capacity constraints violated by the optimal solution of LP(h), $h \ge 0$. As we will see in Section 4 this is not a major difficulty. We exit the cutting plane procedure either with an optimal solution to IP(CVRP), or with an optimal solution $\bar{x}^*$ to the linear relaxation which is not an optimal solution to IP(CVRP). There are at least two ways to execute the branching process. The first, and most often used one, is to choose a variable which is fractional, say $\bar{x}^*_{e^*}$ (for simplicity let us assume that x_e is restricted to only two values, 0 and 1). Then consider, on one subproblem, the solutions in which $x_{e^*} = 0$ and, on the other, those in which $x_{e^*} = 1$. An alternative, and, as we will see in Section 5, a preferable way to perform branching is to use a set S^* such that $\left\lceil \frac{d(S^*)}{C} \right\rceil = t$ and the value $x^*(\delta(S^*))$ is close to $2t + 1$. We can branch by considering on one subproblem those solutions for which $x(\delta(S^*)) = 2t$ and on the other those for which $x(\delta(S^*)) \ge 2t + 2$. As we will see, and explain why later in Section 5, we will favor the case where $t = 1$.

Branch-and-Cut has been very successful in solving many combinatorial optimization problems (see Caprara and Fischetti [9]); however it may give poor performances in case some of its components is too weak. This unpleasant situation happens, for example, when

(i) we do not have a good algorithm to perform the cutting plane phase;

(ii) the number of iterations of the cutting plane phase is too high;

(iii) the linear program becomes unsolvable because of its size;

(iv) the tree generated by the branching procedure becomes too large and termination seems unlikely within a reasonable amount of time.

There is no remedy for (i) and (ii), except that, for (i), Branch-and-Cut, as already mentioned, does not necessarily require that the relaxed LP be solved to optimality, we can go into the branching phase even if that LP has not been solved exactly. Therefore it is not strictly necessary to have an exact separation procedure; a good heuristic usually suffices.

Problem (iii) can be avoided most of the time with some extra effort such as a regular clean up of the linear program by deleting inactive constraints. For more details see the above-mentioned

references on Branch-and-Cut. If we do so, for CVRP and STSP instances this is never a problem. However, this is not the case, for example, in dealing with linear relaxations of problems related with large uncapacitated plant location problems, where the variable upper-bounding constraints of the type $x_{i,j} \leq y_j$ although polynomial in number pose a serious problem.

We are left with Problem (iv), which is the central problem of Branch-and-Cut. Most failures are due to this situation. For IP(CVRP), as we will see, we have a good separation algorithm for the rounded capacity constraints, and none of the three first problems leads us to failure. Nevertheless, Branch-and-Cut applied as described here is doomed to failure even on quite small instances. As we will see, one solution is to change LP(∞), either by modifying some inequalities or by adding others. Why this helps is now explained.

There is a direct relationship between Problem (iv) and the gap between the solution value of the integer program and that of its linear relaxation. Note that this is also true for Branch-and-Bound with linear bounding. So the only possible fix to Problem (iv) is to *strengthen* the linear relaxation, i.e., to add some linear inequalities that are satisfied by all solutions¿ These inequalities, although unnecessary in the integer formulation, force the optimal solution value of the linear relaxation to get "closer" to an optimal solution of the integer program. Finding such inequalities is in general not an easy task and is part of the so called *polyhedral study* of the problem, which is the subject of the next section.

Therefore, in the cutting plane phase we will not only be looking for violated rounded capacity constraints but also for many other type of violated inequalities.

3. Polyhedral Studies

We assume the reader to be familiar with the basic elements of polyhedral analysis. For a reference see, e.g., Nemhauser and Wolsey [36].

The *CVRP Polytope* $CVRP(n, d, C, K)$ is the convex hull of the representative vectors of all the K-routes.

Like any polytope, it has a linear description given by a finite number of linear inequalities. This linear description is the most we can strengthen the LP relaxation of IP(CVRP), since solving that strengthened relaxation amounts to solving the CVRP. Unfortunately, we are far from knowing this complete linear description. However, many successful applications of polyhedral analysis to the design of algorithms for solving hard combinatorial problems to optimality rely on a very small subset of the inequalities that provide a complete linear descriptions of a given polyhedron.

Unlike the Symmetric Traveling Salesman Polytope (STSP Polytope), the CVRP Polytope has not received much attention. Moreover, most of the known results have not been published yet. Therefore, in this section we try to give a survey of the current results, to the best of our knowledge.

Like the STSP Polytope, the CVRP Polytope is not of full dimension; however, unlike the STSP Polytope, the dimension of the CVRP Polytope is not a simple function of the problem size (number of nodes) but depends, in a complex way, on all the terms of the quadruple (n, d, C, K). In the case of the STSP Polytope, we know that the only equations satisfied by all the feasible solutions are the degree constraints and all their linear combinations. In the case of the CVRP Polytope these equalities also hold, but in general many others too. For example, it may be the case that in all the K-routes, the clients of a given set S are served by exactly t vehicles, and therefore the equality $x(\delta(S)) = 2t$ holds for the representative vectors of all the K-routes. In the extreme case, assuming that K is large enough to guarantee the existence of at least one

solution, this solution may be unique and thus the dimension of $CVRP(n, d, C, K)$ is 0 (this is the case when only one K-partition exists and all its sets have cardinality 1 or 2).

The difficulty in determining the dimension of the polytope makes the task of proving that an inequality induces a facet particularly hard. Therefore, we concentrate on describing inequalities that are shown to be valid for the CVRP Polytope; determining the conditions under which they also define facets it is often still an open issue or it requires taking into account a great deal of tedious technical conditions. Whether these inequalities are "powerful" or not, from a computational point of view, can be decided in various ways. For example, via computational experiments, that is we can check if using them we get better performances for an exact solution algorithm. Unfortunately, this method may produce erroneous conclusions: it is not infrequent that some inequalities are totally useless for certain instances but very useful for others.

An alternative approach is to prove the facet inducing property with respect to a relaxation of the CVRP Polytope, that is a polytope that contains the CVRP as a face and is possibly full dimensional, thus making the task of proving the facet-inducing property more manageable. We could then consider "powerful" an inequality that induces a facet of the relaxation, although it might happen that the same does not hold true for the CVRP Polytope.

Two relaxations have been used in the literature. The first one comes directly from the most successful relaxation of the STSP Polytope, the *graphical relaxation* (see Cornuéjols, Fonlupt, and Naddef [13], Naddef and Rinaldi [32], [33]). In this relaxation, introduced for the CVRP by Cornuéjols and Harche [14], a vehicle may visit any client without making a delivery, and may do so an arbitrary number of times for that same client. It may also make the delivery for a client and then visit it again later in the trip. To be more formal, let us extend the definition of a *route* by introducing the notion of a *partial closed walk*, defined as a multiset R of edges (the same edge may appear several times in R) such that the multigraph built on R, replacing an edge by as many its copies as there are in R, is Eulerian and contains node 0. A *K-walk* is the union of K partial closed walks with the additional condition that it induces a feasible K-partition. With every K-walk we associate a representative vector, the components of which give the number of times each edge is present in it. The *Graphical CVRP Polyhedron* (GCVRP) is the convex hull of the representative vectors of all the K-walks of the graph. This polyhedron is unbounded and has the nice feature of being of full dimension. Note that if we know the multiset of edges of a K-walk we do not have enough information to produce a solution to the problem. Actually, we must find a feasible assignment of the clients to the partial closed walks and this task turns out to be NP-hard. If the triangular inequality is satisfied by the cost vector c, then there always exists an optimal K-walk that is a K-route. Therefore, in this case, rather than CVRP, one can solve *tout court* its graphical relaxation.

The second relaxation is due to Araque, Hall, and Magnanti [3] and aims at removing the degree equations from the formulation. Each feasible solution of the relaxation are sets of disjoint paths of the subgraph of G induced by V_0, each with associated total demand not exceeding C. It is easy to see that each feasible solution of this problem corresponds to a feasible solution of the CVRP (and the cost of the two solutions always differs by the constant amount $2\sum_i c_{0i}$), after we replace the cost of each edge c_{ij} with the Clarke and Wright savings (see [11]) $-c_{0i} - c_{0j} + c_{ij}$. In order to avoid situations where a set of clients is served by a fixed number of vehicles in all feasible solutions, which would imply that the representative vectors of all the feasible solution satisfy the same equation, as mentioned earlier, the number of vehicles is also relaxed to be arbitrarily large. The convex hull of the set of all representative vectors of such sets of disjoint paths forms a full dimensional polyhedron, assuming that there are no pairs of clients whose total demand exceeds the capacity C. Indeed, if i and j are such that

$d_i + d_j > C$ then $x_{ij} = 0$ is an equation satisfied by all the solutions.

3.1. Capacity Constraints

The capacity inequalities for CVRP Polytope play, in some sense, the same role as the subtour elimination inequalities for the STSP Polytope: they are also exponentially many (there is one for each subset S of V_0) and all are necessary to define IP(CVRP). However, while all the subtour inequalities define facets of the STSP Polytope, the same does not always hold for the capacity inequalities.

There is actually a hierarchy of capacity inequalities, all sharing the same left hand side, but having right hand sides of nondecreasing value. The higher the right hand side value the stronger is the inequality but the harder is the corresponding separation problem. Observe that the left hand side of the inequality, evaluated at the representative vector of a K-route, gives the number of edges of the K-route that belong also to the cut $\delta(S)$. Roughly speaking, this is the number of times that the vehicles in the K-route cross the boundary of the set S.

We survey all the inequalities of the hierarchy going from the weakest to strongest.

A *fractional capacity inequality* has as a right hand side

$$2\frac{d(S)}{C}. \tag{8}$$

The separation problem for these inequalities is quite easy and actually solvable in polynomial time (see Section 4), however they are almost never supporting for the CVRP Polytope since their left hand sides, evaluated at one of its vertices, have integral value. Nevertheless, it is not difficult to see that IP(CVRP), where the right hand side of (4) is replaced by (8), still is a valid integer linear programming formulation of the problem. Thus its linear programming relaxation, obtained by dropping the integrality requirements, yields a bound that can be computed in polynomial time.

A *rounded capacity inequality* is obtained by rounding the right hand side (8) to the nearest larger integer. The resulting inequality is (4) and the associated separation problem is quite more difficult than the one of the fractional inequality, although still computationally affordable (see Section 4). This inequality may not be supporting yet in some cases. A better lower bound on the left hand side of the inequality is given by taking twice the minimum number of bins of capacity C needed to pack the items of the set S, whose sizes are given by the vector d. We call this number of bins $r(S)$. When $\left\lceil \frac{d(S)}{C} \right\rceil \neq r(S)$, then the rounded capacity inequality relative to S is not supporting.

A *weak capacity inequality* has $2r(S)$ as a right hand side. Since it is NP-hard to compute $r(S)$, the separation problem for these inequalities is difficult (see [16]). Nonetheless, although it might sound surprising, a weak capacity inequality is, in general, not supporting because it does not take into account the demands outside the set S under consideration.

Let $\mathcal{P}$ denote the set of all feasible K-partitions of V_0. For any nonempty set $S \subseteq V_0$ and for any K-partition $P = \{S_1, \ldots, S_K\}$, define

$$\beta(P, S) = |\{i : S_i \cap S \neq \emptyset\}|.$$

This quantity is obviously equal to the number of vehicles needed to satisfy the demands of all clients in S in the K-partition P.

The function $R : 2^{V_0} \to Z_+$ defined by

$$R(S) = \min_{P \in \mathcal{P}} \beta(P, S) \tag{9}$$

clearly gives the minimum number of vehicles needed to satisfy all client demands in S in a feasible K-partition. Any $P \in \mathcal{P}$ for which the minimum in (9) is attained is called a *tight* feasible K-partition relative to S.

The reason why the weak capacity inequality may not be supporting is that there may be no tight feasible K partition P for which $\beta(P, S) = r(S)$, although it is possible to pack the items of set S into $r(S)$ bins. This situation calls for the ultimate version of the inequality.

A *capacity inequality* is the one having $2R(S)$ as a right hand side. This inequality is supporting by definition. Whether the inequality defines a facet of the CVRP Polytope still depends (for complete graphs) on the structure of the vector d and on the values K and C. This issue has been investigated in Cornuéjols and Harche [14] where conditions are given under which the capacity inequality induces a facet of the CVRP Polytope. Since the computation of $r(S)$ is a special case of the computation of $R(S)$, the separation for the capacity inequalities is hard.

An example, due to Cornuéjols and Harche [14], where the last three right hand sides are different, has 8 clients, $K = 4$, $C = 7$, $d = \{5, 3, 3, 3, 4, 4, 4, 2\}$, and S is given by the first four clients. The resulting right hand sides are

$$R(S) = 4 > r(S) = 3 > \left\lceil \frac{d(S)}{C} \right\rceil = 2.$$

An inequality is *tight* for a given vector $x^* \in \mathbb{R}^E$ if the left hand side of the inequality evaluated at x^* equals the right hand side. We say that a K-route is tight for a set S if it contains exactly $2R(S)$ edges of the coboundary $\delta(S)$ of S. That is the capacity inequality is tight for the representative vector of that K-route. We will also use the term tight when $R(S)$ is replaced by $r(S)$.

3.2. Generalized Capacity Constraints

Let us consider now a set $\mathcal{S} = \{S_1, \ldots, S_t\}$ of $t > 1$ disjoint subsets of V_0. If we add up the t capacity inequalities defined by each subset, clearly we obtain a valid inequality that is dominated by each of the capacity inequalities used in this operation. On the other hand, it may happen that no feasible K-partition tight relative to, say, S_1 is also tight for all the remaining $t - 1$ sets. In this case we can increase the right hand side of the inequality by at least 2 units, while preserving its validity, obtaining a new inequality that is not dominated by any capacity constraint. How much can the right hand side be increased without losing the validity? The largest value of the right hand side is given by

$$2R(\mathcal{S}) = 2 \min\{\sum_{i=1}^{t} \beta(P, S_i) : P \in \mathcal{P}\}.$$

The resulting inequality

$$\sum_{i=1}^{t} x(\delta(S_i)) \geq 2R(\mathcal{S}) \tag{10}$$

is called the *generalized capacity inequality* and was defined in De Vitis, Harche and Rinaldi [15], where sufficient conditions for it to be facet defining for the GCVRP Polyhedron as well as for the CVRP Polytope are given.

Since the function $R(\cdot)$ is difficult to compute, as a more tractable valid inequality for a cutting plane algorithm we consider a weak version of the generalized capacity inequality. Let H be a

subset of V_0 containing all the subsets in $\mathcal{S}$ and assume that $d(S_i) \leq C$ holds for $i = 1, \ldots, t$. Then we define $r(H \mid S_1, S_2, \ldots, S_t)$ to be the bin packing solution of the problem having bin capacity of C, one item with size $d(u)$ for each node u in $H \setminus \bigcup_{i=1}^{t} S_i$, and one item having size $d(S)$ for each $S \in \mathcal{S}$. Alternatively, $r(H \mid S_1, S_2, \ldots, S_t)$ can be seen as the optimal solution of a bin packing problem with one item of size $d(u)$ for each node u of G, where all items in each subset $S \in \mathcal{S}$ are constrained to stay together in the same bin.

In the case $H = V_0$, we define the *weak generalized capacity inequality* to be the following

$$x(\delta(V_0)) + \sum_{i=1}^{t} x(\delta(S_i)) \geq 2t + 2r(V_0 \mid S_1, S_2, \ldots, S_t), \tag{11}$$

or, equivalently, as $x(\delta(V_0)) = 2K$ holds,

$$\sum_{i=1}^{t} x(\delta(S_i)) \geq 2t + 2(r(V_0 \mid S_1, S_2, \ldots, S_t) - K).$$

The validity of this inequality is implied by the definition of its right hand side.

While it is very difficult to check if a generalized capacity inequality is violated, because of the function $R(\cdot)$, checking if its weak version is violated amounts to solving a bin packing problem to compute $r(V_0 \mid S_1, S_2, \ldots, S_t)$. Although bin-packing is NP-hard, it is recognized to be computationally affordable for a wide number of instances (see also the remark of Section 3.5).

3.3. Framed Capacity Constraints

Let $\mathcal{S}$ be as in the previous section. If in the inequality (11) we replace V_0 by any of its subsets H containing all the sets S_i for $i = 1, 2, \ldots, t$, then we get the new inequality

$$x(\delta(H)) + \sum_{i=1}^{t} x(\delta(S_i)) \geq 2t + 2r(H \mid S_1, S_2, \ldots, S_t) \tag{12}$$

that is a generalization of (11) which we call the *framed capacity constraint*. This inequality was proposed by Pochet [38] and Augerat [4]. The proof of its validity is a special case of the proof of validity for the path-bin inequalities of Section 3.5.

Observe that it might be the case that this inequality be violated while the weak generalized capacity might not be.

As an example of a framed capacity constraint, consider an instance were $t = 4$, the sets S_1, S_2, S_3, and S_4 have 8, 7, 7, 7 clients, respectively, all with unit demands, and $C = 10$. Moreover the set $H \setminus \bigcup_{i=1}^{4} S_i$ has only two clients with demands 5 and 6. For this case $r(H \mid S_1, S_2, S_3, S_4) = 6$, hence $x(\delta(H)) + \sum_{i=1}^{t} x(\delta(S_i)) \geq 20$ is valid. Note that the sum of the five weak capacity constraints would yield a right hand side of 16 instead of 20. Indeed, there are many other framed capacity constraints in this case, for example $\mathcal{S} = \{S_1, S_2\}$, $H = S_1 \cup S_2 \cup \{\text{the client of demand 5}\}$, or $\mathcal{S} = \{S_3, S_4\}$, $H = S_3 \cup S_4 \cup \{\text{the client of demand 6}\}$, both have 10 as a right hand side, while 8 is the sum of right hand sides of the corresponding three weak capacity constraints.

3.4. Valid Inequalities from STSP

A K-route and a Hamiltonian cycle have very close structures: they are both connected subgraphs of G where all nodes in V_0 have degree 2. The only difference occurs at node 0, which

has different degree in the two subgraphs. Indeed, a K-route is a special case of a *tour* that is a feasible solution of the already cited graphical relaxation of STSP. A *tour* (see Cornuéjols, Fonlupt, and Naddef [13]) is defined as a multiset Θ of edges such that the multigraph obtained from Θ is Eulerian, connected, and spans G. Of course any inequality valid for all tours of G is also valid for all its K-routes (and for all its Hamiltonian cycles). What about the reverse? E.g., is a valid inequality for STSP also valid for its graphical relaxation? Remember that the incidence vectors of all the Hamiltonian cycles satisfy all the degree equations. Therefore, if we add any linear combination of the degree equations to a valid inequality we obtain again a valid inequality that defines the same face of the STSP Polytope. The two inequalities are said to be *equivalent*. Thus, we usually say that a valid inequality for the polytope can be written in different (equivalent) forms. Naddef and Rinaldi [33] have studied the connections between STSP and its graphical relaxation; in particular they showed that any valid inequality for the STSP, written in a certain form, is also valid for all tours, and therefore for all K-routes. We make this more precise now.

An inequality $ax \geq a_0$ (where a and x are vectors of $\mathbb{R}^E$) is in *tight triangular form* (*TT form*), if for all triples of distinct nodes $i, j, k \in V$, we have $a_{jk} \leq a_{ij} + a_{ik}$ and for every $i \in V$ there exists $j(i) \in V$ and $k(i) \in V$ such that $a_{j(i)k(i)} = a_{ij(i)} + a_{ik(i)}$.

The tight triangular form of the STSP valid inequalities was introduced by Naddef and Rinaldi [33] and became, for many reasons, the standard form of STSP inequalities. One of these reasons, is that the representation of a facet of the STSP Polytope by an inequality in TT form is unique (up to scaling by a non-negative constant). It is also shown by Naddef and Rinaldi [33] how to transform any STSP valid inequality into its equivalent TT form.

We will give a direct proof of the following theorem.

Theorem 3.1. *All the valid inequalities for the STSP Polytope, written in tight triangular form, are valid for the CVRP Polytope.*

Proof. Assume a valid inequality $ax \geq a_0$ for the STSP Polytope in TT form is not valid for the CVRP Polytope. Let R be a K-route, the representative vector x^R of which satisfies $ax^R < a_0$. Choose two edges $(0, i)$ and $(0, j)$ incident with the depot and belonging to different routes in R. Since the triangular inequality holds, removing these two edges and adding edge (i, j) yields a $(K-1)$-route R' (perhaps violating some capacity constraints) with $ax^{R'} < a_0$. Repeating this process we end up with a Hamiltonian cycle Γ such that $ax^\Gamma < a_0$, which contradicts the fact that $ax \geq a_0$ is a valid inequality for the STSP Polytope. □

A survey of the most well-known STSP valid inequalities with an intuitive idea of their validity can be found in Naddef and Pochet [31].

The feasible solutions of CVRP inherit in a complex way the structure of the Hamiltonian cycles and those of the bin packing solutions. The inequalities of the previous section deal only with the "bin packing part" of the problem, while inequalities coming from STSP only deal with the "routing part". Therefore, it is not surprising if often STSP inequalities define faces of the CVRP Polytope that are not of high dimension. However, these inequalities can be strengthened if we take into account the "bin packing part" of the problem.

To understand better how this can be done, it is useful to understand first the role that some classes of STSP inequality play in the definition of the STSP Polytope. As an example we will consider the well-known class of comb inequalities. For further classes see Naddef and Pochet [31].

A *comb* is defined by a handle H and an odd number of teeth $T_1, T_2, \ldots, T_t$ satisfying the

following conditions

$$\begin{array}{l} H, T_1, T_2, \ldots, T_t \subseteq V \\ T_j \setminus H \neq \emptyset \quad \text{for} \quad 1 \leq j \leq t \\ T_j \cap H \neq \emptyset \quad \text{for} \quad 1 \leq j \leq t \\ T_i \cap T_j = \emptyset \quad \text{for} \quad 1 \leq i < j \leq t \\ t \geq 3 \text{ and odd.} \end{array}$$

The *comb inequality* in TT form is:

$$x(\delta(H)) + \sum_{i=1}^{t} x(\delta(T_i)) \geq 3t + 1$$

or equivalently:

$$x(\delta(H)) \geq (t+1) - \sum_{i=1}^{t} (x(\delta(T_i)) - 2).$$

We give now an informal proof of the validity for the comb inequalities for the STSP Polytope. In the following, Hamiltonian cycle can be replaced by closed walk or by K-route. Take any Hamiltonian cycle Γ such that $|\Gamma \cap \delta(T_i)| = 2$ for all $i = 1, \ldots, t$. Then $|\Gamma \cap \delta(H)| \geq t$ since there must be at least one edge of Γ in $(T_i \setminus H : T_i \cap H)$ for each tooth T_i. Now since a cycle must intersect the coboundary of any node set in a even number of edges and since t is odd, we must have $|\Gamma \cap \delta(H)| \geq t + 1$. So all such Hamiltonian cycles satisfy the inequality. As explained in Naddef and Pochet [31], in most inequalities described in term of handles and teeth, the role of the teeth is in some sense to force a prescribed number of edges out of the handles. Does the inequality remain valid for a Hamiltonian cycle Γ such that $|\Gamma \cap \delta(T_i)| = 2 + 2s_i$ for $i = 1, \ldots, t$? It can be shown that the minimum number of edges of $\Gamma \cap \delta(H)$ is not reduced by more than $2\sum_{i=1}^{t} s_i$ over the former minimum of $3t+1$. Thus the inequality is valid for all the Hamiltonian cycles.

To see how comb inequalities can be rather weak for the CVRP Polytope, consider the following three simple observations.

1. If the handle does not contain the depot and $R(H) \geq (t+1)/2$, then obviously the comb inequality will be implied by the capacity constraint on the handle H. If the handle contains the depot and $R(V \setminus H) \geq (t+1)/2$, then the same holds.

2. If all teeth have a bin packing value of 1, but no two can be entirely picked up by a same vehicle, then there are no K-routes for which the comb inequality is tight. Section 3.5 will show how to transform such an inequality to one that at least has some K-routes on the induced face.

3. If a tooth T_i has a bin packing value higher than 1, or if it contains the depot and the bin packing value of its complement is more than 1, then all the K-routes will have at least 4 edges in $\delta(T_i)$ and therefore the comb inequality will be dominated by another inequality.

We now try to strengthen a comb inequality by taking the capacity constraint into account. While the intersection of the coboundary of a node set S (a handle or a tooth) with a Hamiltonian cycle has cardinality at least 2, the intersection with a K-route is at least $2R(S)$. In general, to describe a strong inequality, $2R(S)$ is the value to be considered; however, as done for the capacity constraints, for the sake of computational tractability we will replace it by its relaxation $r(S)$, or even by $\left\lceil \frac{d(S)}{C} \right\rceil$.

In view of the explanation given before on how comb inequalities can be derived, we try now to see how we can "force" edges out of the handle also using the capacity constraints. In order to do so, we need that, when the nodes of a tooth T_i are visited by the minimum number of

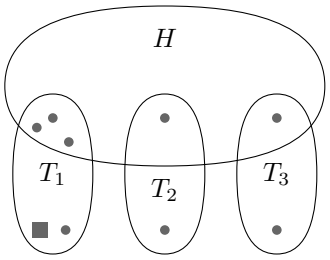

Figure 1: An example with depot in tooth

necessary vehicles, at least one edge of $(T_i \setminus H : T_i \cap H)$ be used. The following condition, due to Laporte and Nobert [26], is given precisely in this spirit:

Let $H, T_1, T_2, \ldots, T_t$ define a comb such that no tooth contains the depot and

$$r(T_i \setminus H) + r(T_i \cap H) > r(T_i) \tag{13}$$

then the following inequality is valid (Laporte and Nobert [26]):

$$x(\delta(H)) + \sum_{i=1}^{t} x(\delta(T_i)) \geq t + 1 + 2\sum_{i=1}^{t} r(T_i) \tag{14}$$

Note that if $r(T_i) = 1$ for all i, then the condition (13) obviously always holds, and we just get the comb inequality. Therefore 14 is a proper generalization of a comb inequality.

What about the case in which a tooth, say T_1, contains the depot? Since a comb inequality is equivalent to the inequality defined by taking the complement of its handle, we may assume, without loss of generality, that $0 \in T_1 \setminus H$. Of course, the demands of the nodes outside T_1 now play a role, since they force a minimum number of edges in the coboundary of T_1. Therefore, in this case, assuming that for all the other teeth $r(T_i) = 1$ holds, the right hand side of the comb inequality would be $3t - 1 + 2r(V \setminus T_1)$.

So far we have considered some simple situations when the plain comb inequality is not supporting for the CVRP Polytope and the right hand side can be modified to strengthen it. Deciding in general whether a comb inequality is supporting, finding the correction in the right hand side to make it supporting, deciding if the strengthened version is faced defining, and, finally, in case it is not, determining the correction in the coefficients to make it facet defining seem all to be difficult tasks and of increasing complexity. For example, for the case considered earlier of a comb with the depot in a tooth, it is likely that not only the bin packing value of $V \setminus T_1$ should be taken into account, but also that of $H \cup (\bigcup_{i=2}^{t} T_i)$, a node set that contains the clients of $H \cap T_1$. Figure 1 shows an example with each node of demand 1 (the depot is represented by a squared node), $C = 3$, $R(V \setminus T_1) = r(V \setminus T_1) = 2$. The right hand side of the strengthened comb inequality would be 12, but no K-path is tight for that inequality. The lowest value we can reach on the left hand side is 14. Note that replacing $r(V \setminus T_1)$ by $R(V \setminus T_1)$ does not help here, since we assumed that these two values coincide. One could then increase the right hand side to 14, but still what we get does not define a facet: all K-routes that satisfy the new inequality to equality have exactly 4 edges in $\delta(T_1)$ and therefore the face that it induces is contained in the one induced by the capacity inequality associated with $V \setminus T_1$. All this is to say that things are not easy.

A vehicle routing problem can be reduced to a pure traveling salesman problem by the following operation: Replace the depot by a set D of K nodes. Since we deal with complete graphs, we

set $x_e = 0$ for each edge e having both endpoints in D, or alternatively, as done in Ralphs, Kopman, Pulleyblank, and Trotter [40], we just assume that there are no edges between any two nodes in D. Every node of D is linked to every client node by an edge of same cost as in the original problem. A K-route yields a Hamiltonian cycle in the new graph. Conversely a Hamiltonian cycle such that the sum of the demands of the clients on every path between two consecutive visits to the nodes in D is at most C, yields a K-route. Take a comb inequality in the new graph. If we contract all the nodes of D we are back to the original graph, but the combs may now have teeth that intersect in the depot. Therefore it is quite natural to address combs with intersecting teeth in the depot as possible valid inequalities for CVRP.

A *comb* can be defined more generally by a handle H and an odd number of teeth $T_1, T_2, \ldots, T_r, T_{r+1}, \ldots T_t$ satisfying the following conditions:

$$\begin{array}{ll} H, T_1, T_2, \ldots, T_t \subseteq V & \\ T_j \setminus H \neq \emptyset & \text{for} \quad 1 \leq j \leq t \\ T_j \cap H \neq \emptyset & \text{for} \quad 1 \leq j \leq t \\ T_i \cap T_j = \emptyset & \text{for} \quad 1 \leq i < j \leq r \\ T_i \cap T_j = \{0\} & \text{for} \quad r+1 \leq i < j \leq t \\ t \geq 3 \text{ and odd}, & \end{array}$$

where r may be any value between 0 (all teeth intersect) and t (no teeth intersect nor contain the depot). The teeth T_{r+1} to T_t intersect in the depot; if $r = t-1$, only one tooth does. Moreover, assume that the full set of K vehicles is needed to satisfy the demand of all the nodes outside a tooth containing the depot, that is $R(V \setminus T_j) = K$, for all $j = r+1, \ldots, t$. Then the following comb inequality is valid:

$$x(\delta(H)) + \sum_{i=1}^{t} x(\delta(T_i)) \geq t + 1 + 2r + 2K(t-r)$$

We do not give a formal proof of validity, but only an intuitive explanation. If a K-route is going to intersect the coboundary of a tooth T_j containing the depot only $2K$ times, then at least one edge of $(H \cap T_j : T_j \setminus H)$ must be in that K-route. The argument then proceeds as in the case of regular comb inequalities. Note that if some teeth not containing the depot cannot be served by a single vehicle and satisfies the Laporte-Nobert condition, then we can strengthen the inequality like we have already seen.

3.5. Valid Inequalities combining Bin-Packing and STSP

As we saw, the "plain" STSP inequalities ignore the demands, while the capacity inequalities ignore the routing conditions. Here we try to combine the two aspects of the problem, as we started to do with the framed capacity inequalities of Section 3.3 and with the comb inequalities.

The *path-bin inequalities* defined below are a generalization of the framed capacity inequalities defined in Section 3.3 and of the comb inequalities defined in Section 3.4. They were defined by Pochet [38] and Augerat [4]. Since this material is not readily available at the time of writing this chapter, we will include the necessary proofs.

A *path-bin support* is defined by a *handle* H, by *teeth* $T_1, T_2, \ldots, T_t$, and by *spots* $S_1, S_2, \ldots, S_s$. For notational simplicity, we let T_{t+i} stand for S_i.

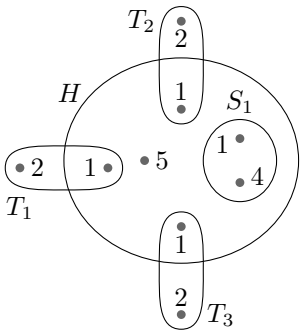

Figure 2: A path-bin support with three teeth and one spot

These sets satisfy the following conditions:

$$
\begin{array}{lll}
H, T_1, T_2, \ldots, T_{t+s} \subset V_0 & & \\
d(T_j) \leq C & \text{for} & 1 \leq j \leq t+s \\
S_j \subset H & \text{for} & 1 \leq j \leq s \\
T_j \cap H \neq \emptyset & \text{for} & 1 \leq j \leq t \\
T_j \setminus H \neq \emptyset & \text{for} & 1 \leq j \leq t \\
T_i \cap T_j = \emptyset & \text{for} & 1 \leq i < j \leq t+s \\
t+s \geq 1. & &
\end{array}
$$

The difference with respect to a comb is that, for a path-bin support, we impose that the total demand of a tooth or of a spot be less or equal to the vehicle capacity C and there is no parity requirement on the numbers s or t. Figure 2 represents a path-bin support with three teeth and one spot.

Associated with the path-bin support, we define a path-bin subproblem which is the following constrained bin packing problem. Let I be the set of items. Each tooth $T_j,\ 1 \leq j \leq t$, and spot $S_j,\ 1 \leq j \leq s$, defines one item of size $d(T_j)$ and $d(S_j)$, respectively. Each node $v \in H \setminus (\bigcup_{j=1}^{t+s} T_j)$ defines one item of size d_v. The bin size is the vehicle capacity C. Let $r'(H \mid T_1, \ldots, T_{t+s})$ be the minimum number of bins needed to contain all items of I with the additional constraint that a bin can contain the items corresponding to at most two teeth.

To link this constrained bin-packing subproblem to the solutions of CVRP, we define an *H-path* of a K-route of CVRP as a connected component of the intersection of one of the routes with $E(H)$. In this definition, a single route can define several H-paths but in all cases, the number of edges of the K-route in $\delta(H)$ equals two times the number of its H-paths. This holds because the depot does not belong to H and thus each H-path contains two edges of $\delta(H)$ in the K-route.

The purpose of the constrained bin-packing subproblem is to compute the minimum number of H-paths, over all K-routes of CVRP, under the condition that the demand of a tooth or spot must be satisfied by a single vehicle, i.e., those K-routes use exactly two edges of each tooth or spot coboundary. Such an H-path can contain at most two teeth and each such H-path corresponds to a feasible bin. More precisely, as the bin packing subproblem only considers the demand in $H \cup (\bigcup_{j=1}^{t} T_j)$, it only uses local information and will only provide a lower bound on this minimum in the same sense as $r(S)$ is a lower bound on $R(S)$. The first example of the Figure 3 illustrates this correspondence between the H-paths and the bin-packing subproblem. This result is formalized in the following proposition.

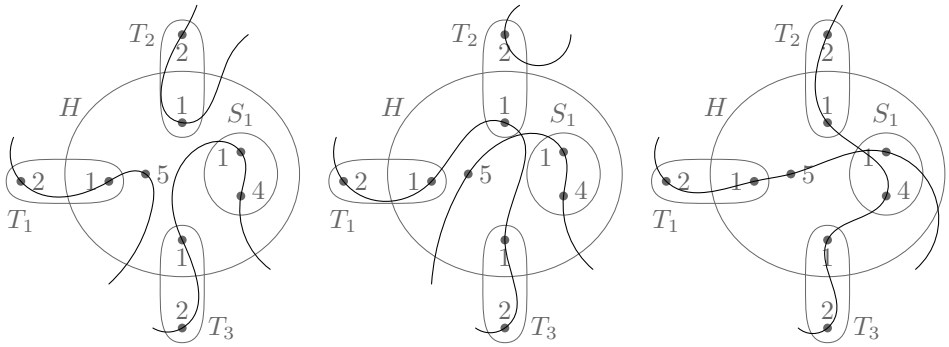

Figure 3: Tight solutions for a path-bin inequality (C=10)

Proposition 3.2. *If $\bar{x}$ is a solution of CVRP satisfying $\bar{x}(\delta(T_j)) = 2$ for $1 \leq j \leq t+s$, then the following inequality holds*

$$\bar{x}(\delta(H)) \geq 2r'(H \mid T_1, \ldots, T_{t+s}), \tag{15}$$

where $r'(H \mid T_1, \ldots, T_{t+s})$ is the value of the constrained bin packing problem.

Proof. Let Θ be a K-route which contains the minimum number of edges in $\delta(H)$, among all K-routes satisfying the conditions of the proposition. Consider the set F of edges of Θ with both the endpoints in H. The number of connected components of $G' = (H, F)$ is exactly half the number of edges of Θ in $\delta(H)$. Assume that the side condition of the bin packing problem is violated, i.e., that a connected component intersects three teeth, say $T_i, T_j, and T_p$ in that order (going from one extremity to the other). Since T_j has nodes outside the handle H, Θ must intersect $\delta(T_j)$ in at least 4 edges, a contradiction with our choice of Θ. □

The path-bin inequality transforms the inequality (15) into a valid inequality for CVRP. It is based on a well-known property of the bin packing problem, which says that splitting an item into two parts cannot decrease the optimal number of bins by more than one. This property is translated in terms of solutions of CVRP to build the valid inequality. Putting the item T_j (tooth or spot) in one bin means satisfying the demand $d(T_j)$ by only one vehicle therefore $x(\delta(T_j)) = 2$. Splitting the item T_j into two parts and allowing these two parts to be in different bins means allowing the demand $d(T_j)$ to be satisfied by two vehicles and $x(\delta(T_j)) = 4$. So, as $\sum_{j=1}^{t+s} x(\delta(T_j))$ is an even integer, each time $\sum_{j=1}^{t+s} \Big[x(\delta(T_j)) - 2\Big]$ is increased by 2, the number of H-paths can not be reduced by more than 1 and the coboundary of H can not be reduced by more than 2. Consequently, we have the following

Theorem 3.3. *For any path-bin support $(H, T_1, \ldots, T_t, S_1, \ldots, S_s)$, the associated path-bin inequality*

$$x(\delta(H)) \geq 2r'(H \mid T_1, \ldots, T_{t+s}) - \sum_{j=1}^{t+s} (x(\delta(T_j)) - 2) \tag{16}$$

is valid for the CVRP Polytope.

Figure 3 represents three tight solutions of the path-bin inequality using the same support as in Figure 2, the first one corresponding to a constrained bin-packing solution, the second one with a tooth split into two parts and the last one with a spot split into two parts.

Computing $r'(H \mid T_1, \ldots, T_t, S_1, \ldots, S_s)$ is harder than computing $r(H)$ which is NP-hard. For small sets H, the exact bin-packing algorithm of Martello and Toth [27] can be adapted to deal with the additional constraint on the teeth. For larger sets H, the lower bounding procedure of Martello and Toth [27] can be adapted. Two trivial lower bounds on $r'(H \mid T_1, \ldots, T_t, S_1, \ldots, S_s)$ are $\lceil t/2 \rceil$ and $\left\lceil d(H \cup (\bigcup_{j=1}^{t+s} T_j))/C \right\rceil$. As $r'(H \mid T_1, \ldots, T_t, S_1, \ldots, S_s)$ only appears at the right hand side of the path-bin inequality (16), replacing it by any lower bound still yields a valid inequality.

As $x(\delta(T_j)) \geq 2$ in any feasible solution of CVRP, a set T_j or S_j will only be introduced in the path-bin inequality if its removal would reduce the value of the bin-packing subproblem, for otherwise the inequality without this set T_j dominates the inequality with the set T_j.

Note that if some tooth T_i is such that $r(T_i) > 1$ but satisfies Condition (13) of Laporte and Nobert, then it is easy to modify the path-bin inequality to take this case into account.

3.6. Valid Inequalities from the Stable Set Problem

The following inequality generalizes Araque's [2] *star inequalities* (not to be confused with the star inequalities for the STSP Polytope defined in Fleischmann [18] nor with the multistar inequalities mentioned in Section 4) for the special case of CVRP occurring when all demands are equal.

A *clique cluster* (see Pochet [38] and Augerat [4]) is defined by $W_1, \ldots, W_w$ subsets of V_0 such that:

$$\begin{array}{lll} W_i \cap W_j = \{v\} & \text{for} & 1 \leq i < j \leq w \\ r(W_i) = 1 & \text{for} & 1 \leq i \leq w \\ r(W_i \cup W_j) > 1 & \text{for} & 1 \leq i < j \leq w. \end{array}$$

Thus any two sets intersect in the same node v, each set can be served by one vehicle, but not the union of two of them. The node v is called the *nucleus* of the cluster.

The capacity conditions imply that at most one set W_i can have a coboundary that intersects a K-route in exactly two edges. Therefore the following inequality is valid:

$$\sum_{i=1}^{w} x(\delta(W_i)) \geq 4w - 2. \tag{17}$$

Note that the nucleus cannot consist of more than one node. Assume $W_i \cap W_j = N$ for all $1 \leq i < j \leq w$, with $|N| > 1$, then any tight K-tour for the clique cluster inequality is tight for the capacity constraint on N, i.e., for $x(\delta(N)) \geq 2$, therefore the face induced by (17) would be contained in the face defined by the capacity constraint on the set N. In the case of a nucleus N with more than one node, one can modify an inequality proposed by Araque, Hall, and Magnanti [3] for the unit capacity CVRP. The valid inequality in this case is (see [38]):

$$\sum_{i=1}^{w} x(\delta(W_i)) - (w-2)x(\delta(N)) \geq 2w + 2.$$

The star inequality is a particular case of a much larger class of inequalities. Consider a rank inequality facet defining for the Stable Set Polytope. A rank inequalities is defined for some special graph G_s; each variable is associated with a node of G_s and has a coefficient of 1, while the right hand side is the stability number of G_s. Special graphs are, for example, cliques or chordless cycles of odd length. A rank inequality for the Stable Set Polytope can be translated

into a valid inequality for the CVRP Polytope in the following way. The nodes of G_s correspond to subsets of clients of CVRP with total demand not exceeding the vehicle capacity but such that the union of any two of them requires two vehicles to be served. If two nodes of G_s are adjacent, then the corresponding sets intersect, let us say, for simplicity, in a single node. Assume the stable set inequality has p as right hand side, this means that at most p sets of clients can have a coboundary that intersects a K-route in exactly two edges, yielding the inequality:

$$\sum_{i=1}^{w} x(\delta(W_i)) \geq 4w - 2p.$$

Well-known rank inequalities for the Stable Set Polytope are the clique constraints which say that at most one node of a clique can belong to a stable set. These yield the clique cluster inequalities. As long as the sets intersect pairwise we get a clique, therefore one inequality for the stable set polytope may yield several valid inequalities for the CVRP. Many, of course will not even be supporting for the CVRP polytope.

Another well known inequality is the *odd hole* inequality, which says that at most t nodes of a chordless cycle of length $2t+1$ can belong to a stable set. This would correspond to taking $2t+1$ subsets $W_0, \ldots, W_{2t}$ of V_0 such that:

$$\begin{array}{lll} W_i \cap W_{i+1} = \{v_i\} & \text{for} & 0 \leq i \leq 2t \pmod{2t+1} \\ W_i \cap W_j = \emptyset & \text{for} & 0 \leq i, j \leq 2t,\ |i-j| > 1 \\ r(W_i) = 1 & \text{for} & 0 \leq i \leq 2t \\ r(W_i \cup W_{i+1}) > 1 & \text{for} & 0 \leq i \leq 2t \pmod{2t+1}. \end{array}$$

That is the sets form an "odd hole", the intersection of two "consecutive" sets is just a node and two "consecutive" sets (modulo $2t+1$) cannot be served by the same vehicle. Then at most t sets W_i can have $x(\delta(W_i)) = 2$ for any representative vector x of a K-route. Therefore the following inequality is valid:

$$\sum_{i=1}^{w} x(\delta(W_i)) \geq 6t + 4. \tag{18}$$

It should now be obvious to the reader that the facial structure of the CVRP polytope is extremely complex and that there is still a lot to investigate in this field. For more on this topic we refer the reader to the papers cited above.

4. Separation Procedures

Throughout this section we assume that $\bar{x} \in \mathbb{R}^E$ is a fractional point satisfying all the constraints (2), (3), (5), and (6). We aim at describing efficient (possibly of polynomial time complexity) procedures to find a valid inequality for the CVRP Polytope, belonging to one of the classes described in the previous section, that is violated by $\bar{x}$.

4.1. Exact Separation of Capacity Constraints

We only address the problem of separating the fractional and the rounded capacity inequalities.

The separation problem for the fractional capacity inequality is solvable in polynomial time as shown by McCormick, Rao, and Rinaldi [29] by reducing it to a network flow problem. Using the same idea, Blasum and Hochstättler [8] provide a polynomial separation for a generalization of the fractional capacity inequalities that they call *multistar inequalities*.

Let $G_{\bar{x}}$ be the weighted graph induced by the edges whose associated components of $\bar{x}$ are strictly positive and by all edges incident to node $\{0\}$. Each edge of $G_{\bar{x}}$ has the corresponding component of $\bar{x}$ as a weight. Produce a new graph $G'_{\bar{x}}$ by replacing the weight $\bar{x}_e$ by $\bar{x}_e - d_i/C$ for all edges $e = (0, i)$ with $i \in V_0$. It easy to see that if in $G'_{\bar{x}}$ there is a minimum cut with negative weight, then the shore of this cut that does not contain node 0 defines a fractional capacity inequality violated by $\bar{x}$. Unfortunately, a standard (polynomial time) algorithm for computing the minimum cut of $G'_{\bar{x}}$ cannot be used, because such a graph may have edges with a negative weight. To overcome this problem, consider an auxiliary graph $G''_{\bar{x}}$ obtained from $G_{\bar{x}}$ by adding a node $n+1$ connected to all nodes $1, 2, \ldots, n$. All edges of $G''_{\bar{x}}$ with both endnodes in $\{1, 2, \ldots, n\}$ have the same weight as the corresponding edges of $G_{\bar{x}}$. For $i = 1, 2, \ldots, n$ the edges $(0, i)$ and $(i, n+1)$ get weights $\max\{0, \bar{x}_{(0,i)} - d_i/C\}$ and $\max\{0, d_i/C - \bar{x}_{(0,i)}\}$, respectively. As the network $G'_{\bar{x}}$ has nonnegative weights, the minimum weight cut separating 0 from $n+1$ can be computed in polynomial time. In McCormick, Rao, and Rinaldi [29] it is shown that the minimum cut F'' separating 0 from $n+1$ in $G''_{\bar{x}}$ yields a minimum cut F' for $G'_{\bar{x}}$ by removing the edges incident with $n+1$. Moreover, the weight of F'' exceeds the weight of F' by the amount $Z = \sum_{i=1}^{n} \max\{0, d_i/C - \bar{x}_{(0,i)}\}$. Thus, if the weight of F'' is strictly less than Z, then its shore not containing $n+1$ defines a violated fractional capacity inequality. It this weight is nonnegative but strictly less than $Z+2$, then all fractional capacity inequalities are satisfied, but it may be possible that the defined rounded inequality be violated. If $\bar{w}$ is more than $Z+2$ no rounded capacity inequalities are violated.

To explore the second situation, De Vitis, Queyranne, and Rinaldi [16] use a procedure that produces all the cuts of $G''_{\bar{x}}$ separating 0 from $n+1$ in increasing order of their weights. The procedure stops as soon as a cut is produced that has weight greater that or equal to $Z+2$. For each cut generated by the procedure a check for violation of the corresponding rounded inequality is performed. It is then proved that if we require an amount of violation of at least ϵ, then the number of cuts produced is polynomial in n for any fixed ϵ. Since the complexity of producing every new cut is polynomial in n, the whole separation procedure is polynomial in n for a fixed ϵ.

4.2. Heuristics for capacity and related constraints

The previous algorithm for the rounded capacity inequalities may be too slow. We describe here a heuristic separation procedure for them as well for the generalized capacity constraints.

Given $\bar{x}$ and $S \subset V$ and $v \notin S$, we call $\bar{x}(\{v\} : S)$ the amount by which "*v sees S*". Given $S \subset V$, if one wants to add a node v to S and have $\bar{x}(\delta(S \cup \{v\}))$ as small as possible, then, because of the degree equations, one has to choose the node v that sees S by the largest amount. Following the terminology of some authors, we say that a set S is built by *max-back order* starting at S_0, if initially $S = S_0$, and at each iteration a node that sees the current set by the largest amount is added.

Greedy rounded capacity heuristic: Let S_0 be a set such that $\bar{x}(\delta(S_0)) = 2$. For example, S_0 could be just a node or the set of nodes of a path of edges with weight equal to 1 having maximal length. Grow S starting from S_0 until it reaches V_0 by max-back order. At each step check if the corresponding rounded capacity constraint is violated. Note that, due to the degree equations, it is easy to keep track of $\bar{x}(\delta(S))$ just knowing by how much the entering node v sees $S \setminus \{v\}$. Repeat such a procedure for all possible choices of S_0.

In [5], a tabu search procedure based on the previous idea is given and is shown to give good results.

Ralphs, Kopman, Pulleyblank, and Trotter in [40] suggest another approach when the previous greedy heuristic fails. First of all they work on the modified problem in which the depot is replaced by a set D of K nodes as described earlier. They try to decompose the solution $\bar{x}$ into a convex combination of Hamiltonian cycles. We will not go into the details of how this can be done. If the decomposition succeeds, then they look at all the Hamiltonian cycles of that convex combination and check whether one of the paths between two consecutive visits to D yields a set S such that the corresponding rounded capacity constraint is violated. Note that if we use such a sophisticated separation algorithm, it may also be worth the effort to check the bin packing value of these sets. The authors report success in this approach. Our experience is that when we encountered such a convex combination, we rarely found a violated capacity inequality. The following example of Figure 4, to which we will come back later, is taken from a fractional solution to `E-n31-k3` (also known as `eil31`) (see Section 6), the edges e in solid line have $\bar{x}_e = 1$, those in dotted line have $\bar{x}_e = 0.5$. The capacity of the vehicles is $C = 4500$ and $K = 3$. The solution is a convex combination of the two 3-routes displayed in Figure 5 each

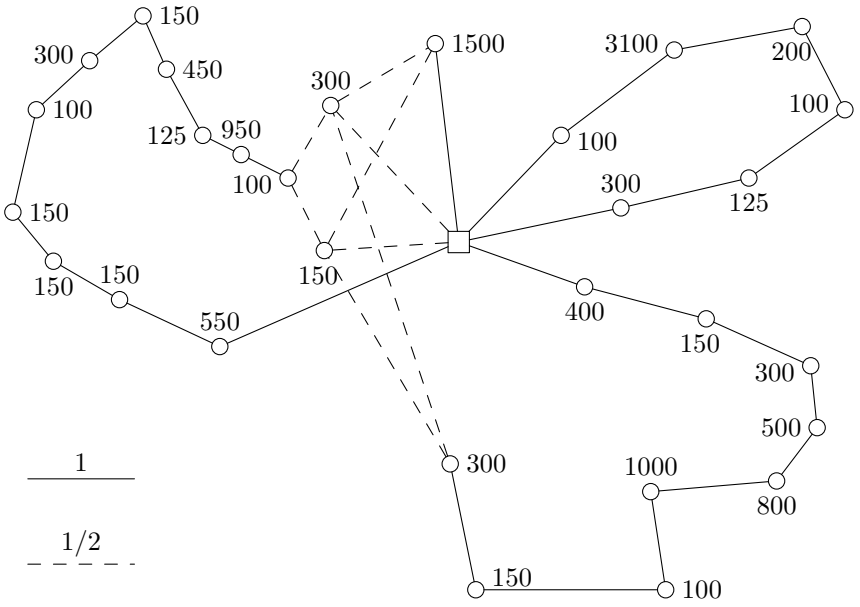

Figure 4: An example of fractional solution

with coefficient 0.5. It is easy to check that none of the sets of nodes of each route violates a capacity constraint in the fractional solution $\bar{x}$ since only one route, in each solution, goes over capacity, but it has a $\bar{x}$-coboundary of 4.

Heuristic for the weak generalized capacity constraints: This heuristic starts with a partition of V_0 into (hopefully maximal) sets of $\bar{x}$-coboundary equal to 2. If one has shrunk (recursively) each path of edges with weight equal to 1 to a single node, we can start with a partition where each set is a node of the shrunk graph (although there may still be sets of more than one node having coboundary of weight 2). Note that finding a partition into maximal sets of coboundary of weight 2 can be done in polynomial time, since finding all sets of minimum coboundary in an undirected graph is polynomial (see [23]), but the previously

described partition is in general good enough to start our heuristic. At each step of the following procedure we check whether the current partition violates the generalized capacity constraint. This amounts to solving a bin packing problem, which, for the size of instances we are dealing with, can be done in a moderate amount of time.

If the check fails, we merge the two sets S_i and S_j of the partition which "see" each other by the largest amount, i.e., such that $\bar{x}(S_i : S_j) = \max_{r,s} \bar{x}(S_r : S_s)$.

Going back to the example of Figure 4, as it is shown in Figure 6, the initial partition yields already a violated weak generalized capacity constraint. The partition is given by the circled sets. The capacity of the vehicles is $C = 4500$ and $K = 3$. The demands of the sets of the partition are 3175, 3700, 1500, 300, 150, and 3925 and all have a $\bar{x}$-coboundary value of 2. The bin packing value is 4 since the item of size 1500 cannot fit with any of the first two nor with the last one.

Note that the nodes in the set S_6 do not play a significant role in this inequality. In fact there is also a framed capacity constraint violated by the same amount. It is defined by the handle $H = V \setminus (S_6 \cup \{0\})$ and the two sets S_1 and S_2. We have $\bar{x}(\delta(H)) = 4$ but if only two vehicles serve H, then S_1 and S_2 cannot be served each by a single vehicle, as it is currently the case since $\bar{x}(\delta(S_1)) = \bar{x}(\delta(S_2)) = 2$, because then none of these vehicles can serve the node with demand 1500.

4.3. STSP Constraints

Naddef and Thienel [34], [35] give separation routines for the STSP that can be very easily adapted to the CVRP. Moreover these routines can easily be adapted to take into account the Laporte-Nobert conditions (13). The computational results given in this paper are done with an earlier version of these routines developed by Clochard and Naddef [12].

5. Branching Strategies

We devote a special section to branching since it is a critical component of any Branch-and-Cut implementation.

As described in Section 2, there are several ways to perform a branching.

The first one is to choose an edge e^* for which the corresponding variable is fractional in the current LP optimal solution, and split the set of solutions into those that use the edge ($x_{e^*} = 1$) and those that do not ($x_{e^*} = 0$) (assuming, of course, that x_{e^*} can take only these two values). We will call this method *edge* or *variable branching*.

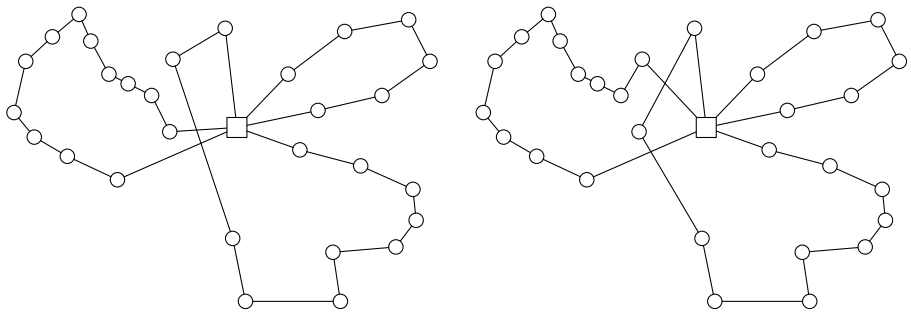

Figure 5: The two solutions of the convex combination

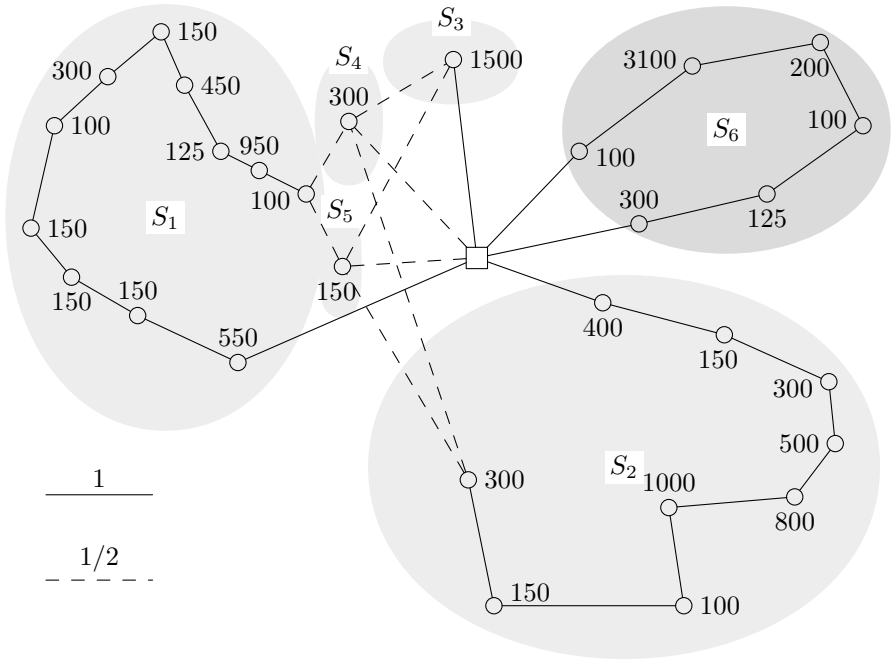

Figure 6: An example of violated generalized capacity constraint

A crucial problem is to choose the variable on which such a branching should be performed. For the case of STSP, a general rule, proposed for the first time by Padberg and Rinaldi [37], is to choose a variable $\bar{x}_{e^*}$ whose value falls into a small interval centered at 0.5 and, in case this happens for more variables, to choose the one among them with largest cost. Note that the branching on variables is very asymmetric in the sense that setting a variable to 1 amounts to choosing one of the $n + K$ edges of a K-route, while setting a variable to 0 corresponds to deciding that one out of $O(n^2)$ edges is not in a K-route. This made some researchers think that it could be a better choice to select a variable with a higher fractional value, say 0.75. Although some authors report no success with such a choice for the STSP, we have tried it in the case of the CVRP.

Some other researchers propose to choose a variable such that, if set to 1, would extend an existing path of edges already set to 1. Among all possible choices, the one which leads to a path of largest total demand should be preferred. The idea behind this strategy is that it implies many other variables to be set to 0: those associated to the edges incident with an internal node of the path and those incident with the extremities of the path and with a node whose demand is not compatible with the total demand of the path. The asymmetry problem is now even more evident and therefore this does not seem to be a good choice to us.

Applegate et al. [1] suggested to choose a candidate set of variable for branching and to select the best among them by *LP-testing*, i.e., by solving both linear programs induced by the two possible values of the variable. The best variable is then chosen as the one for which the the minimum of the two objective function values is the largest.

To give an example on how the different strategies behave for CVRP, we report on some computational tests performed on 15 instances chosen from the literature, using the Branch-and-Cut code developed by Augerat et al. [6] with a depth first search procedure and limiting the depth of the search tree to 30. Because the behavior of the Branch-and-Cut algorithm

depends on so many implementation details, as it has been already observed, and due to the small number of experiments, the results have to be taken, of course, with some caution.

As a first experiment three possible strategies were tested for branching variable selection:

A1) choice of the variable with largest cost among those with value close to 0.5;

A2) choice of the variable with largest cost among those with value close to 0.75;

A3) the best in an LP-test performed on 10 variables of values between 0.45 and 0.65.

We give the number of problems solved, total times (in hours:minutes) and the number of times the strategy led to a smaller search tree. Unfortunately, the unsolved problems bias the times since, contrary to what one may expect, because of the depth first search strategy and our limit of the tree depth, these take less time than the others.

strategy	solved instances	time	# best
A1	13	24:30	0
A2	12	18:40	0
A3	15	14:00	15

Clochard and Naddef [12] proposed, for the first time in Branch-and-Cut, to use an alternative branching strategy, which we will call *branching on an inequality*. It was implemented for the STSP using inequalities dealing with the coboundaries of sets. Any closed walk must use a positive even number of edges of any coboundary. Let S be a node set such that $\bar{x}(\delta(S)) \approx 2t+1$. Then we can decompose the problem into two subproblems: one where $x(\delta(S)) \leq 2t$ holds and on the other where $x(\delta(S)) \geq 2t+2$ holds. They report significant improvements in difficult STSP instances over the classical variable branching. For example, Naddef and Thienel [35] get an improvement by a factor of 3 in solving the difficult `ts225` instance of the library collected by Reinelt [41].

Branching on inequalities was used for the first time in Augerat [4] and Augerat et al. [6] for the CVRP. In this case the most interesting situation happens when there is a node set S for which $\bar{x}(\delta(S)) \approx 3$. Like in the case of branching on variables there is an unbalance between the side for which we impose $x(\delta(S)) = 2$ and the one where we impose $x(\delta(S)) \geq 4$. Imposing that all the clients of set S are served by the same vehicle amounts to considering S as a unique client with demand equal to the sum of the demands in S. If its total demand is high enough, then one of the K-routes is almost fixed and therefore this side of the search tree will be solved faster than the other side.

The numerical results of this section are taken again from Augerat [4] where 6 strategies were tested:

B1) select S for which $x(\delta(S))$ is the closest to 3.0;

B2) select S for which $x(\delta(S))$ is the closest to 2.85;

B3) select S for which $x(\delta(S))$ is the closest to 3.15;

B4) select S for which $2.75 \leq x(\delta(S)) \leq 3$ and $d(S)$ maximum;

B5) select S for which $2.75 \leq x(\delta(S)) \leq 3$ and the distance from S to depot is maximum (ties are split by total demand); by distance we mean the sum of the distances to the depot of the two nodes of S that are the closest to it.

B6) select S for which $2.75 \leq x(\delta(S)) \leq 3$ and contains the largest number of "supernodes"; a "supernode" is a set that has been identified as having in the current solution a coboundary of weight 2, for example, the nodes of a maximal path of edges of weight 1, or the node sets the coboundary of which have been set to 2 in previous branchings.

The seventh parameter in the choice of set S, namely its cardinality, would have been worth testing. This parameter was not considered to keep the selection procedure as simple as possible. The reason why the cardinality of S might be important is that the smaller S the easier should be the subproblem where $x(\delta(S)) \geq 4$.

The sets S are built by the same procedure that is used to heuristically find violated capacity constraints as described in the previous section. The results are reported in the following table.

strategy	solved instances	time	# best
B1	14	14:55	7
B2	13	26:10	1
B3	13	26:25	1
B4	14	5:40	4
B5	13	5:30	3
B6	13	5:25	2

Note that several strategies may have given trees of the same sizes for the same instance; for this reason the entries of the last column of the table do not add up to 15.

It is clear from these results, that because of the asymmetry issue mentioned earlier it is better to choose a set of coboundary strictly less than 3. It is also evident that the last three strategies are the clear winners.

It was also experimented if it is worthwhile spending time on looking for a good branching set by LP-testing. The selection was done by LP-testing from sets with coboundary weight falling into three intervals. The results are summarized in the following table.

condition	solved instances	time
$2.50 \leq \delta(S) \leq 2.85$	14	17:00
$2.75 \leq \delta(S) \leq 3.00$	14	5:50
$2.85 \leq \delta(S) \leq 3.10$	14	18:00

A final experiment was carried out selecting by LP-testing and according to three different criteria: edge branching, branching on inequalities, and a mix of the first two. Edges were selected from a set made of the one with value closest to 0.75 and the ten ones with value closest to 0.5. Sets were selected considering one representative from those satisfying each of the following three conditions: $x(\delta(S))$ closest to 3, 2.85, and 3.15, respectively. Finally, three more sets S were selected among those with $x(\delta(S))$ falling into the interval $[2.75, 3]$: the one with largest total demand, the one with largest cardinality, and the one with maximum distance from the depot. The following table summarizes the results:

strategy	time	# best	average BC nodes
test on edges	14:00	4	76
test on sets	4:57	10	50
test on edges+sets	5:51	6	35

It seems clear that the two last strategies are the best. More extensive computational experiments have shown that the extra time spent by LP-testing in choosing a proper branchings almost always pays off: it may cut by a factor of more than 2 the total time to solve harder instances, although it may double the time for the very easy ones. However the latter situation does not affect the evaluation as we believe that a reasonable measure of performance for an exact algorithm should be "how many instances from a test set can we solve with a limited time for each?".

To conclude, we list a set of rules for the selection of S, derived also by analyzing the above experimental results:

- $2.75 \leq x(\delta(S)) \leq 3.0$ and $d(S) > C/2$;
- S "far" from the depot;
- $|S|$ small ($W \subset S$ with $x(\delta(W))$ set to 2 in previous branchings is counted as a unique node);
- S is contained or at least intersects a former branching set.

The idea behind these conditions is that if the set S contains a few nodes, then with the last rule we will soon have partitioned its nodes between different vehicles. Moreover, the assignment to the vehicles of clients which are far from the depot seems to be critical; the second rule aims at making such an assignment as soon as possible.

6. Numerical Results

The computational study reported here is the one of Augerat et al. [6]. It was made with a Branch-and-Cut algorithm that makes use of many of the separation procedures and of the strategies described in the previous sections. As the algorithm was developed by three groups of researchers, its implementation was not done with the purpose of being efficient. Rather than a state of the art software, the code is a kind of experimental environment that can easily accommodate various separation routines and algorithmic strategies with the purpose of making comparison testing readily available. Another source of extensive computational experimentation can be found in Kopman, Ralphs, Pulleyblank and Trotter [40].

The main drawback of such a code is the lack of several components that are common to most of Branch-and-Cut codes like, among the others, the pool management and the possibility of having only subsets of variables active in the solution of the linear programs. In addition, the visit of the enumeration tree is done using the *depth-first*, which is the easiest to implement, but also the least effective. Last but not least, the algorithm was implemented via independent pieces of code communicating through files written in the mass storage. Such an algorithmic design provided indeed some flexibility to the developers but has, of course, a price in terms of efficiency.

Due to these facts the numerical results and the performance indicators reported in [6] are not to be taken as a reliable evidence of the actual potential of the technique. Nevertheless, the algorithm was able to find for the first time an optimal solution and proved its optimality for two instances of 135 nodes proposed by Fisher [17]. To the best of our knowledge, these are still the largest instances for which a certified optimal solution has been computed.

The numerical results of Augerat et al. [6] are summarized in the tables 1 and 2. The instances that form the test bed for the study are all taken from the literature (most of them are available

Table 1

Name	Upper Bound	Lower Bound	Gap	# Cuts	# LP's	CPU (*)
E-n22-k4	375.	375.	0.	100	21	2
E-n23-k3	569.	569.	0.	43	9	1
E-n30-k3	534.	534.	0.	243	50	17
E-n33-k4	835.	835.	0.	389	27	8
F-n45-k4	724.	724.	0.	168	24	7
f-n45-k4	748.	723.541	0.	173	28	19
E-n51-k5	521.	517.581	0.66	737	62	31
F-n72-k4	238.	235.	0.84	443	69	82
f-n72-k4(a)	269.241	240.408	0.65	300	50	208
f-n72-k4(b)	241.974	240.408	0.65	300	50	29
E-n76-k10	832.	793.545	4.85	1949	111	761
E-n76-k7	683.	664.361	2.81	1283	77	236
E-n76-k8	735.	713.601	3.00	1686	101	351
E-n76-k14	1032.	953.794	8.20	2157	72	466
E-n101-k8	815.	799.398	1.95	1749	111	494
M-n101-k10	820.	820.	0.	1706	48	472
F-n135-k7	1165.	1159.27	0.24	4481	136	1428
f-n135-k7	1162.96	1159.28	0.32	3397	138	1098

(*) seconds in a Sun Sparc 20

from the electronic library TSPLIB of Reinelt [41]). The naming convention for the instances is as follow. The name is a string `X-nxx-kyy`, where `xx` is the number of nodes, i.e., the number of clients plus 1, `yy` is the number of vehicles. If `X` is equal to "E" the instance is from Christofides and Eilon [10] and can be found in the TSPLIB under the prefix `eil`, if it is equal to "F" or "f" the instance comes from Fisher [17], finally if it is equal to "M" the source is Mingozzi et al. [30].

All these instances are of Euclidean type. As it is now customary, for the computation of the distances the convention of TSPLIB is adopted, i.e., the real Euclidean distance between any pair of nodes is rounded to the nearest integer. In order to compare the results with those of Fisher [17] an exception is made for the instances whose name starts with "f" that are obtained from the corresponding ones with name starting with "F", by taking only the first three decimal digits of the real Euclidean distances.

In Table 1 we summarizes the results concerning the computation at the end of the root node of the enumeration tree. The values of the column labeled "Upper Bound" are taken from the literature, while those of the column "Lower Bound" are value obtained after adding the cutting planes. Each value of the column "Gap" is the ratio of the difference between the optimal value and the lower bound over the lower bound. For the cases were the optimal value is not computed, the upper bound is used instead. The columns labeled "# Cuts", "# LP's", and "CPU" give the total number of valid inequalities generated, the number of LP calls, and the total cpu time, respectively. Seven instances were solved to optimality at the root node.

Table 2 summarizes the results for the five instances that were solved to optimality by performing some enumeration steps. The columns of the table report the optimal value, the total

Table 2

Name	Optimum	# Cuts	# LP's	# B&C nodes	CPU(*)
E-n51-k5	521.	908	129	7	54
F-n72-k4	237.	603	390	51	180
f-n72-k4(a)	241.973	1670	1862	239	4622
f-n72-k4(b)	241.973	484	280	31	98
F-n135-k7	1162.	13482	3086	423	20570
f-n135-k7	1162.95	10450	4833	633	15774

(*) seconds in a Sun Sparc 20

number of cuts generated, the total number of LP iterations, and the total cpu time, respectively. The last four values do not include those concerning the root node that are reported in Table 1.

For the instances with 76 and 101 nodes the algorithm was not able to terminate the computation in a reasonable amount of time.

Table 3

Name	Optimal Value	# B&C nodes	# processors	CPU(*)
E-n76-k7	682	115 991	9	278 613
E-n76-k8	735	484 245	60	1 927 422
E-n101-k8	815	244 968	80	1 900 671

(*) seconds in a network of IBM RS/6000 (from 120 to 135 Mhz)

In a further computational study Ralphs, Kopman, Pulleyblank, and Trotter [40] (see also [39] and [24]) implemented a parallel Branch-and-Cut algorithm that exploits the ideas already mentioned in Section 3.4. Such an algorithm was able to find an optimal solution of value 815 (and prove its optimality) for the instances E-n101-k8, improving by two units the best known solution. Moreover, for the first time it proved the optimality of the best known solution for E-n76-k8 and improved the best known solution for E-n76-k7 by one unit, providing a proof of optimality. Table 3 summarizes these results.

Another more recent study is reported by Blasum and Hochstättler [8], who developed an algorithm using the same branching strategy and separation procedures as in [6] with some modifications. For example, they developed a heuristic procedure for separating the rounded capacity inequalities based on their algorithm for the separation of the multistar inequalities already mentioned in Section 4. However, they used the state-of-the-art Branch-and-Cut framework ABACUS, developed by Jünger and Thienel [22]. The results are comparable with those of the tables 1 and 2, taking into account, of course, the differences in computer speed and LP solver efficiency. It is remarkable, though, that the algorithm was able to solve two difficult 76-node problems to optimality with computing times considerably shorter than those reported in Table 3. The instances E-n76-k7 and E-n76-k8 where solved with 6 717 and 6 259 nodes, respectively, in 27 550 and 35 466 seconds, respectively, on a 400 Mhz Sun-Ultrasparc II. No proof of optimality is reported for the third difficult problem of Table 3.

7. Conclusions

Branch-and-Cut to solve the CVRP is at the beginning of its developments. We believe that a better understanding of the underlying polytope and further effort in designing efficient separation routines should yield much better computational results than those reported here. Various groups around the world are currently working on the subject and new results should appear very soon. In particular other formulations have been studied that yield polytopes which are different from the one studied in this chapter. See, for example, the *two-commodity network flow formulation* studied by Baldacci, Mingozzi and Hadjiconstantinou [7]. This formulation, described in Chapter 1, at the time of writing does not provide yet better results than those reported here.

Some variants of the vehicle routing can appear also in the literature as particular cases of more general problems. For example the unit demand ($d_i = 1$ for all i) CVRP can be seen as a particular case of the Black and White traveling salesman problem of Ghiani, Laporte and Semet [19], therefore so can be the CVRP in which we split the client demands, since in that case every client with demand d_i can be replaced by d_i clients with unit demand, the distance between the copies of the same client being 0. A polyhedral study of the *split demand CVRP* has been carried out by Martinez, Mota, and Rinaldi [28]. Finally, for further material on the linear relaxation of CVRP, we refer the reader to the survey of Laporte [25].

References

[1] D. Applegate, R. Bixby, V. Chvátal, and W. Cook, "Solving traveling salesman problems," 1994. 15th International Symposium on Mathematical Programming, University of Michigan, USA.

[2] J. Araque, "Contributions to the polyhedral approach to vehicle routing," Discussion Paper 90-74, CORE, University of Louvain La Neuve, 1990.

[3] J. Araque, L. Hall, and T. Magnanti, "Capacitated trees, capacitated routing and associated polyhedra," Discussion Paper 90-61, CORE, University of Louvain La Neuve, 1990.

[4] P. Augerat, *Approche Polyèdrale du Problème de Tournées de Véhicules*. PhD thesis, Institut National Polytechnique de Grenoble, 1995.

[5] P. Augerat, J. Belenguer, E. Benavent, A. Corberán, and D. Naddef, "Separating capacity inequalities in the CVRP using tabu search," *European Journal of Operational Research*, vol. 106, pp. 546–557, 1999.

[6] P. Augerat, J. Belenguer, E. Benavent, A. Corberán, D. Naddef, and G. Rinaldi, "Computational results with a branch and cut code for the capacitated vehicle routing problem," Tech. Rep. RR 949-M, Université Joseph Fourier, Grenoble, 1995.

[7] R. Baldacci, A. Mingozzi, and E. Hadjiconstantinou, "An exact algorithm for the capacitated vehicle routing problem based on a two-commodity network flow formulation," Tech. Rep. 16, Department of Mathematics, University of Bologna, 1999.

[8] U. Blasum and W. Hochstättler, "Application of the branch and cut method to the vehicle routing problem," Tech. Rep. ZPR2000-386, ZPR, Universität zu Köln, 2000. URL: `http://www.zaik.uni-koeln.de/ paper`.

[9] A. Caprara and M. Fischetti, "Branch-and-cut algorithms," in *Annotated bibliographies in combinatorial optimization* (M. Dell'Amico, F. Maffioli, and S. Martello, eds.), pp. 45–64, Wiley, 1997.

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

[11] G. Clarke and J. Wright, "Scheduling of vehicles from a central depot to a number of delivery points," *Operations Research*, vol. 12, no. 4, pp. 568–581, 1964.

[12] J.-M. Clochard and D. Naddef, "Use of path inequalities for TSP," in *Proceedings of the Third Workshop on Integer Programming and Combinatorial Optimization* (G. Rinaldi and L. Wolsey, eds.), pp. 291–312, 1993.

[13] G. Cornuéjols, J. Fonlupt, and D. Naddef, "The traveling salesman problem on a graph and some related integer polyhedra," *Mathematical Programming*, vol. 33, pp. 1–27, 1985.

[14] G. Cornuéjols and F. Harche, "Polyhedral study of the capacitated vehicle routing," *Mathematical Programming*, vol. 60, pp. 21–52, 1993.

[15] A. De Vitis, F. Harche, and G. Rinaldi, "Generalized capacity inequalities for vehicle routing problems." In preparation, 2000.

[16] A. De Vitis, M. Queyranne, and G. Rinaldi, "Separating the capacity inequalities of the vehicle routing problem in polynomial time." In preparation, 2000.

[17] M. Fisher, "Optimal Solution of Vehicle Routing Problems Using Minimum k-Trees," *Operations Research*, vol. 42, no. 4, pp. 626–642, 1994.

[18] B. Fleischmann, "A new class of cutting planes for the symmetric traveling salesman problem," *Mathematical Programming*, vol. 40, pp. 225–246, 1988.

[19] G. Ghiani, G. Laporte, and F. Semet, "Black and white traveling salesman problem," Tech. Rep. 99-47, CRT, Montreal, 1999.

[20] M. Jünger, G. Reinelt, and G. Rinaldi, "The traveling salesman problem," in *Network Models* (M. Ball, T. Magnanti, C. Monma, and G. Nemhauser, eds.), vol. 7 of *Handbooks in Operations Research and Management Science*, ch. 4, pp. 225–330, Amsterdam: North-Holland, 1995.

[21] M. Jünger, G. Reinelt, and S. Thienel, "Practical problem solving with cutting plane algorithms in combinatorial optimization," in *Combinatorial Optimization* (W. Cook, L. Lovász, and P. Seymour, eds.), DIMACS Series in Discrete Mathematics and Theoretical Computer Science, pp. 111–152, American Mathematical Society, 1995.

[22] M. Jünger and S. Thienel, "Introduction to ABACUS—A Branch-And-CUt System," *Operations Research Letters*, vol. 22, pp. 83–95, 1998.

[23] A. Karzanov and E. Timofeev, "Efficient algorithm for finding all minimal edge cuts of a nonoriented graph," *Cybernetics*, pp. 156–162, 1986.

[24] L. Kopman, *A New Generic Separation Routine and its Application in a Branch and Cut Algorithm for the Capacitated Vehicle Routing Problem*. PhD thesis, Field of Operations Research, Cornell University, Ithaca, N.Y., U.S.A., 1999.

[25] G. Laporte, "The vehicle routing problem: An overview of exact and approximate algorithms," *European Journal of Operational Research*, vol. 59, pp. 345–358, 1992.

[26] G. Laporte and Y. Nobert, "Comb inequalities for the vehicle routing problem," *Methods of Operations Research*, vol. 51, pp. 271–276, 1984.

[27] S. Martello and P. Toth, "Lower bounds and reduction procedures for the bin packing problem," *Discrete Applied Mathematics*, vol. 28, pp. 59–70, 1990.

[28] M. Martinez, E. Mota, and G. Rinaldi, "The split delivery vehicle routing polyhedron: new families of facet defining inequalities." Preprint, Departamento de Estadística e I.O., Universitat de València, Spain, 1999.

[29] T. McCormick, M. Rao, and G. Rinaldi, "When is min cut with negative edges easy to solve? easy and difficult objective functions for max cut." IASI-CNR Reseach Report. In preparation, 2000.

[30] A. Mingozzi, N. Christofides, and E. Hadjiconstantinou, "An exact algorithm for the vehicle routing problem based on the set partitioning formulation." Unpublished manuscript, June 1994.

[31] D. Naddef and Y. Pochet, "The traveling salesman polytope revisited," Discussion Paper 98-42, CORE, University of Louvain-la-Neuve, 1998.

[32] D. Naddef and G. Rinaldi, "The symmetric traveling salesman polytope and its graphical relaxation: Composition of valid inequalities," *Mathematical Programming*, vol. 51, pp. 359–400, 1991.

[33] D. Naddef and G. Rinaldi, "The graphical relaxation: A new framework for the symmetric traveling salesman polytope," *Mathematical Programming*, vol. 58, pp. 53–88, 1993.

[34] D. Naddef and S. Thienel, "Efficient separation routines for the symmetric traveling salesman problem I: general tools and comb separation," tech. rep., Universität zu Köln, 1998. to appear.

[35] D. Naddef and S. Thienel, "Efficient separation routines for the symmetric traveling salesman problem II: separating multi handle inequalities," tech. rep., Universität zu Köln, 1998. to appear.

[36] G. Nemhauser and L. Wolsey, *Integer and Combinatorial Optimization*. Chichester: John Wiley & Sons, 1988.

[37] M. Padberg and G. Rinaldi, "A branch and cut algorithm for the resolution of large-scale symmetric traveling salesman problems," *SIAM Review*, vol. 33, pp. 60–100, 1991.

[38] Y. Pochet, "New valid inequalities for the vehicle routing problem." in preparation, 1998.

[39] T. Ralphs, *Parallel Branch and Cut for Vehicle Routing*. PhD thesis, Field of Operations Research, Cornell University, Ithaca, N.Y., U.S.A., 1995.

[40] T. Ralphs, L. Kopman, W. Pulleyblank, and L. Trotter, "A branch and cut algorithm for the vehicle routing problem (preliminary draft)." URL: `ftp://branchandcut.org/pub/reference/vrp.ps`.

[41] G. Reinelt, "A traveling salesman problem library," *ORSA Journal on Computing*, vol. 3, pp. 376–384, 1991.

[42] S. Thienel, *ABACUS: A Branch and Cut System*. PhD thesis, Universität zu Köln, 1995.