Robust Branch-and-Cut-and-Price for the Capacitated Vehicle Routing Problem

Ricardo Fukasawa, Marcus Poggi de Aragão,
Marcelo Reis, Eduardo Uchoa

Relatórios de Pesquisa em Engenharia de Produção
RPEP Vol.3 no.8 (2003)

Niterói, 10 de setembro de 2003

Robust Branch-and-Cut-and-Price for the Capacitated Vehicle Routing Problem

Ricardo Fukasawa [1], Marcus Poggi de Aragão [2], Marcelo Reis [2], Eduardo Uchoa [3]
[1] Gapso Inc.
R. Jardim Botânico 674, sala 614, Rio de Janeiro, Brasil, 22461-000
rfukasawa@gapso.com.br
[2] Departamento de Informática, PUC-Rio
R. Marquês de São Vicente, 225, Rio de Janeiro, Brasil, 22453-900
{mreis,poggi}@inf.puc-rio.br
[3] Departamento de Engenharia de Produção, Universidade Federal Fluminense
R. Passo da Pátria, 156, Bloco E, sala 440, Niterói, Brasil, 24210-240
uchoa@producao.uff.br (corresponding author)

September 10th, 2003

Abstract

During the eigthies and early nineties, the best exact algorithms for the Capacitated Vehicle Routing Problem (CVRP) utilized lower bounds obtained by Lagrangean relaxation or column generation. Next, the advances in the polyhedral description of the CVRP yielded branch-and-cut algorithms giving better results. However, several instances in the range of 50–80 vertices, some proposed more than 30 years ago, can not be solved with current known techniques. This paper presents an algorithm utilizing a lower bound obtained by minimizing over the intersection of the polytopes associated to a traditional Lagrangean relaxation over q-routes and the one defined by bounds, degree and the capacity constraints. This is equivalent to a linear program with an exponential number of both variables and constraints. Computational experiments show the new lower bound to be superior to the previous ones, specially when the number of vehicles is large. The resulting branch-and-cut-and-price could solve to optimality almost all instances from the literature up to 100 vertices, nearly doubling the size of the instances that can be consistently solved. Further progress in this algorithm may be soon obtained by also using other known families of inequalities.

1 Introduction

The Capacitated Vehicle Routing Problem (CVRP) consists of given an undirected graph $G = (V, E)$ with vertices numbered as $\{0, 1, \ldots, n\}$ (vertex 0 represents the depot and the remaining vertices represent clients), client demands $d(1), \ldots, d(n)$, lengths $\ell(e)$ associated to edges in E and K vehicles with capacity C; determine routes for each vehicle satisfying the following constraints: (i) each route starts and ends at the depot, (ii) each client is visited by a single vehicle, and (iii) the total demand of clients visited in a route is at most C. The objective is to minimize the sum of the routes length. This classical NP-hard problem was first proposed by Dantzig and Ramsey in 1959 [19] and have received a lot of attention from the optimization community since then, due to its widespread applications, and also because it is a natural generalization of the Travelling Salesperson Problem (TSP).

A landmark exact algorithm for the CVRP was presented in 1981 by Christofides, Mingozzi and Toth [16] using a Lagrangean bound obtained by solving minimum q-route problems. A q-route starts at the depot and traverses a sequence of clients limiting the demand accumulated to at most C and returns to the depot. It is not necessarily a simple path, for some clients may be visited more than once. Therefore, the set of valid CVRP routes is contained in the set of q-routes. The resulting branch-and-bound could solve instances up to 25 vertices, a quite respectful size at that time.

Several other branch-and-bound algorithms using Lagrangean bounds appear in the literature. This same article [16] also describes a lower bound based on k-degree center trees, minimum spanning trees having degree $K \leq k \leq 2K$ on the depot, plus $2K - k$ least cost edges. Other authors propose Lagrangean bounds based on K-trees, which are sets of $n + K$ edges spanning G, like Fisher [21] and Martinhon, Lucena and Maculan [29]. There is also an algorithm based on minimum b-matchings having degree $2K$ at the depot and 2 on the remaining vertices by Miller [30]. The Lagrangean bounds can be improved by dualizing capacity inequalities [21, 30] and also combs and multistar inequalities [29].

Another kind of exact algorithms have its stem on the formulation of the CVRP as a set partition problem by Balinsky and Quandt [9]. In that formulation, a column covers a set of vertices S with total demand not exceeding C and have the cost of a minimum route over $\{0\} \cup S$. Bramel and Simchi-Levi [13] proved that for certain natural classes of instances, the ratio between the lower bounds given by that formulation and the optimal solution values asymptotically approaches 1 as the number of clients grows. However, that formulation in itself is not practical because pricing over the exponential number of columns require the solution of capacitated prize-collecting TSPs, a problem almost as difficult as the CVRP itself. Agarwal, Marthur and Salkin [3] proposed a column generation algorithm on a modified set partition where column costs are given by a linear function over the vertices yielding a lower bound on the actual route cost. Columns with the modified cost can be priced by solving easy knapsack problems. Hadjconstantinou et al. [23] derive lower bounds from heuristic solutions to the dual of the set partitioning formulation. Those dual solutions are obtained by the so-called additive approach, combining the q-path with the K-shortest path relaxations.

For further information and some comparative results on the above mentioned algorithms, we refer the reader to the survey by Toth and Vigo [37].

Starting in the 90's, most of the research effort on the CVRP is now concentrated on the polyhedral description of convex hull of the edge incidence vectors that correspond to K feasible routes and on the development of effective separation algorithms for the families of valid inequalities identified (for instance, [4, 14, 18, 6, 7, 2, 31, 27]). In particular, Araque et al. [5], Augerat et al. [8], Blasum and Hochstattler [12], Ralphs et al. [36], Achutan, Caccetta and Hill [1] and Lysgard, Letchford and Eglese [28] describe complete branch-and-cut algorithms, some including sophisticated inequalities such as framed capacity, strengthened combs, multistar, among others. (The taxonomy and nomenclature of valid inequalities for the CVRP is not uniform in the literature, we are following [28] in this article.)

Although those are the best exact algorithms currently available for the CVRP, the lower bounds obtained at the root nodes, even after adding all those cuts, are not very tight for many instances with as little as 40 vertices. The quality of those bounds is specially problematic for larger values of K, say $K \geq 7$. Many nodes in a branch-and-cut tree may have to be explored in order to close the resulting duality gaps. Even resorting to massive computational power (up to 80 processors running in parallel in a recent work by Ralphs [36, 35]) several instances with less than 80 vertices, including some proposed more than 30 years ago by Christofides and Eilon [15], can not be solved at all. In fact, it seems that branch-and-cut algorithms for the CVRP are experimenting a "diminishing returns" phenomenon, where substantial theoretical and implementation efforts leads to practical results that are only marginally better than those of previous works.

This work presents a new exact algorithm for the CVRP that seems to break through this situation. The main idea is to combine the branch-and-cut approach with the old q-routes approach (which we interpret as a column generation instead of the original Lagrangean relaxation)

to derive superior lower bounds. Since the resulting formulation has an exponential number of both columns and rows, this leads to a branch-and-cut-and-price algorithm. Computational experiments over the main instances from the literature show that this algorithm can consistently solve instances with up to 100 vertices. Seventeen open instances were solved for the first time. Those results were obtained using only bound, degree and capacity inequalities. An improved algorithm is soon expected by also separating framed capacity, strengthened combs, multistar, partial multistar and extended hypotour inequalities, but this was not done yet due to the complexity of implementing the corresponding separation heuristics. [1]

The idea of combining column and cut generation in order to achieve improved lower bounds have existed since the eighties. The difficulty, pointed out for rarely performing that, was the fact that the new dual variables corresponding to separated cuts could change the structure of the pricing subproblem, leading to an intractable pricing (Barnhart et al.[11] and Wilhelm[39] comment on this matter). Recently, several researchers [38, 24, 25, 20, 10] have noted that cuts expressed in terms of variables from a suitable original formulation can be incorporated to the column generation without disturbing the pricing. We use the term "robust branch-and-cut-and-price" to refer to such algorithms: branch-and-bounds over linear programs with an exponential number of both columns and rows and where neither branching nor separation ever change the structure of the pricing subproblems.

The article [33] is a detailed discussion on that matter. In particular, it proposes some reformulation techniques that extend the applicability of robust branch-and-cut-and-price algorithms to virtually any combinatorial optimization problem. Moreover, it is argued that such algorithms may lead to advances on a wide variety of classic problems, from TSP to graph coloring. The present article on the CVRP is part of a larger effort to support that claim. Very good results were also already obtained on the capacitated minimum spanning tree problem [22] and on the generalized assignment problem [32].

2 The New Formulation

A classical formulation for the CVRP [26] represents by x_{ij} the number of times a vehicle traverses edge $(i, j) \in E$. Let V_+ be the set $\{1, \ldots, n\}$ of client vertices. Given a set $S \subseteq V_+$, let $d(S)$ be the sum of the demands of all vertices in S and $\delta(S)$ denote the cut-set defined by S. Let also $k(S) = \lceil d(S)/C \rceil$. Consider the polytope in $R^{|E|}$ next defined:

$$
P_1 \quad = \quad \left\{ \begin{array}{rcll}
\sum\limits_{e \in \delta(\{i\})} x_e & = & 2 & \forall\, i \in V_+ \quad (1) \\
\sum\limits_{e \in \delta(\{0\})} x_e & = & 2.K & \quad (2) \\
\sum\limits_{e \in \delta(S)} x_e & \geq & 2.k(S) & \forall\, S \subseteq V_+ \quad (3) \\
x_e & \leq & 1 & \forall\, e \in E \setminus \delta(\{0\}) \quad (4) \\
x_e & \geq & 0 & \forall\, e \in E
\end{array} \right.
$$

Constraints (1) state that each non-depot vertex is visited once by a vehicle and constraints (2) that K vehicles must go out and in the depot. Constraints (3) are the capacity inequalities requiring that all subsets are served by enough vehicles. Constraints (4) ensure that each edge non adjacent to the depot can be traversed no more than once. Edges adjacent to the depot can be used twice, this is the case where a route serves only one client. The integer vectors x in P_1 define all feasible solutions for the CVRP.

Due to the exponential number of inequalities (3), the lower bound given by

$$L_1 = \min\{\sum_{e \in E} \ell_e.x_e : x \in P_1\}$$

[1] J. Lysgard kindly promised to let us experiment with his implementation of those heuristics, described in [27, 28], in a near future.

has to be calculated by a cutting plane algorithm. In fact, as separating capacity inequalities is NP-hard, one usually resorts to separation heuristics. In that case, the actual bounds obtained may be a little worse than L_1. Modern branch-and-cut algorithms for the CVRP, like [5, 8, 12, 36, 1, 28], may improve this bound by also separating several other known families of inequalities.

Formulations with an exponential number of columns can be obtained by defining variables (columns) that correspond to valid CVRP routes, as first proposed by Balinski and Quandt [9]. Those formulations are not practical since pricing such columns amounts to solving a capacitated prize-collecting TSP, a strongly NP-hard problem. One way of dealing with this difficulty is by enlarging the set of the columns to correspond to q-routes.

As already mentioned in the introduction, a q-route is a closed path visiting a sequence $\{0, v_1, \ldots, v_r, 0\}$ of vertices such that $\sum_{i=1}^{r} d(v_i) \leq C$ and $v_i \neq 0$ for each i in $\{1, \ldots, r\}$, $r \geq 1$. Note that vertices can appear more than once in the q-route. The cost of a q-route is given by $\ell(0, v_1) + \sum_{i=1}^{r-1} \ell(v_i, v_{i+1}) + \ell(v_r, 0)$. Finding a minimum cost q-route can be done in pseudo-polynomial time, $O(n^2.C)$. Christofides, Mingozzi and Toth [16] has shown that restricting the set of q-routes to those without 2-cycles (subpaths $(i, j), (j, i)$) does not change this complexity. Let Q be a $m \times p$ matrix where the columns are the edge incidence vectors of all p possible q-routes with no 2-cycles (except for those corresponding to single client routes). Denote by q_j^e the coefficient associated to edge e in the j^{th} column of Q and consider the following polytope in $R^{p+|E|}$:

$$
P_2 \quad = \quad \left\{ \begin{array}{rcccl}
\sum_{j=1}^{p} q_j^e.\lambda_j & - & x_e & = 0 & \forall e \in E \qquad (5) \\
\sum_{j=1}^{p} \lambda_j & & & = K & \qquad (6) \\
& & \sum_{e \in \delta(\{i\})} x_e & = 2 & \forall\, i \in V_+ \qquad (1) \\
& & x_e & \geq 0 & \forall\, e \in E \\
\lambda_j & & & \geq 0 & \forall\, j = 1, \ldots, p
\end{array} \right.
$$

Constraints (5) define the coupling between variables x and λ. Constraint (6) states the number of vehicles to utilized. Since the definition of the columns already imposes the capacity constraints, it remains to add the degree constraints (1). The integer vectors in P_2 also define all feasible solutions for the CVRP. This is because setting to 1 a variable λ_j corresponding to a q-route that is not a valid CVRP route, violates some degree constraints (1).

Due to the exponential number of variables λ, the lower bound given by

$$L_2 = \min\{\sum_{e \in E} \ell_e.x_e : x \in \mathrm{Proj}_x(P_2)\}$$

has to be calculated by a column generation algorithm, or equivalently, by Lagrangean relaxation. A branch-and-bound using such relaxation [16] was the first really successful exact algorithm for the CVRP.

The alternative way of describing the polyhedra associated to a column generation or to a Lagrangean relaxation in terms of two sets of variables, λ and x, used in the definition of P_2, is called Explicit Master in [33]. The main contribution of this article is to propose a formulation that amounts to optimizing over the intersection of polytopes P_1 and $\mathrm{Proj}_x(P_2)$, which follows. The Explicit Master format makes easy to see that such formulation must be the following.

$$P_1 \bigcap \mathrm{Proj}_x(P_2) = \mathrm{Proj}_x \left\{ \begin{array}{rcl} \sum\limits_{e\in\delta(\{i\})} x_e & = & 2 \quad \forall\, i \in V_+ \quad (1) \\ \sum\limits_{e\in\delta(\{0\})} x_e & = & 2.K \quad \forall\, S \subseteq V_+ \quad (2) \\ \sum\limits_{e\in\delta(S)} x_e & \geq & 2.k(S) \quad \forall\, S \subseteq V_+ \quad (3) \\ x_e & \leq & 1 \quad \forall\, e \in E \setminus \delta(\{0\}) \quad (4) \\ \sum\limits_{j=1}^{p} q_j^e.\lambda_j \; - \; x_e & = & 0 \quad \forall e \in E \quad (5) \\ \sum\limits_{j=1}^{p} \lambda_j & = & K \quad (6) \\ x_e & \geq & 0 \quad \forall\, e \in E \\ \lambda_j & \geq & 0 \quad \forall\, j = 1, \ldots, p \end{array} \right.$$

Remark that constraints (6) can be discarded. They are implied by constraints (2) and (5). The improved lower bound is, then, given by:

$$L_3 = \min\{\sum_{e\in E} \ell_e.x_e : x \in P_1 \bigcap \mathrm{Proj}_x(P_2)\}.$$

This bound can be calculated by solving a linear program with an exponential number of both variables and constraints. A more compact equivalent linear program can be obtained by eliminating the x variables substituting the x_e values given by (5) in constraints (1)-(4). We will refer to the resulting LP as the Dantzig-Wolfe Master problem (DWM).

$$DWM \quad = \quad \left\{ \begin{array}{lrcll} L_3 = \min & \sum\limits_{j=1}^{p} \sum\limits_{e\in E} \ell_e.q_j^e.\lambda_j & & & (7) \\ s.t. & \sum\limits_{j=1}^{p} \sum\limits_{e\in\delta(i)} q_j^e.\lambda_j & = & 2 \quad \forall\, i \in V_+ \quad (8) \\ & \sum\limits_{j=1}^{p} \sum\limits_{e\in\delta(\{0\})} q_j^e.\lambda_j & = & 2.K \quad (9) \\ & \sum\limits_{j=1}^{p} \sum\limits_{e\in\delta(S)} q_j^e.\lambda_j & \geq & 2.k(S) \quad \forall\, S \subseteq V_+ \quad (10) \\ & \sum\limits_{j=1}^{p} q_j^e.\lambda_j & \leq & 1 \quad \forall\, e \in E \setminus \delta(\{0\}) \quad (11) \\ & \lambda_j & \geq & 0 \quad \forall\, j = 1, \ldots, p \end{array} \right.$$

Remark that bound constraints (11) are really necessary, because imposing upper bounds on the x variables can not be done only by upper bounds on λ variables.

3 The Branch-and-Cut-and-Price

3.1 Initialization

We first try to tighten the upper bounds on the variables associated to edges incident to the depot. If $K-1$ vehicles are not enough to serve the $n-1$ clients in the set $V^+ \setminus \{i\}$ (a bin-packing problem) then the upper bound of edge $(0, i)$ can be reduced to 1. We do not actually solve the bin-packing, the simple lower bound $\lceil d(V^+ \setminus \{i\})/C \rceil$ is used instead.

The DWM corresponding to the root node is initialized with a set of artificial variables of high cost covering constraints (8) and (9). Constraints (10) and (11) are omitted.

3.2 Column Generation

The pricing subproblem amounts to determining whether there is a q-route corresponding to a column with a negative reduced cost. In particular, we want to find the q-route with the most negative reduced cost and, if there exists, several others negative reduced cost routes as different as possible from each other. We remind that only 2-cycle free q-routes are being considered.

Let μ, ν, π and ω be the dual variables associated to constraints (8), (9), (10) and (11), respectively. The reduced cost associated to edge e is given by

$$\bar{c}_e = \begin{cases} \ell_e - \mu_i - \mu_j - \sum\limits_{S|\delta(S)\ni e} \pi_S - \omega_e & e = \{i,j\} \in E \setminus \delta(\{0\}) \\ \ell_e - \nu - \mu_j - \sum\limits_{S|\delta(S)\ni e} \pi_S & e = \{0,j\} \in \delta(\{0\}) \end{cases}$$

The reduced cost of a column (λ variable) is given by the sum of the reduced costs of the edges in the q-route associated to it. A minimum reduced cost q-route can be obtained by dynamic programming in $O(n^2.C)$ time. We resort the reader to [16, 17].

We generate columns to DWM after the root initialization, after some cut from (10) or (11) is added to DWM or, in the non-root nodes, after a branching.

3.3 Cut Generation

Given a fractional solution $\bar{\lambda}$ to DWM, the corresponding fractional solution $\bar{x}$ is calculated by using equalities (5). All separation routines work over $\bar{x}$.

Bound constraints (11) are easily separated by inspection. Remark that having all those constraints in the current LP would add $|E|$ rows and too many non-zeros to it.

Capacity constraints (10) are heuristically separated using a set of six routines written by T. Ralphs as part of a demo implementation of a branch-and-cut for the CVRP on the Symphony framework [34]. Those routines are pretty fast, but they are not as good on finding violated capacity constraints as state-of-art separating routines, like those described on [8, 7, 12, 36, 28]. For that reason, we also use an exact separation routine using the MIP solver embedded in CPLEX. Let $\bar{x}$ be a fractional solution. Define y_i as a binary variable equal to 1 iff vertex i belongs to set S and w_{ij} be a variable that is equal to 1 if edge (i,j) belongs to $\delta(S)$. For each value of M in $\{1, \ldots, \lceil d(V^+)/C \rceil - 1\}$ we solve:

$$\begin{array}{lrcll} z = \min & \multicolumn{3}{l}{\sum\limits_{(i,j)\in E} \bar{x}_{ij} w_{ij}} & \\ s.t. & w_{ij} & \geq & y_j - y_i & \forall\, (i,j) \in E \\ & w_{ij} & \geq & y_i - y_j & \forall\, (i,j) \in E \\ & \sum\limits_{i\in V^+} d(i) y_i & \geq & M.C + 1 & \\ & y_0 & = & 0 & \\ & y_i & \in & \{0,1\} & \forall\, i \in V^+ \\ & w_{ij} & \geq & 0 & \forall\, (i,j) \in E \end{array}$$

If $z \leq 2(M+1)$, then the capacity inequality over set S is violated.

Solving those MIPs is computationally costly. In level 0 of the branch-and-bound tree (the root node), we separate capacity constraints exactly, calling the MIP solver whenever the heuristic separation fails, as many times as necessary to be sure that no capacity constraint is violated. In levels 1 up to 4 we perform only one round of exact separation. In deeper levels, only the heuristic separation is performed.

3.4 Branching

We branched by calculating $\bar{x}$ as above and selecting the variable with value closest to 0.5. Note that if for some edge $(0,j)$, $1 < \bar{x}_{0j} < 2$, then for some other edge (i,j), $0 < \bar{x}_{ij} < 1$. In other words, we never need to branch over a fractional variable with value greater than 1.

Fixing a variable x_e to 0 or 1 can be enforced in the DWM by a constraint similar to (11). Edges fixed to 0 can also be removed from the pricing subproblem.

4 Computational Experiments

We present our computation results on 44 instances from the literature available at http://www.branchandcut.org/VRP , a site maintained by T. Ralphs. Those instances are: (i) the 12 instances from series A with no less than 50 vertices; (ii) the 13 instances from series B with no less than 50 vertices; (iii) the 7 instances from series E with no less than 50 vertices; (iv) the 5 instances from series M; and (v) the 7 open instances from series P. On all those instances the lengths are obtained by the Euclidian distance between vertex coordinates rounded to the nearest integer, following the TSPLIB convention. Most of the recent literature on the CVRP also adopt that convention.

Table 1 is a comparison of lower bounds. The first column gives the instance name, indicating its series, the number of vertices and the value of K. For instance, A-n53-k7 is an instance from series A, with 53 vertices (52 clients, plus the depot) and 7 vehicles. The following columns are the bounds L1, L2 and L3. Bound L1 is calculated by a cutting plane algorithm that separates capacity constraints exactly, bound L2 is from a column generation over the q-paths and bound L3 comes from solving DWM to optimality. Column **LLE03** gives the lower bounds shown in [28], on most instances those are the best bounds already obtained by a cutting plane algorithm using several constraints besides capacities. Column **BUB** are the best known upper bounds, already including the improved bounds found in this work. Values in bold indicate upper bounds proved to be optimal. Finally, **Time L3** gives the time in seconds spent by our code to calculate L3 (that is, the time to solve the root node of our branch-and-cut-and-price) on a Pentium 2GHz processor and 512MB of RAM, using CPLEX 7.1 as LP solver.

Some comments about results on Table 1.

- On most instances the bound L3 is significantly better than bounds L2 and LLE03.
- Bound L2 was shown to be unstable. On a number of instances, it can be much better than LLE03 and quite close to L3. This is the case of instance E-n76-k14. However, on some instances it can be very poor, much worse than L1. This happens on all tested instances from series B.
- On 4 instances (B-n56-k7, B-n64-k9, E-n51-k7 and E-n101-k8) bound LLE03 is a little better than L3. This is a sure indication that adding the families of inequalities used by [28] to DWM would give even better bounds.
- The times to calculate L3 are quite reasonable, except for instances from series M. We remark that the exact separation of capacity constraints by solving MIPs responds for more than 70% of the L3 Time on average.

Table 2 presents results of our complete branch-and-cut-and-price algorithm. Columns **L3** and **Root Time** repeat information already given in the previous table. Column **Total Time** is the total cpu time spent. Column **Tree Size** is the number of nodes explored in the branch-and-bound tree. Column **PUB** is the previous best known upper bound found in the literature or at http://www.branchandcut.org/VRP. Values in bold indicate upper bounds already proved to be optimal. Column **OUB** is the upper bound found by our branch-and-cut-and-price. Again, values in bold indicate optimal solutions. Rows with a dash (-) indicate that the complete algorithm was not run for that instance. Some comments follows.

- Our branch-and-cut-and-price could solve 17 previously open instances to optimality, including the classical instances E-n76-k10 and E-n76-k14 proposed by [15]. Our algorithm seems to be significantly better than recent branch-and-cut implementations on instances with many vehicles.
- We believe that our algorithm (perhaps with minor amendments) can consistently solve instances with up to 100 vertices.

- We stopped running instance B-n68-k9 because we felt it would take too much time to finish. Anyway, a solution improving upon the previous best known solution on 3 units was found. This instance has a special structure: its clients are split into 9 very compact clusters and the cluster capacities do not match vehicle capacity. This makes our branching on edges very ineffective. Suppose e is a fractional edge with endpoints on clusters A and B. Fixing e to 0 barely changes the problem, because another edge f also with endpoints in A and B will take the role of e on another fractional solution with almost the same cost. The solution here is branching using constraints on the total value of edges that goes from one cluster to another.

  By the way, instance M-n101-k10 is very easy because its clients are split into 10 clusters and the cluster capacities match vehicle capacity.

- We believe that E-n101-k14 can be solved with our current code, but that would take some weeks of cpu time. We decided that it is better to improve our code before spending so much cpu time.

5 Conclusion

We believe that the good performance of branch-and-cut-and-price algorithms will motivate further polyhedral research on the CVRP in the next years. We take open instance E-n101-k14 as example. A branch-and-cut only using the capacity inequalities gets the lower bound of 1009 in the root node. Adding many other families of inequalities, [28] could improve this bound to 1027. As the best known upper bound for this instance is 1071, this substantial improvement makes little difference in practice: closing a gap of 44 units is impossible anyway. Our branch-and-cut-and-price with only capacity inequalities gets a lower bound of 1052. Closing this gap of 19 units would probably take weeks of cpu time. But now, every unit of gap reduction achieved by adding other families of inequalities makes much practical difference, possibly reducing running time to days or even hours of cpu. To summarize things, now that we know that the "nasty part" of the CVRP polyhedra can somehow be treated by the column generation over the q-routes, our knowledge on the "remaining parts" of the CVRP polyhedra and the availability of good separation heuristics are likely to play a decisive role in the solution of larger and harder instances.

References

[1] N. Achuthan, L. Caccetta, and S. Hill. An improved branch and cut algorithm for the capacitated vehicle routing problem. *Transportation Science*. To appear.

[2] N. Achuthan, L. Caccetta, and S. Hill. Capacited vehicle routing problem: Some new cutting planes. *Asia-Pacific Journal of Operational Research*, 15:109–123, 1998.

[3] Y. Agarwal, K. Mathur, and H. Salkin. A set-partitioning based exact algorithm for the vehicle routing problem. *Networks*, 19:731–739, 1989.

[4] J. Araque, L. Hall, and T. Magnanti. Capacitated trees, capacitated routing and polyhedra. Technical Report SOR-90-12, Princeton University, 1990.

[5] J. Araque, G. Kudva, T. Morin, and J. Pekny. A branch-and-cut algorithm for the vehicle routing problem. *Annals of operations research*, 50:37–59, 1994.

[6] P. Augerat. *Approche polyèdrale du problème de tournées de véhicles*. PhD thesis, Institut National Polytechnique de Grenoble, 1995.

[7] P. Augerat, J. Belenguer, E. Benavent, A. Corbern, and D. Naddef. Separating capacity constraints in the cvrp using tabu search. *European Journal on Operational Research*, (106):546–557, 1995.

[8] P. Augerat, J. Belenguer, E. Benavent, A. Corbern, D. Naddef, and G. Rinaldi. Computational results with a branch and cut code for the capacitated vehicle routing problem. Technical Report 949-M, Universite Joseph Fourier, Grenoble, France, 1995.

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

[10] C. Barnhart, C. Hane, and P. Vance. Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Operations Research*, 40:318–326, 2000.

[11] C. Barnhart, E. Johnson, G. Nemhauser, M. Savelsbergh, and P. Vance. Branch-and-price: column generation for solving huge integer programs. *Operations Research*, 46:316–329, 1998.

[12] U. Blasum and W. Hochstattler. Application of the branch and cut method to the vehicle routing problem. Technical Report zpr2000-386, Zentrum fur Angewandte Informatik Koln, 2000.

[13] J. Bramel and D. Simchi-Levi. On the effectiveness of the set partitioning formulation for the vehicle routing problem. *Operations Research*, 45:295–301, 1997.

[14] V. Campos, A. Corberán, and E. Mota. Polyhedral results for a vehicle routing problem. *European Journal of Operational Research*, 52:75–85, 1991.

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

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

[17] N. Christofides, A. Mingozzi, and P. Toth. State-space relaxation procedures for the computation of bounds to routing problems. *Networks*, 11:145–164, 1981.

[18] G. Cornuéjols and F. Harche. Polyhedral study of the capacitated vehicle routing problem. *Mathematical Programming*, 60:21–52, 1993.

[19] G. Dantzig and R. Ramser. The truck dispatching problem. *Management Science*, 6:80–91, 1959.

[20] G. Felici, C. Gentile, and G. Rinaldi. Solving large MIP models in supply chain management by branch & cut. Technical report, Istituto di Analisi dei Sistemi ed Informatica del CNR, Italy, 2000.

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

[22] R. Fukasawa, M. Poggi de Aragão, O. Porto, and E. Uchoa. Robust branch-and-cut-and-price fot the capacitated minimum spanning tree problem. In *Proceedings of the International Network Optimization Conference*, 2003. To appear.

[23] E. Hadjiconstantinou, N. Christofides, and A. Mingozzi. A new exact algorithm from the vehicle routing problem based on q-paths and k-shortest paths relaxations. In G. Laporte and Michel Gendreau, editors, *Freight Transportation*, number 61 in Annals of Operations Research, pages 21–44. Baltzer Science Publishers, 1995.

[24] D. Kim, C. Barnhart, K. Ware, and G. Reinhardt. Multimodal express package delivery: A service network design application. *Transportation Science*, 33:391–407, 1999.

[25] N. Kohl, J. Desrosiers, O. Madsen, M. Solomon, and F. Soumis. 2-path cuts for the vehicle routing with time windows. *Transportation Science*, 33, 1999.

[26] G. Laporte and Y. Norbert. A branch and bound algorithm for the capacitated vehicle routing problem. *Operations Research Spektrum*, 5:77–85, 1983.

[27] A. Letchford, R. Eglese, and J. Lysgaard. Multistars, partial multistars and the capacitated vehicle routing problem. *Mathematical Programming*, 94:21–40, 2002.

[28] J. Lysgaard, A. Letchford, and R. Eglese. A new branch-and-cut algorithm for capacitated vehicle routing problems. *Mathematical Programming*, May 2003. Submitted to publication.

[29] C. Martinhon, A. Lucena, and N. Maculan. A relax and cut algorithm for the vehicle routing problem. *European Journal of Operational Research*. To appear.

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

[31] D. Naddef and G. Rinaldi. Branch-and-cut algorithms for the capacitated vrp. In P. Toth and D. Vigo, editors, *The vehicle routing problem*, chapter 3, pages 53–84. SIAM, 2002.

[32] A. Pigatti. Modelos e algoritmos para o problema de alocação generalizada e aplicações. Master's thesis, Pontifícia Universidade Católica do Rio de Janeiro, Brazil, July 2003.

[33] M. Poggi de Aragão and E. Uchoa. Integer program reformulation for robust branch-and-cut-and-price. August 2003. Working paper. Presented at ISMP'03 Copenhagen, slides available at www.inf.puc-rio.br/∼poggi.

[34] T. Ralphs. Symphony version 2.8 user's guide, 2002. available at www.branchandcut.org/SYMPHONY.

[35] T. Ralphs. Parallel branch and cut for capacitated vehicle routing. *Parallel Computing*, 29:607–629, 2003.

[36] T. Ralphs, L. Kopman, W. Pulleyblank, and L. Trotter Jr. On the capacitated vehicle routing problem. *Mathematical Programming*, 94:343–359, 2003.

[37] P. Toth and D. Vigo. Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 123:487–512, 2002.

[38] F. Vanderbeck. Lot-sizing with start-up times. *Management Science*, 44:1409–1425, 1998.

[39] W. Wilhelm. A technical review of column generation in integer programming. *Optimization and Engineering*, 2:159–200, 2001.

Instance	L1	L2	L3	LLE03	BUB	Time L3 (s)
A-n53-k7	996.6	978.5	1002.2	998.7	**1010**	157
A-n54-k7	1130.7	1114.0	1150.0	1135.3	**1167**	61
A-n55-k9	1055.9	1025.4	1066.4	1058.3	**1073**	289
A-n60-k9	1316.4	1305.6	1341.6	1319.6	**1354**	129
A-n61-k9	1004.9	996.8	1018.6	1010.2	**1034**	129
A-n62-k8	1244.1	1222.7	1274.1	1251.7	**1288**	105
A-n63-k9	1572.2	1564.8	1603.5	1580.7	**1616**	158
A-n63-k10	1262.2	1267.4	1294.5	1266.6	**1314**	148
A-n64-k9	1340.1	1353.3	1378.9	1351.6	**1401**	84
A-n65-k9	1151.1	1133.0	1163.4	1155.2	**1174**	120
A-n69-k9	1106.7	1113.2	1138.4	1114.4	**1159**	118
A-n80-k10	1699.6	1712.2	1749.7	1709.6	**1763**	327
B-n50-k7	740.0	664.8	741.0	741.0	**741**	10
B-n50-k8	1279.3	1217.5	1291.8	1281.1	**1312**	53
B-n51-k7	1024.6	918.4	1025.9	1025.6	**1032**	42
B-n52-k7	745.0	640.2	746.3	746.0	**747**	99
B-n56-k7	703.4	606.9	704.5	705.0	**707**	83
B-n57-k7	1148.7	1058.5	1150.9	1150.1	**1153**	1347
B-n57-k9	1568.0	1511.5	1595.2	1589.2	**1598**	113
B-n63-k10	1478.9	1418.4	1484.2	1481.0	**1496**	191
B-n64-k9	858.5	769.3	860.1	860.5	**861**	2086
B-n66-k9	1279.8	1223.1	1302.6	1298.5	**1316**	144
B-n67-k10	1023.8	984.5	1026.4	1024.8	**1032**	278
B-n68-k9	1256.4	1163.9	1261.5	1258.1	1272	126
B-n78-k10	1201.2	1124.5	1212.5	1205.6	**1221**	846
E-n51-k5	514.5	512.9	518.0	519.0	**521**	33
E-n76-k7	661.4	663.3	668.4	666.4	**682**	44
E-n76-k8	711.2	716.7	725.1	717.9	**735**	191
E-n76-k10	789.5	811.4	816.5	799.9	**830**	160
E-n76-k14	948.1	999.6	1004.8	969.6	**1021**	94
E-n101-k8	796.3	786.4	801.8	802.6	**815**	645
E-n101-k14	1008.3	1045.1	1051.6	1026.9	1071	271
M-n101-k10	819.5	798.1	820.0	820.0	**820**	189
M-n121-k7	1009.7	1013.0	1030.9	1017.4	**1034**	9295
M-n151-k12	967.2	991.5	996.4	-	1053	10142
M-n200-k16	1187.8	1240.4	1246.7	-	-	12342
M-n200-k17	1195.5	1243.3	1249.2	-	1373	13547
P-n50-k8	596.9	612.5	615.3	-	**631**	70
P-n55-k10	646.7	677.2	680.1	-	**694**	27
P-n55-k15	895.1	963.0	967.5	-	**989**	110
P-n60-k10	708.3	733.5	737.2	-	**744**	26
P-n60-k15	903.3	955.6	961.2	-	**968**	50
P-n65-k10	756.5	779.8	785.2	-	**792**	34
P-n70-k10	786.9	808.3	812.7	-	**827**	74

Table 1: Comparison of bounds.

Instance	**L3**	**Root Time** (s)	**Total Time** (s)	**Tree Size**	**PUB**	**OUB**
A-n53-k7	1002.2	157	2017	185	**1010**	**1010**
A-n54-k7	1150.0	61	11220	2467	**1167**	**1167**
A-n55-k9	1066.4	289	1769	197	**1073**	**1073**
A-n60-k9	1341.6	129	71995	8389	1354	**1354**
A-n61-k9	1018.6	129	17640	4235	**1034**	**1034**
A-n62-k8	1274.1	105	80291	9033	1290	**1288**
A-n63-k9	1603.5	158	38590	3549	1616	**1616**
A-n63-k10	1294.5	148	276259	126840	1315	**1314**
A-n64-k9	1378.9	84	667339	155761	1402	**1401**
A-n65-k9	1163.4	120	2820	301	**1174**	**1174**
A-n69-k9	1138.4	118	288028	31541	1159	**1159**
A-n80-k10	1749.7	327	101331	15609	1763	**1763**
B-n50-k7	741.0	10	10	1	**741**	**741**
B-n50-k8	1291.8	53	-	-	**1312**	-
B-n51-k7	1025.9	42	400	1035	**1032**	**1032**
B-n52-k7	746.3	99	229	3	**747**	**747**
B-n56-k7	704.5	83	358	15	**707**	**707**
B-n57-k7	1150.9	1347	1650	13	**1153**	**1153**
B-n57-k9	1595.2	113	1529	83	**1598**	**1598**
B-n63-k10	1484.2	191	4080	537	**1496**	**1496**
B-n64-k9	860.1	2086	1169	3	**861**	**861**
B-n66-k9	1302.6	144	-	-	**1316**	-
B-n67-k10	1026.4	278	14520	1043	**1032**	**1032**
B-n68-k9	1261.5	126	> 569212	> 141044	1275	1272
B-n78-k10	1212.5	846	9000	15797	**1221**	**1221**
E-n51-k5	518.0	33	188	21	**521**	**521**
E-n76-k7	668.4	44	203352	24799	**682**	**682**
E-n76-k8	725.1	191	94990	7786	**735**	**735**
E-n76-k10	816.5	160	160640	48449	830	**830**
E-n76-k14	1004.8	94	244455	85738	1021	**1021**
E-n101-k8	801.8	645	-	-	**815**	-
E-n101-k14	1051.6	271	-	-	1071	-
M-n101-k10	820.0	189	520	3	**820**	**820**
M-n121-k7	1030.9	9295	206244	529	1034	**1034**
M-n151-k12	996.4	10142	-	-	1053	-
M-n200-k16	1246.7	12342	-	-	-	-
M-n200-k17	1249.2	13547	-	-	1373	-
P-n50-k8	615.3	70	10251	12287	649	**631**
P-n55-k10	680.1	27	65036	19535	696	**694**
P-n55-k15	967.5	110	7136	15161	993	**989**
P-n60-k10	737.2	26	2471	173	756	**744**
P-n60-k15	961.2	50	20454	2403	1033	**968**
P-n65-k10	785.2	34	12394	337	792	**792**
P-n70-k10	812.7	74	413296	79480	834	**827**

Table 2: Branch-and-cut-and-price results.