WORKING PAPER L-2008-02

WP

Jens Lysgaard

The Pyramidal Capacitated Vehicle Routing Problem

CORAL - Centre for Operations Research Applications in Logistics

The Pyramidal Capacitated Vehicle Routing Problem

Jens Lysgaard
CORAL
Department of Business Studies
Aarhus School of Business
University of Aarhus
Denmark
e-mail: lys@asb.dk

March 25, 2008

Abstract

This paper introduces the Pyramidal Capacitated Vehicle Routing Problem (PCVRP) as a restricted version of the Capacitated Vehicle Routing Problem (CVRP). In the PCVRP each route is required to be *pyramidal* in a sense generalized from the Pyramidal Traveling Salesman Problem (PTSP). A pyramidal route is defined as a route on which the vehicle first visits customers in increasing order of customer index, and on the remaining part of the route visits customers in decreasing order of customer index.

Provided that customers are indexed in nondecreasing order of distance from the depot, the shape of a pyramidal route is such that its traversal can be divided in two parts, where on the first part of the route, customers are visited in nondecreasing distance from the depot, and on the remaining part of the route, customers are visited in nonincreasing distance from the depot. Such a route shape is indeed found in many optimal solutions to CVRP instances. An optimal solution to the PCVRP may therefore be useful in itself as a heuristic solution to the CVRP. Further, an attempt can be made to find an even better CVRP solution by solving a TSP, possibly leading to a non-pyramidal route, for each of the routes in the PCVRP solution.

This paper develops an exact branch-and-cut-and-price (BCP) algorithm for the PCVRP. At the pricing stage, elementary routes can be computed in pseudo-polynomial time in the PCVRP, unlike in the CVRP. We have therefore implemented pricing algorithms that generate only *elementary* routes. Computational results suggest that PCVRP solutions are highly useful for obtaining near-optimal solutions to the CVRP. Moreover, pricing of pyramidal routes may due to its efficiency prove to be very useful in column generation for the CVRP.

Key words: vehicle routing, pyramidal traveling salesman, branch-and-cut-and-price.

1 Introduction

This paper introduces the Pyramidal Capacitated Vehicle Routing Problem (PCVRP) as a restricted version of the Capacitated Vehicle Routing Problem (CVRP). In the PCVRP each route is required to be *pyramidal* in a sense generalized from the Pyramidal Traveling Salesman Problem (PTSP). For a formal definition of the PCVRP, it is useful first to consider the CVRP and the PTSP.

The CVRP can be defined as follows ([16, 14, 6]). Let $G = (V, E)$ be a complete undirected graph, with $V = \{0, \ldots, n\}$. Vertex 0 represents a depot, whereas each of the vertices in $V_c = \{1, \ldots, n\}$ represents a customer. The symmetric cost of travel between vertices i and j is denoted by c_{ij}. A unlimited number of identical vehicles, each of capacity $Q > 0$, is available. Each customer i has an integer demand q_i, with $0 < q_i \leq Q$. Each customer must be served by a single vehicle and no vehicle can serve a set of customers whose demand exceeds its capacity. The task is to find a set of vehicle routes of minimum cost, where each vehicle used leaves from and returns to the depot.

The PTSP can be defined as follows ([8, 11]). Let G' be a directed graph with vertex set $\{1, \ldots, n\}$ and a set of arcs with given lengths. A *pyramidal* traveling salesman route in G' is a hamiltonian circuit $(1, i_1, \ldots, i_\alpha, n, j_\beta, \ldots, j_1, 1)$ such that $i_1 < \ldots < i_\alpha$ and $j_1 < \ldots < j_\beta$. The PTSP is to determine a shortest pyramidal traveling salesman route in G'.

In relation to the PTSP, the concepts of *valleys* and *peaks* are useful. In a given hamiltonian circuit containing the subsequence $(\ldots, i, j, k, \ldots)$, vertex j is called a *valley* if and only if $i > j$ and $k > j$, and j is called a *peak* if and only if $i < j$ and $k < j$. In a feasible solution to the PTSP, vertex 1 is the only valley, and vertex n is the only peak. The requirement that a traveling salesman route in G' must be pyramidal is equivalent to requiring that vertex 1 (vertex n) must be the only valley (peak).

We now consider the definition of the PCVRP. First, for the CVRP we define a *pyramidal* route as given by definition 1.

Definition 1 *In the CVRP, a pyramidal route is a sequence of vertices* $(0, i_1, \ldots, i_\alpha, j_\beta, \ldots, j_1, 0)$ *which contains* $\alpha + \beta$ *distinct customers, such that* $i_1 < \ldots < i_\alpha$ *and* $j_1 < \ldots < j_\beta$.

We will use the terms *valley* and *peak* in relation to the CVRP. Their definitions are equivalent to those for the TSP. As such, in a feasible solution to the CVRP, the depot is a valley on each route, and customer n is a peak on one of the routes. Moreover, no customer is a valley, and each route contains exactly one peak. By definition, a pyramidal route is elementary.

Given definition 1, we can now define the PCVRP.

Definition 2 *The PCVRP is the CVRP with the additional restriction that each route must be pyramidal.*

Regarding the complexity of the PCVRP, it is useful to consider its relation to other combinatorial problems. Just as the CVRP may be viewed as conceptually lying in the intersection of the Bin Packing Problem (BPP) and the TSP ([17]), the PCVRP may be viewed as conceptually lying in the intersection of the BPP and the PTSP.

The PCVRP with $c_{ij} = 0$ for $i, j = 1, \dots, n$, and $c_{0j} = \frac{1}{2}$ for $j = 1, \dots, n$, is equivalent to the BPP with item sizes given by the customer demands, and bin capacity Q. As such, the PCVRP generalizes the BPP, which is strongly NP-hard [7, 15]. The PCVRP is therefore strongly NP-hard.

However, the routing aspect is much easier for the PCVRP than for the CVRP. For a given set of customers to be visited on a route, an optimal pyramidal route can be computed in polynomial time. In the general case, the PTSP on n vertices can be solved to optimality in $O(n^2)$ time ([8]).

Based on the relative ease of solving PTSPs, this paper develops specialized column generation algorithms for a Set Partitioning Problem (SPP) formulation of the PCVRP, in which variables (columns) represent pyramidal routes. These algorithms are then embedded in branch-and-price (BP) and branch-and-cut-and-price (BCP) algorithms.

The paper is organized as follows. Section 2 introduces the *Capacitated Prize-Collecting PTSP* which arises as the column generation subproblem for the SPP formulation, along with an exact algorithm as well as a heuristic for solving this problem. Section 3 presents a specialized assignment heuristic for solving the PCVRP. Sections 4 and 5 present our BP and BCP algorithm, respectively. Computional results are presented in Section 6, and conclusions are made in Section 7.

2 The Capacitated Prize-Collecting PTSP

The Capacitated Prize-Collecting PTSP can be defined as a restricted version of the Capacitated Prize-Collecting TSP ([2, 5]), the restriction being that the route must be pyramidal.

More formally, let $G' = (V', E')$ be an undirected graph with vertices $V' = \{0, 1, \dots, n\}$ and edge set E'. Each edge $\{i, j\}$ has a given length d_{ij}. Each vertex i is associated with a nonnegative weight w_i and a prize p_i, where $w_0 = p_0 = 0$. Finally, a weight limit W is given. We emphasize that by Definition 1, any pyramidal route in G' which visits at least two customers is a simple cycle containing vertex 0.

The cost $C(\bar{E}')$ of any simple cycle $\bar{E}' \subseteq E'$ containing vertices $V(\bar{E}')$ is given by the total length of the edges on the cycle, minus the sum of the collected prizes:

$$C(\bar{E}') = \sum_{\{i,j\} \in \bar{E}'} d_{ij} - \sum_{i \in V(\bar{E}')} p_i. \tag{1}$$

Moreover, let the total weight of a cycle be the sum of the weights of the vertices on the cycle.

For completeness, since a route visiting only one customer is not a *simple* cycle, we need to define the cost and weight of a single-customer route, respectively: a route of the form $0 - i - 0$ has cost $2d_{0i} - p_i$ and weight w_i. We can now define the CPCPTSP.

Definition 3 *The CPCPTSP is the problem of finding a minimum-cost pyramidal route in G', subject to the restriction that its total weight must be less than or equal to W.*

The importance of the CPCPTSP is due to its role as the column generation subproblem in the set partitioning problem formulation of the PCVRP.

We now show that the CPCPTSP can be formulated as a weight constrained shortest path problem on an appropriately defined graph. A key concept is that of a *pyramidal path*.

Definition 4 *A pyramidal path is a sequence of vertices $(i_\alpha, \ldots, i_1, 0, j_1, \ldots, j_\beta)$ which contains $\alpha + \beta$ distinct customers, such that $i_1 < \ldots < i_\alpha$ and $j_1 < \ldots < j_\beta$.*

If follows from definitions 1 and 4 that any pyramidal route can be constructed by adding the edge between the two end vertices of a pyramidal path. We exploit this relation for representing the CPCPTSP as a weight constrained shortest path problem in an appropriately defined acyclic graph, which is now described.

Let $G_P = (V_P, A_P)$ be a graph with vertex set V_P and arc set A_P. Each vertex $v(i,j) \in V_P, i < j$ represents all pyramidal paths from i to j. In addition, V_P contains the source vertex $v(0,0)$ and the sink vertex t. The arc set is constructed as follows. For each $j \in V_c$, A_P contains an arc from $v(0,0)$ to $v(0,j)$ having a cost of $d_{0j} - p_j$ and a weight of w_j. From each $v(i,j), i < j$, and for each $k = j+1, \ldots, n$ such that $w_i + w_j + w_k \leq Q$, A_P contains an arc of length $d_{jk} - p_k$ and weight w_k to vertex $v(i,k)$, and an arc of length $d_{ik} - p_k$ and weight w_k to vertex $v(j,k)$. Finally, from each $v(i,j), i < j$, A_P contains an arc of length d_{ij} and weight 0 to vertex t.

The traversal in G_P of an arc from $v(i,j)$ to $v(j,k)$ (where $k > \max\{i,j\}$) represents the addition of edge $\{i,k\}$ to a pyramidal path from i to j. Since $k > \max\{i,j\}$, this results in a pyramidal path from k to j. Moreover, we note that the arc from $v(i,j)$ to t represents the edge between the two end vertices i and j of a pyramidal path.

From any $v(i,j)$, A_P contains an arc to each $k > \max\{i,j\}$ from both i and j. As such, all possible pyramidal paths from i to j are represented by a path in G_P from $v(0,0)$ to $v(i,j)$. Obviously, the sum of the arc weights of the path in G_P equals the sum of the vertex weights of the corresponding pyramidal path.

Consequently, the CPCPTSP is equivalent to the weight constrained shortest path problem (WCSPP) defined on G_P.

In relation to the computational complexity we note that each vertex $v(i,j)$ in G_P is the origin of $O(n)$ arcs, and the number of vertices in G_P is $O(n^2)$, which implies that G_P contains $O(n^3)$ arcs. The computing time for finding a shortest path in an acyclic graph with m arcs is $O(m)$ ([1, Section 4.4]), so a shortest path in G_P can be determined in $O(n^3)$ time.

Obviously, including the weight constraint increases the complexity. Unless all arc weights are equal or all arc costs are equal, the WCSPP is NP-hard, even on acyclic graphs ([4]). A review of the area of the WCSPP is given in [4].

We have implemented two algorithms for the WCSPP. One is an exact pseudopolynomial dynamic programming algorithm, the other is a heuristic based on Lagrangean relaxation. The two algorithms are briefly described in the following subsections.

2.1 An exact dynamic programming algorithm

The weight constraint can be taken into account by including the weight resource in node labels. Let $L(i,j,w)$ denote the length of the shortest path in G_P from $v(0,0)$ to $v(i,j)$, on which the total arc weight equals w. The value of $L(i,j,w)$, for all combinations of i, j (with $i<j$) and $0<w\leq W$, can be found by dynamic programming using the following recurrence relation:

$$L(i,j,w) = \min\left\{\min_{i<k<j}\{L(i,k,w-w_j)+d_{kj}-p_j\}, \min_{0\leq k<i}\{L(k,i,w-w_j)+d_{kj}-p_j\}\right\}. \quad (2)$$

The recurrence is initialized by setting $L(0,0,0)=0$. The shortest path from $v(0,0)$ to t is given by i^*, j^* and w^* according to (3).

$$L(i^*,j^*,w^*)+d_{i^*j^*} = \min_{i,j,w}\{L(i,j,w)+d_{ij}\}. \quad (3)$$

Solving this dynamic programming problem corresponds to finding a shortest path in a particular acyclic graph. Indeed, the structure of G_P reflects the recurrence relation in (2). In our implementation we create $O(W)$ duplicates of each vertex in G_P, one $v(i,j,w)$ for each possible accumulated weight on a path from the source to $v(i,j)$, and create copies of arcs accordingly so that they connect the proper weight indices. In this expanded graph, $L(i,j,w)$ is the length of the shortest path from $v(0,0,0)$ to $v(i,j,w)$.

We note that the number of arcs in this expanded (acyclic) graph is $O(n^3W)$, which therefore also is the running time for finding the shortest path from $v(0,0,0)$ to all $v(i,j,w)$. To conclude, we can solve the CPCPTSP by dynamic programming in $O(n^3W)$ time.

Our implementation is a pulling algorithm in which we consider vertices $v(i,j)$ in lexicographic order. For a given $v(i,j)$, we calculate $L(i,j,w)$ for $w=1,\ldots,W$, and then attempt to eliminate some of these labels before proceeding to the next $v(i,j)$. The elimination is done based on the dominance relation that $L(i,j,w_1)$ dominates $L(i,j,w_2)$ if $L(i,j,w_1)\leq L(i,j,w_2)$ and $w_1<w_2$. According to our experience, this may eliminate a significant part of the labels and may therefore also increase the overall computational speed of the dynamic programming algorithm.

2.2 A Lagrangean relaxation heuristic

The weight constraint may be taken into account in a Lagrangean fashion by redefining the arc lengths in G_P. Specifically, the length of each arc may be increased by an amount proportional to its weight. Given a multiplier u, we increase the length of any arc arriving at $v(i,k)$ (where $i < k$) in G_P by the amount uw_k. Letting $\Lambda(u)$ denote the length of the shortest path in G_P from $v(0,0)$ to t using the modified arc lengths, the Lagrangean dual is the problem of finding a multiplier $u^* \geq 0$ that maximizes $\Lambda(u) - uW$.

This particular Lagrangean dual can be solved by specialized algorithms taking into account that we have dualized only a single constraint. We have implemented the algorithm in [10] for finding u^*.

3 An assignment heuristic for the PCVRP

The special structure of the PCVRP, including the symmetry of costs, makes it possible to use a particular Linear Assignment Problem (LAP) for generating columns which possess certain attractive characteristics. Our purpose of using LAP is to be able to generate promising columns at the early stage of a BP algorithm.

The idea is as follows. We heuristically choose a set of peak customers, and solve the PCVRP without the capacity constraint subject to that fixed set of peak customers. This can be done in polynomial time using the LAP. If the LAP solution contains a route with excess demand, we choose one of its customers as a new peak, and resolve the LAP with the expanded set of peak customers. When all routes are feasible, we use these as columns in the BP algorithm. An attractive characteristic of each route in the LAP solution is that its cost is minimized, i.e., the customers on each route are sequenced optimally under the pyramidal constraint.

In the following we firstly describe how the LAP is solved for a given set of peak customers, and secondly we describe how we choose the set of peak customers.

Let $F = (f_1, \ldots, f_{|F|})$, where $\{n\} \in F$, denote a given set of peak customers. We seek an optimal solution to the PCVRP without the capacity constraint, containing $|F|$ routes such that each route has one of the customers in F as its peak.

The corresponding LAP is given by the cost matrix $\Gamma = (\gamma_{ij})$ containing $n + |F|$ rows and $n + |F|$ columns. Each customer $i \in V_c \setminus F$ is represented by row i and column i, each peak customer f_i is represented by rows f_i and $n + i$, and the depot is represented by columns $f_1, f_2, \ldots, f_{|F|}$ and $n + 1, \ldots, n + |F|$. The costs are defined as follows:

$$\gamma_{ij} = \begin{cases} c_{ij} & \text{for } j \in V_c \setminus F,\ j < i \leq n \\ \infty & \text{for } j \in V_c \setminus F,\ i \leq j \\ c_{i0} & \text{for } i \leq n,\ j \in F \cup \{n+1, \ldots, n+|F|\} \\ \gamma_{f_{i-n}j} & \text{for } i \in \{n+1, \ldots, n+|F|\} \end{cases} \tag{4}$$

By construction of the cost matrix, any LAP solution of finite cost contains two paths from each peak to the depot, all $2|F|$ paths being customer disjoint. Reversing the direction of one of the paths from each peak yields a collection of $|F|$ pyramidal routes. This can be done without changing the total cost, as the PCVRP is defined as being undirected. Moreover, subtours are not possible in a finite cost solution, so the $|F|$ routes collectively visit all customers.

Using a straightforward implementation of the Hungarian algorithm, for example the APC code in [3], the LAP can be solved in $O(n^3)$ time. However, we have specialized the code for solving the LAP, taking into account the special structure of the cost matrix. The resulting algorithm, which is a specialized version of the APC code, runs in $O(|F|n^2)$ time. Specifically, we first solve the LAP for $F = \{n\}$ and then iteratively add the remaining peaks to F.

To solve the LAP with $F = \{n\}$, we first identify the LAP solution for only one path from customer n to the depot. The solution to this LAP is trivially $(1, n), (2, 1), (3, 2), \ldots, (n, n-1)$. Letting u and v denote the dual variables associated with rows and columns, respectively, we have the following optimal dual values: $u_n = \gamma_{n,n-1}$; $u_i = u_{i+1} + \gamma_{i,i-1}$ for $i = n-1, \ldots, 2$; $u_1 = u_2 + \gamma_{1,n}$; $v_{n-1} = 0$; $v_n = -u_2$; and $v_j = -u_{j+2}$ for $j = 1, \ldots, n-2$. Then we add row $n+1$ representing customer n, and column $n+1$ representing the depot, and perform a single iteration of the Hungarian algorithm in $O(n^2)$ time. (We note that this is in fact an alternative $O(n^2)$ algorithm for solving the PTSP with symmetric costs.)

Subsequently we repeatedly add a peak customer until the resulting assignment solution is feasible, i.e., such that the demand on each route does not exceed the vehicle capacity. This process contains two main operations, one for choosing a peak from a given set of candidates, and another for choosing the candidate sets. A description of these two operations follows.

A bit more notation is needed for the following. For any set of customers $S \subseteq V_c$, let $k(S)$ denote $\lceil \sum_{i \in S} q_i / Q \rceil$. Clearly, $k(S)$ is a lower bound on the number of vehicles needed to service all customers in S.

The choice of candidate sets is divided into two stages. In the first stage, we choose $k(V_c) - 1$ additional peaks as follows. For $p = 2, \ldots, k(V_c)$, let θ_p denote the largest customer index such that $k(\{\theta_p, \ldots, n\}) = p$, and let $\Theta_p = \{\theta_p, \ldots, n\}$. Each of the p (or more) routes visiting at least one customer in Θ_p has its peak customer in Θ_p. As such, Θ_p contains at least p peaks in any feasible PCVRP solution. As a consequence of this observation, we choose for $p = 2, \ldots, k(V_c)$ a peak among those customers in Θ_p which have not already been chosen as a peak.

In the second stage, we repeatedly choose, among the routes in the latest LAP solution, the route with largest demand, as long as this demand exceeds the vehicle capacity. Suppose that the route has a demand which exceeds Q, let R denote the set of customers on the route, and let p denote its peak. We begin with the set $S = R \setminus \{p\}$ and then repeatedly remove the smallest indexed customer from S as long as the resulting S satisfies $k(S \cup \{p\}) > 1$. The final S is then our candidate set.

The second main operation is the choice of a peak from a given candidate set S. We implemented a randomized choice of the peak, such that repeated use of the overall procedure may lead to different solutions. A central part of this operation is to assign probabilities to candidate nodes. For this, the following cost measure for each candidate node $j \in S$ is used:

$$\Upsilon_j = \min_{i \in \{j+1, \ldots, n\}} c_{ij} + c_{j0} - c_{i0}. \tag{5}$$

The value of Υ_j is taken as a measure of the cost of inserting customer j into a pyramidal route such that customer j is not the peak. As such, a large value of Υ_j suggests that it would be sensible to choose customer j as a peak. For completeness, we define $\Upsilon'_j = \max\{\epsilon, \Upsilon_j\}$ in order to handle nonpositive insertion costs, where ϵ is some small positive number. Finally, we let the probability of choosing node j be proportional to Υ'_j.

4 Branch-and-price

The PCVRP can be formulated as an SPP, in which variables (columns) represent pyramidal routes.

We call a route *feasible* if and only if the total demand on the route does not exceed the vehicle capacity. Let $\Re$ denote the set of all feasible pyramidal routes. Further, let α_{ir} be a parameter of value 1 if route r visits customer i and 0 otherwise, let λ_r be a variable of value 1 if route r is chosen and 0 otherwise, and let d_r be the cost of route r. This leads to the following SPP formulation:

$$\min \quad \sum_{r\in\Re} d_r\lambda_r \tag{6}$$

$$\text{s.t.:} \quad \sum_{r\in\Re} \alpha_{ir}\lambda_r = 1 \quad \forall i \in V_c \tag{7}$$

$$\lambda_r \in \{0,1\} \qquad \forall r \in \Re \tag{8}$$

This model can be solved by branch-and-bound, where the LP relaxation in each subproblem is solved by column generation (CG). The resulting algorithm is a branch-and-price (BP) algorithm.

The column generation subproblem is the problem of determining a feasible pyramidal route of minimum reduced cost, for a given vector of dual prices associated with (7). This is exactly the CPCPTSP as presented in Section 2, where the dual prices of (7) are used as prizes, and customer demands are used as vertex weights.

The initial LP contains the n single-customer routes. Further, we initially make n repetitions of the LAP heuristic, and add all resulting columns (after removal of duplicates) to the LP.

Subsequently, for column generation after each LP reoptimization, we first call the Lagrangean relaxation heuristic, and only if this fails do we call the exact dynamic programming algorithm.

For branching we choose an edge $\{i, j\}$ in the underlying flow network which minimizes the expression $|x^*_{ij} - \frac{1}{2}||i-j|$, where x^*_{ij} is the current flow along edge $\{i, j\}$. The untraditional multiplier $|i-j|$ is due to the structure of pyramidal routes. In particular, if $i = j+1$, prohibiting this edge implies that i and j cannot be visited on the same route unless either i) the peak of the route is visited in between i and j, or ii) i or j is the peak of the route. This makes the usually weaker *0-branch* appear more tightly constrained than with ordinary routes.

Our node selection strategy is 'best-bound', i.e., when choosing a subproblem to be processed next, we choose the subproblem with the smallest lower bound.

When the BP algorithm has found an optimal PCVRP solution, we attempt to find an even better CVRP solution by solving a TSP for each route in the PCVRP solution. For this we have implemented a simple branch-and-cut algorithm for the symmetric TSP, in which only subtour elimination constraints are separated at each subproblem. The computational burden of this postprocessing step is negligible relative to the overall BP algorithm.

5 Branch-and-cut-and-price

For our BCP algorithm we use a formulation of the PCVRP which is obtained by modifying the CVRP formulation used in [6]. The key difference is our definition of routes which involves the pyramidality constraint. We use only one class of cuts, namely the rounded capacity inequalities which are separated using the CVRPSEP package in [13].

In our implementation, each subproblem is processed by first doing column generation until no more columns are generated, after which we call the cut separation routine. If this succeeds in finding at least one violated cut, we reoptimize the LP and repeat the process, again beginning with column generation. The process stops when both column generation and separation fail on the same LP solution.

The other ingredients, including the node selection strategy and the choice of branching, are the same as those in the BP algorithm. Although it would be possible to branch on sets as in [14, 6], we preferred to use the same branching strategy as in the BP algorithm in order to isolate the effect of adding cuts.

6 Computational results

We have applied our BP and BCP algorithms on a set of test instances all of which are available from `www.branchandcut.org`. More specifically, we give results for all `A` and `E` instances and for two `M` instances. The optimal CVRP solution is known for all these instances, which makes it possible to investigate the deviation from CVRP optimality that results from PCVRP solutions.

In our experiments we fix the number of routes at the minimum possible by adding a constraint on the sum of all route variables to the SPP formulation.

The computations were done on a PC with a 2.0 GHz Intel Centrino Duo processor and 512 MB RAM running under Microsoft Windows XP. We used the ILOG CPLEX 9.1 callable library and the Microsoft Visual Studio 2005 C/C++ compiler.

Table 1 shows the results. Column 'Name' gives the name of the instance, and column 'CVRP OPT' gives the optimal CVRP objective value. Column 'PCVRP OPT' gives the optimal PCVRP objective value, and column 'PCVRP OPT+TSP' gives the objective value after resequencing the routes in the PCVRP solution, i.e., the value obtained by solving a TSP for the customer set (and the depot) of each route in the PCVRP solution. Column 'Dev.' gives the deviation between 'CVRP OPT' and 'PCVRP OPT+TSP', i.e., a measure of quality of the 'PCVRP OPT+TSP' solution when viewed as a heuristic solution to the CVRP (a blank entry represents a deviation of zero). A '—' in columns 'CVRP OPT', 'PCVRP OPT+TSP', and 'Dev.' represents that the problem was not solved to optimality.

The next five columns give the results for the BP algorithm. Column 'Lagr.' gives the objective value obtained at the root node of the search tree when the Lagrangean relaxation heuristic fails for the first time, i.e., without using the exact dynamic programming algorithm for column generation. Columns 'Root LB' and 'Root time' give the final lower bound, and the computing time in seconds, for the root node of the search tree. Columns 'BB time' and 'BB tree size' give the total computing time in seconds for solving the problem to optimality, and the size of the search tree (branch-and-bound tree). A '—' in column 'BB time' represents that the problem was not solved within a time limit of two CPU hours.

The final four columns, which show the results for the BCP algorithm, have the same meaning as the last four under the 'Branch-and-price' heading.

We draw two main conclusions from Table 1. First, the 'PCVRP OPT+TSP' solution does indeed come very close to the CVRP optimum, and in reasonable time. On many instances it coincides with the CVRP optimum, and the difference is very small on a considerable part of the remaining instances. This, in turn, suggests that the column generation procedures that are presented in this paper may also be useful as heuristics in column generation for the CVRP. Second, it is evident that the BCP algorithm is consistently faster than the BP algorithm (with the only exceptions being the easily solved `E-22-4` and `E-23-3`). As such, the addition of cuts has proven useful although only *elementary* routes are involved in the BP algorithm.

7 Conclusions

This paper has introduced the Pyramidal Capacitated Vehicle Routing Problem (PCVRP) as a restricted version of the Capacitated Vehicle Routing Problem (CVRP). The added pyramidality restriction has led to pseudo-polynomial column generation of elementary routes. Computational results show that the optimal PCVRP solution tends to be very close to the optimal CVRP solution, in particular after resequencing the customers on each route in the PCVRP solution. The results also show that the BCP algorithm performs consistently better than the BP algorithm.

The encouraging results in this paper suggest two directions for further research. First, the introduced pseudo-polynomial algorithm for column generation of pyramidal routes could be used as a heuristic in an exact column generation of elementary routes for the CVRP. The results in this paper indicate that such a heuristic would indeed have a very good performance. Second, a similar restriction of route shape could be considered for other related routing problems. In particular, a similarly restricted version of the Capacitated Open Vehicle Routing Problem (COVRP) [12] could be obtained by requiring that customers are visited in nondecreasing order of distance from the depot. In that case it would be possible to obtain a dynamic programming algorithm for exact column generation that runs in $O(n^2Q)$ time, i.e., a factor of n faster than for closed routes. We leave these and related issues for further research.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows: Theory, Algorithms, and Applications.* Prentice-Hall, 1993.

[2] A. E. Bixby. *Polyhedral Analysis and Effective Algorithms for the Capacitated Vehicle Routing Problem.* PhD thesis, Northwestern University, Evanston, Illinois, 1999.

[3] G. Carpaneto, S. Martello, and P. Toth. Algorithms and codes for the assignment problem. *Annals of Operations Research*, 13(1):193–223, 1988.

[4] I. Dumitrescu and N. Boland. Algorithms for the weight constrained shortest path problem. *International Transactions in Operational Research*, 8(1):15–29, 2001.

[5] M. Fischetti, J.-J. Salazar-González, and P. Toth. The generalized traveling salesman and orienteering problems. In Gutin and Punnen [9], chapter 13.

[6] R. Fukasawa, H. Longo, J. Lysgaard, M. Poggi de Aragão, M. Reis, E. Uchoa, and R. F. Werneck. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical Programming*, 106(3):491–511, 2006.

[7] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, 1979.

[8] P. C. Gilmore, E. L. Lawler, and D. B. Shmoys. Well-solved special cases. In E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, editors, *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, chapter 4. Wiley, 1985.

[9] G. Gutin and A. P. Punnen, editors. *The Traveling Salesman Problem and Its Variations.* Kluwer, 2002.

[10] G. Y. Handler and I. Zang. A dual algorithm for the constrained shortest path problem. *Networks*, 10(4):293–309, 1980.

[11] S. N. Kabadi. Polynomially solvable cases of the TSP. In Gutin and Punnen [9], chapter 11.

[12] A. N. Letchford, J. Lysgaard, and R. W. Eglese. A branch-and-cut algorithm for the capacitated open vehicle routing problem. *Journal of the Operational Research Society*, 58(12):1642–1651, 2007.

[13] J. Lysgaard. CVRPSEP: A package of separation routines for the capacitated vehicle routing problem. Working Paper 03-04, Department of Management Science and Logistics, Aarhus School of Business. Available at www.hha.dk/∼lys, 2003.

[14] J. Lysgaard, A. N. Letchford, and R. W. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, 100(2):423–445, 2004.

[15] S. Martello and P. Toth. *Knapsack Problems: Algorithms and Computer Implementations*. Wiley, 1990.

[16] 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. SIAM Monographs on Discrete Mathematics and Applications, 2002.

[17] T. K. Ralphs, L. Kopman, W. R. Pulleyblank, and L. E. Trotter. On the capacitated vehicle routing problem. *Mathematical Programming*, 94(2–3):343–359, 2003.

					Branch-and-price					Branch-and-cut-and-price			
Name	CVRP OPT	PCVRP OPT	PCVRP OPT+TSP	Dev.	Lagr.	Root LB	Root time	BB time	BB tree size	Root LB	Root time	BB time	BB tree size
A-32-5	784	784	784		839.0	771.1	0.2	0.7	17	784*	0.3	0.3	1
A-33-5	661	665	661		727.4	659.8	0.2	0.5	27	665*	0.2	0.2	1
A-33-6	742	744	742		804.4	734.5	0.2	1.4	139	743.5	0.2	0.3	3
A-34-5	778	792	787	9	798.3	763.2	0.2	395.9	3897	787.9	0.4	1.3	23
A-36-5	799	814	813	14	836.8	789.7	0.2	5713.5	12757	809.3	0.6	2.0	33
A-37-5	669	673	673	4	701.7	671.2	0.4	0.5	9	673*	0.3	0.3	1
A-37-6	949	962	949		1014.3	942.4	0.5	20.3	663	957.9	0.9	1.5	13
A-38-5	730	738	730		813.5	706.9	0.7	263.1	3187	731.9	1.0	2.1	15
A-39-5	822	825	825	3	914.0	812.2	0.6	15.0	429	821.9	0.8	2.3	37
A-39-6	831	838	833	2	889.2	814.7	0.3	44.6	875	838*	0.7	0.7	1
A-44-6	937	948	942	5	1022.1	938.8	1.0	4.3	129	945.9	1.2	2.2	9
A-45-6	944	971	967	23	1208.5	940.0	1.1	288.3	2409	959.9	1.7	6.8	49
A-45-7	1146	1146	1146		1280.8	1129.7	0.9	75.6	2017	1143.7	1.5	3.3	25
A-46-7	914	927	923	9	979.0	917.5	0.4	3.9	239	927*	0.7	0.7	1
A-48-7	1073	1073	1073		1180.4	1065.4	0.7	5.2	99	1073*	0.9	0.9	1
A-53-7	1010	1019	1010		1082.5	1004.6	1.1	29.5	453	1013.6	2.3	9.0	21
A-54-7	1167	1175	1167		1276.9	1150.8	3.0	—	18053	1165.0	6.3	23.0	77
A-55-9	1073	1078	1073		1100.8	1065.0	1.1	28.0	875	1072.4	1.7	5.1	25
A-60-9	1354	1357	1356	2	1537.5	1330.1	2.0	—	25487	1347.8	5.3	71.1	255
A-61-9	1034	1040	1038	4	1137.7	1017.2	3.9	458.5	2543	1028.8	5.2	20.1	79
A-62-8	1288	1294	1288		1431.3	1259.3	3.1	—	16133	1288.0	7.3	55.3	209
A-63-10	1314	1319	1316	2	1418.9	1290.3	2.1	—	29575	1308.0	3.1	818.0	1183
A-63-9	1616	1626	1616		1741.1	1588.6	4.4	—	20241	1610.6	7.2	1365.3	1109
A-64-9	1401	1419	1418	17	1550.8	1391.6	3.2	—	14097	1404.5	4.1	1553.8	1433
A-65-9	1174	1183	1177	3	1331.9	1155.5	4.6	—	26635	1172.5	7.3	20.8	29
A-69-9	1159	1184	1165	6	1240.9	1151.9	3.8	—	17687	1165.0	6.7	6763.9	2877
A-80-10	1763	1763	1763		1919.8	1738.4	9.4	—	14317	1758.7	14.9	32.4	33
E-22-4	375	375	375		399.7	373.9	0.1	0.1	5	375*	0.2	0.2	1
E-23-3	569	652	652	83	—	652*	0.1	0.1	1	652*	0.2	0.2	1
E-30-3	534	538	534		679.1	501.0	0.2	572.6	4993	538*	0.3	0.3	1
E-33-4	835	835	835		885.1	825.5	0.3	13.2	469	835*	0.6	0.6	1
E-51-5	521	540	521		563.0	534.7	2.1	6.7	59	535.5	2.4	5.2	27
E-76-7	682	702	696	14	701.2	686.8	2.9	—	6895	692.9	4.7	67.1	159
E-76-8	735	743	740	5	753.0	729.6	8.5	—	11693	736.5	18.9	135.9	115
E-76-10	830	—	—	—	871.6	819.4	7.6	—	22719	823.2	10.0	—	2385
E-76-14	1021	1021	1021		1063.9	1007.8	4.6	228.6	4559	1010.4	4.9	267.2	993
E-101-8	815	—	—	—	829.7	811.2	28.3	—	5889	820.9	81.2	—	1357
E-101-14	1067	—	—	—	1121.4	1054.3	10.4	—	19437	1058.6	14.0	—	2947
M-101-10	820	846	821	1	876.4	842.4	19.2	28.9	63	845.5	21.6	31.9	3
M-121-7	1034	1065	1037	3	1280.4	1062.5	529.7	931.5	99	1065*	621.8	621.9	1

Table 1: Computational results.

Working Papers from CORAL – Centre for Operations Research Applications in Logistics

L-2008-02	Jens Lysgaard: The Pyramidal Capacitated Vehicle Routing Problem.
L-2008-01	Jens Lysgaard & Janni Løber: Scheduling participants of Assessment Centres.
L-2007-03	Christian Larsen: Note: Comments on the paper by Rosling (2002).
L-2007-02	Christian Larsen, Claus Hoe Seiding, Christian Teller & Anders Thorstenson: An inventory control project in a major Danish company using compound renewal demand models.
L-2007-01	Christian Larsen: The *Q(s,S)* control policy for the joint replenishment problem extended to the case of correlation among item-demands.
L-2006-11	Daniele Pretolani, Lars Relund Nielsen & Kim Allan Andersen: A note on “Multicriteria adaptive paths in stochastic, time-varying networks”.
L-2006-10	Lars Relund Nielsen, Kim Allan Andersen & Daniele Pretolani: Bicriterion a priori route choice in stochastic time-dependent networks.
L-2006-09	Christian Larsen & Gudrun P. Kiesmüller: Developing a closed-form cost expression for an *(R,s,nQ)* policy where the demand process is compound generalized Erlang.
L-2006-08	Eduardo Uchoa, Ricardo Fukasawa, Jens Lysgaard, Artur Pessoa, Marcus Poggi de Aragão, Diogo Andrade: Robust Branch-Cut-and-Price for the Capacitated Minimum Spanning Tree Problem over a Large Extended Formulation.
L-2006-07	Geir Brønmo, Bjørn Nygreen & Jens Lysgaard: Column generation approaches to ship scheduling with flexible cargo sizes.
L-2006-06	Adam N. Letchford, Jens Lysgaard & Richard W. Eglese: A Branch-and-Cut Algorithm for the Capacitated Open Vehicle Routing Problem.
L-2006-05	Ole Mortensen & Olga W. Lemoine: Business integration between manufacturing and transport-logistics firms.
L-2006-04	Christian H. Christiansen & Jens Lysgaard: A column generation approach to the capacitated vehicle routing problem with stochastic demands.
L-2006-03	Christian Larsen: Computation of order and volume fill rates for a base stock inventory control system with heterogeneous demand to investigate which customer class gets the best service.

L-2006-02	Søren Glud Johansen & Anders Thorstenson: Note: Optimal base-stock policy for the inventory system with periodic review, backorders and sequential lead times.
L-2006-01	Christian Larsen & Anders Thorstenson: A comparison between the order and the volume fill rates for a base-stock inventory control system under a compound renewal demand process.
L-2005-02	Michael M. Sørensen: Polyhedral computations for the simple graph partitioning problem.
L-2005-01	Ole Mortensen: Transportkoncepter og IT-støtte: et undersøgelsesoplæg og nogle foreløbige resultater.
L-2004-05	Lars Relund Nielsen, Daniele Pretolani & Kim Allan Andersen: *K* shortest paths in stochastic time-dependent networks.
L-2004-04	Lars Relund Nielsen, Daniele Pretolani & Kim Allan Andersen: Finding the *K* shortest hyperpaths using reoptimization.
L-2004-03	Søren Glud Johansen & Anders Thorstenson: The (r,q) policy for the lost-sales inventory system when more than one order may be outstanding.
L-2004-02	Erland Hejn Nielsen: Streams of events and performance of queuing systems: The basic anatomy of arrival/departure processes, when focus is set on autocorrelation.
L-2004-01	Jens Lysgaard: Reachability cuts for the vehicle routing problem with time windows.

Handelshøjskolen i Århus

Aarhus
School of Business

ISBN 9788778823151

Department of Business Studies

Aarhus School of Business
University of Aarhus
Fuglesangs Allé 4
DK-8210 Aarhus V - Denmark

Tel. +45 89 48 66 88
Fax +45 86 15 01 88

www.asb.dk