# An exact solution framework for a broad class of vehicle routing problems

**Baldacci, Roberto; Bartolini, Enrico; Mingozzi, Aristide; Roberti, Roberto**

*Published in:*
Computational Management Science

*Link to article, DOI:*
10.1007/s10287-009-0118-3

*Publication date:*
2010

*Document Version*
Publisher's PDF, also known as Version of record

*Citation (APA):*
Baldacci, R., Bartolini, E., Mingozzi, A., & Roberti, R. (2010). An exact solution framework for a broad class of vehicle routing problems. *Computational Management Science*, *7*(3), 229-268. https://doi.org/10.1007/s10287-009-0118-3

Comput Manag Sci (2010) 7:229–268
DOI 10.1007/s10287-009-0118-3

ORIGINAL PAPER

# An exact solution framework for a broad class of vehicle routing problems

**Roberto Baldacci · Enrico Bartolini · Aristide Mingozzi · Roberto Roberti**

Received: 8 May 2009 / Accepted: 4 December 2009 / Published online: 1 January 2010

**Abstract** This paper presents an exact solution framework for solving some variants of the vehicle routing problem (VRP) that can be modeled as set partitioning (SP) problems with additional constraints. The method consists in combining different dual ascent procedures to find a near optimal dual solution of the SP model. Then, a column-and-cut generation algorithm attempts to close the integrality gap left by the dual ascent procedures by adding valid inequalities to the SP formulation. The final dual solution is used to generate a reduced problem containing all optimal integer solutions that is solved by an integer programming solver. In this paper, we describe how this solution framework can be extended to solve different variants of the VRP by tailoring the different bounding procedures to deal with the constraints of the specific variant. We describe how this solution framework has been recently used to derive exact algorithms for a broad class of VRPs such as the capacitated VRP, the VRP with time windows, the pickup and delivery problem with time windows,

R. Baldacci (✉)
Department of Electronics, Computer Science and Systems (DEIS),
University of Bologna, Via Venezia 52, 47521 Cesena, Italy
e-mail: r.baldacci@unibo.it

E. Bartolini
Department of Computer Science, University of Bologna,
Mura Anteo Zamboni 7, 40127 Bologna, Italy
e-mail: enrico.bartolini2@unibo.it

A. Mingozzi
Department of Mathematics, University of Bologna, Via Sacchi 3, 47521 Cesena, Italy
e-mail: mingozzi@csr.unibo.it

R. Roberti
Department of Electronics, Computer Science and Systems (DEIS),
University of Bologna, Viale Risorgimento 2, 40136 Bologna, Italy
e-mail: roberto.roberti6@unibo.it

all types of heterogeneous VRP including the multi depot VRP, and the period VRP. The computational results show that the exact algorithm derived for each of these VRP variants outperforms all other exact methods published so far and can solve several test instances that were previously unsolved.

**Keywords** Vehicle routing · Set partitioning · Dual ascent · Valid inequalities

**Mathematics Subject Classification (2000)** 90-02 · 90C27 · 49M29 · 90C39

## 1 Introduction

The problems of supplying customers from a central depot (or depots) using vehicles and drivers based at the depot(s) are generally known as *vehicle routing problems* (VRPs). The solution of a VRP calls for the design of a set of *routes*, each performed by a single vehicle starting and ending at its own depot, such that all customers are serviced, a set of *operational constraints* are satisfied, and the total distribution cost is minimized.

Real-world VRPs often include complications that depend on the nature of the transported goods, the quality of service required, and the characteristics of customers and vehicles. Some typical complications are: heterogeneous vehicles located at different depots, customers incompatible with certain vehicle types, customers accepting delivery within specified time windows, multi-commodity demands that have to be loaded into separate vehicle compartments, a multiple-day planning horizon, vehicles performing multiple routes.

In all cases, the objective is to supply the customers at minimum cost. This cost includes the costs of vehicles used, the cost of the routes performed, and a "penalty" for each customer left unserviced within its due date.

The two simplest and most studied members of the VRP family are the capacitated VRP (CVRP) and the VRP with time windows (VRPTW). In the CVRP, a fleet of identical vehicles located at a central depot has to be optimally routed to supply a set of customers with known demands. Each vehicle can perform at most one route, and the total demand of the customers visited by a route cannot exceed the vehicle capacity. The VRPTW generalizes the CVRP imposing that each customer is to be visited within a specified time interval, called *time window*.

Several heuristics but few exact methods have been proposed in the literature for the VRP. Surveys of both exact and heuristic methods can be found in Toth and Vigo (2002), Cordeau et al. (2007) and Baldacci et al. (2007).

Concerning exact methods, most of the research effort has been devoted to the CVRP and the VRTPW. Two main approaches have been used: branch-and-cut (BC) methods based on a two-index flow model, and branch-and-cut-and-price (BCP) methods based on a set partitioning (SP) model where SP columns correspond to feasible routes. In both cases, valid inequalities are used to strengthen the LP-relaxation of the corresponding model. Computational results indicate that BCP methods usually dominate BC methods. Indeed, these latter methods perform well only on instances involving low vehicle utilization and a large number of customers per route.

Almost all BCP algorithms use the simplex algorithm to compute lower bounds. The main drawback of this approach is that the LP-relaxation of the master problem is usually highly degenerate, yet degeneracy implies alternative optimal dual solutions. As a result, the generation of new columns may not change the value of the objective function of the master, making the master large and the overall method time-consuming.

Mingozzi et al. (1994) proposed a method for avoiding the drawbacks of column generation based on the simplex algorithm that combines in an additive way different dual heuristics to find a near optimal dual solution of the LP-relaxation of the SP model. These dual heuristics are based on a bounding method proposed by Christofides et al. (1981c) and Christofides and Mingozzi (1989). The method was successfully extended by Mingozzi et al. (1999) for the VRP with backhauls, Baldacci et al. (2004b) for the car pooling problem, Baldacci et al. (2006) for the multiple facilities rollon-rolloff VRP, and Boschetti et al. (2004) for the multi depot crew scheduling problem.

Recently, Baldacci et al. (2008b) proposed a new exact method for the CVRP based on the SP model that provides a general solution framework for designing exact algorithms for a broad class of VRPs.

In Sect. 2, we describe the main characteristics of this method. In Sects. 3 and 4, we present the extensions of the method for solving the VRPTW and the pickup and delivery problem with time window (PDPTW), respectively. In Sect. 5, we describe a generalization of the method for solving the Heterogeneous VRP (HVRP). It is worth mentioning that the HVRP model considered contains as special cases all variants of the HVRP presented in the literature, the site-dependent vehicle routing problem (SDVRP), and the multi-depot vehicle routing problem (MDVRP). In Sect. 6, we describe an exact method for the Period VRP (PVRP). In each section, we report computational results showing the effectiveness of the proposed exact algorithms for each VRP variant considered.

## 2 The capacitated vehicle routing problem

The CVRP can be described as follows. It is given an undirected graph $G = (V', E)$ where $V' = \{0, 1, \ldots, n\}$ is the set of $n + 1$ vertices and $E$ is the set of edges. Vertex 0 represents the depot, and the vertex set $V = V' \setminus \{0\}$ corresponds to $n$ customers. A nonnegative cost $d_{ij}$ is associated with each edge $\{i, j\} \in E$. Each customer $i \in V$ requires a supply of $q_i$ units from depot 0 (we assume $q_0 = 0$), and a set of $m$ identical vehicles of capacity $Q$ stationed at depot 0 must be used to supply the customers. A *route* is defined as a least cost simple cycle in $G$ passing through the depot 0 and such that the total demand of the customers visited does not exceed the vehicle capacity $Q$.

The objective of the CVRP is to design at most $m$ routes so that all customers are visited exactly once and the sum of the route costs is minimized.

The CVRP is $\mathcal{NP}$-hard as it is a natural generalization of the traveling salesman problem (TSP).

Different heuristics have been proposed in the literature for the CVRP and its variants. Among the various surveys on heuristic algorithms for the CVRP, we mention

Laporte and Semet (2002) and Gendreau et al (2002) in the book edited by Toth and Vigo (2002), and the more recent update by Cordeau et al. (2007).

The most effective exact algorithms for the CVRP are due to Baldacci et al. (2004a, 2008b), Lysgaard et al. (2004) and Fukasawa et al. (2006). Baldacci et al. (2004a) described a BC algorithm based on a Two-Commodity network flow formulation of the CVRP. Lysgaard et al. (2004) proposed a BC algorithm improving the method proposed by Augerat et al. (1995); they used a variety of valid inequalities including capacity, framed capacity, comb, partial multistar, hypotour and Gomory cuts.

Fukasawa et al. (2006) described a BCP for solving the SP model of the CVRP strengthened by the valid inequalities introduced by Lysgaard et al. (2004). The lower bound is computed by a column-and-cut generation method that uses $q$-routes (see Christofides et al. 1981a) instead of feasible CVRP routes. Because this method may not be competitive with the BC of Lysgaard et al. (2004), they combined these two methods. Thus, the resulting algorithm of Fukasawa et al. (2006) decides at the root node to use either the BC of Lysgaard et al. (2004) or the new BCP. The computational results show that 26 out of 74 CVRP instances are solved using the BC algorithm, and the remaining 48 instances are solved by the new BCP algorithm.

Baldacci et al. (2008b) improved the method of Mingozzi et al. (1994) and derived a new exact method for the CVRP based on the SP model that on average outperforms all other exact methods on the main CVRP instances from the literature. They proposed an additive bounding procedure that combines two dual ascent heuristics, called $H^1$ and $H^2$, to derive a near optimal dual solution that is used by a column-and-cut generation algorithm, called $H^3$, to initialize the master problem. $H^3$ attempts to close the integrality gap by adding in a cutting plane fashion both generalized capacity and clique constraints. The final dual solution achieved is then used to generate a reduced SP problem containing only the routes whose reduced cost is smaller than the gap between an upper bound and the lower bound obtained. The resulting reduced problem is then solved by an integer programming (IP) solver.

The key components of this method are: (i) the dual ascent scheme used by the bounding procedures $H^1$ and $H^2$, (ii) the use of a state-space relaxation (see Christofides et al. 1981b) to extend the route set with a relaxation of feasible routes that is easier to compute, and (iii) the use of bounding functions to reduce the state space graph computed by dynamic programming when solving the pricing problem and generating the final SP model.

Recently, Baldacci et al. (2009d) further improved the method of Baldacci et al. (2008b) using new ideas introduced by Baldacci et al. (2009c) for solving the VRPTW (see Sect. 3 for further details). In particular, they used a new route relaxation, called $ng$-route, that strongly improves other relaxations of feasible routes proposed in the literature for the CVRP and increases the efficiency of the pricing algorithms. They improved procedure $H^3$ using Subset-Row inequalities (see Jepsen et al. 2008) and a novel strategy for solving the pricing subproblem. Finally, a new fathoming criterion based on the dual solution achieved by $H^2$ is used to speed up the solution of the pricing subproblems and reduce the size of the final SP model.

The method described above provides a general solution framework that can be specialized to solve a number of VRP variants. In this section, we review this method focusing on its key components.

## 2.1 SP model of the CVRP with additional cuts

Let $\mathscr{R}$ be the index set of all feasible routes, and let $a_{i\ell}$ be a binary coefficient that is equal to 1 if vertex $i \in V'$ belongs to route $\ell \in \mathscr{R}$ and takes value 0 otherwise (note that $a_{0\ell} = 1, \forall \ell \in \mathscr{R}$). In the following, $R_\ell = \{i \in V' : a_{i\ell} = 1\}$ indicates the subset of vertices visited by route $\ell \in \mathscr{R}$. Each route $\ell \in \mathscr{R}$ has an associated cost $c_\ell$.

Let $x_\ell$ be a (0–1) binary variable equal to 1 if and only if route $\ell \in \mathscr{R}$ belongs to the optimal solution. The SP formulation of the CVRP is as follows:

$$(F) \quad z(F) = \min \sum_{\ell \in \mathscr{R}} c_\ell x_\ell \tag{1}$$

$$\text{s.t.} \sum_{\ell \in \mathscr{R}} a_{i\ell} x_\ell = 1, \quad \forall i \in V, \tag{2}$$

$$\sum_{\ell \in \mathscr{R}} x_\ell \le m, \tag{3}$$

$$x_\ell \in \{0, 1\}, \quad \forall \ell \in \mathscr{R}. \tag{4}$$

Constraints (2) specify that each customer $i \in V$ must be covered by one route, and constraint (3) requires that at most $m$ routes are selected.

The optimal solution cost $z(F)$ of the LP-relaxation of $F$ (called $LF$) provides a tight lower bound on the CVRP. Moreover, the lower bound can be strengthened by adding to $LF$ valid inequalities in a cutting plane fashion.

For a subset $S \subseteq V$, let $q(S)$ be the total demand of the customers in $S$, and let $\mathscr{S} = \{S : S \subseteq V, |S| \ge 2\}$. The following inequalities, called *strengthened capacity constraints*, are added to $LF$:

$$\sum_{\ell \in \mathscr{R}(S)} x_\ell \ge \lceil q(S)/Q \rceil, \quad \forall S \in \mathscr{S}, \tag{5}$$

where $\mathscr{R}(S)$ represents the index subset of routes visiting at least one customer of set $S \in \mathscr{S}$. Relaxation $LF$ can be further improved by adding any valid inequality for the SP problem. Let $H = (\mathscr{R}, \mathscr{E})$ be the *conflict graph* where each vertex corresponds to a route and the edge set $\mathscr{E}$ contains every pair $\{\ell, \ell'\}$, $\forall \ell, \ell' \in \mathscr{R}, \ell < \ell'$, such that $R_\ell \cap R_{\ell'} \ne \{0\}$. Let $\mathscr{C}$ be the set of all cliques of $H$. The following *clique inequalities* are added to $LF$:

$$\sum_{\ell \in C} x_\ell \le 1, \quad \forall C \in \mathscr{C}. \tag{6}$$

Let $\overline{LF}$ be the problem obtained by adding to $LF$ inequalities (5) and (6).

Let $\mathbf{u} = (u_0, u_1, \ldots, u_n)$ be a vector of dual variables, where $u_i, i \in V$ and $u_0$ are associated with constraints (2) and (3), respectively. Moreover, let $v_S$, $S \in \mathscr{S}$ and $g_C$, $C \in \mathscr{C}$, be the dual variables of constraints (5) and (6), respectively. The dual $\overline{DF}$ of $\overline{LF}$ is as follows:

$$(\overline{DF}) \quad z(\overline{DF}) = \max \sum_{i\in V} u_i + mu_0 + \sum_{S\in\mathscr{S}} \lceil q(S)/Q \rceil v_S + \sum_{C\in\mathscr{C}} g_C \tag{7}$$

$$\text{s.t.} \quad \sum_{i\in V'} a_{i\ell} u_i + \sum_{S\in\mathscr{S}} b_{S\ell} v_S + \sum_{C\in\mathscr{C}_\ell} g_C \le c_\ell, \quad \forall \ell \in \mathscr{R}, \tag{8}$$

$$u_i \in \mathbb{R}, \quad \forall i \in V, \tag{9}$$

$$u_0 \le 0, \tag{10}$$

$$v_S \ge 0, \quad \forall S \in \mathscr{S}, \tag{11}$$

$$g_C \le 0, \quad \forall C \in \mathscr{C}, \tag{12}$$

where $\mathscr{C}_\ell = \{C \in \mathscr{C} : \ell \in C\}$, and $b_{S\ell} \in \{0, 1\}$ equal to 1, $\forall \ell \in \mathscr{R}$, if $R_\ell \cap S \ne \emptyset$.

## 2.2 An exact method for the CVRP

The core of the exact method is a bounding procedure that uses in sequence three dual heuristics, $H^1$, $H^2$ and $H^3$, to obtain a near optimal $\overline{DF}$ solution $(\mathbf{u}', \mathbf{v}', \mathbf{g}')$ of cost $z'$ without generating all routes and constraints (5) and (6).

The exact method performs the following two steps.

1. Define the reduced problem $\hat{F}$ resulting from $F$ as follows:
   (a) replace the route set $\mathscr{R}$ with the largest subset $\hat{\mathscr{R}} \subseteq \mathscr{R}$ such that $c'_\ell < z(UB) - z', \forall \ell \in \hat{\mathscr{R}}$, where $c'_\ell$ is the reduced cost of route $\ell$ with respect to $(\mathbf{u}', \mathbf{v}', \mathbf{g}')$ and $z(UB)$ is a valid upper bound to $z(F)$;
   (b) add all constraints (5) and (6) saturated by the final $\overline{LF}$ solution (i.e., all constraints whose associated slacks are not in the basis).
2. Solve problem $\hat{F}$ using a general purpose IP solver.

The effectiveness of the method relies on the quality of the dual solution $(\mathbf{u}', \mathbf{v}', \mathbf{g}')$ achieved as the size of subset $\hat{\mathscr{R}}$ depends on the gap $z(UB) - z'$.

## 2.3 Bounding procedures $H^1$, $H^2$ and $H^3$

The procedure used to compute a near-optimal solution of problem $\overline{DF}$ is an additive bounding method that computes a lower bound on the $CVRP$ by combining three bounding methods, called $H^1$, $H^2$ and $H^3$. The three methods are used in sequence and do not require the a priori generation of the entire route set $\mathscr{R}$.

The first two procedures $H^1$ and $H^2$ ignore clique inequalities (6) and compute lower bounds $LB1$ and $LB2$ as the cost of two near-optimal dual solutions $(\mathbf{u}^1, \mathbf{v}^1, \mathbf{g}^1)$ and $(\mathbf{u}^2, \mathbf{v}^2, \mathbf{g}^2)$, where $\mathbf{g}^1 = \mathbf{g}^2 = \mathbf{0}$, respectively. Both $H^1$ and $H^2$ are based on the following theorem.

**Theorem 1** *Associate penalties* $\lambda_i \in \mathbb{R}, i \in V$, *with constraints* (2), $\lambda_0 \le 0$ *with constraint* (3), *and* $\sigma_S \ge 0$, $S \in \mathscr{S}$, *with constraints* (5). *Define*

$$b_i = q_i \min_{\ell\in\mathscr{R}_i} \left\{ \frac{c_\ell - \lambda(R_\ell) - \sigma(R_\ell)}{\sum_{i\in V} a_{i\ell} q_i} \right\}, \quad \forall i \in V, \tag{13}$$

*where $\mathscr{R}_i \subseteq \mathscr{R}$ is the index subset of the routes visiting customer $i \in V$, $\lambda(R_\ell) = \sum_{i \in V} a_{i\ell}\lambda_i$ and $\sigma(R_\ell) = \sum_{S \in \mathscr{S}} b_{S\ell}\sigma_S$.*

*A feasible solution* $(\mathbf{u}, \mathbf{v}, \mathbf{g})$ *of problem* $\overline{DF}$ *of cost* $z(\overline{DF}(\boldsymbol{\lambda}, \boldsymbol{\sigma}))$ *is given by setting* $\mathbf{g} = \mathbf{0}$ *and computing* $\mathbf{u}$ *and* $\mathbf{v}$ *according to the following expressions:*

$$u_i = b_i + \lambda_i, \quad \forall i \in V, \quad u_0 = \lambda_0 \quad \textit{and} \quad v_S = \sigma_S, \quad S \in \mathscr{S}. \tag{14}$$

A lower bound on the CVRP is then given by $\max_{\boldsymbol{\lambda},\boldsymbol{\sigma}}\{z(\overline{DF}(\boldsymbol{\lambda}, \boldsymbol{\sigma}))\}$ and can be computed using subgradient optimization. Let $\ell(i) \in \mathscr{R}$ be the index of the route producing $b_i$ in expression (13, and let $\rho_{ji}$ be the number of times that customer $j \in V$ is visited by route $\ell(i)$. It can be shown that a valid subgradient of the function $z(\overline{DF}(\boldsymbol{\lambda}, \boldsymbol{\sigma}))$, at point $(\boldsymbol{\lambda}, \boldsymbol{\sigma})$, is given by the vectors $\boldsymbol{\theta} = (\theta_0, \theta_1, \dots, \theta_n)$ and $\boldsymbol{\delta}$ computed as follows:

$$\theta_j = 1 - \sum_{i \in V} \rho_{ji} q_i / q(R_{\ell(i)}), \quad \forall j \in V, \quad \theta_0 = m - \sum_{i \in V} q_i / q(R_{\ell(i)}), \tag{15}$$

and

$$\delta_S = \lceil q(S)/Q \rceil - \sum_{j \in V:\, \ell(j) \in \mathscr{R}(S)} \sum_{i \in V} \rho_{ij} q_i / q(R_{\ell(j)}), \quad \forall S \in \mathscr{S}. \tag{16}$$

Even for moderate size CVRPs, expression (13) cannot be solved directly because the size of the set $\mathscr{R}$ is exponential. $H^1$ and $H^2$ use the same column generation algorithm, called $CG$, to solve equations (13). $CG$ differs from standard column generation methods as it uses subgradient optimization to compute $\max_{\boldsymbol{\lambda},\boldsymbol{\sigma}}\{z(\overline{DF}(\boldsymbol{\lambda}, \boldsymbol{\sigma}))\}$.

### *2.3.1 Algorithm $CG$*

The initial master problem of $CG$ is initialized by generating a subset of routes $\overline{\mathscr{R}} \subseteq \mathscr{R}$ containing the $\Delta^{\min}$-routes of minimum reduced cost with respect to an initial $\overline{DF}$ solution, where $\Delta^{\min}$ is a parameter defined a priori. The set $\mathscr{S}$ is initialized by setting $\mathscr{S} = \overline{\mathscr{S}}$, where $\overline{\mathscr{S}}$ is a set of capacity constraints generated a priori. As clique inequalities (6) are ignored, $CG$ sets $\mathscr{C} = \emptyset$.

$CG$ initializes $LCG = 0$ and executes an a priori defined number of macro iterations. At each macro iteration, the following two steps are performed:

1. *Solve the master problem*. The master is obtained from $\overline{LF}$ by replacing $\mathscr{R}$ with $\overline{\mathscr{R}}$ and $\mathscr{S}$ with $\overline{\mathscr{S}}$. A near-optimal solution $(\overline{\mathbf{u}}, \overline{\mathbf{v}})$ of cost $\overline{z}$ of the master problem is obtained by an iterative method that performs $Maxt2$ iterations. At each iteration the dual solution $(\overline{\mathbf{u}}, \overline{\mathbf{v}})$ is computed with respect to the current $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ using expressions (13) and (14), where $\mathscr{R}$ and $\mathscr{S}$ are replaced with $\overline{\mathscr{R}}$ and $\overline{\mathscr{S}}$, respectively.
   The initial values of $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are set equal to the best values achieved at the previous macro iteration. Penalties $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are then modified using the subgradient method after computing a subgradient $(\boldsymbol{\theta}, \boldsymbol{\delta})$ according to expressions (15) and (16).

2. *Check if* $(\bar{\mathbf{u}}, \bar{\mathbf{v}})$ *is a feasible* $\overline{DF}$ *solution*. Generate the largest subset $\mathscr{N}$ of routes having minimum reduced cost with respect to the dual master solution $(\bar{\mathbf{u}}, \bar{\mathbf{v}})$ and such that $|\mathscr{N}| \leq \Delta^a$ ($\Delta^a$ is a parameter defined a priori). If $\mathscr{N} = \emptyset$ and $\bar{z}$ is greater than $LCG$, then update $LCG = \bar{z}$, $\mathbf{u}^* = \bar{\mathbf{u}}$, $\mathbf{v}^* = \bar{\mathbf{v}}$, $\boldsymbol{\lambda}^* = \boldsymbol{\lambda}$ and $\boldsymbol{\sigma}^* = \boldsymbol{\sigma}$; otherwise, update $\overline{\mathscr{R}} = \overline{\mathscr{R}} \cup \mathscr{N}$.

At termination, $(\mathbf{u}^*, \mathbf{v}^*)$ is a $\overline{DF}$ solution of cost $LCG = z(\overline{DF}(\boldsymbol{\lambda}^*, \boldsymbol{\sigma}^*))$.

Algorithm $CG$ is faster than standard simplex-based methods as it avoids their typical degeneracy.

The exact method uses an additive bounding procedure that combines different versions of procedure $CG$, called $H^1$ and $H^2$.

### 2.3.2 Procedure $H^1$

$H^1$ corresponds to $CG$ where the route set $\mathscr{R}$ is replaced with the set $\overline{\mathscr{R}} \supseteq \mathscr{R}$ of $q$-routes (see Christofides et al. 1981a). A $q$-route is a not-necessarily elementary circuit of $G$ passing through the depot and such that the total demand of the customers visited is equal to $q$. Let $\Phi(q, i)$ be the cost of the least cost $q$-route passing through customer $i$. $H^1$ is initialized by setting $\boldsymbol{\lambda} = \mathbf{0}, \boldsymbol{\sigma} = \mathbf{0}$, and by generating for each customer $i$ the $q$-route of cost $\Phi(\hat{q}, i)$ such that $\Phi(\hat{q}, i)/\hat{q} = \min_{q_i \leq q \leq Q}[\Phi(q, i)/q]$. These $q$-routes form the initial route set $\overline{\mathscr{R}}$.

At each iteration, $H^1$ generates for each $i \in V$ the $q$-route passing through $i$ having the most negative reduced cost. Note that Theorem 1 remains valid if we set $b_i = \Phi(\hat{q}, i)/\hat{q}$ in expression (13). Let $(\mathbf{u}^1, \mathbf{v}^1)$ be the final $\overline{DF}$ solution of cost $LB1$ obtained by $H^1$ using penalties $(\boldsymbol{\lambda}^1, \boldsymbol{\sigma}^1)$.

### 2.3.3 Procedure $H^2$

$H^2$ is executed after $H^1$ and is based on elementary routes. $H^2$ starts by setting $\boldsymbol{\lambda} = \boldsymbol{\lambda}^1$ and $\boldsymbol{\sigma} = \boldsymbol{\sigma}^1$, and by using the dual solution $(\mathbf{u}^1, \mathbf{v}^1)$ obtained by $H^1$ to initialize the master route subset $\overline{\mathscr{R}}$. Both sets $\overline{\mathscr{R}}$ and $\mathscr{N}$ are generated by procedure GENROUTE described in Sect. 2.4.

$H^2$ computes a near optimal $\overline{DF}$ solution $(\mathbf{u}^2, \mathbf{v}^2)$ of cost $LB2$.

### 2.3.4 Procedure $H^3$

$H^3$ is a column-and-cut generation procedure based on the simplex method that solves $\overline{LF}$. The initial master route set $\overline{\mathscr{R}}$ of $H^3$ is generated using GENROUTE with respect to the $\overline{DF}$ solution $(\mathbf{u}^2, \mathbf{v}^2, \mathbf{g}^2)$, where $\mathbf{g}^2 = \mathbf{0}$.

In procedure $H^3$, capacity constraints (5) are heuristically separated using the package CVRPSEP of Lysgaard (2003), and clique inequalities are separated using the CLIQUER 1.1 package of Niskanen and Östergård (2003). At each iteration, the pricing problem is solved using procedure GENROUTE.

We denote by $(\mathbf{u}^3, \mathbf{v}^3, \mathbf{g}^3)$ the $\overline{DF}$ solution of cost $LB3$ obtained by $H^3$.

## 2.4 Route generation algorithm GENROUTE

GENROUTE is used by $H^2$, $H^3$ and the exact algorithm to generate feasible CVRP routes. Given a $\overline{DF}$ solution $(\hat{\mathbf{u}}, \hat{\mathbf{v}}, \hat{\mathbf{g}})$ of cost $\hat{z}(\overline{DF})$ and two user defined parameters $\Delta$ and $\gamma$, it provides the largest subset $\mathscr{B} \subseteq \mathscr{R}$ such that:

$$\left.\begin{array}{l} \max_{\ell\in\mathscr{B}}\{\hat{c}_\ell\} \leq \min_{\ell\in\mathscr{R}\setminus\mathscr{B}}\{\hat{c}_\ell\}, \\ |\mathscr{B}| \leq \Delta, \\ \max_{\ell\in\mathscr{B}}\{\hat{c}_\ell\} < \gamma, \end{array}\right\} \tag{17}$$

where $\hat{c}_\ell$ is the reduced cost of route $\ell$ with respect to $(\hat{\mathbf{u}}, \hat{\mathbf{v}}, \hat{\mathbf{g}})$ and $\Delta$ and $\gamma$ are defined according to the type of subset $\mathscr{B}$ that must be generated. By setting $\Delta = \Delta^a$ and $\gamma = 0$, GENROUTE produces the set $\mathscr{N}$ required at each iteration by $CG$ and $H^3$. Setting $\Delta = \infty$ and $\gamma = z(UB) - LB3$ GENROUTE produces the set $\hat{\mathscr{R}}$ required by the exact method.

GENROUTE is a two-phase procedure based on the following observation.

Let $\mathscr{P}_i$ be the set of all simple paths of minimum cost from the depot to vertex $i \in V$ and such that $q(P) \leq Q/2+q_i$, $\forall P \in \mathscr{P}_i$, where $q(P) = \sum_{i\in V(P)} q_i$ and $V(P) \subseteq V$ is the set of vertices visited by path $P$. Every route visiting customer $i$ can be obtained combining a pair $P, \overline{P} \in \mathscr{P}_i$ that are internally disjoint (i.e., $V(P) \cap V(\overline{P}) = \{0, i\}$) and such that $q(P) + q(\overline{P}) \leq Q/2 + q_i$.

Define the modified edge costs $d'_{ij}$ with respect to the dual vector $(\hat{\mathbf{u}}, \hat{\mathbf{v}})$ as $d'_{ij} = d_{ij} - \frac{1}{2}\hat{u}_i - \frac{1}{2}\hat{u}_j - \frac{1}{2}\sum_{S\in\overline{\mathscr{S}}(i,j)} \hat{v}_S$, $\forall\{i, j\} \in E$, where $\overline{\mathscr{S}}(i, j) = \{S \in \overline{\mathscr{S}} : \{i, j\} \in \delta(S)\}$ and $\delta(S)$ is the *cutset* defined by $S$.

Let $c'_\ell$ be the reduced cost of route $\ell$ with respect to $(\hat{\mathbf{u}}, \hat{\mathbf{v}})$ (i.e., ignoring $\hat{\mathbf{g}}$). We have $c'_\ell = \sum_{\{i,j\}\in E(R_\ell)} d'_{ij}$, where $E(R_\ell)$ is the set of edges traversed by route $\ell$. As $\hat{g} \leq 0$, it is easy to observe that $c'_\ell \leq \hat{c}_\ell$, $\forall \ell \in \mathscr{R}$.

The two phases of GENROUTE are described below.

*Phase 1* Using the modified edge costs $d'_{ij}$, let $c'(P)$ be the reduced cost of path $P$, and let $LB(P)$ be a lower bound on the cost $c'_\ell$ of any route $\ell \in \mathscr{R}$ that contains path $P$. Lower bound $LB(P)$ can be computed using the $q$-path relaxation described by Christofides et al. (1981a) as follows.

Let $F(q, i)$ be the cost of the least cost $q$-path ending at vertex $i$ of load less than or equal to $q$. Thus, we have $LB(P) = c'(P) + F(Q - q(P) + q_i, e(P))$, where $e(P)$ is the ending vertex of path $P$.

Phase 1 is a Dijkstra-like algorithm that generates the set $\mathscr{P}$ of paths, where each $P \in \mathscr{P}$ satisfies the following conditions: $LB(P) \leq \gamma$, $q(P) \leq Q/2 + q_{e(P)}$, and $P$ is not *dominated*, i.e., $c'(P) \leq c'(P')$, for each path $P'$ such that $V(P) = V(P')$ and $e(P') = e(P)$.

*Phase 2* This phase starts by setting $\mathscr{B} = \emptyset$ and iteratively adds to $\mathscr{B}$ the route $\ell$ of minimum reduced cost $c'_\ell$ with respect to $(\hat{\mathbf{u}}, \hat{\mathbf{v}})$ and such that its reduced cost with respect to $(\hat{\mathbf{u}}, \hat{\mathbf{v}}, \hat{\mathbf{g}})$ is less than $\gamma$. Phase 2 dynamically generates a set $\mathscr{T}$ containing a subset of all possible pairs $(P, \overline{P}) \in \mathscr{P}$ such that $e(P) = e(\overline{P})$. At each iteration, it

**Table 1** Summary results for the CVRP

| | *np* | Fukasawa et al. (2006) | | | | | | Baldacci et al. (2008b) | | | | Baldacci et al. (2009d) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | nopt | nopt$_1$ | nopt$_2$ | %LB | $t_{LB}$ | $t_{TOT}$ | nopt | %LB | $t_{LB}$ | $t_{TOT}$ | nopt | %LB | $t_{LB}$ | $t_{TOT}$ |
| A | 22 | 22 | 20 | 2 | 99.2 | 183 | 1,961 | 22 | 99.8 | 41 | 118 | 22 | 99.7 | 16 | 22 |
| B | 20 | 20 | 6 | 14 | 99.5 | 84 | 4,763 | 20 | 99.8 | 119 | 417 | 20 | 99.9 | 49 | 66 |
| E-M | 11 | 9 | 7 | 2 | 98.8 | 957 | 126,987 | 8 | 99.4 | 185 | 1,025 | 9 | 99.5 | 220 | 249 |
| P | 24 | 24 | 16 | 8 | 99.1 | 136 | 2,892 | 22 | 99.7 | 34 | 186 | 24 | 99.8 | 53 | 54 |
| All | 77 | 75 | 49 | 26 | | | | 72 | | | | 75 | | | |
| Avg. | | | | | 99.2 | 234 | 18,009 | | 99.7 | 77 | 322 | | 99.7 | 61 | 71 |

*nopt*$_1$ number of instances solved using the BC algorithm; *nopt*$_2$ number of instances solved using the BCP algorithm

selects from $\mathscr{T}$ the path pair $(P, \overline{P})$ of minimum cost $c'(P) + c'(\overline{P})$ and, if these two paths correspond to a feasible route $R_\ell$ not dominated by any other route in $\mathscr{B}$, then it adds $R_\ell$ to $\mathscr{B}$. The selected pair $(P, \overline{P})$ is then used to expand $\mathscr{T}$.

GENROUTE terminates if either $c'(P) + c'(\overline{P}) \geq \max_{\ell \in \mathscr{B}}\{\hat{c}_\ell\}$ and $|\mathscr{B}| = \Delta$ or $c'(P) + c'(\overline{P}) \geq \gamma$. Notice that if $|\mathscr{B}| = \Delta$, then a route $\ell$ of cost $\hat{c}_\ell$ enters $\mathscr{B}$ only if $\hat{c}_\ell < \max_{r \in \mathscr{B}}\{\hat{c}_r\}$.

## 2.5 Computational results for the CVRP

In this section, we report a computational comparison of the results obtained by Lysgaard et al. (2004), Fukasawa et al. (2006) and Baldacci et al. (2008b, 2009d) on six classes of CVRP instances from the literature, called A, B, E, M and P. These instances are available at http://branchandcut.org/VRP/data. Classes A, B and P were proposed by Augerat (1995). Instance class M was proposed by Christofides et al. (1979), and class E was produced by Christofides and Eilon (1969).

The algorithm of Lysgaard et al. (2004) was run on an Intel Celeron at 700 MHz whereas the algorithms of Baldacci et al. (2008b) and Fukasawa et al. (2006) used Pentium 4 processors running at 2.6 and 2.4 Ghz, respectively. The algorithm of Baldacci et al. (2009d) was run on an Intel Core 2 Duo P8400 at 2.26 GHz. According to the SPEC benchmarks, the machine used by Baldacci et al. (2008b) is about 10% faster and at least five times faster than those used by Fukasawa et al. (2006) and by Lysgaard et al. (2004), respectively, while the machine used by Baldacci et al. (2009d) is about twice as fast as the machine used by Baldacci et al. (2008b).

Table 1 reports a summary of the computational results obtained by the exact methods of Baldacci et al. (2008b, 2009d) and Fukasawa et al. (2006). Column *np* of this table reports the total number of instances in the corresponding class.

For each method and for each class, the table reports the following data: number of instances solved to optimality ($nopt$), average percentage ratio of the lower bound with respect to the optimal solution value ($\%LB$), average time in seconds for computing the lower bound ($t_{LB}$) and average total time in seconds ($t_{TOT}$) both computed over all the instances solved to optimality.

**Table 2** Results on difficult CVRP instances

| Name | $z^*$ | Lysgaard et al. (2004) | | | Fukasawa et al. (2006) | | | Baldacci et al. (2008b) | | | Baldacci et al. (2009d) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | %LB | $t_{LB}$ | $t_{TOT}$ | %LB | $t_{LB}$ | $t_{TOT}$ | %LB | $t_{LB}$ | $t_{TOT}$ | %LB | $t_{LB}$ | $t_{TOT}$ |
| A-n54-k7 | 1,167 | 97.3 | 30 | 7,246 | 98.9 | 125 | 1,409 | 99.5 | 43 | 86 | 99.8 | 10 | 10 |
| A-n64-k9 | 1,401 | 96.5 | 132 | *tl* | 98.9 | 265 | 11,254 | 99.5 | 32 | 120 | 99.6 | 18 | 22 |
| A-n80-k10 | 1,763 | 97.0 | 201 | *tl* | 99.5 | 1,120 | 6,464 | 99.6 | 117 | 194 | 99.7 | 96 | 105 |
| B-n50-k8 | 1,312 | 97.6 | 26 | *tl* | 98.7 | 97 | 2,845 | 99.7 | 640 | 662 | 99.8 | 169 | 204 |
| B-n66-k9 | 1,316 | 98.7 | 80 | *tl* | 99.4 | 145 | 1,778 | 99.5 | 216 | 227 | 99.9 | 66 | 72 |
| B-n68-k9 | 1,272 | 98.9 | 65 | *tl* | 99.3 | 260 | 87,436 | 99.5 | 254 | 6,168 | 99.7 | 465 | 702 |
| E-n51-k5 | 521 | 99.6 | 24 | 59 | 99.5 | 51 | 65 | 100.0 | 13 | 13 | 100.0 | 3 | 3 |
| E-n76-k7 | 682 | 97.7 | 72 | 118,683 | 98.2 | 264 | 46,520 | 99.0 | 146 | 3,371 | 99.8 | 123 | 125 |
| E-n76-k8 | 735 | 97.7 | 136 | *tl* | 98.8 | 277 | 22,891 | 99.3 | 104 | 873 | 99.9 | 60 | 61 |
| E-n76-k10 | 830 | 96.4 | 158 | *tl* | 98.5 | 354 | 80,722 | 99.5 | 60 | 174 | 99.5 | 18 | 27 |
| E-n76-k14 | 1,021 | 95.0 | 181 | *tl* | 98.6 | 224 | 48,637 | 99.6 | 17 | 45 | 99.4 | 8 | 12 |
| E-n101-k8 | 815 | 98.5 | 222 | *tl* | 98.8 | 1,068 | 801,963 | 99.0 | 250 | – | 100.0 | 395 | 395 |
| E-n101-k14 | 1,067 | 96.2 | 555 | *tl* | 98.8 | 658 | 116,284 | 99.7 | 154 | 1,230 | 99.6 | 110 | 352 |
| M-n101-k10 | 820 | 100.0 | 33 | 33 | 100.0 | 119 | 119 | 100.0 | 47 | 47 | 100.0 | 24 | 24 |
| M-n121-k7 | 1,034 | 98.4 | 979 | *tl* | 99.7 | 5,594 | 25,678 | 99.8 | 944 | 2,448 | 99.9 | 1,238 | 1,240 |
| P-n50-k8 | 631 | 95.4 | 28 | *tl* | 97.7 | 102 | 9,272 | 99.0 | 12 | 596 | 99.1 | 5 | 7 |
| P-n55-k10 | 694 | 95.4 | 53 | *tl* | 98.2 | 107 | 9,076 | 99.3 | 16 | 66 | 99.3 | 5 | 6 |
| P-n70-k10 | 827 | 96.2 | 90 | *tl* | 98.5 | 292 | 24,039 | 99.4 | 35 | 774 | 99.5 | 11 | 17 |
| P-n76-k5 | 627 | 98.5 | 92 | 10,970 | 98.4 | 273 | 14,546 | 98.8 | 122 | – | 100.0 | 224 | 224 |
| P-n101-k4 | 681 | 99.6 | 127 | 281 | 99.6 | 1,055 | 1,253 | 99.4 | 371 | – | 100.0 | 884 | 884 |

For the exact method of Fukasawa et al. (2006), column $nopt_1$ reports the number of instances solved to optimality using the BC of Lysgaard et al. (2004), and column $nopt_2$ reports the number of instances solved by the BCP of Fukasawa et al. (2006).

The last two lines of Table 1 report the total number of instances solved by each method and the averages of lower bounds and computing times over all classes. The method of Baldacci et al. (2008b) was not able to solve to optimality three instances solved by Fukasawa et al. (2006) and by Baldacci et al. (2009d).

Table 1 indicates that the lower bounds of Baldacci et al. (2008b, 2009d) are on average superior to the lower bounds of Fukasawa et al. (2006) in all classes of instances considered. Notice that the method of Fukasawa et al. (2006) solved to optimality 75 instances, but 26 of them were solved using the BC of Lysgaard et al. (2004). Taking the different computers used into account, Table 1 indicates that the method of Baldacci et al. (2009d) is on average faster than the method of Fukasawa et al. (2006).

Table 2 reports a detailed comparison of the methods of Lysgaard et al. (2004), Fukasawa et al. (2006) and Baldacci et al. (2008b, 2009d) on difficult CVRP instances. Columns $\%LB$, $t_{LB}$ and $t_{TOT}$ of this table have the same meaning as in Table 1. Column $z^*$ reports the cost of the optimal solution of each instance. For the exact method of Lysgaard et al. (2004), "$tl$" indicates that the time limit has been reached, and for the exact method of Baldacci et al. (2008b), "–" denotes that the memory limit has been reached.

## 3 The vehicle routing problem with time windows

The vehicle routing problem with time windows (VRPTW) is a CVRP variant where each customer $i \in V$ has to be visited within a given time window $[e_i, l_i]$. If a vehicle arrives at $i$ before $e_i$, the service starts exactly at $e_i$. The VRPTW is defined on a complete digraph $G = (V', A)$ where the vertex set $V'$ is defined as in Sect. 2 for the CVRP and $A$ is the arc set. With each arc $(i, j) \in A$ are associated a travel cost $d_{ij}$ and a travel time $t_{ij}$, the latter including the time for servicing customer $i$. In the following, we denote with $\Gamma_i$ and $\Gamma_i^{-1}$ the sets of successor and predecessor vertices of $i \in V'$, respectively.

It is assumed that all vehicles leave the depot at time $e_0$ and return to it before $l_0$. Each vehicle can perform at most one route that visits a set of customers within their time windows and has cost equal to the sum of the travel costs of the arcs traversed. The objective is to design at most $m$ routes of minimum total cost visiting each customer exactly once.

A number of heuristic algorithms can be found in the literature, but few exact methods have been proposed. Reviews of heuristic and exact algorithms can be found in Toth and Vigo (2002) and Braysy and Gendreau (2005a,b).

Desrochers et al. (1992) were the first to propose an exact branch-and-price algorithm. They succeeded in solving instances involving up to 100 customers. Jepsen et al. (2008) described a BCP method where the pricing problem is solved by dynamic programming. For tightening the LP-relaxation of the SP formulation of the VRPTW, they introduced the Subset-Row (SR) inequalities that significantly improved the lower bounds. Desaulniers et al. (2008) proposed a BCP method that solves the pricing problem by either tabu search heuristics or dynamic programming. The computational results show that their method is faster than the BCP of Jepsen et al. (2008). Moreover, they solved 51 out of 56 100-customer Solomon instances (5 of which were previously unsolved).

Recently, Baldacci et al. (2009c) proposed a new algorithm for the VRPTW that uses the same SP like formulation of the CVRP. However, the VRPTW cannot be solved directly using the general method described in Sect. 2 as it needs an ad-hoc algorithm for generating VRPTW routes. Moreover, the method described for the CVRP can be ineffective in solving instances with loose time window and capacity constraints.

To overcome these drawbacks, they introduced several new ideas. They proposed a novel state-space relaxation that improves the lower bounds computed by procedure $H^1$ and increases the efficiency of the pricing algorithm in procedure $H^2$. Both procedures $H^1$ and $H^2$ are based on relaxation $LF$. In procedure $H^3$, they used Subset-Row inequalities (introduced by Jepsen et al. 2008) instead of clique inequalities and introduced a new method for fathoming states within the pricing algorithm using the near-optimal dual solution achieved by $H^2$. Finally, they introduced a better performing pricing scheme for procedure $H^3$. The algorithm of Baldacci et al. (2009c) was able to solve to optimality all but one Solomon benchmark instances outperforming all other exact algorithms published so far.

In this section, we briefly describe the new features of this algorithm.

## 3.1 Procedure $H^1$ based on $ng$-route relaxation

The procedure $H^1$, which was introduced for the CVRP and is based on $q$-route relaxation, produces weak lower bounds if applied to the VRPTW - particularly, for instances with loose time window and vehicle capacity constraints.

To obtain stronger lower bounds, Baldacci et al. (2009c) introduced a new relaxation of feasible routes called $ng$-routes. Procedure $H^1$ is similar to that described in Sect. 2.3.2 for the CVRP but uses $ng$-routes instead of $q$-routes.

Let us define an $ng$-path $(NG, t, i)$ as a not-necessarily simple path starting from $i$ at time $t$, ending at the depot at time $l_0$, and such that: (i) it visits customers within their time window, and (ii) it visits at least once each customer $j \in NG \subseteq V$. An $ng$-route $(NG, t, i)$ starting from $i$ is obtained by adding arc $(0, i)$ to an $ng$-path $(NG, t, i)$.

Let $\bar{\mathbf{u}}$ be the vector of dual variables associated with constraints (2) and (3) of problem $LF$. We denote by $f(NG, t, i)$ the cost of the least cost $ng$-path $(NG, t, i)$ with respect to modified arc costs $d'_{ij} = d_{ij} - \frac{1}{2}\bar{u}_i - \frac{1}{2}\bar{u}_j$.

Define for each customer $i$ the neighbor set $N_i \subseteq V$ as follows. $N_i = N_i^- \cup \{i\}$, where $N_i^- \subseteq \Gamma_i$ satisfies: $|N_i^-| \leq \Delta^{ng}$ ($\Delta^{ng}$ defined a priori) and $\max_{j \in N_i^-}\{d'_{ij}\} \leq \min_{j \in \Gamma_i \setminus N_i^-}\{d'_{ij}\}$. Functions $f(NG, t, i), \forall NG \subseteq N_i, \forall e_i \leq t \leq l_i, \forall i \in V$, and the corresponding $ng$-paths used to compute the subsets $\hat{\mathscr{R}}$ and $\mathscr{N}$ of $H^1$ are computed through the following dynamic programming recursion:

$$f(NG, t, i) = \begin{cases} \min\limits_{\substack{j \in \Gamma_i \\ t + t_{ij} \leq t' \leq l_j \\ NG' \cap N_i = NG \setminus \{i\}}} \left\{ f(NG', t', j) + d'_{ij} \right\} & \text{if } t = l_i \\ \min\limits_{\substack{j \in \Gamma_i \\ NG' \cap N_i = NG \setminus \{i\}}} \left\{ f(NG', t + t_{ij}, j) + d'_{ij} \right\} & \text{if } e_i \leq t < l_i \end{cases}$$

## 3.2 Procedure $H^2$

Procedure $H^2$ uses feasible VRPTW routes. It uses a forward dynamic programming algorithm for generating both route sets $\overline{\mathscr{R}}$ and $\mathscr{N}$. At each iteration, $H^2$ computes functions $f(NG, t, i), \forall NG \subseteq N_i, e_0 \leq t \leq l_0, \forall i \in V$, with respect to the current dual master solution $\bar{\mathbf{u}}$ and uses them to reduce the state-space graph when generating VRPTW routes.

Each state corresponds to a forward path $P$ starting at 0, ending in $i \in V$ at time $e_i \leq t \leq l_i$ and visiting a set of vertices $V(P) \subseteq V$. Thus, a lower bound on the reduced cost of the least cost route containing $P$ can be computed as:

$$LB(P) = c'(P) + \min_{\substack{NG \cap V(P) = \{i\} \\ t' \geq t}} \{f(NG, t', i)\}, \tag{18}$$

where $c'(P)$ is the cost of path $P$ with respect to costs $d'_{ij} = d_{ij} - \frac{1}{2}\bar{u}_i - \frac{1}{2}\bar{u}_j$. If $LB(P) > 0$, then path $P$ can be fathomed as it cannot produce any route of $\mathscr{N}$.

$H^2$ computes a $DF$ solution $\mathbf{u}^2$ of cost $LB2$.

## 3.3 Procedure $H^3$

Procedure $H^3$ solves formulation $LF$ tightened by additional cuts. Unlike procedure $H^3$ described in Sect. 2.3, this procedure does not use capacity constraints (5) and replaces clique inequalities (6) with the Subset-Row (SR) inequalities introduced by Jepsen et al. (2008).

Only a subset of SR inequalities are used. Let $\mathscr{C} = \{C \subseteq V : |C| = 3\}$ be the set of all customer triplets, and let $\mathscr{R}(C) \subseteq \mathscr{R}$ be the subset of routes servicing at least two customers in $C$, i.e., $\mathscr{R}(C) = \{\ell \in \mathscr{R} : |R_\ell \cap C| \geq 2\}$. The following inequalities are valid for $LF$:

$$\sum_{\ell \in \mathscr{R}(C)} x_\ell \leq 1, \quad \forall C \in \mathscr{C}. \tag{19}$$

In this section, we denote by $\overline{LF}$ the problem obtained by adding to $LF$ SR inequalities (19), and by $\overline{DF}$ its dual.

Inequalities (19) can be separated in polynomial time by complete enumeration, and their duals can be easily taken into account in the pricing problem.

### *3.3.1 Pricing scheme of $H^3$*

$H^3$ solves $\overline{DF}$ by column-and-cut generation and uses an adaptation of procedure GENROUTE (see Sect. 2.4) for generating feasible VRPTW routes that is briefly described in Sect. 3.3.2. With respect to procedure $H^3$ of Sect. 2.3, the pricing scheme of Baldacci et al. (2009c) is somewhat more involved.

To speed up the solution process, before starting procedure $H^3$ an attempt is made to generate the set $\hat{\mathscr{R}}$ containing all routes having reduced cost smaller than the gap $z(UB) - LB2$, where $z(UB)$ is an upper bound on the VRPTW. If this attempt succeeds, $H^3$ turns into a cutting plane method as it is guaranteed that $\hat{\mathscr{R}}$ contains an optimal integer solution, and GENROUTE is not called again within $H^3$. Nevertheless, if $|\hat{\mathscr{R}}|$ is large it may be computationally convenient to initialize the master of $H^3$ with a limited subset $\overline{\mathscr{R}} \subseteq \hat{\mathscr{R}}$ and simply price out columns in $\hat{\mathscr{R}} \setminus \overline{\mathscr{R}}$ at each iteration by inspection.

On the contrary, if the method is unable to generate all routes, two cases are considered. (a) GENROUTE could generate all simple paths but was unable to generate the whole set $\hat{\mathscr{R}}$ by combining them: in this case, it suffices to combine these paths to generate new routes at each iteration of $H^3$. (b) GENROUTE could not compute all the simple paths required to generate $|\hat{\mathscr{R}}|$: in this case, GENROUTE must be called at each iteration of $H^3$. However, it is possible to avoid using GENROUTE at each iteration by combining previously generated paths for a certain number of iterations, until no more routes can be generated.

As a matter of fact, even for solving the most difficult VRPTW instances, the method of Baldacci et al. (2009c) never required to call GENROUTE more than 3 times after starting $H^3$.

### *3.3.2* GENROUTE *and pricing by different dual solutions*

The algorithm GENROUTE described for the CVRP was modified to take into account the time window constraints and to improve the pricing scheme. The first modification concerns the fact that any VRPTW route $R$ can be decomposed, for each $i \in V(R)$, into a *forward* path $P$ starting from depot 0 and ending at vertex $e(P) = i$ at time $t(P)$, and a *backward* path $\overline{P}$ starting at $e(\overline{P}) = i$ at time $t(\overline{P})$ and ending at depot 0 before time $l_0$. Thus, in the first phase GENROUTE generates the two sets $\mathscr{P}$ and $\mathscr{P}^{-1}$ of forward and backward paths, and in the second phase it combines every pair $(P, \overline{P})$, $P \in \mathscr{P}$, $\overline{P} \in \mathscr{P}^{-1}$, to derive the route set $\mathscr{B}$.

GENROUTE fathoms paths and routes according to the following observations. Let $\bar{\mathbf{u}}$ be the current dual master solution at a given iteration of $H^3$, and let $\ell \in \mathscr{R}$ be a route having reduced cost $\bar{c}_\ell < 0$ with respect to $\bar{\mathbf{u}}$. If $\mathbf{u}^2$ is a feasible dual solution of cost $LB2$, route $\ell$ can still be fathomed if it has reduced cost greater than $z(UB) - LB2$ with respect to $\mathbf{u}^2$.

Similarly, the size of path sets $\mathscr{P}$ and $\mathscr{P}^{-1}$ can be reduced using both $\bar{\mathbf{u}}$ and $\mathbf{u}^2$ as follows. Let $LB'(P)$ and $LB^2(P)$ be the lower bounds associated with path $P \in \mathscr{P}$ using expression (18) with respect to the two bounding functions given by $\bar{\mathbf{u}}$ and $\mathbf{u}^2$, respectively. Path $P$ cannot be contained in any route of set $\mathscr{B}$ if either $LB'(P) > 0$ or $LB^2(P) \geq z(UB) - LB2$. Similar observations are used to reduce the size of $\mathscr{P}^{-1}$.

## 3.4 Computational results for the VRPTW

This section reports a summary of the computational results of the method proposed by Baldacci et al. (2009c). The algorithm was tested on the VRPTW benchmark instances created by Solomon (see Solomon 1987) and available at http://web.cba.neu.edu/~msolomon/problems.htm. The whole benchmark set is made up of 168 instances divided into three groups having 25, 50 and 100 customers. Each group is divided into six classes (C1, RC1, R1, C2, RC2 and R2) differing by the geographical distribution of the customers and/or the width of the time windows.

The algorithm of Baldacci et al. (2009c) (BMR in the following) was coded in Fortran 77, compiled with the Intel Fortran 2008, and run on an Intel Xeon E5310 Workstation (1.6 GHz with 8 Gb of RAM). CPLEX 11.0 was the LP and IP solver used in $H^3$. The method is compared with Jepsen et al. (2008) (JPPS in the following) and Desaulniers et al. (2008) (DHL in the following). According to SPEC tables (http://www.spec.org), the machine used by BMR is 5–10% faster than that used by DHL and 35–40% faster than that of JPPS.

Table 3 reports the computational time for a selected set of instances. The columns report the name of the instance (Name), the number of customers ($|V|$), and the computational time in seconds for the methods compared ($t_{TOT}$ of BMR, JPPS, DHL). In this table, an entry "n.a." means data not available, and a dash "–" means that the problem was not solved to optimality.

Tables 4 and 5 report a summary of the computational results obtained on all instances with 50 and 100 customers. The columns of this table report the class of instances considered (Class), the number of instances of the class ($np$), and for each

**Table 3** Computational results for the most difficult VRPTW Solomon instances

| Name | $\|V\|$ | $t_{TOT}$ | | | Name | $\|V\|$ | $t_{TOT}$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | | BMR | JPPS | DHL | | | BMR | JPPS | DHL |
| C204 | 50 | 48 | – | n.a. | RC204 | 100 | 1,657 | – | – |
| RC204 | 50 | 61 | – | n.a. | RC207 | 100 | 5,439 | – | 91,405 |
| R204 | 50 | 72 | – | n.a. | RC208 | 100 | 877 | – | – |
| R207 | 50 | 248 | 34,406 | n.a. | R202 | 100 | 828 | 8,282 | 1,663 |
| R208 | 50 | 550 | – | n.a. | R203 | 100 | 1,060 | 54,187 | 641 |
| R210 | 50 | 53 | 18,545 | n.a. | R204 | 100 | 362,661 | – | – |
| R211 | 50 | 32 | 10,543 | n.a. | R205 | 100 | 2,035 | – | 6,904 |
| RC104 | 100 | 979 | 65,806 | 11,773 | R206 | 100 | 7,996 | – | 60,608 |
| RC106 | 100 | 173 | 15,891 | 3,916 | R207 | 100 | 2,690 | – | 11,228 |
| R104 | 100 | 400 | 32,343 | 3,103 | R209 | 100 | 13,921 | 78,560 | 22,514 |
| R111 | 100 | 213 | 83,931 | 5,738 | R210 | 100 | 32,528 | – | 400,904 |
| R112 | 100 | 645 | 202,803 | 16,073 | R211 | 100 | 18,162 | – | – |
| C204 | 100 | 319 | – | 16,416 | | | | | |

**Table 4** Summary for 50-customer VRPTW instances

| Class | $np$ | $nopt$ | | $t_{TOT}$ | |
|---|---|---|---|---|---|
| | | BMR | JPPS | BMR | JPPS |
| C1 | 9 | 9 | 9 | 5 | 175 |
| RC1 | 8 | 8 | 8 | 14 | 41 |
| R1 | 12 | 12 | 12 | 33 | 139 |
| All | 29 | 29 | 29 | 19 | 123 |
| C2 | 8 | 8 | 7 | 17 | 79 |
| RC2 | 8 | 8 | 7 | 24 | 268 |
| R2 | 11 | 11 | 9 | 95 | 7,086 |
| All | 27 | 27 | 23 | 51 | 2,879 |
| Solved by JPPS | | 23 | 23 | 28 | 2,879 |

method: the number of problems solved to optimality ($nopt$), and the average computing time ($t_{TOT}$) over all instances solved to optimality. Lines labeled “All” report for each method the total number of problems solved and the average computing time over the instances solved. Lines “Solved by JPPS” and “Solved by DHL” report for each method the number of instances solved among those solved by JPPS and DHL, respectively, and the corresponding average computational time.

The tables show that BMR solved all the 56 50-customer instances. JPPS could not solve 4 of them. On average, for instances with wide time windows (classes C2, RC2 and R2) BMR is 60–65 times faster than JPPS. All compared methods solved 100-customer instances with tight time windows (classes C1, RC1 and R1), but BMR is significantly faster than the others. Instances with 100 customers and wide time windows are clearly the most difficult ones. BMR succeeded in solving all but one

**Table 5** Summary for 100-customer VRPTW instances

| Class | *np* | *nopt* | | | $t_{TOT}$ | | |
|---|---|---|---|---|---|---|---|
| | | BMR | JPPS | DHL | BMR | JPPS | DHL |
| C1 | 9 | 9 | 9 | 9 | 15 | 468 | 18 |
| RC1 | 8 | 8 | 8 | 8 | 230 | 11,004 | 2,150 |
| R1 | 12 | 12 | 12 | 12 | 162 | 27,412 | 2,327 |
| All | 29 | 29 | 29 | 29 | 135 | 14,524 | 1,562 |
| C2 | 8 | 8 | 7 | 8 | 71 | 2,795 | 2,093 |
| RC2 | 8 | 8 | 5 | 6 | 1,035 | 3,204 | 15,394 |
| R2 | 11 | 10 | 4 | 8 | 44,191 | 35,292 | 63,068 |
| All | 27 | 26 | 16 | 22 | 17,337 | 11,047 | 27,893 |
| Solved by JPPS | | 16 | 16 | 16 | 1,025 | 11,047 | 1,637 |
| Solved by DHL | | 21 | | 21 | 3,064 | | 27,893 |

of them. It solved for the first time ever instances RC204, RC208, R204 and R211. The only instance that remains open is R208. BMR outperformed JPPS and DHL: on instances solved by JPPS it is 5–6 times faster than JPPS itself, and on instances solved by DHL it is 7–8 times faster than DHL.

## 4 The pickup and delivery problem with time windows

The PDPTW is a generalization of the VRPTW where the set of vertices $V$ is partitioned as $V = P \cup D \cup \{0\}$. Vertex 0 represents the depot, and the subsets $P = \{1, \dots, n\}$ and $D = \{n + 1, \dots, 2n\}$ represent $n$ pickup and delivery vertices. Each pickup $i \in P$ is associated with a delivery $n + i \in D$, and each pair $(i, n + i)$ defines a transportation request $i, i = 1, \dots, n$, requiring that a load $q_i$ is delivered from pickup $i$ to delivery $n + i$. Hereafter, the set $P$ will also be used to represent the set of the $n$ transportation requests. With each vertex $i \in V$, there is associated a time window $[e_i, l_i]$. A load $q_i > 0$ is associated with each pickup $i \in P$, whereas $q_i = -q_{i-n}, \forall i \in D$ (we assume $q_0 = 0$).

The problem requires to design the routes of at most $m$ vehicles of capacity $Q$ that are based at the depot to satisfy the $n$ transportation requests. Let $S(R) = V(R) \cap P$ be the subset of requests visited by a route $R$. Each route $R$ must be simple, must visit vertices according to their time windows, and must satisfy the following two constraints: (i) $R$ visits the delivery $n + i$ after having visited the pickup $i \in S(R)$, and (ii) the total load of vertices visited after leaving each vertex $i \in S(R)$ must be smaller than or equal to $Q$.

In the literature, two different PDPTW objective functions have been considered. The first objective, hereafter called o1, is to minimize the sum of the route costs. The second objective, hereafter called o2, involves a fixed cost $W$ that is associated with each vehicle and is to minimize the sum of fixed costs and route costs. Notice that if $W$ is very large (i.e., larger than the route cost of any feasible solution), then objective o2 minimizes first the number of vehicles used and second the sum of the route costs.

In the following, we denote by PDPTW-o1 and PDPTW-o2 the variants of the PDPTW where the objective is to minimize the objective functions o1 and o2, respectively.

Recent surveys on the PDPTW can be found in Cordeau et al. (2008) and Parragh et al. (2008). Exact algorithms for the PDPTW have been proposed by Dumas et al. (1991), Savelsbergh and Sol (1998), Lu and Dessouky (2004), Ropke et al. (2007) and recently by Ropke and Cordeau (2009) and Baldacci et al. (2009a). The method of Ropke and Cordeau is a BCP algorithm that uses different classes of valid inequalities to improve the lower bound and two different methods for solving the pricing problem. The method of Baldacci et al. (2009a) relies on the general framework described in Sect. 2.

In this section, we highlight the key differences of the algorithm described in Baldacci et al. (2009a) for solving the PDPTW with respect to the general solution method described in Sect. 2.

The PDPTW-o1 can be formulated mathematically using the same SP like formulation $F$ of the CVRP as all the constraints that are specific to the PDPTW can be implicitly imposed in the definition of the route set $\mathscr{R}$. The same formulation can also be used to model the PDPTW-o2 as any PDPTW-o2 instance can be transformed into a PDPTW-o1 instance by adding the fixed cost $W$ to each outgoing arc from the depot. Notice that, since each pickup must be visited by the same route of the corresponding delivery, it is sufficient to impose the SP constraints (2) for each pickup vertex only.

The method described in Sect. 2 cannot be used directly to solve the PDPTW as it needs an ad-hoc algorithm for generating PDPTW routes. Moreover, it can be ineffective for those PDPTW-o2 instances where the fixed cost $W$ is large because in this case relaxation $LF$ provides a weak lower bound.

In the following, we describe the forward dynamic programming algorithm of Baldacci et al. (2009a) for generating PDPTW routes, called GENR, that uses problem-specific bounding functions to reduce the size of the state-space. Moreover, we describe an adaptation of the exact method of Sect. 2 for the PDPTW-o1 and a new exact algorithm for the PDPTW-o2.

## 4.1 Solving the PDPTW-o1

Relaxation $LF$ provides a tight lower bound on the PDPTW-o1, so the problem can be solved using an adaptation of the exact method of Sect. 2 where the bounding procedures $H^1$, $H^2$ and $H^3$, are tailored to the PDPTW-o1.

$H^1$ uses a relaxation of PDPTW routes called $(t, i)$-*routes*. A $(t, i)$-route is a not necessarily simple circuit that starts and ends at the depot, satisfies time window constraints of the vertices visited, and such that $i \in D$ is the last vertex visited before the depot. $(t, i)$-routes are computed as described in Sect. 4.3.

$H^2$ uses feasible PDPTW routes that are generated using algorithm GENR described in Sect. 4.3.

$H^3$ is a column-and-cut generation method that solves a relaxation $\overline{LF}$ obtained by adding SR inequalities to $LF$ (see Sect. 3.3). In this section, we denote by $\overline{DF}$ the dual of $\overline{LF}$.

The exact method for the PDPTW-o1 combines the exact method of Sect. 2.2 with a BCP algorithm where the lower bound at each node is computed by solving problem $\overline{LF}$. The method is obtained by substituting Step 2 of the exact method of Sect. 2.2 with the following:

2. We have two cases:
   (a) if $|\hat{\mathscr{R}}| < \Delta^{\max}$: solve problem $\hat{F}$ using a general purpose IP solver;
   (b) if $|\hat{\mathscr{R}}| = \Delta^{\max}$: solve problem $F$ using the BCP algorithm described below;

where $\Delta^{\max}$ is an a priori defined parameter.

#### *4.1.1 BCP algorithm for PDPTW-o1*

The BCP algorithm solves problem $\overline{LF}$ using procedure $H^3$ at each node of the enumerative tree. Given an $\overline{LF}$ solution $\bar{\mathbf{x}}$ involving a set of routes $\overline{\mathscr{R}}$, let $\bar{\omega}_{ij} = \sum_{\ell \in \overline{\mathscr{R}}_{ij}} \overline{x}_{\ell}, \forall (i, j) \in A$, where $\overline{\mathscr{R}}_{ij} \subseteq \overline{\mathscr{R}}$ is the subset of routes traversing arc $(i, j)$. When $H^3$ terminates with a fractional solution $\bar{\mathbf{x}}$, the algorithm selects an arc $(i, j)$ having the value $\bar{\omega}_{ij}$ closest to 0.5 (in case of ties, the arc having the smallest modified arc cost $d_{ij} - u^3_j$ is chosen). Then, it creates two branches imposing the disjunction $\overline{\omega}_{ij} = 0 \vee \overline{\omega}_{ij} = 1$. The nodes are processed according to a best bound node selection rule.

The algorithm uses a pool of routes $\mathscr{R}^0 \subseteq \mathscr{R}$ and a pool of cuts $\mathscr{C}^0$, that are initialized by setting $\mathscr{R}^0 = \hat{\mathscr{R}}$ and $\mathscr{C}^0 = \hat{\mathscr{C}}$. At each node of the enumerative tree, the pools $\mathscr{R}^0$ and $\mathscr{C}^0$ are used to initialize the initial master problem of $H^3$. The route subset $\overline{\mathscr{R}}$ is obtained by extracting from $\mathscr{R}^0$ the largest set of routes satisfying the branching conditions. The subset of triplets $\overline{\mathscr{C}}$ is initialized by setting $\overline{\mathscr{C}} = \mathscr{C}^0$. The route pool $\mathscr{R}^0$ and the cut pool $\mathscr{C}^0$ are enlarged at each node during the execution of $H^3$ by setting $\mathscr{R}^0 = \mathscr{R}^0 \cup \mathscr{N}$ and $\mathscr{C}^0 = \mathscr{C}^0 \cup \mathscr{C}'$, where $\mathscr{N}$ and $\mathscr{C}'$ are the route subset and the cut subset generated at each iteration of $H^3$, respectively.

### 4.2 Solving the PDPTW-o2

The exact method of Sect. 2 can be ineffective for those PDPTW-o2 instances where the fixed cost $W$ is large. The exact algorithm of Baldacci et al. (2009a) decomposes a PDPTW-o2 instance into $m_{UB} - m_{LB} + 1$ PDPTW-o1 instances obtained from the original PDPTW-o2 by setting $W = 0$, where $m_{UB}$ and $m_{LB}$ represent an upper bound and a lower bound on the number of vehicles required, respectively.

With each PDPTW-o1 instance involving $\kappa$ vehicles, $\kappa = m_{LB}, \ldots, m_{UB}$, is associated a problem $F(\kappa)$ derived from $F$ by setting $m = \kappa$ and $W = 0$. Each problem $F(\kappa)$ is solved using the exact method described in Sect. 4.1.

The values of $m_{LB}$ and $m_{UB}$ are computed as follows.

- Let $z(UB)$ be an upper bound on the original PDPTW-o2 instance. The value $m_{UB}$ can be computed as $m_{UB} = \lfloor z(UB)/W \rfloor$. However, a better estimate of $m_{UB}$ is obtained by computing a valid lower bound $r_{LB}$ on problem $F(\lfloor z(UB)/W \rfloor)$ and setting $m_{UB} = \lfloor \frac{z(UB) - r_{LB}}{W} \rfloor$.

- Lower bound $m_{LB}$ can be set equal to the smallest integer $\kappa$ such that problem $F(\kappa)$ has a feasible solution and can be computed as follows. Let $LB(\kappa)$ be a lower bound on $F(\kappa)$ (we assume $LB(\kappa) = \infty$ in case $LF(\kappa)$ has no feasible solution). Lower bound $LB(\kappa)$ can be computed by solving the dual of $LF(\kappa)$ using procedure $H^3$. Then $m_{LB}$ can be set equal to the smallest integer $\kappa$ such that $LB(\kappa) < \infty$.

After computing $m_{UB}$ and $m_{LB}$, the root node lower bound $z(LB)$ on the PDPTW-o2 is computed as $z(LB) = \min\limits_{m_{LB} \leq \kappa \leq m_{UB}} \{W \times \kappa + LB(\kappa)\}$. The exact method iteratively solves problems $F(\kappa)$, $\kappa = m_{LB}, \ldots, m_{UB}$, but terminates prematurely at iteration $\kappa \leq m_{UB}$ if $W \times \kappa + z^*(F(\kappa)) \leq W \times \bar{\kappa} + LB(\bar{\kappa})$, $\bar{\kappa} = \kappa + 1, \ldots, m_{UB}$, where $z^* F(\kappa)$ is the optimal solution cost of $F(\kappa)$.

## 4.3 Procedure GENR

GENR is a dynamic programming algorithm for generating PDPTW routes as required by $H^1$, $H^2$ and the exact algorithms. Given a $\overline{DF}$ solution $(\hat{\mathbf{u}}, \hat{\mathbf{g}})$ and two user defined parameters $\Delta$ and $\gamma$, GENR computes the largest subset $\mathscr{B} \subseteq \mathscr{R}$ satisfying conditions (17). GENR dynamically generates a state-space graph where each vertex corresponds to a PDPTW *forward path L*.

A PDPTW forward path $L = (0, i_1, \ldots, i_h)$ is a simple path in $G$ starting from the depot, visiting a subset of vertices $V(L)$, and ending at a vertex $e(L) \in V$ at time $\tau(L)$. Let $P(L)$ and $D(L)$ be the subsets of pickups and deliveries visited by $L$, that is $D(L) = V(L) \cap D$ and $P(L) = V(L) \cap P$. $L$ must satisfy the following conditions: $(i)$ each vertex $i \in V(L)$ is visited according to its time window; $(ii)$ $i - n \in P(L), \forall i \in D(L)$, and each delivery $i \in D(L)$ is visited after the corresponding pickup $i - n$; $(iii)$ the total load of the vehicle $\sum_{j=1,\ldots,k} q_{i_k}$ after having visited customer $i_k$ is smaller than or equal to the vehicle capacity $Q$. We denote by $\widetilde{P}(L) \subseteq P(L)$ the set of requests whose deliveries are not visited by $L$.

GENR uses different bounding functions called $f(t, i)$, $g(t, i)$ and $f(\widetilde{D}, t, i)$ to compute a lower bound $LB(L)$ on the reduced cost of any feasible route that contains a forward path $L$. The state-space graph is reduced by removing all paths $L$ such that $LB(L) > \gamma$.

## 4.4 Bounding functions $f(t, i)$ and $g(t, i)$

Let a backward $(t, i)$-path be a not-necessarily simple path in $G$ that starts from vertex $i \in V$ at time $e_i \leq t \leq l_i$, ends at the depot 0, and arrives at each vertex visited at a time within its time window.

Function $f(t, i)$ is defined as the cost of the least cost backward $(t, i)$-path starting from vertex $i$ at time $t$, with respect to the modified arc costs $d'_{ij} = d_{ij} - \hat{u}_j$ given a $\overline{DF}$ solution $(\hat{\mathbf{u}}, \hat{\mathbf{g}})$. Let $\pi(t, i)$ be the vertex just after $i$ in the $(t, i)$-path of cost $f(t, i)$. Define $g(t, i)$ as the cost of the least cost backward $(t, i)$-path starting from vertex $i$ at time $t$ and such that the vertex after $i$ is different from $\pi(t, i)$. Functions $f(t, i)$ and $g(t, i)$ can be computed in pseudo-polynomial time (i.e., in time $O(T\ n^2)$ where

$T = l_0 - e_0$) using the dynamic programming procedure described in Christofides et al. (1981c,b). This procedure imposes the restriction that a backward $(t, i)$-path does not contain loops of two consecutive vertices.

Let $L$ be a forward path, and let $c'(L)$ be its cost with respect to costs $d'_{ij}$. As the duals $\hat{\mathbf{g}}$ of inequalities (19) are non-positive, a valid lower bound $LB_1(L)$ on the cost of any route containing $L$ is computed as follows:

$$LB_1(L) = c'(L) + \min_{\tau(L) \le t \le l_{e(L)}} \begin{cases} f(t, e(L)), & \text{if } \pi(t,i) \notin V(L), \\ g(t, e(L)), & \text{otherwise.} \end{cases} \tag{20}$$

## 4.5 Bounding functions $f(\widetilde{D}, t, i)$

Let $\widetilde{D} \subseteq P$ be a subset of requests. A backward $(\widetilde{D}, t, i)$-path is a not necessarily simple path that starts from vertex $i \in V$ at time $e_i \le t \le l_i$, visits the deliveries of the requests in $\widetilde{D}$, but does not visit the corresponding pickups. Moreover, a backward $(\widetilde{D}, t, i)$-path must satisfy the following conditions: (a) it must visit vertices according to their time window; (b) if it visits a pickup $i$, then it must also visit the corresponding delivery $n + i$ after $i$; (c) the total load of vertices visited after leaving each vertex must be greater than or equal to $-Q$ and less than or equal to $Q$.

Let $f(\widetilde{D}, t, i)$ be the cost of the least cost $(\widetilde{D}, t, i)$-path, with respect to the modified arc costs $d'_{ij}$. Functions $f(\widetilde{D}, t, i)$ can be computed using a backward dynamic programming procedure where each vertex of the state-space graph corresponds to a backward $(\widetilde{D}, t, i)$-path. Even if $(\widetilde{D}, t, i)$-paths can be non-simple, it is easy to impose that after visiting a delivery vertex the same delivery is not visited again before the corresponding pickup. In fact, it suffices to forbid that a $(\widetilde{D}, t, i)$-path visits any delivery whose corresponding request is in $\widetilde{D}$. This condition is trivially satisfied by any simple path and forbids loops of two consecutive vertices in non-simple $(\widetilde{D}, t, i)$-paths.

Consider a forward path $L$ of cost $c'(L)$ with respect to costs $d'_{ij}$, and let $L^-$ be the sub-path that is obtained from $L$ by removing the last vertex $e(L)$. A lower bound $LB_2(L)$ on the reduced cost of any route containing a forward path $L$ is given by:

$$LB_2(L) = c'(L) + \min_{\substack{\tau(L) \le t \le l_{e(L)} \\ \widetilde{D} = \widetilde{P}(L^-)}} \left\{ f\left(\widetilde{D}, t, e(L)\right) \right\}. \tag{21}$$

In computing functions $f(\widetilde{D}, t, i)$, $H^3$ ignores the duals $\hat{\mathbf{g}}$ of inequalities (19). As $\hat{\mathbf{g}} \le 0$, then $LB_2(L)$ is a valid lower bound with respect to any feasible $\overline{DF}$ solution $(\hat{\mathbf{u}}, \hat{\mathbf{g}})$.

## 4.6 Computational results for the PDPTW

In this section, we report a summary of the computational results obtained by the exact algorithm of Baldacci et al. (2009a).

The exact algorithm of Baldacci et al. (2009a) is compared with the BCP of Ropke and Cordeau (2009) on two sets of PDPTW instances from the literature called Class

**Table 6** Summary results for the PDPTW

| | $np$ | Ropke and Cordeau (2009) | | | Baldacci et al. (2009a) | | |
|---|---|---|---|---|---|---|---|
| | | $\%LB$ | nopt | $t_{TOT}$ | %LB | nopt | $t_{TOT}$ |
| Class 1 | | | | | | | |
| AA | 10 | – | 9 | 361.3 | 99.975 | 10 | 85.8 |
| BB | 10 | – | 9 | 291.2 | 99.989 | 9 | 57.4 |
| CC | 10 | – | 6 | 865.4 | 99.981 | 10 | 46.2 |
| DD | 10 | – | 6 | 771.9 | 99.970 | 10 | 37.1 |
| | 40 | – | 30 | 572.5 | 99.979 | 39 | 56.6 |
| Class 2 | | | | | | | |
| LC1 | 10 | 99.961 | 8 | 870.0 | 100.000 | 9 | 28.8 |
| LR1 | 10 | 99.964 | 5 | 312.2 | 100.000 | 7 | 44.2 |
| LRC1 | 10 | 99.925 | 4 | 402.9 | 100.000 | 7 | 93.6 |
| LL500 | 6 | 99.998 | 3 | 1,049.9 | 100.000 | 3 | 143.8 |
| | 36 | 99.962 | 20 | 658.7 | 100.000 | 26 | 77.6 |

1 and Class 2 instances, respectively. Class 1 contains PDPTW-o2 instances that were introduced by Ropke and Cordeau (2009) and are available at http://www.diku.dk/~sropke/. Class 2 contains PDPTW-o1 instances proposed by Li and Lim (2001) that are publicly available at http://www.top.sintef.no/. Both classes are divided into 4 subclasses.

Baldacci et al. (2009a) used an Intel Xeon E5310 Workstation clocked at 1.6 GHz with 8 Gb RAM running Windows Server 2003 × 64 Edition. The BCP of Ropke and Cordeau used an AMD Opteron 250 (2.4 GHz) running Linux (∼5–10% slower than the Xeon according to SPEC2000 benchmarks). A time limit of 10 h was imposed on the exact algorithm of Baldacci et al. (2009a), whereas a time limit of 2 h was used by the BCP of opke and Cordeau.

Table 6 reports a summary of the computational results obtained by the two methods for the two classes of instances. Table 6 shows the following columns: the number of instances ($np$); the average percentage ratio of the lower bound at the root node, computed over all instances solved by both algorithms ($\%LB$); the number of instances solved ($nopt$); and the average computing time in seconds, over all instances solved by both algorithms ($t_{TOT}$).

Table 7 reports more detailed results on the most difficult instances that could be solved to optimality by at least one of the two algorithms. Column $z^*$ of this table reports the value of the optimal solution, column $\%LB$ reports the percentage ratio of the lower bound at the root node and $t_{TOT}$ the total computing time.

For PDPTW-o2 instances, we do not report column $\%LB$ for the BCP algorithm of Ropke and Cordeau as lower bound at the root node of their BCP cannot be directly compared with lower bound $z(LB)$ of Baldacci et al. (2009a) (see Sect. 4.2).

The results on the two classes of instances considered indicate that both the lower bounds and the exact method of Baldacci et al. (2009a) are superior to those of Ropke

**Table 7** Results on difficult PDPTW instances

| Name | $z^*$ | Ropke and Cordeau (2009) | | Baldacci et al. (2009a) | |
|---|---|---|---|---|---|
| | | %LB | $t_{TOT}$ | %LB | $t_{TOT}$ |
| AA75 | 52,461.6 | – | *tl* | 99.981 | 6,442.4 |
| CC50 | 41,685.3 | – | 1,962.3 | 99.981 | 38.9 |
| CC55 | 41,836.3 | – | 2,729.2 | 99.970 | 185.2 |
| CC60 | 42,009.3 | – | *tl* | 99.943 | 2,128.2 |
| CC65 | 42,164.0 | – | *tl* | 99.929 | 7,111.2 |
| CC70 | 52,201.7 | – | *tl* | 81.137 | 5,565.7 |
| CC75 | 52,359.0 | – | *tl* | 99.978 | 259.6 |
| DD50 | 31,600.9 | – | 1,976.0 | 99.958 | 96.7 |
| DD55 | 31,743.3 | – | 1,178.5 | 99.971 | 36.5 |
| DD60 | 32,069.2 | – | *tl* | 99.685 | 13,048.3 |
| DD65 | 42,107.3 | – | *tl* | 99.935 | 25,929.1 |
| DD70 | 42,214.2 | – | *tl* | 99.943 | 20,737.5 |
| DD75 | 42,359.9 | – | *tl* | 99.937 | 34,718.6 |
| lc1_2_9 | 2,724.2 | 99.684 | 6,628.6 | 100.000 | 55.3 |
| lc1_2_10 | 2,741.6 | 99.756 | *tl* | 100.000 | 137.1 |
| lr1_2_3 | 3,486.8 | 99.923 | *tl* | 100.000 | 3,690.8 |
| lr1_2_6 | 3,763.0 | 100.000 | 1,041.6 | 100.000 | 180.9 |
| lr1_2_10 | 3,386.3 | 99.702 | *tl* | 100.000 | 1,376.7 |
| lrc1_2_2 | 3,292.4 | 99.830 | 1,053.3 | 100.000 | 322.3 |
| lrc1_2_7 | 3,317.7 | 99.412 | *tl* | 99.941 | 408.2 |
| lrc1_2_8 | 3,086.5 | 98.033 | *tl* | 99.733 | 1,562.7 |
| lrc1_2_9 | 3,053.8 | 98.091 | *tl* | 99.360 | 1,757.2 |
| lr1101 | 56,744.9 | 99.993 | 1,682.3 | 100.000 | 233.1 |

and Cordeau. In particular, the exact algorithm of Baldacci et al. (2009a) is on average 9 times faster on Class 1 instances and solves to optimality 9 problems previously unsolved. It also outperforms the BCP of Ropke and Cordeau on Class 2 instances, being on average 8 times faster and solving 6 problems of Class 2 previously unsolved.

## 5 The heterogenous vehicle routing problem

The heterogenous vehicle routing problem (HVRP) is a generalization of the CVRP where the vehicle fleet is composed of a set $M = \{1, \dots, m\}$ of $m$ different vehicle types. The HVRP can be described as follows.

For each type $k \in M$, $U_k$ vehicles are available at the depot, each having a *capacity* equal to $Q_k$. With each vehicle type is also associated a *fixed cost* $F_k$ modeling, e.g., rental or capital amortization costs. In addition, for each edge $\{i, j\} \in E$ and for each vehicle type $k \in M$, it is given a *routing cost* $d_{ij}^k$ that represents the cost for traversing edge $\{i, j\}$ with a vehicle of type $k$.

A route $R = (0, i_1, \dots, i_r, 0)$ performed by a vehicle of type $k$ is a simple cycle in $G$ passing through the depot and customers $\{i_1, \dots, i_r\} \subseteq V$, with $r \geq 1$, such that the total demand of the customers visited does not exceed the vehicle capacity $Q_k$ (i.e., $\sum_{h=1}^{r} q_{i_h} \leq Q_k$). The cost of a route is equal to the sum of the routing costs plus the fixed cost of the associated vehicle.

The HVRP consists of designing a set of feasible routes of minimum total cost such that each customer is visited by exactly one route and the number of routes performed by the vehicles of type $k$ is not greater than $U_k$, $k \in M$.

This model subsumes the following classes of VRP.

(1) The CVRP corresponds to the HVRP where $m = 1$, $Q_1 = Q$, $F_1 = 0$ and $U_1 = p$.

(2) The fleet size and mix CVRP with fixed vehicle costs, unlimited number of vehicles, and independent routing costs (FSMF). This problem is obtained by setting $U_k = n, \forall k \in M, d^r_{ij} = d^s_{ij}, \forall r, s \in M, r \neq s$.

(3) The fleet size and mix CVRP with fixed vehicle costs, unlimited number of vehicles, and vehicle dependent routing costs (FSMFD) is obtained by setting $U_k = n, \forall k \in M$.

(4) The heterogeneous CVRP with no fixed vehicle costs and vehicle dependent routing costs (HD) is obtained by setting $F_k = 0, \forall k \in M$.

(5) The fleet size and mix CVRP with no fixed vehicle costs, unlimited number of vehicles, and vehicle dependent routing costs (FSMD). This problem is obtained by setting $F_k = 0, \forall k \in M, U_k = n, \forall k \in M$.

(6) The Site-Dependent CVRP (SDVRP). In the SDVRP a customer $i \in V$ can only be serviced by a subset of vehicle types $M_i \subseteq M$. The routing costs are vehicle independent and are represented by a symmetric matrix $[d_{ij}]$. No fixed costs are associated with the vehicles.

Any SDVRP instance can be converted into an equivalent HD instance by setting for each vehicle type $k \in M$:

$$F_k = 0 \quad \text{and} \quad d^k_{ij} = \begin{cases} d_{ij}, & \text{if } k \in M_i \cap M_j \\ \infty, & \text{otherwise} \end{cases}, \quad \forall \{i, j\} \in E,$$

where $M_0 = M$.

(7) The MDVRP. This problem is an extension of the CVRP where a customer can be serviced by an unlimited fleet of identical vehicles of capacity $Q$ that are located at $p$ depots. Inter-depot routes are not allowed.

Let $[\hat{d}_{ij}]$ be a $(n + p) \times (n + p)$ symmetric cost matrix, where $\hat{d}_{n+k\ i}$ is the travel cost for going from depot $k = 1, \dots, p$ to customer $i \in V$. Any MDVRP instance can be converted into an equivalent HVRP instance involving $m = p$ different vehicle types and setting for each vehicle type $k \in M$:

$$Q_k = Q, \quad U_k = n, \quad F_k = 0 \quad \text{and} \quad d^k_{ij} = \begin{cases} \hat{d}_{n+k\ j}, & \text{if } i = 0, \\ \hat{d}_{ij}, & \text{otherwise}, \end{cases} \quad \forall \{i, j\} \in E.$$

All problems described above are $\mathcal{NP}$-hard as they generalize the CVRP. Their main characteristics are summarized in Table 8.

**Table 8** Characteristics of the different problems

| Problem | Vehicle fixed costs | Vehicle dependent routing costs | Heterogenous vehicle fleet | Limited fleet |
|---|---|---|---|---|
| HVRP | Yes | Yes | Yes | Yes |
| CVRP | No | No | No | Yes |
| FSMF | Yes | No | Yes | No |
| FSMFD | Yes | Yes | Yes | No |
| HD/SDVRP | No | Yes | Yes | Yes |
| FSMD | No | Yes | Yes | No |
| MDVRP | No | Yes | No | No |

Many different heuristics have been proposed in the literature for the HVRP and its variants. A recent survey of lower bounds for the HVRP and its variants can be found in Baldacci et al. (2008a). An exact method for the FSMF, FSMFD and FSMD variants was recently proposed by Pessoa et al. (2007). These authors presented an exact BCP method based on the one proposed by Fukasawa et al. (2006) for the CVRP. To our knowledge, only three exact algorithms have been proposed for the MDVRP. Laporte et al. (1984, 1988) have developed exact branch-and-bound algorithms, but these only work well on relatively small instances (see Crevier et al. 2007). Baldacci and Mingozzi (2009) present an exact algorithm for the HVRP that generalizes the bounding procedures and the exact method for the CVRP described in Sect. 2. They introduce new bounding methods that are particularly effective when the vehicle fixed cost contribution to the total cost is relevant. The exact algorithm proposed for the HVRP is able to solve all VRP shown in Table 8. The computational results show that the proposed lower bound is superior to the lower bounds presented in the literature. Moreover, the exact algorithm of Baldacci and Mingozzi (2009) outperforms the exact method of Pessoa et al. (2007) and can solve for the first time several test instances of all problem types considered.

In this section, we review the method of Baldacci and Mingozzi (2009).

## 5.1 Mathematical formulation of the HVRP

Let $\mathscr{R}^k$ be the index set of all feasible routes of vehicle type $k \in M$, and let $\mathscr{R} = \bigcup_{k\in M} \mathscr{R}^k$. With each route $\ell \in \mathscr{R}^k$ it is associated a routing cost $c^k_\ell$. Let $\mathscr{R}^k_i \subset \mathscr{R}^k$ be the index subset of the routes of a vehicle of type $k$ visiting customer $i \in V$. In the following, we use $R^k_\ell$ to indicate the subset of customers visited by route $\ell \in \mathscr{R}^k$.

Let $x^k_\ell$ be a (0–1) binary variable equal to 1 if and only if route $\ell \in \mathscr{R}^k$ is chosen in the solution.

$$(F) \quad z(F) = \min \sum_{k\in M} \sum_{\ell\in\mathscr{R}^k} (F_k + c^k_\ell)x^k_\ell \tag{22}$$

$$\text{s.t.} \sum_{k \in M} \sum_{\ell \in \mathscr{R}_i^k} x_\ell^k = 1, \quad \forall i \in V, \tag{23}$$

$$\sum_{\ell \in \mathscr{R}^k} x_\ell^k \leq U_k, \quad \forall k \in M, \tag{24}$$

$$x_\ell^k \in \{0, 1\}, \quad \forall \ell \in \mathscr{R}^k, \forall k \in M. \tag{25}$$

Constraints (23) specify that each customer $i \in V$ must be covered by exactly one route. Constraints (24) impose an upper bound on the number of vehicles of each type that can be used.

In the following, we describe the three relaxations of problem $F$ used by Baldacci and Mingozzi (2009) to derive different lower bounds as well as different methods for reducing the size of sets $\mathscr{R}^k$, $k \in M$, by eliminating those routes that cannot belong to any optimal HVRP solution.

## 5.2 Relaxation $LF$ and procedures $H^1$ and $H^2$

Let $LF$ be the LP-relaxation of problem $F$, and let $z(LF)$ be its optimal solution cost. We denote by $DF$ the dual of problem $LF$. Let $\mathbf{u} = (u_1, \ldots, u_n)$ and $\mathbf{v} = (v_1, \ldots, v_m)$ be the dual variable vectors associated with constraints (23) and (24), respectively.

A near-optimal $DF$ solution can be obtained by the following theorem, which is an extension to the HVRP of Theorem 1 introduced in Sect. 2.3.

**Theorem 2** *Associate penalties $\lambda_i \in \mathbb{R}, \forall i \in V$, with constraints* (23) *and penalties $\mu_k \leq 0, \forall k \in M$, with constraints* (24). *Define*

$$b_{ik} = q_i \min_{\ell \in \mathscr{R}_i^k} \left\{ \frac{c_\ell^k + F_k - \lambda\left(R_l^k\right) - \mu_k}{q\left(R_l^k\right)} \right\}, \quad \forall i \in V, \ \forall k \in M, \tag{26}$$

*where $\lambda(R_\ell^k) = \sum_{i \in R_\ell^k} \lambda_i$ and $q\left(R_l^k\right) = \sum_{i \in R_\ell^k} q_i$.*

*A feasible $DF$ solution* $(\mathbf{u}, \mathbf{v})$ *of cost $z(DF(\boldsymbol{\lambda}, \boldsymbol{\mu}))$ is given by setting*:

$$\left.\begin{array}{ll} u_i = \min_{k \in M}\{b_{ik}\} + \lambda_i, \ \forall i \in V, & (a) \\ v_k = \mu_k, \ \forall k \in M. & (b) \end{array}\right\} \tag{27}$$

Using Theorem 2, the bounding procedures $H^1$ and $H^2$ described in Sect. 2.3 for the CVRP can be extended to derive lower bounds $LB1$ and $LB2$, respectively.

## 5.3 Relaxation $RP$ and bounding procedures $DP^1$ and $DP^2$

A second relaxation of the HVRP corresponds to an integer problem, called $RP$, that can provide a better lower bound than $z(LF)$ for those HVRP s where the vehicle fixed cost contribution to the optimal cost is relevant or dominates the routing cost contribution.

$RP$ involves two types of integer variables: $\xi_{ik} \in \{0, 1\}, i \in V, k \in M$, and $y_k \in \mathbb{Z}^+, k \in M$. Variable $\xi_{ik}$ is equal to 1 if and only if customer $i \in V$ is serviced by a vehicle of type $k \in M$. Variable $y_k$ represents the number of vehicles of type $k$ used in the solution.

Let $\beta_{ik}$ be the *marginal routing cost* for servicing customer $i \in V$ with a vehicle of type $k \in M$. We assume that the values $\beta_{ik}, i \in V, k \in M$, satisfy the following inequalities:

$$\sum_{i\in R_\ell^k} \beta_{ik} \le c_\ell^k, \quad \forall \ell \in \mathscr{R}^k, \quad \forall k \in M. \tag{28}$$

It can be shown that the following integer problem $RP$ provides a valid lower bound on the HVRP for any solution $\beta_{ik}$ of inequalities (28).

$$(RP) \quad z(RP) = \min \sum_{k\in M}\sum_{i\in V} \beta_{ik}\xi_{ik} + \sum_{k\in M} F_k y_k \tag{29}$$

$$\text{s.t.} \sum_{k\in M}\sum_{i\in V} q_i\xi_{ik} = q(V), \tag{30}$$

$$\sum_{i\in V} q_i\xi_{ik} \le Q_k y_k, \quad \forall k \in M, \tag{31}$$

$$y_k \le U_k, \quad \forall k \in M, \tag{32}$$

$$\xi_{ik} \in \{0, 1\}, \quad \forall i \in V, \forall k \in M, \tag{33}$$

$$y_k \in \mathbb{Z}^+, \quad \forall k \in M. \tag{34}$$

Relaxation $RP$ is used by two bounding procedures, called $DP^1$ and $DP^2$, that correspond to two different methods for computing $\beta_{ik}$ satisfying inequalities (28). $DP^1$ uses $q$-route relaxation while $DP^2$ uses column generation. Both procedures are based on the dual ascent procedure $CG$ (see Sect. 2.3) and solve problem $RP$ by dynamic programming.

The lower bounds $LD1$ and $LD2$ correspond to the cost $z(RP)$ of the $RP$ solution achieved by $DP^1$ and $DP^2$, respectively.

### *5.3.1 Relaxation $\overline{LF}$ and procedure $H^3$*

A better relaxation than $LF$, called $\overline{LF}$, is obtained by adding to $LF$ a generalization to the HVRP of the *strengthened capacity constraints* and of the *clique inequalities* described in Sect. 2.3.4 for the CVRP.

Let $z(\overline{LF})$ be the optimal solution cost of $\overline{LF}$. Relaxation $\overline{LF}$ is solved by means of a standard column-and-cut generation method, called $CG$, that is described in Sect. 2.3. The initial master problem is generated by using either the dual solution $(\mathbf{u}^2, \mathbf{v}^2)$ given by $H^2$ or the marginal routing cost $\beta_{ik}^2$ obtained by $DP^2$ as described in previous section. The master problem is then solved by using a simplex algorithm where, at each iteration, a limited subset of *strengthened capacity constraints* and *clique inequalities* that are violated by the current fractional solution are added to the master. $LB3$ corresponds to the cost $z(\overline{LF})$ of the final $\overline{LF}$ solution achieved by $H^3$.

It is easy to observe that $LB3 \geq z(LF)$. No dominance relation exists between $LB3$ and $z(RP)$. Furthermore, $LD1$ and $LD2$ can be greater than $LB3$.

### *5.3.2 An exact method for solving the HVRP*

The exact algorithm for solving the HVRP generalizes the method described in Sect. 2.2 for the CVRP. The method consists of finding, by means of a general purpose IP solver, an optimal integer solution of a reduced problem $\hat{F}$ obtained from $F$ by replacing each set $\mathscr{R}^k, k \in M$, with a subset $\hat{\mathscr{R}}^k$, and adding two subsets $\hat{\mathscr{S}} \subset \mathscr{S}$ and $\hat{\mathscr{C}} \subset \mathscr{C}$ of *strengthened capacity constraints* and *clique inequalities*, respectively. The subsets $\hat{\mathscr{R}}^k, \forall k \in M$, are generated in such a way that any optimal $\hat{F}$ solution is also optimal for $F$.

The core of the algorithm is the bounding method that combines different bounding procedures based on the three relaxations described in the previous section.

#### *Bounding method*

1. Execute in sequence $H^1$ and $H^2$. Let $LB2$ be the cost of the $DF$ solution $(\mathbf{u}^2, \mathbf{v}^2)$ obtained by $H^2$. If $F_k = 0, \forall k \in M$, set $LD1 = 0, LD2 = 0$ and go to Step 3, otherwise go to Step 2.
2. Execute $DP^1$. If $LD1 \geq LB2$, execute $DP^2$ producing the marginal routing costs $\beta^2_{ik}$ and the lower bound $LD2$. If $LD1 < LB2$ set $LD2 = 0$.
3. Execute bounding procedure $H^3$. Compute the final lower bound $LB = \max\{LD2, LB3\}$.

#### *Generating the reduced problem $\hat{F}$*

The route subsets $\hat{\mathscr{R}}^k, k \in M$, are generated either using the dual solution of $\overline{LF}$ obtained by procedure $H^3$ or the marginal routing costs $\beta^2_{ik}$ obtained by $DP^2$. We have the following two cases:

(a) $LB = LB3$. We replace the route sets $\mathscr{R}^k, k \in M$, with the route subsets $\hat{\mathscr{R}}^k, k \in M$, containing all routes whose reduced costs with respect to the dual solution of $\overline{LF}$ achieved by $H^3$ is smaller than the gap $z(UB) - LB3$, where $z(UB)$ is a valid upper bound on the HVRP. We consider only those constraints generated by $H^3$ that have zero slack in the final $\overline{LF}$ solution.

(b) $LB = LD2$. We generate for each $k \in M$ the subset $\hat{\mathscr{R}}^k$ containing all routes satisfying the following inequalities:

$$c^k_\ell - \sum_{i \in R^k_\ell} \beta^2_{ik} < z(UB) - LD2, \quad \forall \ell \in \hat{\mathscr{R}}^k. \tag{35}$$

It is quite easy to show that any route $\ell \in \mathscr{R}^k \setminus \hat{\mathscr{R}}^k, k \in M$, cannot belong to an optimal HVRP solution of cost smaller than $z(UB)$.

**Table 9** Summary results for the HVRP

| Variant | $np$ | Lower bounds | | | | | Exact methods | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Pessoa et al. (2007) | | Choi and Tcha (2007) | Baldacci and Mingozzi (2009) | | Pessoa et al. (2007) | | Baldacci and Mingozzi (2009) | |
| | | $\%LB$ | $t_{LB}$ | $\%LB$ | %LB | $t_{LB}$ | nopt | $t_{TOT}$ | nopt | $t_{TOT}$ |
| HVRP | 12 | – | – | – | 99.6 | 224.8 | – | – | 10 | 259.9 |
| FSMF | 12 | 99.6 | 229.5 | 98.4 | 99.8 | 147.0 | 9 | 4,741.2 | 11 | 125.4 |
| FSMFD | 12 | 99.7 | 243.8 | 98.6 | 99.7 | 143.5 | 10 | 963.46 | 11 | 172.9 |
| HD | 8 | – | – | – | 99.2 | 128.9 | – | – | 7 | 564.8 |
| FSMD | 12 | 99.2 | 330.5 | 98.1 | 99.5 | 81.9 | 10 | 2,309.0 | 12 | 281.1 |
| SDVRP | 13 | – | – | – | 99.1 | 183.9 | – | – | 9 | 880.6 |
| MDVRP | 9 | – | – | – | 99.2 | 310.5 | – | – | 7 | 875.3 |
| MDVRP | 8 | – | – | – | 99.7 | 189.5 | – | – | 7 | 4,788.6 |
| | 86 | | | | 99.5 | 176.2 | | | 74 | 993.6 |

In both cases, the route sets $\hat{\mathscr{R}}^k$, $k \in M$, are generated by using procedure GENROUTE described in Sect. 2.4 for the CVRP.

## 5.4 Computational results for the HVRP

In this section, we report a comparison of the results obtained by Baldacci and Mingozzi (2009) with those of Choi and Tcha (2007) and Pessoa et al. (2007) on three main sets of instances from the literature. These instances correspond to the HVRP variants, SDVRP and MDVRP. The complete details of the instances can be found in Baldacci and Mingozzi (2009).

The algorithms described in Baldacci and Mingozzi (2009) were coded in Fortran 77, and the experiments were run on a personal computer with an AMD Athlon 64 X2 Dual Core 4200+ processor at 2.6 GHz and 3 GB of RAM. CPLEX 10.1 was used as the LP solver in procedure H3 and as the integer linear programming solver in the exact method. The computing times of the method of Pessoa et al. (2007) are seconds of a Pentium Core 2 Duo at 2.13 GHz.

Table 9 summarizes the results obtained over all the variants considered by the different methods. This table reports the following columns: the total number of instances in each class ($np$), the average percentage ratio of the lower bound with respect to the optimal solution value ($\%LB$), the average running time in seconds for computing the lower bound ($t_{LB}$), the number of instances solved to optimality by each method ($nopt$), and the average running time in seconds of the exact method computed over all the instances solved to optimality by all methods ($t_{TOT}$).

Table 9 shows that the exact method of Baldacci and Mingozzi (2009) was able to solve to optimality 74 out of 86 instances considered. The table shows that the average percentage ratio of lower bound $\%LB$ obtained by the method, computed over all 86 instances, is equal to 99.5 and that the corresponding average computing time is 176.2 s.

## 6 The period vehicle routing problem

The period vehicle routing problem (PVRP) is defined on an undirected graph $G = (V', E)$ and a set $P = \{1, \dots, p\}$ that represents a $p$-day planning horizon. With each day $k \in P$ it is associated a non-negative cost matrix $[d^k_{ij}]$ where $d^k_{ij}$ represents the cost for traversing edge $\{i, j\} \in E$ on day $k$. A fleet of $m_k$ vehicles is available on each day $k$ to supply the customers $V = V' \setminus \{0\}$.

Each customer $i$ specifies a service frequency $f_i$ and a quantity $q_i$ of product that must be received from the depot at each visit. The visits of customer $i$ can only occur in one of a given set $C_i$ of allowable *day-combinations* of $f_i$ days. The $f_i$ visit days of a day-combination are represented by a column of a (0–1) matrix $[a_{ks}]$ of $p$ rows where $a_{ks} = 1$ if day $k$ is an allowable visit day in day-combination $s$. Hereafter, we assume that $C_i$ is the index set of those columns of matrix $[a_{ks}]$ corresponding to allowable day-combinations of customer $i \in V$.

Every day, each vehicle can perform at most one route. Each route starts and finishes at the depot and services a total customer demand smaller than or equal to the vehicle capacity $Q$. The cost of a route on day $k$ is given by the sum of the costs $d^k_{ij}$ of the edges traversed by the route.

The PVRP has several practical applications in the grocery industry, the soft drink industry, the automotive industry, industrial gases distribution and refuse collection, and admits several variants in terms of the objectives and the specific constraints (see Mourgaya and Vanderbeck 2006).

The PVRP contains as special cases the CVRP, the MDVRP, and the tactical planning vehicle routing problem (TPVRP). The PVRP corresponds to the CVRP when the planning period is a single day (i.e., $p = 1$) and every customer must be visited from the depot exactly once (i.e., $f_i = 1, \forall i \in V$).

Any MDVRP instance with $p$ depots and $m_k$ vehicles located at depot $k$ can be converted into a PVRP instance defined on a $p$-day period, where each day corresponds to a depot, the (0–1) matrix $[a_{ks}]$ is the identity matrix of order $(p \times p)$, and each day combination of $C_i$ represents a depot that can service customer $i$.

The TPVRP is a problem with practical applications in different fields, such as food and beverage distribution and maintenance of logistics activities in the field force planning.

In the TPVRP, each customer $i$ has frequency $f_i = 1$ and requires to be visited on any day $k$ of a specified *day-window* $[e_i, l_i]$ of a short term horizon of $p$-days. Visiting customer $i$ on day $k \in [e_i, l_i]$ involves a service cost $\tau_i(k)$ that is proportional to the number of days of delay with respect to the first day of his day-window. The TPVRP requires to assign a visit day $k \in [e_i, l_i]$ to each customer $i$ and to design at most $m_k$ routes for each day of the period in order to visit each customer exactly once while minimizing the sum of the routing costs and service costs. Any TPVRP solution having all customers serviced on their first allowable day has service cost equal to zero. However, this solution can be too expensive in terms of routing costs or infeasible as it could require too many vehicles on some day of the period.

The TPVRP can be modeled as a PVRP as follows. Let $[a_{ks}]$ be the $(p \times p)$ identity matrix. Associate with each customer the set of day-combinations $C_i = \{s : s \in [e_i, l_i]\}$ and define the edge cost matrix $[d^k_{ij}]$ as $d^k_{ij} = \hat{d}_{ij} + \tau_i(k)/2 + \tau_j(k)/2, \forall k \in$

$P, \forall \{i, j\} \in E$, where $\hat{d}_{ij}$ is the travel cost associated with each edge $\{i, j\} \in E$ (we assume that $\tau_0(k) = 0, \forall k \in P$).

All papers on the PVRP in the literature present heuristic methods. A recent survey of the PVRP and its extensions, discussing modeling and heuristic methods, can be found in the chapter of Francis et al. (2008). To our knowledge, no exact methods have been proposed in the literature for both the PVRP and the TPVRP so far. The paper recently proposed by Baldacci et al. (2009b) is the first one presenting lower bounds and an exact method that can solve the PVRP and its variants. In this section, we shortly describe the mathematical formulation, the lower bounds, and the exact method proposed by Baldacci et al. (2009b) for the PVRP.

## 6.1 Mathematical formulation of the PVRP

In this section, we describe a SP-like formulation of the PVRP and three relaxations used to derive valid lower bounds on the PVRP.

The customer set $V$ can be partitioned as $V = V^1 \cup V^2$, where $V^1$ contains the customers having frequency equal to one (i.e., $V^1 = \{i \in V : f_i = 1\}$), and $V^2$ contains the customers having frequency greater than or equal to two (i.e., $V^2 = \{i \in V : f_i \geq 2\}$). We denote by $V_k \subseteq V$ the subset of customers that can be visited on day $k \in P$ (i.e., $V_k = \{i \in V : \sum_{s \in C_i} a_{ks} \geq 1\}$).

Let $\mathscr{R}^k$ be the index set of all routes of day $k \in P$ visiting the customers in $V_k$ on day $k$, and let $\mathscr{R}_i^k \subseteq \mathscr{R}^k$ be the index set of the route subset covering customer $i \in V_k$. We use $R_\ell^k$ and $c_\ell^k$ to indicate the subset of customers and the cost of route $\ell \in \mathscr{R}^k$ on day $k \in P$, respectively. Let $y_{is}$ be a (0–1) binary variable equal to 1 if and only if day-combination $s \in C_i$ is assigned to customer $i \in V^2$. Define a (0–1) binary variable $x_\ell^k$ that is equal to 1 if and only if route $\ell \in \mathscr{R}^k$ of day $k \in P$ is in solution. The PVRP is as follows:

$$(F) \quad z(F) = \min \sum_{k \in P} \sum_{\ell \in \mathscr{R}^k} c_\ell^k x_\ell^k \tag{36}$$

$$\text{s.t.} \sum_{k \in P} \sum_{\ell \in \mathscr{R}_i^k} x_\ell^k = f_i, \quad \forall i \in V, \tag{37}$$

$$\sum_{\ell \in \mathscr{R}_i^k} x_\ell^k - \sum_{s \in C_i} a_{ks} y_{is} = 0, \quad \forall i \in V^2, \forall k \in P, \tag{38}$$

$$\sum_{\ell \in \mathscr{R}^k} x_\ell^k \leq m_k, \quad \forall k \in P, \tag{39}$$

$$x_\ell^k \in \{0, 1\}, \quad \forall \ell \in \mathscr{R}^k, \forall k \in P, \tag{40}$$

$$y_{is} \in \{0, 1\}, \quad \forall s \in C_i, \forall i \in V^2. \tag{41}$$

Constraints (37) impose that each customer $i$ is visited exactly $f_i$ times. In particular, as a consequence of the definition of customer subsets $V_k$ and of the route sets $\mathscr{R}^k$, $k \in P$, every customer $i \in V^1$ is visited exactly once in one of its allowable days.

Constraints (38) impose that every customer $i \in V^2$ is visited exactly $f_i$ times during the $f_i$ days of the day-combination assigned to the customer. Constraints (39) force the solution to contain at most $m_k$ routes on day $k \in P$. Finally, constraints (40) and (41) are the integrality constraints for the decision variables $x^k_\ell$ and $y_{is}$, respectively.

Notice that constraints (37) and (38) imply that one day combination $s \in C_i$ is assigned to each customer $i \in V^2$, that is $\sum_{s \in C_i} y_{is} = 1, \forall i \in V^2$. In fact, summing all constraints (38) associated with a customer $i \in V^2$ over the $p$-days of the period and considering that $\sum_{k \in P} a_{ks} = f_i, \forall s \in C_i$, we obtain $\sum_{k \in P} \sum_{\ell \in \mathscr{R}^k_i} x^k_\ell = \sum_{s \in C_i} f_i y_{is}$. From the latter expression and equations (37), we derive $\sum_{s \in C_i} f_i y_{is} = f_i$, that is, $\sum_{s \in C_i} y_{is} = 1$.

Let $LF$ be the LP-relaxation of problem $F$. We denote by $z(LF)$ the optimal solution cost of $LF$. In the following sections, we describe three relaxations of problem $F$ used by five procedures for computing different lower bounds on the PVRP. These procedures are based on the three bounding procedures $H^1$, $H^2$ and $H^3$ described in Sect. 2 for the CVRP.

## 6.2 Relaxation $RF$ and procedures $H^1$ and $H^2$

Relaxation $RF$ corresponds to an integer problem whose optimal cost $z(RF)$ is a valid lower bound on $z(F)$. Problem $RF$ can be described as follows.

Let us associate with each edge $\{i, j\} \in E$ the cost $\overline{d}_{ij} = \min_{k \in P}\{d^k_{ij}\}$. Let $I$ be a $(n \times n)$ (0–1) matrix, where $I_{ij} = 0$ if and only if there exists at least one day $k \in P$ on which both $i$ and $j$ can be visited (i.e., $i, j \in V_k$, for some $k \in P$).

It is quite easy to observe that, by using the edge costs $\overline{d}_{ij}$, the cost of the least cost route in $G$ visiting any subset of customers $S$ is a lower bound on the cost $c^k_\ell$ of any route $\ell \in \mathscr{R}^k$, $k \in P$, such that $R^k_\ell = S$. Let $\overline{\mathscr{R}}$ be the index set of all least cost routes in $G$ with respect to the modified edge costs $\overline{d}_{ij}$ and satisfying, in addition to the capacity constraints, the constraint $\sum_{i \in \overline{R}_\ell} \sum_{j \in \overline{R}_\ell} I_{ij} = 0, \quad \forall \ell \in \overline{\mathscr{R}}$, where $\overline{R}_\ell$ is the subset of customers visited by route $\ell$. We denote by $\overline{c}_\ell$ the cost of route $\ell \in \overline{\mathscr{R}}$ and by $\overline{\mathscr{R}}_i \subseteq \overline{\mathscr{R}}$ the index subset of the routes visiting customer $i \in V$. Problem $RF$ is to select at most $\overline{m} = \sum_{k \in P} m_k$ routes from $\overline{\mathscr{R}}$, where each route can be in a solution more than once, and each customer $i \in V$ is visited exactly $f_i$ times.

Let $\overline{x}_\ell$ be a non-negative integer variable representing the number of times that route $\ell \in \overline{\mathscr{R}}$ is in the solution. Problem $RF$ is the following.

$$(RF) \quad z(RF) = \min \sum_{\ell \in \overline{\mathscr{R}}} \overline{c}_\ell \overline{x}_\ell \tag{42}$$

$$\text{s.t.} \sum_{\ell \in \overline{\mathscr{R}}_i} \overline{x}_\ell = f_i, \quad \forall i \in V, \tag{43}$$

$$\sum_{\ell \in \overline{\mathscr{R}}} \overline{x}_\ell \leq \overline{m}, \tag{44}$$

$$\overline{x}_\ell \geq 0 \text{ integer}, \quad \forall \ell \in \overline{\mathscr{R}}. \tag{45}$$

$z(RF)$ provides a valid lower bound on the PVRP as any route $\ell \in \mathscr{R}^k$, for some $k \in P$, corresponds to a route $\ell' \in \overline{\mathscr{R}}$ of cost $\bar{c}_{\ell'} \leq c^k_\ell$. Thus, any solution $(\mathbf{x}', \mathbf{y}')$ of problem $F$ of cost $z'(F)$ can be transformed into a feasible $RF$ solution $\bar{\mathbf{x}}$ of cost $\bar{z}(RF) \leq z'(F)$.

We denote by $DRF$ the dual of $RF$ and by $\mathbf{w} = (w_0, w_1, \ldots, w_n)$ the vector of $n+1$ variables, where variable $w_0$ is associated with constraint (44) and variables $w_i$, $\forall i \in V$, are associated with constraints (43).

Notice that problem $DRF$ differs from the dual *DF* of the SP formulation of the CVRP described in Sect. 2 only for the coefficients of the dual variables in the objective function. Thus, the procedures $H^1$ and $H^2$ described in Sect. 2.3 can be used in sequence to find two near-optimal $DRF$ solutions $\mathbf{w}^1$ and $\mathbf{w}^2$ of cost $LB1$ and $LB2$, respectively.

## 6.3 Relaxation $LF$ and procedures $HF^1$ and $HF^2$

A second relaxation of the PVRP is based on the dual problem of $LF$, called *DF*. Associate dual variables $v_i, i \in V$, with constraints (37), $u_{ik}, i \in V^2, k \in P$, with constraints (38), and $\sigma_k, k \in P$, with constraints (39). Let $B^k_\ell = R^k_\ell \cap V^2$ be the subset of customers visited by route $\ell \in \mathscr{R}^k$ having frequency greater than one.

The following theorem, that is a generalization of Theorem 1 described in Sect. 2, provides a method for computing a near-optimal *DF* solution without generating all route sets $\mathscr{R}^k, k \in P$.

**Theorem 3** *Associate penalties $\mu_i \in \mathbb{R}$ with each customer $i \in V$, penalties $\lambda_{ik} \in \mathbb{R}$ with each customer $i \in V^2$ and day $k \in P$, and penalties $\gamma_k$ with each day $k \in P$. For each route $\ell \in \mathscr{R}^k, k \in P$, define $\mu(R^k_\ell) = \sum_{i \in R^k_\ell} \mu_i$, $\lambda(R^k_\ell) = \sum_{i \in B^k_\ell} \lambda_{ik}$ and $q(R^k_\ell) = \sum_{i \in R^k_\ell} q_i$. Compute*

$$b_{ik} = q_i \min_{\ell \in \mathscr{R}^k_i} \{(c^k_\ell - \mu(R^k_\ell) - \lambda(R^k_\ell) - \gamma_k)/q(R^k_\ell)\} + \mu_i + \lambda_{ik}, \quad \forall i \in V, \ \forall k \in P. \tag{46}$$

*A feasible solution $(\mathbf{u}, \mathbf{v}, \boldsymbol{\sigma})$ of cost $z(DF(\boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\gamma}))$ of problem DF can be obtained as follows:*

$$\left.\begin{array}{llll} v_i = \min_{k \in P}\{b_{ik}\}, & \forall i \in V^1, & (a) & \\ v_i = \frac{1}{f_i} \min_{s \in C_i} \left\{\sum_{k \in P} a_{ks} b_{ik}\right\}, & \forall i \in V^2, & (b) & \\ u_{ik} = b_{ik} - v_i, & & \forall i \in V^2, \ \forall k \in P, & (c) \\ \sigma_k = \gamma_k, & & \forall k \in P. & (d) \end{array}\right\} \tag{47}$$

Problem *DF* is solved using two bounding procedures, called $HF^1$ and $HF^2$, that are extensions of procedures $H^1$ and $H^2$ described in Sect. 2 for the CVRP. $HF^1$ and $HF^2$ use procedure $CG$ (see Sect. 2) for computing expressions (46) and (47).

## 6.4 Relaxation $\overline{LF}$ and bounding procedure $H^3$

A better relaxation than $LF$, called $\overline{LF}$, is obtained by adding to $LF$ the following extensions of two well-known valid inequalities designed for the CVRP.

(a) *Generalized Capacity Constraints*. Let $\overline{P} \subseteq P$, and let $\hat{q}_i = q_i \min_{s \in C_i} \{\sum_{k \in \overline{P}} a_{ks}\}$ be a lower bound on the total demand delivered to customer $i \in V$ during days in $\overline{P}$. Any feasible $F$ solution must satisfy the following inequalities:

$$\sum_{k \in \overline{P}} \sum_{\ell \in \mathscr{R}^k(S)} x^k_\ell \geq \left\lceil \sum_{i \in S} \hat{q}_i / Q \right\rceil, \quad \forall S \in \mathscr{S}, \tag{48}$$

where $\mathscr{R}^k(S) = \{\ell \in \mathscr{R}^k : R^k_\ell \cap S \neq \emptyset\}$.

(b) *Clique Inequalities*. Let $H = (\mathscr{R}, \mathscr{E})$ be the *conflict graph* associated with the routes of sets $\mathscr{R}^k, \forall k \in P$. Node $i$ of graph $H$ represents route $\ell(i)$ of day $k(i)$. The edge set $\mathscr{E}$ contains every pair $\{i, j\}, i < j$, such that the set of conflicting customers $S = R^{k(i)}_{\ell(i)} \cap R^{k(j)}_{\ell(j)} \neq \emptyset$ and one of the following two conditions is satisfied: (a) $k(i) = k(j)$; (b) $k(i) \neq k(j)$ or at least one conflicting customer either has frequency equal to one or cannot be serviced on both days $k(i)$ and $k(j)$ according to the customer day combinations (i.e., either $\min_{h \in S}\{f_h\} = 1$ or $a_{k(i)s} + a_{k(j)s} \leq 1, \forall s \in C_h$, for some $h \in S$). Then, any feasible $F$ solution must satisfy the following inequalities:

$$\sum_{i \in C} x^{k(i)}_{\ell(i)} \leq 1, \quad \forall C \in \mathscr{C}, \tag{49}$$

where $\mathscr{C}$ is the set of all cliques of graph $H$.

$\overline{LF}$ is solved using a column-and-cut generation method, called $H^3$, that is based on the simplex algorithm. The initial master problem of $H^3$ is generated using the best dual solution computed by $H^1$, $H^2$, $HF^1$ and $HF^2$.

We denote by $LB3$ the optimal solution cost of $\overline{LF}$ achieved by $H^3$.

## 6.5 An exact algorithm for solving the PVRP

The exact algorithm for the PVRP is based on the exact method described in Sect. 2. It generates the reduced problem $\hat{F}$ obtained from $F$ by replacing each $\mathscr{R}^k$ with the subset $\hat{\mathscr{R}}^k$ containing all routes of reduced cost smaller than $z(UB) - LB3$, with respect to the final dual solution of $H^3$, and adding the subsets of generalized capacity and clique inequalities generated by $H^3$. Then, problem $\hat{F}$ is solved using an integer programming solver.

A valid lower bound on the PVRP can be obtained by executing in sequence procedures $H^1$, $H^2$, $HF^1$, $HF^2$ and $H^3$, where each procedure starts from the dual solution found by the previous one. However, this method can be time consuming and can be improved without affecting the quality of the final lower bound $LB3$ by removing from the sequence either $H^1$ and $H^2$ or $HF^1$ and $HF^2$, according to the type of PVRP instance considered.

In our computational experience, we observed that $LB2$ is almost equal to $LBF2$ for PVRP instances having the following characteristics: (a) the edge costs of graph $G$ do not depend on the day of the period; and (b) the customer frequencies and the day-combinations are such that every customer can be serviced on any day of the period. For these type of instances it is worth executing the sequence $H^1$, $H^2$ and $H^3$, where the initial master of $H^3$ is generated using the dual solution of cost $LB2$ produced by $H^2$.

On the other hand, we observed that for any PVRP instances not satisfying at least one of the two conditions (a) and (b) both $LBF1$ and $LBF2$ are better than $LB1$ and $LB2$, respectively. In particular, whenever condition (a) is not satisfied, then $LBF2$ is significantly greater than $LB2$. In these cases, it is computationally convenient to execute in sequence $HF1$, $HF2$ and $H^3$. Notice that conditions (a) and (b) are satisfied by all PVRP instances proposed in the literature, but they are not satisfied by the PVRP instances corresponding to the MDVRP and the TPVRP.

## 6.6 Computational results for the PVRP

The exact algorithm of Baldacci et al. (2009b) was coded in Fortran 77, compiled with the Intel Visual Fortran 10.1 compiler and linked with the C source codes of the packages CVRPSEP (see Lysgaard 2003) and CLIQUER (see Niskanen and Östergård 2003). CPLEX 11.0 was used as the LP solver in procedure $H^3$ and as the integer programming solver in the exact method. All the experiments were run on a Fujitsu Siemens Primergy TX200S3 running an Intel Xeon E5310 processor at 1.6 GHz with 8 Gb of RAM. The algorithm was tested on two sets of instances.

(a) 28 PVRP instances from the literature that are publicly available at http://neumann.hec.ca/chairedistributique/data/pvrp/. These instances were proposed by Christofides and Beasley (1984), Russel and Igo (1979), Russel and Gribbin (1991) and Chao et al. (1995).

(b) 20 TPVRP instances introduced by Baldacci et al. (2009b). These instances were derived from five CVRP instances, namely E-n51-k5, E-n76-k10, E-n101-k8, M-n121-k7, M-n151-k12 and M-n200-k16 and are available at http://branchandcut.org/VRP/data. In these instances, each customer $i$ has frequency $f_i = 1$, $p = 5$, and day windows have width of at most 3 days. A service cost is imposed for visiting a customer at a later day than the first one in its day-window.

Table 10 reports computational results over those PVRP instances from the literature that could be solved to optimality or for which the best known upper bound was improved by the exact method of Baldacci et al. (2009b). The first three columns report the name of the instance, the number of customers involved, and the number of days of the planning period, respectively. Column $z^*$ reports the value of the optimal solution or the value of the best known upper bound. In this column, "a" indicates that the solution found by the exact algorithm could not be proved to be optimal, and values in bold indicate that the solution found improves the previously best known upper bound. Columns $\%LB1$, $\%LB2$ and $\%LB3$ report the percentage ratio $100.0LB_x/z^*$ of lower bounds

**Table 10** Summary results for the PVRP

| Name | $\lvert V \rvert$ | $p$ | $z^*$ | Bounding procedure | | | | Exact method | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | %LB1 | %LB2 | %LB3 | $t_{BP}$ | $\lvert\hat{\mathscr{R}}\rvert$ | $t_{TOT}$ |
| p01 | 52 | 2 | 524.61 | 98.5 | 99.3 | 100.0 | 4.4 | 0 | 4.4 |
| p03 | 51 | 5 | 524.61 | 98.5 | 99.3 | 100.0 | 4.7 | 0 | 4.7 |
| p04 | 76 | 2 | 835.26 | 97.6 | 97.8 | 99.5 | 81.3 | 19,399 | 335.6 |
| p06 | 76 | 10 | **835.26** | 97.6 | 97.8 | 99.5 | 81.2 | 27,644 | 670.3 |
| p14 | 21 | 4 | 954.81 | 89.3 | 100.0 | 100.0 | 0.7 | 0 | 0.7 |
| p15 | 39 | 4 | 1,862.63 | 95.2 | 100.0 | 100.0 | 3.5 | 0 | 3.5 |
| p16 | 57 | 4 | 2,875.24 | 97.1 | 100.0 | 100.0 | 6.8 | 0 | 6.8 |
| p17 | 41 | 4 | 1,597.75 | 92.3 | 99.8 | 100.0 | 2.6 | 0 | 2.6 |
| p21 | 61 | 4 | 2,170.61 | 92.5 | 99.6 | 99.9 | 145.1 | 27,005 | 163.5 |
| p24 | 52 | 6 | 3,687.46 | 86.0 | 99.8 | 99.8 | 2.3 | 5,628 | 61.6 |
| p25 | 52 | 6 | 3,777.15 | 88.3 | 99.4 | 99.4 | 4.2 | 18,390 | 223.2 |
| p26 | 52 | 6 | 3,795.32 | 90.6 | 99.9 | 99.9 | 1.9 | 1,500 | 3.8 |
| p27 | 103 | 6 | **21,912.85**[a] | 77.5 | 99.4 | 99.4 | 81.5 | 300,000 | *tl* |
| p28 | 103 | 6 | **22,242.51** | 80.0 | 99.8 | 99.8 | 12.5 | 141,246 | 13,866.8 |
| p29 | 103 | 6 | **22,543.76** | 85.9 | 99.9 | 99.9 | 8.2 | 7,680 | 6,458.8 |
| p31 | 154 | 6 | **76,322.04**[a] | 78.1 | 99.3 | 99.3 | 186.8 | 300,000 | *tl* |
| | | | | 90.3 | 99.5 | 99.8 | 39.2 | | |

$LB1$, $LB2$ and $LB3$, respectively. Column $t_{BP}$ reports the total computing time in seconds used by procedures $H^1$, $H^2$ and $H^3$. Finally, columns $|\hat{\mathscr{R}}|$ and $t_{TOT}$ report the total number of routes in the final route subsets and the total computing time, respectively.

A time limit of 14,400 s was imposed on CPLEX for solving the reduced problem $\hat{F}$. An entry "$tl$" under column $t_{TOT}$ indicates that the algorithm terminated prematurely as CPLEX reached the imposed time limit.

Table 11 reports the results obtained on the new TPVRP instances. In this table, columns $|V|$, $Q$ and $m$ report for each problem the number of customers, the vehicle capacity, and the number of vehicles available daily. Columns $\%LBF1$, $\%LBF2$ and $\%LB3$ report the percentage ratio of lower bounds $LBF1$, $LBF2$ and $LB3$. Column $t_{BP}$ reports the total computing time of procedures $HF^1$, $HF^2$ and $H^3$. Columns $z^*$, $|\hat{\mathscr{R}}|$ and $t_{TOT}$ have the same meaning as in Table 10 except that a time limit of 7,200 s was imposed on CPLEX for these instances. For the TPVRP instances in this table, the upper bounds were obtained by Baldacci et al. (2009b) using a tabu search heuristic. Values in bold in column $z^*$ indicate that the solution found by the exact algorithm improves this upper bound.

Table 10 shows that the exact method of Baldacci et al. (2009b) was able to optimally solve for the first time 14 out of the 28 PVRP instances from the literature and to improve the best known upper bound of 5 instances. On these instances, the final lower bound computed is close to optimality and required on average 40 s of computing time.

**Table 11** Summary results for the TPVRP

| Name | $\|V\|$ | $Q$ | $m$ | $z^*$ | Bounding procedure | | | | Exact method | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | %LBF1 | %LBF2 | %LB3 | $t_{BP}$ | $\|\hat{\mathscr{R}}\|$ | $t_{TOT}$ |
| E-n51-k5 | 51 | 80 | 4 | 988.98 | 95.4 | 99.8 | 100.0 | 6.7 | 0 | 6.7 |
| E-n76-k10 | 76 | 80 | 6 | 1,574.10 | 93.0 | 99.5 | 100.0 | 8.4 | 0 | 8.4 |
| E-n101-k8 | 101 | 80 | 5 | 1,791.78 | 89.5 | 98.5 | 99.5 | 34.4 | 20,752 | 58.0 |
| M-n151-k12 | 151 | 80 | 7 | 2,648.73 | 86.3 | 98.7 | 99.4 | 78.3 | 73,913 | 1,211.9 |
| M-n200-k16 | 200 | 80 | 9 | **3,507.50**[a] | 83.4 | 98.6 | 99.2 | 82.0 | 300,000 | tl |
| E-n51-k5 | 51 | 100 | 3 | 937.96 | 91.5 | 99.4 | 100.0 | 7.9 | 0 | 7.9 |
| E-n76-k10 | 76 | 100 | 5 | 1,418.71 | 92.6 | 98.9 | 99.8 | 14.7 | 11,384 | 15.4 |
| E-n101-k8 | 101 | 100 | 5 | 1,517.89 | 93.8 | 98.9 | 99.6 | 34.4 | 20,996 | 54.8 |
| M-n151-k12 | 151 | 100 | 6 | **2,246.07**[a] | 89.6 | 98.4 | 99.1 | 85.5 | 300,000 | tl |
| M-n200-k16 | 200 | 100 | 8 | **2,877.23**[a] | 87.6 | 97.3 | 97.9 | 96.9 | 300,000 | tl |
| E-n51-k5 | 51 | 140 | 3 | 869.24 | 86.7 | 97.0 | 99.7 | 498.5 | 16,744 | 639.3 |
| E-n76-k10 | 76 | 140 | 4 | 1,203.91 | 93.9 | 98.9 | 99.5 | 25.5 | 14,541 | 27.5 |
| E-n101-k8 | 101 | 140 | 4 | 1,330.74 | 91.9 | 98.9 | 99.8 | 619.3 | 20,892 | 956.1 |
| M-n151-k12 | 151 | 140 | 5 | **1,867.06**[a] | 91.4 | 97.4 | 98.1 | 252.8 | 300,000 | tl |
| M-n200-k16 | 200 | 140 | 6 | 2,317.54[a] | 90.3 | 98.6 | 98.8 | 804.7 | 300,000 | tl |
| E-n51-k5 | 51 | 160 | 3 | 839.05 | 86.0 | 96.9 | 99.6 | 624.2 | 17,003 | 1,012.1 |
| E-n76-k10 | 76 | 160 | 4 | 1,151.66 | 93.6 | 99.1 | 100.0 | 33.3 | 0 | 33.3 |
| E-n101-k8 | 101 | 160 | 3 | 1,292.41 | 90.3 | 98.3 | 98.9 | 1,120.9 | 267,839 | 10,666.0 |
| M-n151-k12 | 151 | 160 | 4 | 1,772.27[a] | 91.2 | 97.9 | 98.2 | 929.2 | 300,000 | tl |
| M-n200-k16 | 200 | 160 | 5 | 2,241.47[a] | 87.4 | 96.3 | 96.9 | 730.3 | 300,000 | tl |
| | | | | | 90.3 | 98.4 | 99.2 | 304.4 | | |

Table 11 shows that the lower bound achieved on the TPVRP is on average within one percent of the optimal solution. The exact method was able to solve to optimality 13 out of the 20 TPVRP instances considered and all the TPVRP instances involving up to 100 customers.

## 7 Conclusions

In this paper, we presented an exact solution framework for solving some variants of the VRP that can be modeled as SP problems with additional constraints.

We described how the framework has been used to derive exact algorithms for the CVRP, the VRPTW, the PDPTW, all types of HVRP including the MDVRP, and the PVRP.

For each VRP variant, we reported a computational comparison of the results obtained with the best methods presented in the literature. The computational results show that the exact algorithm derived for each of these VRP variants outperforms all other exact methods published so far and can solve several test instances that were previously unsolved.

**Acknowledgments** The authors thank two anonymous referees for helpful comments.

## References

Augerat P (1995) Approche polyédrale du problème de tournées de véhicules. PhD thesis, Institut National Polytechnique de Grenoble

Augerat P, Belenguer JM, Benavent E, Corberán A, Naddef D, Rinaldi G (1995) Computational results with a branch and cut code for the capacitated vehicle routing problem. Technical Report 1 RR949-M, ARTEMIS-IMAG, Grenoble, France

Baldacci R, Mingozzi A (2009) A unified exact method for solving different classes of vehicle routing problems. Math Program Ser A 120(2):347–380

Baldacci R, Hadjiconstantinou EA, Mingozzi A (2004a) An exact algorithm for the capacitated vehicle routing problem based on a two-commodity network flow formulation. Oper Res 52:723–738

Baldacci R, Maniezzo V, Mingozzi A (2004b) An exact method for the car pooling problem based on lagrangean column generation. Oper Res 52:422–439

Baldacci R, Bodin LD, Mingozzi A (2006) The multiple disposal facilities and multiple inventory locations rollon-rolloff vehicle routing problem. Comput Oper Res 33:2667–2702

Baldacci R, Toth P, Vigo D (2007) Recent advances in vehicle routing exact algorithms. 4OR: Q J Oper Res 5(4):269–298

Baldacci R, Battarra M, Vigo D (2008a) Routing a heterogeneous fleet of vehicles. In: Golden BL, Raghavan S, Wasil E (eds) The vehicle routing problem: latest advances and new challenges, vol 43. Springer, Berlin

Baldacci R, Christofides N, Mingozzi A (2008b) An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. Math Program Ser A 115(2):351–385

Baldacci R, Bartolini E, Mingozzi A (2009a) An exact algorithm for the pickup and delivery problem with time windows (submitted)

Baldacci R, Bartolini E, Mingozzi A, Valletta A (2009b) An exact algorithm for the period routing problem (submitted)

Baldacci R, Mingozzi A, Roberti R (2009c) Solving the vehicle routing problem with time windows using new state space relaxation and pricing strategies (submitted)

Baldacci R, Mingozzi A, Roberti R (2009d) New benchmarks results for the capacitated vehicle routing problem. Working paper

Boschetti MA, Mingozzi A, Ricciardelli S (2004) An exact algorithm for the simplyfied multi depot crew scheduling problem. Ann Oper Res 127:177–201

Braysy O, Gendreau M (2005a) Vehicle routing problem with time windows, part II: metaheuristics. Transp Sci 39(1):119–139

Braysy O, Gendreau M (2005b) Vehicle routing problem with time windows, part I: route construction and local search algorithms. Transp Sci 39(1):104–118

Chao IM, Golden BL, Wasil E (1995) An improved heuristic for the period vehicle-routing problem. Networks 26(1):25–44

Choi E, Tcha DW (2007) A column generation approach to the heterogeneous fleet vehicle routing problem. Comput Oper Res 34:2080–2095

Christofides N, Beasley JE (1984) The period routing problem. Networks 14:237–256

Christofides N, Eilon S (1969) An algorithm for the vehicle dispatching problem. Oper Res Q 20:309–318

Christofides N, Mingozzi A (1989) Vehicle routing: practical and algorithmic aspects. In: van Rijn CFH (ed) Logistics: where ends have to meet. Pergamon Press, New York pp 30–48

Christofides N, Mingozzi A, Toth P (1979) The vehicle routing problem. In: Christofides N, Mingozzi A, Toth P, Sandi C (eds) Combinatorial optimization. Wiley, Chichester pp 315–338

Christofides N, Mingozzi A, Toth P (1981a) Exact algorithms for the vehicle routing problem based on spanning tree and shortest path relaxation. Math Program 10:255–280

Christofides N, Mingozzi A, Toth P (1981b) State-space relaxation procedures for the computation of bounds to routing problems. Networks 11:145–164

Christofides N, Mingozzi A, Toth P (1981c) Exact algorithms for the vehicle routing problem based on spanning tree and shortest path relaxations. Math Program 20(20):255–282

Cordeau J-F, Laporte G, Savelsbergh MWP, Vigo D (2007) Vehicle routing. In: Barnhart C, Laporte G (eds) Transportation, handbooks in operations research and management science, vol 14. Elsevier, Amsterdam, pp 367–428

Cordeau J-F, Laporte G, Ropke S (2008) Recent models and algorithms for one-to-one pickup and delivery problems. In: Golden B, Raghavan S, Wasil E (eds) The vehicle routing problem: latest advances and new challenges, vol 43. Springer, Berlin, pp 327–357

Crevier B, Cordeau JF, Laporte G (2007) The multi-depot vehicle routing problem with inter-depot routes. Eur J Oper Res 176:756–773

Desaulniers G, Lessard F, Hadjar A (2008) Tabu search, partial elementarity, and generalized k-path inequalities for the vehicle routing problem with time windows. Transp Sci 42(3):387–404

Desrochers M, Desrosiers J, Solomon M (1992) A new optimization algorithm for the vehicle-routing problem with time windows. Oper Res 40(2):342–354

Dumas Y, Desrosiers J, Soumis F (1991) The pickup and delivery problem with time windows. Eur J Oper Res 54(1):7–22

Francis PM, Smilowitz KR, Tzur M (2008) The period vehicle routing problem and its extensions. In: Golden BL, Raghavan S, Wasil E (eds) The vehicle routing problem: latest advances and new challenges, vol 43. Springer, Berlin

Fukasawa R, Longo H, Lysgaard J, Poggide Aragão M, Reis M, Uchoa E, Werneck RF (eds) (2006) Robust branch-and-cut-and-price for the capacitated vehicle routing problem. Math Program Ser A 106: 491–511

Gendreau M, Laporte G, Potvin J-Y (2002) Metaheuristics for the capacitated VRP. In: Toth P, Vigo D (eds) The vehicle routing problemm, vol 9. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia pp 129–154

Jepsen M, Petersen B, Spoorendonk S, Pisinger D (2008) Subset-row inequalities applied to the vehicle-routing problem with time windows. Oper Res 56(2):497–511

Laporte G, Semet F (2002) Classical heuristics for the capacitated VRP. In: Toth P, Vigo D (eds) The vehicle routing problem, vol 9. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia pp 109–128

Laporte G, Nobert Y, Arpin D (1984) Optimal solutions to capacitated multi depot vehicle routing problems. Congressus Numerantium 44:283–292

Laporte G, Nobert Y, Taillefer S (1988) Solving a family of multi-depot vehicle routing and location-routing problems. Transp Sci 22:161–172

Li H, Lim A (2001) A metaheuristic for the pickup and delivery problem with time windows. In: 13th IEEE international conference on tools with artificial intelligence, ICTAI-2001, Dallas, USA

Lu Q, Dessouky M (2004) An exact algorithm for the multiple vehicle pickup and delivery problem. Transp Sci 38(4):503–514

Lysgaard J (2003) CVRPSEP: a package of separation routines for the capacitated vehicle routing problem. Technical report, Dept. of Mgt. Science and Logistics, Aarhus School of Business

Lysgaard J, Letchford AN, Eglese RW (2004) A new branch-and-cut algorithm for the capacitated vehicle routing problem. Math Program Ser A 100:423–445

Mingozzi A, Christofides N, Hadjiconstantinou EA (1994) An exact algorithm for the vehicle routing problem based on the set partitioning formulation. Technical report, University of Bologna

Mingozzi A, Giorgi S, Baldacci R (1999) An exact method for the vehicle routing problem with backhauls. Transp Sci 33(3):315–329

Mourgaya M, Vanderbeck F (2006) The periodic vehicle routing problem: classification and heuristic. Rairo-Oper Res 40(2):169–194

Niskanen S, Östergård PRJ (2003) Cliquer user's guide. Technical Report 48, Helsinki University of Technology Communications Laboratory

Parragh SN, Doerner KF, Hartl RF (2008) A survey on pickup and delivery problems part II: transportation between pickup and delivery locations. J Betriebswirtschaft 51:81–117

Pessoa A, Poggi de Aragão M, Uchoa E (2007) A robust branch-cut-and-price algorithm for the heterogeneous fleet vehicle routing problem. Lecture notes in computer science, vol 4525. Springer, Berlin 150–160

Ropke S, Cordeau J-F (2009) Branch-and-cut-and-price for the pickup and delivery problem with time windows. Transp Sci (Forthcoming)

Ropke S, Cordeau JF, Laporte G (2007) Models and branch-and-cut algorithms for pickup and delivery problems with time windows. Networks 49(4):258–272

Russel RA, Gribbin D (1991) A multiphase approach to the period routing problem. Networks 21:747–765

Russel RA, Igo W (1979) An assignment routing problem. Networks 9:1–17

Savelsbergh MWP, Sol M (1998) Drive: dynamic routing of independent vehicles. Oper Res 46(4):474–490

Solomon M (1987) Algorithms for the vehicle routing and scheduling problems with the time window constraints. Oper Res 35:254–265

Toth P, Vigo D (eds) (2002) The vehicle routing problem. Monographs on discrete mathematics and applications. SIAM, Philadelphia