European Journal of Operational Research 272 (2019) 121–131

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Production, Manufacturing and Logistics

A column generation approach for location-routing problems with pickup and delivery

Thomas Capelle[a], Cristián E. Cortés[c,*], Michel Gendreau[b], Pablo A. Rey[d], Louis-Martin Rousseau[b]

[a] INRIA, Université Grenoble Alpes, 655 Avenue de l'Europe, 38334 Montbonnot, France
[b] CIRRELT and MAGI, Polytechnique Montréal, C.P. 6079, succ. Centre-ville, Montréal, QC, H3C 3A7, Canada
[c] Department of Civil Engineering, Universidad de Chile, Blanco Encalada 2002, Santiago, Chile
[d] Departamento de Industria and Programa Institucional de Fomento a la Investigación, Desarrollo e Innovación, Universidad Tecnológica Metropolitana, José Pedro Alessandri 1242, Ñuñoa, Santiago, Chile

ARTICLE INFO

Article history:
Received 19 December 2016
Accepted 26 May 2018
Available online 15 June 2018

Keywords:
Routing
Location
Pickup and delivery
Branch and price

ABSTRACT

In this paper we formulate an integer programming model for the Location and Routing Problem with Pickup and Delivery. We propose a column generation scheme and implement, for the subproblem, a label-setting algorithm for the shortest path with pickup and delivery and time windows problem. We also propose a set of heuristics to speed up this process. To validate the model, we implement the column generation scheme and test it on different instances developed in this paper. We also provide an analysis of how the costs of opening depots and the fixed cost of routes affect the optimal solution.

1. Introduction

Distribution costs make up a large fraction of the total cost of supply chains. Because of that, the design of logistics systems has received very significant attention in recent years (Ghiani, Laporte, & Musmanno, 2013). A good logistic design has to efficiently provide solutions at multiple levels of decision making for the following two primary issues: the location of facilities (depots), which act as bases for vehicles, and the assignment and routing of vehicles.

Several authors in the past have considered solving simultaneously facility location and routing problems in so-called location-routing problems (LRPs), because solving independently depot location and routing problems leads to suboptimal solutions.

Most of the LRP literature deals with a routing scheme that involves only deliveries at customer locations or pickups at such locations, but not both. In the present paper, we consider the case where each service request involves picking up some items from a given origin and delivering them to a specified destination. This problem can be defined as an LRP with pickup and delivery (LRP-PDP).

The main objective of this paper is to formulate a model that integrates the PDP and optimum depot location (LRP-PDP). In addition, an efficient solution method is proposed based on a column generation scheme within a B&P framework. Our approach is based on the work of Berger, Coullard, and Daskin (2007), but the pricing subproblem is modified to incorporate the specifics of the PDP.

One embedded difficulty of this integrated methodology is the exact algorithm required to solve the PDP-TW specified at the subproblem level, in which precedence constraints are very difficult to handle efficiently, considering, in addition, that in the present application we are solving the PDP in the context of a LRP. An important contribution of this paper is therefore to propose for the first time an exact formulation for the LRP-PDP, in which the PDP is defined in the way that Savelsbergh and Sol (1995b) do, considering explicitly the precedence constraints on related pickups and deliveries.

The B&P includes an important preprocessing stage, a proper pricing implementation, and a dedicated branching strategy. These branching rules allow the proposed B&P to tackle instances of realistic size, as shown through our computational experiments.

The structure of this article is as follows. Section 2 presents related literature, while Section 3 discusses the column generation scheme for the LRP-PDP, presenting both the master problem and the subproblem; it also provides details on the preprocessing step and the branching strategy. Section 4 presents the label-setting algorithm for the pricing subproblem. Section 5 discusses the

* Corresponding author.
E-mail addresses: thomas.capelle@inria.fr (T. Capelle), ccortes@ing.uchile.cl (C.E. Cortés), michel.gendreau@cirrelt.net (M. Gendreau), prey@utem.cl (P.A. Rey), louis-martin.rousseau@cirrelt.net (L.-M. Rousseau).

https://doi.org/10.1016/j.ejor.2018.05.055
0377-2217/

computational results of the proposed approach. Section 6 concludes the paper.

2. Related work

To the best of our knowledge, Webb (1968) and Christofides and Eilon (1969) were the first authors that considered explicitly the routing costs in the context of location problems. The usual practice at that time was to consider the cost of delivery as a weighted sum of the radial distances to destinations, as would be the result of direct routes from depot to customers. Webb (1968) showed that the optimum location considering the total distances in a set of routes serving several customers can differ greatly from the optimum location when delivery cost is estimated by radial distances from depot to customers. Christofides and Eilon (1969) studied more closely the relationship between the total distances traveled on routes to serve a set of customers and the simplification based on the radial distances.

2.1. Location routing problems

The LRP takes its origin in the roundtrip location problem that considers vehicles that deliver goods directly from one customer to another, although each vehicle is limited to a single pickup and a single delivery only (Chan & Hearn, 1977; Drezner & Wesolowsky, 1982; Kolen, 1985). Several authors have followed up. Laporte, Nobert, and Taillefer (1988) provide a survey of the early literature on the topic, while Prodhon and Prins (2014) and Drexl and Schneider (2015) review the LRP literature up to 2014.

Few authors have developed exact methods to deal with LRP as the combination of location and routing decisions is challenging. Some recent works have proposed lower bounds and exact methods for such problems. Barreto (2004) proposed a lower bound based on a cutting plane approach for the CLRP. Berger et al. (2007) formulated a set-partitioning model of an uncapacitated LRP with distance constraints that was solved through a branch-and-price (B&P) algorithm. This algorithm yielded optimal solutions with reasonable computational time for problems with 10 candidate facilities and 100 customers with different distance constraints. Belenguer, Benavent, Prins, Prodhon, and Wolfler-Calvo (2011) developed a branch-and-cut algorithm based on several families of valid inequalities for the LRP with capacity constraints on both the depots and the vehicles. Their method is able to solve optimally instances with up to 40 or 50 customers. Baldacci, Mingozzi, and Wolfler Calvo (2011b) proposed a branch-and-cut-and-price algorithm to solve a capacitated LRP based on set partitioning, decomposing the problem into a limited set of multi-depot vehicle routing problems (MDVRP). Contardo, Cordeau, and Gendron (2013) developed a branch-and-cut-and price algorithm to solve an LRP formulation, with instances of about 50 customers and 5–10 depots. These are the paper that are the most related to our approach.

As expected, most of the LRP literature has focused on the development of approximate algorithms, including constructive heuristics (Boudahri, Aggoune-Mtalaa, Bennekrouf, & Sari, 2013; Manzour-al Ajdad, Torabi, & Salhi, 2012) and metaheuristics (Contardo, Cordeau, & Gendron, 2014; Derbel, Jarboui, Hanafi, & Chabchoub, 2010; 2012; Hemmelmayr, Cordeau, & Crainic, 2012).

Some extensions of traditional LRPs are classified in Drexl and Schneider (2015), the most relevant being the Generalized LRP, the Prize-Collecting LRP, the Split delivery LRP (Archetti & Speranza, 2008), Stochastic LRPs (Ahmadi-Javid & Seddighi, 2013), and the LRP with simultaneous pickup and delivery LRP-SPD (Karaoglan, Altiparmak, Kara, & Dengiz, 2011; 2012), which is briefly discussed at the end of the next subsection in the context of pickup and delivery models.

Many-to-many location-routing problem (MMLRP) investigates locating hubs to facilitate transshipment from several customers to several customers. Routing aspects are involved, but in general goods do not travel from a customer straight to another customer. Papers on this topic include, among others, (Çetiner, Sepil, & Süral, 2010; Nagy & Salhi, 2007; Wasner & Zäpfel, 2004). A recent reference for many-to-many location routing problems is the paper by Rieck, Ehrenberg, and Zimmermann (2014). In that work, and others on MMLRP, the location of hubs is considered in a multi-echelon network. There is actually the possibility of direct shipping between nodes other than hubs, but here hubs (the nodes that should be located) act more like consolidation nodes or transfer points than depots as in the model developed in this paper.

2.2. Pickup and delivery problems

We aim to investigate LRP models where vehicle routes follow a pickup and delivery organization and satisfy time window constraints (PDP-TW). Formally, the formulated PDP is a generic problem for companies that transport passengers or freight and have a fixed fleet of m vehicles that each have fixed capacity Q (Savelsbergh & Sol, 1995a). In the PDP, each customer request i has an origin location i^+ and a destination location i^-. The objective of the PDP is to find a feasible set of minimum cost routes serving all customer requests.

In general, PDP problems are classified into three groups according to origin-destination relation: many-to-many, one-to-many-to-one and one-to-one (Berbeglia, Cordeau, Gribkovskaia, & Laporte, 2007). PDP-TW problems are also divided into two categories: 1-PDPTW (single-vehicle) and m-PDPTW (multi-vehicle), as pointed out by Dridi, Kammarti, Borne, and Ksouri (2011). PDPs can also be classified depending on their level of dynamism (Berbeglia, Cordeau, & Laporte, 2010): in static approaches, all requests are known in advance, typically one day before the service; in dynamic approaches, not all information is available in advance, but is revealed during the execution of the planned operations, noting that vehicle prepositioning decisions can be made in anticipation of future arrivals (Chou, Chen, & Chen, 2014), in order to generate high-quality solutions (Vonolfen & Affenzeller, 2016).

Examples of PDP-type services include special buses for elderly and handicapped individuals in which passengers are picked up and assisted during their trip. These services are known in the literature as Paratransit or Dial-a-Ride (Cordeau & Laporte, 2003; 2007). Another well-known example of PDP is the delivery of express courier (Mitrović-Minić & Laporte, 2004). The literature on PDPs is extensive (Parragh, Doerner, & Hartl, 2008).

There are diverse PDP variants, including time window constraints at the origin or destination. Desrosiers, Dumas, and Soumis (1986) solved the single-vehicle problem using dynamic programming. Later, Dumas, Desrosiers, and Soumis (1991) solved the multiple-vehicle problem by proposing an exact algorithm through a column generation scheme. More recently, Ropke and Cordeau (2009) proposed a Branch-and-cut-and-price algorithm that considers two subproblems (whether the routes are elementary or not) for the column generation algorithm. Lower bounds are dynamically added, which improves the performance of the column generation scheme. Baldacci, Bartolini, and Mingozzi (2011a) presented an algorithm based on a set partitioning formulation, which, with effective bounds, can be used to solve real size instances under a branch-and-cut-and-price framework.

As far as we know, the LRP problem with PDP routing has not yet been formulated or solved under an integrated scheme using an exact solution algorithm. However, Karaoglan et al. (2011) and Karaoglan, Altiparmak, Kara, and Dengiz (2012) introduce the simultaneous pickup and delivery (LRP-SPD) problem, which at a certain level integrates location decisions with pickups and

deliveries. However, in the LRP-SPD, depot location decisions are made together with the routing of simultaneous pickups and deliveries that have no correspondence, i.e., all deliveries originate from the depots and pickups must be brought back to a depot. While this last problem could look somehow connected to ours, it is in fact substantially different, because in LRP-SPD pickups and deliveries are not linked together.

3. Model and solution approach

This section presents a mathematical formulation of the LRP-PDP as well as a B&P algorithm that can address it. We also further discuss the branching strategy deployed as well as some implementation details.

3.1. Formulation of the LRP-PDP

The LRP-PDP is an extension of the LRP, and our approach is based on the formulation introduced by Berger et al. (2007). This formulation aims at selecting a set of depots and building routes associated with these depots such that the total cost incurred, which includes both fixed costs for opening depots and routing costs for serving customers, is minimized. In the PDP case, customer request i refers to picking up quantity q_i at pickup location i^+ and dropping this quantity off at delivery location i^-. It should be noted that the pickup and the delivery have to be performed by the same vehicle (i.e., we do not allow transfers between vehicles).

The master problem for the LRP, which is based on Berger et al. (2007, Section 1) is formulated as follows:

Formulation LRPPD

$$\min \quad \alpha \sum_{j\in J} f_j X_j + \sum_{j\in J}\sum_{k\in P_j} c_{jk} Y_{jk} \tag{1}$$

$$s.t.: \quad \sum_{j\in J}\sum_{k\in P_j} a_{ijk} Y_{jk} = 1 \qquad \forall i \in I \tag{2}$$

$$X_j - \sum_{k\in P_j} a_{ijk} Y_{jk} \geq 0 \qquad \forall i \in I,\ \forall j \in J \tag{3}$$

$$X_j \in \{0,1\} \qquad \forall j \in J \tag{4}$$

$$Y_{jk} \in \{0,1\} \qquad \forall j \in J, \forall k \in P_j \tag{5}$$

where J is the set of potential depot locations; f_j, $j \in J$, is the fixed cost of opening depot j; X_j, $j \in J$, is a binary decision variable indicating whether or not depot j is open; P_j is the set of feasible routes associated with depot j; c_{jk}, $j \in J$, $k \in P_j$, is the cost of route k of depot j; Y_{jk}, $j \in J$, $k \in P_j$ is a binary decision variable indicating whether or not route k from depot j is selected; I is the set of customer requests; a_{ijk}, $i \in I$, $j \in J$, $k \in P_j$, takes value one if route k from depot j serves request i; and α is a weight parameter for the fixed cost portion of the objective.

Unlike the formulation of Berger et al. (2007), the routes associated with depots in our formulation satisfy the constraints of the PDP-TW, although this is not directly reflected in the formulation of the LRP-PDP master problem but rather in the construction of the routes themselves. For each route $k \in P_j$, the following conditions are satisfied:

- Route k starts and ends at depot j.
- If request i is served by route k, then the precedence is respected: i^+ (pickup) appears before i^- (delivery) on the route.
- If request i is served by route k, then its time windows at i^+ and i^- are respected.
- The total vehicle load anywhere along the route is always lower or equal to its capacity (Q).

Let us consider $P = \{i^+ : i \in I\}$ as the set of pickups and $D = \{i^- : i \in I\}$ as the set of deliveries. For each node $l \in P \cup D$, its time window is denoted as $[a_l, b_l]$ and the transportation demand in each node is denoted as q_l, with $q_{i^+} = q_i, q_{i^-} = -q_i$. The travel time from vertex l to vertex m is denoted t_{lm}.

It should be noted that the set of constraints 3 can be replaced by

$$X_j - Y_{jk} \geq 0, \forall j \in J, k \in P_j. \tag{3'}$$

However, Berger remark that the aggregate form of the constraints (3) yields a tighter relaxation than the disaggregate form (3′).

The LRP-PDP formulation contains an exponential number of variables (Y_{jk}). Therefore, a complete enumeration of routes is not possible for most instances of practical size. In the following section, we describe a B&P framework to solve this type of problem.

3.2. B&P algorithm for the LRP-PDP

The proposed B&P algorithm is based on the formulation of the master problem presented in the previous section and an adaptation of the subproblem in Ropke and Cordeau (2009).

Thus, the master problem is defined by (1)–(5), and the linear relaxation of this master problem is denoted LPM. The set of routes P_j for each depot is not known, and in the B&P scheme, a restriction of LPM is solved with a set $P'_j \subset P_j$ for each depot j. Each of the sets P'_j is generated independently.

The LPM restriction will be referred to as RPM, where only a subset of the route variables is considered. Let us remember that all the location variables X_j are always considered in the RPM. The updating of route sets P'_j is performed by solving the so-called pricing subproblem.

3.3. Formulation of the pricing subproblem

For the B&P scheme, it is necessary to identify the subproblems associated with the master problem LRP-PDP presented in (1)–(5). In this case, for each depot $j \in J$, we solve an independent subproblem. Each of these problems aims at identifying routes that should be added to the set P'_j. Each of these subproblems is a PDP-TW with capacity constraints, which are similar to the problem presented in Dumas et al. (1991).

To formulate the pricing problem, duality theory for linear programming is used to obtain the reduced cost of a route k associated with depot j as follows:

$$\hat{c}_{jk} := c_{jk} - \sum_{i\in I} a_{ijk}(\pi_i - \mu_{ij}) \tag{6}$$

where π is the vector of dual variables associated with constraints (2) and the dual variables μ_{ij} are associated with constraints (3).

The subproblem used to determine routes that depart from depot j is then formulated as an Elementary Shortest Path with Pickup and Delivery and Time Windows Constraints (ESPPDTWC; see Ropke & Cordeau, 2009). For each depot j, we build a graph $D_j = (V_j, A_j)$ that contains depot j itself and all the customer locations. Assuming that $|P| = |D| = N$, the set of vertices $V_j = P \cup D \cup \{0, 2N+1\}$ where node 0 represents depot j as the starting point, and node $2N+1$ also represents depot j as the arrival point of each route. The set of arcs $A_j = [\{0\} \times P] \cup [(P \cup D) \times (P \cup D)] \cup [D \cup \{2N+1\}]$. In the following, we use the convention that the pickup node of request i is denoted i (instead of i^+) and its delivery node $N+i$ (instead of i^-). Therefore, $P = \{1, \ldots, N\}$ and $D = \{N+1, \ldots, 2N\}$.

It is necessary to transfer the information of the dual variables to the arcs such that the cost of a path in this auxiliary network represents the reduced cost (6) of the corresponding route.

Thus, we define the cost of an arc (l, m) in the subproblem network as follows:

$$\hat{d}_{lm} := \begin{cases} d_{l,m} - \pi_m + \mu_{m,j} & \text{if } l \in V_j, m \in P \\ d_{l,m} & \text{if } l \in V_j, m \in D \cup \{2N+1\} \end{cases} \tag{7}$$

Since the subproblems are independent, an auxiliary network is built for each depot j. Each subproblem is solved as a resource constrained shortest-path problem by a label-setting algorithm, which is described in Section 3.

3.3.1. Preprocessing

In the literature, a variety of preprocessing rules have been suggested for the subproblem network. The present work considers the rules corresponding to extensions of the heuristics proposed in Dumas et al. (1991) and Desrochers, Desrosiers, and Solomon (1992).

The first step consists in adjusting the time windows using the partial paths $0 \to i \to N+i$ and $i \to N+i \to 2N+1$ so that they are feasible for all of the values $T_i \in [a_i, b_i]$ and $T_{N+i} \in [a_{N+i}, b_{N+i}]$. The time windows are successively redefined as follows (see Dumas et al. (1991, Section 2.4)):

$$\begin{aligned} b_{N+i} &= \min\{b_{N+i}, b_{2N+1} - t_{i,2N+1}\} \\ b_i &= \min\{b_i, b_{N+i} - t_{i,N+i}\} \\ a_i &= \max\{a_i, a_0 + t_{0,i}\} \\ a_{N+i} &= \max\{a_{N+i}, a_i + t_{i,N+i}\} \end{aligned}$$

Subsequently, rules number 2 and 3 from Desrochers et al. (1992, Section 6.1) are applied for each $k \in \{1, \ldots, 2N\}$:

$$a_k = \max\left\{a_k, \min\left\{b_k, \min_{(k,j)\in E}\{a_j - t_{k,j}\}\right\}\right\}$$

$$b_k = \min\left\{b_k, \max\left\{a_k, \max_{(i,k)\in E}\{b_i + t_{i,k}\}\right\}\right\}$$

Because of the time windows and precedence constraints, several arcs can be removed since they cannot belong to a feasible solution to the problem. Following Dumas et al. (1991), the constraints of the problem are used to remove the following arcs:

- *[Precedence]* The following arcs are removed because they cannot belong to any feasible solution: $(0, N+i)$; $(N+i, i)$; $(2N+1, 0)$; $(2N+1, i)$; $(i, 2N+1)$; and $(2N+1, N+i)$ for $i = 1, \ldots, N$.
- *[Capacity]* The vehicle capacity can never be exceeded; thus, if $q_i + q_j > Q, i, j \in \{1, \ldots, N\}$, $i \neq j$ the following arcs are removed: (i, j); (j, i); $(i, N+j)$; $(j, N+i)$; $(N+i, N+j)$; and $(N+j, N+i)$.
- *[Time windows]* Each node must be reachable within its respective time window; thus, if $a_i + t_{ij} > b_j, i, j \in \{1, \ldots, 2N\}$, then arc (i, j) is removed.
- *[Time windows together with precedence]* If travel times satisfy the triangular inequality, arcs can be removed if they cannot be part of any path that includes both the pickup as well as the delivery for some customers:
  - Arc $(i, N+j)$ is removed if path $j \to i \to N+j \to N+i$ is not feasible for $t_j = a_j$.
  - Arc $(N+i, j)$ is removed if path $i \to N+i \to j \to N+j$ is not feasible for $t_i = a_i$.
  - Arc (i, j) is removed if path $i \to j \to N+i \to N+j$ is not feasible for $t_i = a_i$.
  - Arc $(N+i, N+j)$ is removed if paths $i \to j \to N+i \to N+j$ and $j \to i \to N+i \to N+j$ are not feasible for $t_i = a_i$ and $t_j = a_j$, respectively.

3.4. Branching strategy

An optimum LPM solution can contain variables with non-integer values. It is important to create an adequate branching strategy that is compatible with the pricing problem. For this, Berger et al. (2007) used a strategy where the shortest route structure is maintained in the pricing problem. The fundamental strategy consists of four branching rules for the two types of variables.

First, branching is performed on the location variables X_j. For these variables, the conventional dichotomous branching is adequate. Fixing $X_j = 1$ enforces the use of depot j and solving the pricing problem for depot j. Fixing $X_j = 0$ is achieved by imposing a value of $Y_{jk} = 0$ for all of the routes $k \in P'_j$ in the RPM and no longer solving the pricing subproblem for depot j.

Second, we follow Ropke and Cordeau (2009) and branch on the total number of vehicles used in the solution, if this number is fractional. In one branch, we force the fleet size to be larger than the rounded up value and in the other one, we limit the fleet size to the rounded down value. Note that using this branching rule involves adding two constraints on the size of the fleet in the master program. Dual variables for these constraints thus need to be added to the reduced cost of routes generated by the subproblem to ensure a proper stopping criterion.

Third, in a similar manner, we branch on the number of vehicles leaving a depot. We select the depot for which the fractional part of the total outflow is closest to 0.5.

Finally, we branch on variables related to specific routing decisions. We formalize the strategy proposed by Dumas et al. (1991). Order variables O_{ij} are defined for $i, j \in P \cup \{0, 2N+1\}$:

- $O_{ij} = 1$ indicates that if i and j are on the route, the first pickup node visited after i is j.
- $O_{ij} = 0$ indicates that the first pickup node visited after i cannot be j.

For a route k that serves n_k customer requests, let $(i_0 = 0, i_1, i_2, \ldots, i_{n_k}, i_{n_k+1} = 2N+1)$ be its sequence of pickup and depot nodes. Then, $(n_k + 2)$ branch-and-bound nodes B_l, $l = 0, \ldots, n_k + 1$, are created. Node B_l, $l = 0, \ldots, n_k$ is defined by the constraints

$$\left(\bigwedge_{m=0}^{l-1} O_{i_m i_{m+1}} = 1\right) \bigwedge \left(O_{i_l i_{l+1}} = 0\right). \tag{8}$$

The final node B_{n_k+1} corresponds to the constraints $\bigwedge_{m=0}^{n_k} O_{i_m i_{m+1}} = 1$.

Example 3.1 (from Dumas et al., 1991)**.** If there is a route variable Y_r that is not integer-valued, with the corresponding route given by $0 \to 1 \to 2 \to N+2 \to N+1 \to 2N+1$. The four branches created are the following:

B_0: $O_{01} = 0$;
B_1: $O_{01} = 1,\ O_{12} = 0$;
B_2: $O_{01} = O_{12} = 1,\ O_{2,2N+1} = 0$;
B_3: $O_{01} = O_{12} = O_{2,2N+1} = 1$.

The branching constraints can be transferred to subproblems by adding an additional component in the labels that represents the last visited pickup node. In the case of several fractional routes, we select one route among the shortest ones.

3.5. Implementation

The enumeration tree is explored according to a depth-first search (DFS) strategy, and the list of open problems is implemented as a stack.

As previously mentioned, if the solution at a node is not integer-valued and the node is not pruned, branching is performed

to create new problems. If there is a fractional location variable, branching is performed on the variable with the value closest to 0.5. The next problem considers the branch of zero value, in which the corresponding depot is not present.

When all of the location variables have integer values, if the total number of vehicles is fractional, branching is performed on this quantity. If the total number of vehicles is integer, but the number of routes departing from any of the depots is fractional, branching is performed on this number for one of the depots where this quantity is closest to 0.5. In both of these cases, the next subproblem to be solved corresponds to the rounding up branch.

When neither of the previous branching rules applies, one of the shortest routes with fractional value is chosen. Then, branching as explained above is performed for this route. The problems created are added to the stack in the order B_{n_k+1} to B_0.

The B&P procedure is stopped when the gap between the bounds is below 0.5%.

4. Label-setting algorithm for the pricing subproblems

Shortest path problems are common in column generation schemes for VRPs. In general, these problems are solved through modified versions of the classical algorithms, such as those of Dijkstra or Bellman (Bellman, 1958; Dijkstra, 1959). The general principle involves associating a label to each partial route and extending the label to indicate the feasibility of the resources (time, freight on the vehicle, etc.) until the best possible path has been found. Dominance rules are used to compare the partial routes that arrive to the same node and exclude certain routes. For each node of the graph, a significant amount of labels must be maintained because each comparison considers the consumed resources.

It is common to avoid calculating the set of optimal routes and to prematurely end the solution procedure when a set of columns with negative reduced costs is determined. Finding the minimum cost path is only necessary in the last iteration of the algorithm to verify that there are no routes with negative reduced cost.

In this work, we have implemented an elementary resource-constrained shortest path algorithm, which is quite close to the elementary shortest path algorithm described in Ropke and Cordeau (2009). We refer the reader to this paper for a detailed description of this procedure. In the following, we will focus on the modifications that we had to make to implement the branching strategy described in Section 2.4.

The first modification is the addition to the labels of an extra component, which indicates the last pickup node of the current partial route (as mentioned in Section 2). The addition of this label component forces a change in the label extension rules. To extend a label L to a node j, the last pickup, say i, performed by label L must be checked. If $j \in D$, meaning that j is a delivery, then the extension is performed normally. If $j \in P \cup \{2N+1\}$, then all of the branching rules ending in j must be verified, including the following three cases:

- If $O_{ij} = 1$ is included in the branching constraints, the label can be extended and the last pickup node is now j.
- If $O_{mj} = 1$ for some $m \neq i$, verify if m has already been visited in the route. If it does, the label cannot be extended; otherwise, extend the label.
- If $O_{ij} = 0$, then the extension to j is not allowed.

Furthermore, if there are no rules ending in j that prevents label extension, then the extension should be performed.

The subproblems do not have to be solved to optimality because we only need to identify some routes with negative reduced costs at each iteration of the column generation scheme. In general, the label-setting algorithm finds routes with negative reduced cost before finishing the comparison of all routes. The heuristics presented in the following paragraphs truncate the label-setting algorithm in different ways.

H1 Restricting the label-setting algorithm to a reduced-size network is a common practice. Constructing networks with 30% or 50% of the nodes according to the best arcs with respect to cost was proposed in Dumas et al. (1991). A variant of this restriction consists of defining a network, where each node is connected to a subset of its neighbors. For our problem, we build for each depot an auxiliary network D_m, in which we keep only the m best arcs with respect to d_{ij} out of each node i. Once this has been applied to each node of the graph, the set of arcs is completed with arcs $(0, i)$, $(i, i+N)$, and $(N+i, 2N+1)$. Based on the results presented in Ropke and Cordeau (2009), we consider $m = 5$ and $m = 10$. If it is not possible to determine negative reduced cost routes in D_5, the process is continued with D_{10}.

H2 We limit the number of unprocessed labels throughout the label setting heuristic. We consider different values for the maximum number of these labels λ. At first λ is set to 500; this value increases to 1000 and 1500 in later stages of the algorithm. If this limit is not reached, the subproblem has been solved exactly. These values of λ were shown to work well on a variety of instances by Baldacci et al. (2011a), as well as by Ropke and Cordeau (2009).

At each pricing stage, we solve a subproblem for each depot. We first apply H1 and if no negative reduced cost column is found, then we apply H2 limiting the total number of labels to $\lambda = 500$. If no route is found, then λ is increased, as indicated above, and H2 is run again for each depot subproblem. If not route is found this way, we then solve the exact subproblem.

5. Computational experiments

5.1. Instances description

The instances that were used for our computational experiments were created in a manner similar to the one used for the instances proposed by Ropke and Cordeau (2009) for the pickup and delivery problem, which were, in turn, based on a modification of the generator of Savelsbergh and Sol (1995a).

The nodes of the network, including depots and customers, are located at the vertices of a 50×50 square grid. The demands are uniformly chosen integers in the interval $[5, Q]$, where Q is the vehicle capacity. In our case, $Q = 15$ for all instances.

The complete planning horizon has $T = 1000$ time units. The time window for pickup i is randomly generated with $a_i \sim U(0, 450)$ and $b_i = a_i + W$, with W fixed for all customers to 30 units. For the delivery $i+N$, the time window is defined by $a_{i+N} \sim a_i + U(20, 380)$ and $b_{i+N} = a_{i+N} + W$.

For all instances, seven potential locations are defined for the depots, with one of these in the center of the square and the others randomly generated in the square according to uniform distributions. The same seven locations are used for all instances.

To determine the locations for customer requests, we arbitrarily defined three non-overlapping rectangles over the square grid. A third of the customer requests was assigned to each rectangle and the coordinates for both the pickup and the delivery points were randomly generated according to uniform distributions within the rectangle. By proceeding in this fashion, we were thus able to create six instances with clustered demands. Among the generated instances, we retained one, which seemed the most challenging, to become the *base instance*.

From this base instance, we built other instances by contracting and expanding distances within each cluster. More precisely, the

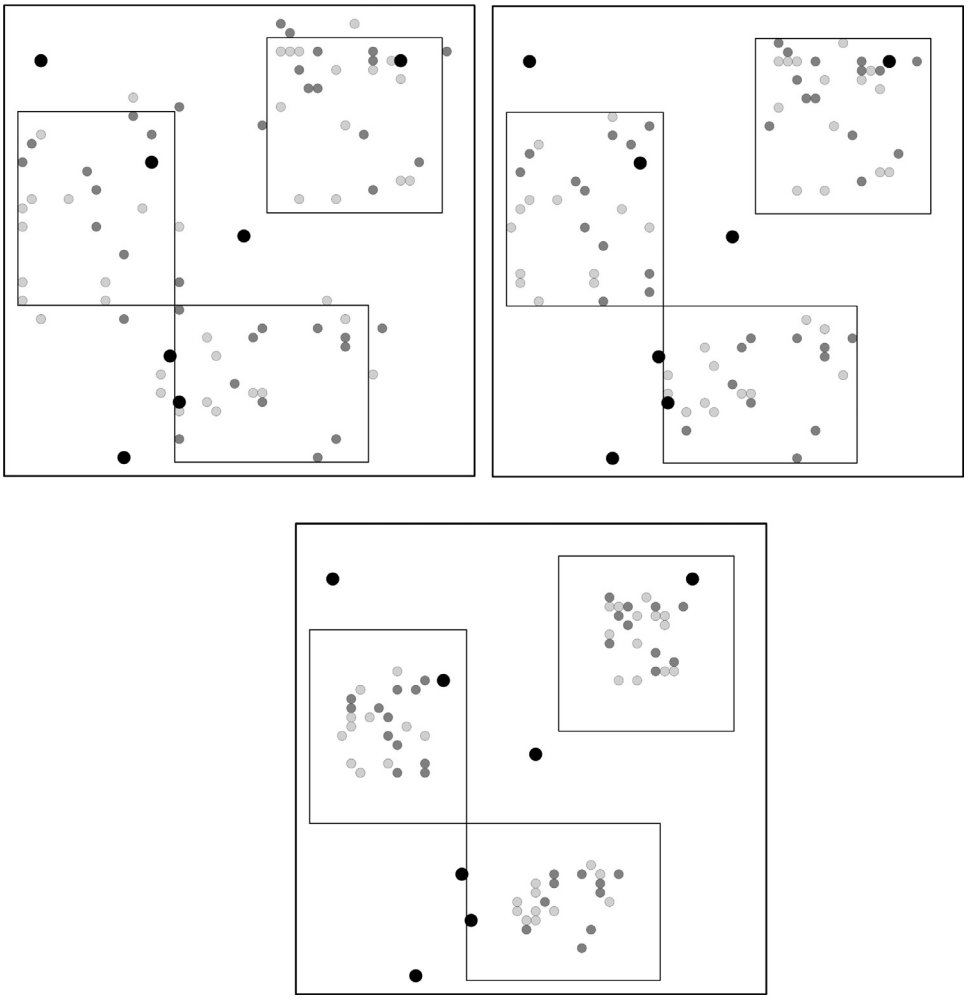

Fig. 1. Location of depots (black nodes) and pickup (light grey) and delivery (dark grey) nodes for configurations 1, 2 (base) and 6.

distance to the center of the corresponding rectangle was multiplied by an inflation factor β for all customer locations.

The following values of β were used: 1.2, 1 (base), 0.87, 0.77, 0.7, 0.55. We tried to generate configurations with larger inflation factors, but we realized that, because of the boundaries of the problem area, those became distorted and did not yield any interesting computational result.

Fig. 1 shows instance 1, which corresponds to the least clustered case (instance 1, factor 1.2); base case (instance 2); and the most clustered case (instance 6; factor 0.55).

Because they were interested in keeping the number of routes to a minimum, Ropke and Cordeau (2009) set a relatively high fixed cost of 10,000 for each route (the routing costs were about 10 times lower than this value). Since, unlike the aforementioned authors, we are interested in the interaction between location costs and routing costs, the fixed costs cannot be arbitrarily high. We considered five different values for the fixed cost for routes: 1, 10, 50, 100, and 200. Similarly, six different values for the depot fixed cost f_j were tested: 0.1, 1, 10, 50, 100, and 200. For each of the six configurations, 30 instances were created by taking all combinations of route and depot fixed costs, thus yielding a total of 180 instances. In the remainder of the paper, we refer to instances by indicating the configuration and the fixed costs as follows: "configuration number-route cost-depot cost". For instance, "3-100-50" is the instance that combines configuration 3 with fixed route cost of 100 and depot cost of 50.

The instances described in the previous subsection cover a wide scope of situations, from instances with highly clustered demand (e.g., instances 5-XX and 6-XX) to other instances with sparse demand (e.g., instances 1-XX).

We also constructed some examples based on the smallest instances described in Ropke and Cordeau (2009). Specifically, we use the location and demands of the instances of classes AA, BB, CC, and DD with 30, 35, and 40 requests and added the same set of depots constructed for our instances. We reduced the original time windows from 60 to 30 units by increasing the lower limits of all time windows by 30 units.

5.2. Computational results

The branch-and-price algorithm was implemented in Python 2.7 by using IBM ILOG CPLEX 12.5. The pricing procedures were implemented in C++ and integrated with the main algorithm as a built-in module of Python. The C/C++ implementation of these components utilizes an interface constructed in C, interacting by means of a wrapper. This wrapper encapsulates the C++ objects that perform the different pricing procedures. All experiments were performed on an Intel i7-5930K computer (3.5 gigahertz) with 16 gigabytes RAM running Ubuntu 14.04.

Computational results are reported in three parts. First, we provide detailed results for 24 selected instances. Aggregated results regarding the performance of the algorithm on the 180 generated instances are reported following that. All generated instances were solved within a gap of 0.6% in a maximum time of 1 hour. Additional results are reported in Section 5.3.

Table 1
Features of instances with depot cost $f_j = 10$ and fixed route cost $r = 1$.

Configuration	z_{IP}	LB	Gap (%)	Nodes B&P	Root time	Total time	Columns	Cols. Root	Cols. Tree	Depots	Routes
1	1019.29	1017.43	0.184	1160	18	2276	4052	2811	1241	4	20
2	910.39	910.39	0.000	0	17	17	2832	2832	0	4	20
3	828.90	828.90	0.000	0	18	19	2864	2864	0	4	20
4	785.98	783.96	0.257	1255	32	1787	3684	2898	786	3	20
5	737.43	737.43	0.000	0	27	27	2907	2907	0	3	19
6	662.28	662.28	0.000	0	25	25	2910	2910	0	3	19

Table 2
Features of instances with depot cost $f_j = 10$ and fixed route cost $r = 10$.

Configuration	z_{IP}	LB	Gap (%)	Nodes B&P	Root time	Total time	Columns	Cols. Root	Cols. Tree	Depots	Routes
1	1199.55	1197.65	0.159	959	35	2369	4292	2839	1453	4	19
2	1083.51	1080.93	0.238	367	28	637	3213	2855	358	4	19
3	1002.51	1001.57	0.093	950	52	2016	3123	2867	256	4	19
4	963.40	958.17	0.545	1035	39	2305	5220	2914	2306	3	19
5	841.15	837.20	0.472	4	288	532	4144	4132	12	4	15
6	831.28	831.28	0.000	0	33	33	2914	2914	0	3	18

Table 3
Features of instances with depot cost $f_j = 10$ and fixed route cost $r = 50$.

Configuration	z_{IP}	LB	Gap (%)	Nodes B&P	Root time	Total time	Columns	Cols. Root	Cols. Tree	Depots	Routes
1	1928.58	1900.73	1.465	763	82	2004	4232	2885	1347	4	16
2	1807.58	1789.03	1.037	1389	84	2798	4330	2937	1393	3	16
3	1709.16	1688.61	1.217	697	75	905	2989	2931	58	4	16
4	1654.19	1632.36	1.337	850	67	1563	3519	2965	554	3	16
5	1576.18	1576.18	0.000	0	71	71	2993	2993	0	3	16
6	1502.00	1502.00	0.000	0	87	87	2976	2976	0	3	16

Table 4
Features of instances with depot cost $f_j = 10$ and fixed route cost $r = 200$.

Configuration	z_{IP}	LB	Gap (%)	Nodes B&P	Root time	Total time	Columns	Cols. Root	Cols. Tree	Depots	Routes
1	4325.72	4287.92	0.882	1250	89	3034	4815	2943	1872	4	16
2	4204.46	4187.46	0.406	754	89	2011	4546	2948	1598	4	16
3	4106.49	4088.61	0.437	845	83	1180	3242	2982	260	4	16
4	4054.19	4032.36	0.541	665	98	1613	3910	3019	891	3	16
5	3924.25	3924.25	0.000	0	111	111	3066	3066	0	3	15
6	3852.04	3852.04	0.000	0	136	136	3044	3044	0	3	15

Table 5
Features of instances with depot cost $f_j = 200$ and fixed route cost $r = 100$.

Configuration	z_{IP}	LB	Gap (%)	Nodes B&P	Root time	Total time	Columns	Cols. Root	Cols. Tree	Depots	Routes
1	3114.26	3106.38	0.254	11	112	976	4594	3002	1592	1	16
2	2995.19	2978.23	0.569	428	119	2280	5252	3039	2213	1	16
3	2896.73	2883.38	0.463	304	119	828	4016	3068	948	1	16
4	2847.47	2837.31	0.358	163	118	445	3416	3109	307	1	16
5	2785.98	2777.53	0.304	2	141	209	3190	3134	56	1	15
6	2702.96	2690.92	0.447	2	160	203	3161	3141	20	1	15

Tables 1–4 report the detailed results for instances with depot fixed cost equal to 10 and route fixed cost equal to 1, 10, 50, and 200. In addition, in Table 5 we show statistics for some cases with depot cost equal to 200 and route cost equal to 100 (both high), because the algorithm generates columns beyond the root node for all maps, even in the most clustered instances. Usually, such instances seem to be easier for the algorithm, as we can see in the results for other cost combinations. In these tables, the first column indicates the configuration of the instances. Columns "z_{IP}", "LB" and "gap" indicate, respectively, the values of the best integer solution, best lower bound and relative gap between these values. Column "Nodes B&P" displays the number of nodes that were created during the branch-and-price process in addition to the root node corresponding to the initial linear relaxation. The two next columns report the execution times (in seconds) for the root node ("Root time") and the complete algorithm ("Total time"). The following four columns describe the total number of route variables added to the master problem ("Columns"), the number of variables that were added to solve the initial linear relaxation ("Cols. Root"), the number of variables in the initial restricted master problem and the number of additional routes generated after branching ("Cols. tree"). The last two columns report the number of open depots ("Depots") and the number of selected routes ("Routes") in the optimal solutions.

From these detailed results, we can can see that the algorithm is working properly, generating in most cases columns in both phases, at the root node first and beyond that as shown in the tables. Most of the instances with low route cost ($r = 1$ as in Table 1) are solved at the root, since in such cases routes tend to be short as they are cheap. Differently, when route costs became higher (Tables 3 and 4) the B&P framework is intensely used, generating many columns beyond the root, as appreciated mainly in more

Table 6
Total number of columns generated after the root node. Averages computed along the different configurations.

		Route fixed cost					
		1	10	50	100	200	Average
Depot fixed cost	0.1	124.17	512.00	614.83	821.17	1079.17	630.27
	1	252.67	441.00	909.83	655.00	757.83	603.27
	10	337.83	730.83	558.67	420.67	770.17	563.63
	50	368.00	498.00	363.83	333.83	647.50	442.23
	100	399.50	733.83	988.83	1087.17	829.50	807.77
	200	241.00	643.00	566.83	856.00	775.50	616.47
	Average	287.19	593.11	667.14	695.64	809.94	610.61

Table 7
Total number of nodes in the enumeration tree. Averages computed along the different configurations.

		Route fixed cost					
		1	10	50	100	200	Average
Depot fixed cost	0.1	305.00	610.67	566.50	647.67	633.33	552.63
	1	339.00	550.33	667.00	661.17	604.83	564.47
	10	402.50	552.50	616.50	515.00	585.67	534.43
	50	528.00	574.00	489.33	410.50	599.33	520.23
	100	124.17	308.33	694.33	386.33	389.50	380.53
	200	144.17	174.67	183.83	151.67	133.00	157.47
	Average	307.14	461.75	536.25	462.06	490.94	451.63

Table 8
Total execution time in seconds. Averages computed along the different configurations.

		Route fixed cost					
		1	10	50	100	200	Average
Depot fixed cost	0.1	482.00	1269.00	1219.50	1468.17	1626.00	1212.93
	1	578.17	1167.67	1482.67	1340.17	1428.83	1199.50
	10	691.83	1315.33	1238.00	1021.67	1347.50	1122.87
	50	865.50	1198.50	950.17	873.33	1333.00	1044.10
	100	421.00	1102.17	1636.50	1272.67	1165.33	1119.53
	200	402.67	946.83	598.33	823.50	761.67	706.60
	Average	573.53	1166.58	1187.53	1133.25	1277.06	1067.59

Table 9
Total execution time in seconds. Averages computed along the different depot fixed costs.

		Route fixed cost					
		1	10	50	100	200	Average
Configuration	1	1407.17	1445.33	1085.67	1107.33	1697.00	1348.50
	2	27.17	683.50	2037.50	1826.17	2521.83	1419.23
	3	24.50	2015.83	1630.33	1368.33	1175.67	1242.93
	4	1903.00	1942.17	2185.17	2238.50	1992.33	2052.23
	5	40.17	865.00	93.83	130.67	137.17	253.37
	6	39.17	47.67	92.67	128.50	138.33	89.27
	Average	573.53	1166.58	1187.53	1133.25	1277.06	1067.59

dispersed configurations (maps 1 and 2). The cases presented in Table 5 are the hardest for the algorithm, generating columns beyond the root in instances 5 and 6, which does not happen in any other case.

Regarding aggregated computational results, on average 3613 variables were generated in total, from which 610 (on average) were generated after the root node. Out of the 3613 routes, 740 were initially included in the set of available routes. These correspond to feasible routes serving one or two customer requests.

We first examine the total number of variables that were generated by the B&P procedure in the tree (beyond the root node) to solve instances depending on the fixed cost of depots and routes. Average values are reported in Table 6.

When examining more closely Table 6, we cannot identify a consistent pattern regarding the relationship between the number of variables after the root and the fixed cost of depots. To the opposite, there is a clear relationship between the number of variables and the fixed cost of routes; problems with higher route fixed cost require significantly more variables after the root, while in cases of low route cost ($r = 1$ for example), all the instances require much less treatment after the root.

Turning to the relationship between the B&B nodes and fixed costs as reported in Table 7, we notice very significant variations in the number of B&B nodes required to solve an instance and the fixed costs. Instances with low route fixed cost are solved very quickly, although most of them perform branching. The situation is quite different with problems with route fixed cost equal to 50, 100 and 200, which require more around 500 B&B nodes on average.

As for the depot fixed cost, problems with larger fixed cost are much easier to solve than those with low fixed cost. Indeed, the

Table 10
Instances from Ropke–Cordeau solved to optimality.

Instance	Requests	TW width	z_{IP}	LB	gap (%)	Nodes B&P	Root time	Total time	Columns	Cols. Root	Cols. Tree	Depots	Routes
AA30	30	30	1294.59	1289.77	0.37	22	200	274	26685	24476	2209	1	4
AA35	35	30	1489.36	1459.29	2.06	241	225	3840	61280	41269	20011	2	4
BB30	30	30	1770.59	1763.45	0.40	54	60	684	6106	5728	378	3	11
BB30	30	45	1818.37	1729.28	5.15	223	92	3834	41050	8479	32571	5	10
BB30	30	60	1707.57	1702.83	0.23	2	105	122	10196	10195	1	6	10

Table 11
Instances from Ropke–Cordeau with no feasible solution.

Instance	Requests	TW width	LB	Nodes B&P	Root time	Total time	Columns	Cols. Root	Cols. Tree
CC30	30	30	1241.72	70	1739	5430	164677	57616	107061
DD30	30	30	1555.13	259	212	3877	56610	30121	26489
AA40	40	30	1667.60	177	270	3885	83822	58942	24880
AA30	30	45	1295.70	370	152	3755	41546	33240	8306
AA35	35	60	1451.01	60	328	3958	88120	58871	29249

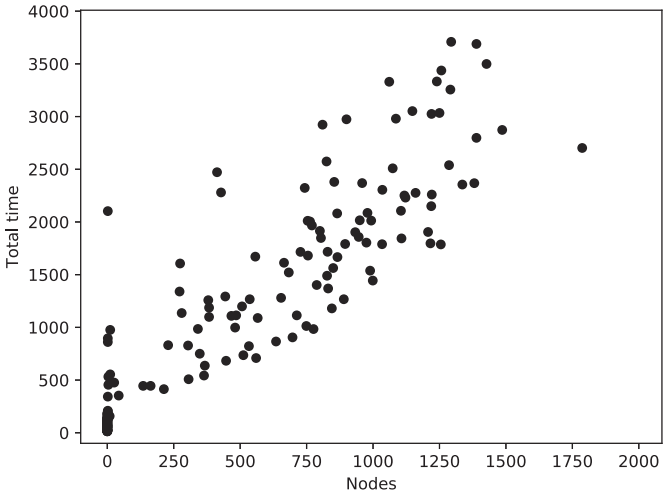

Fig. 2. Scatter plot of the total time vs. the number of nodes in the enumeration tree.

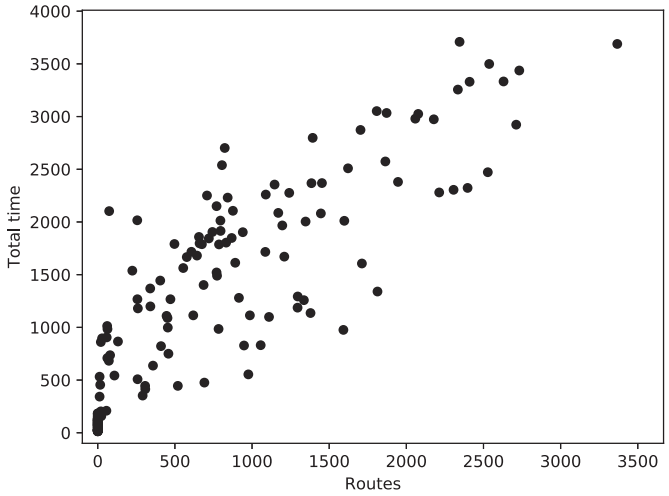

Fig. 3. Scatter plot of the total time vs. the number of routes generated in the tree.

hardest instances are in most cases the one with 0.1 and 1 fixed cost values.

With respect to the connection between the number of B&B nodes and the total number of variables, it seems that in instances with high route fixed cost, the number of columns generated per B&B node is significantly higher. One reason of that is the fact that solutions of instances with high route fixed cost should include long routes as they are expensive, and in those case, we know that the B&P procedure becomes hard as the subproblem turns out to be very complicated.

Examining overall solution times, which are reported in Table 8, they vary in a similar fashion as the number of B&B nodes. The higher the route cost is, the more time is required to solve the B&P procedure. The pattern with respect to depot fixed cost is not as clear as the route fixed cost pattern. In Table 9, we report total execution times but now contrasting by fixed route cost and the different maps, which allows us to identify a special feature of map 4 that makes particularly difficult to solve.

Fig. 2 clearly demonstrates the linear relationship between the solution times of instances and the number of B&B nodes that they require. Fig. 3 displays also a similar relationship between the total execution time and the number of routes generated beyond the root.

Let us now turn to the analysis of the solutions obtained. In all of the instances that we reported upon, the optimal number of open depots ranged from 1 to 5, except one case in which 6 depots are opened. Fig. 4 summarizes the results for the various fixed costs and configurations. As could be expected, solutions for instances in which the fixed costs are high use only 1–2 depots. To the opposite, solutions of most of the instances with depot fixed cost equal to 10 or less use 4 or 5 depots. Regarding these instances, it must be emphasized that the impact on the objective value of the depot fixed cost is almost insignificant; this explains why in some cases one would have 4 open depots while in others, there are 5 or 6 open depots. With respect to the number of routes in the optimal solutions, it varies from 15 to 20. In general, the number of routes is sensitive to the fixed costs, however the limit of 200 was set as results became not interesting beyond this value. We performed additional experiments with higher fixed cost for both routes and depots. In all cases, the solutions obtained were identical to the ones obtained for a fixed cost of 200.

5.3. Benchmark instances

As mentioned in the description of instances, we also modified some of the instances proposed in Ropke and Cordeau (2009) (classes AA, BB, CC, and DD with 30, 35, and 40 requests) by adding the same set of depots constructed for our instances. In addition, we reduced the original time windows from

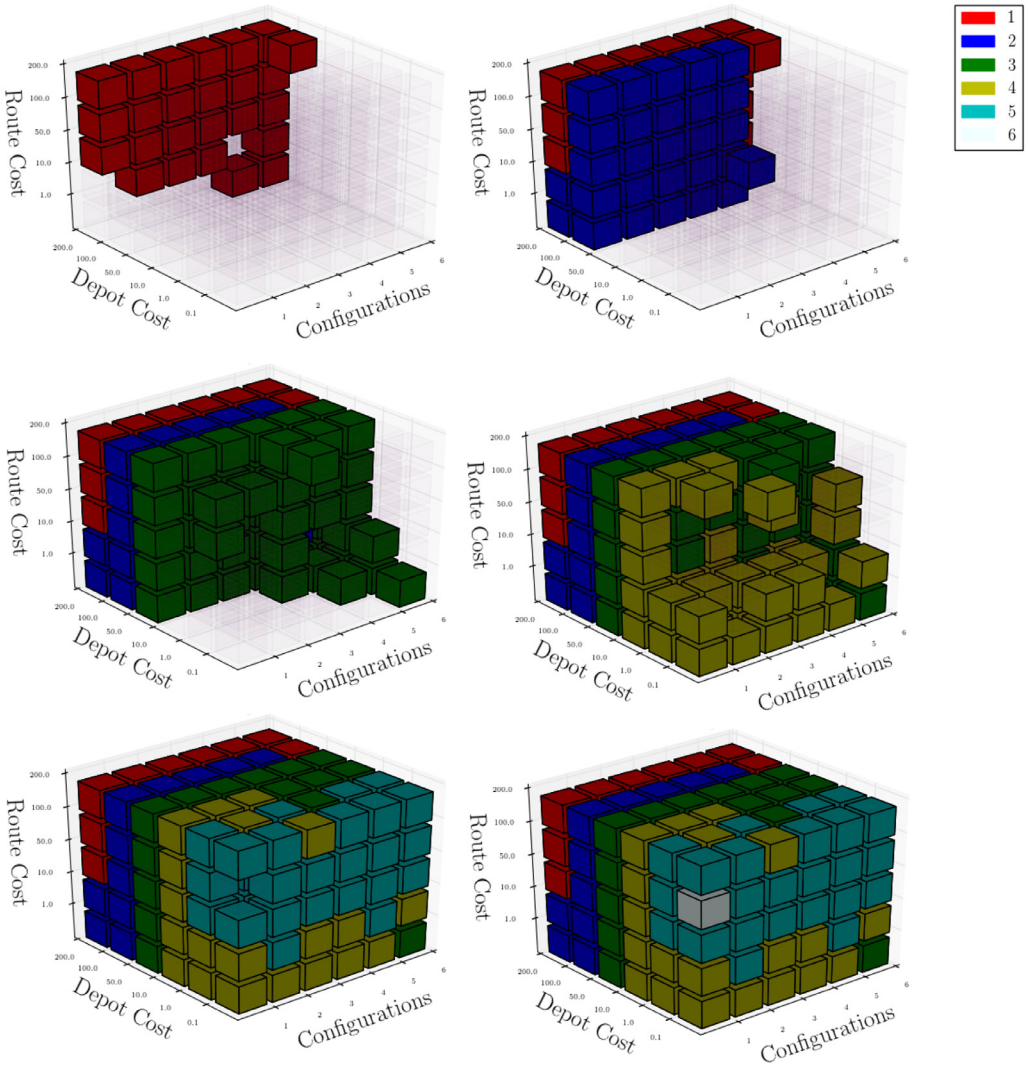

Fig. 4. Number of open depot for instances with different fixed costs and configurations

60 to 30 units by increasing the lower limits of all time windows by 30 units. We were able to solve a subset of such instances to optimality in our integrated scheme (see Table 10). In some cases, we were not able to find a feasible solution, although the algorithm was able to explore the root and tree generating many columns (see Table 11). From these benchmarks, we can determine that our method is working properly, generating many columns and using all the power of the B&P in cases where we know that the instances are difficult for the PDP itself.

6. Conclusions

In this paper, we have formulated an integer programming model that integrates the PDP-TW and optimum depot location decisions (LRP-PDP). We have also proposed an efficient solution method based on column generation within a B&P framework. The solution technique used to solve the pricing subproblem that generates columns has been based on an elementary shortest path procedure as in similar approaches proposed in the literature. This procedure was, however, modified to accommodate the branching strategy of Dumas et al. (1991). The use of this branching strategy in this context is also a contribution of this paper. Computational results on a wide range of instances with different fixed cost values and customer configurations confirm the effectiveness of our proposed solution approach.

Solving much larger instances of the LRP-PDP would probably be difficult with an exact solution scheme, as the one proposed in this paper. To handle such instances, one would have to contemplate defining solution approaches based on metaheuristics or matheuristics ideas.

An interesting extension of the LRP-PDP problem tackled in this paper would involve considering stochastic information on key problem parameters: location of customers, customer demands, fixed cost values, both for depots and routes, and possibly travel times. Obviously, including one or the other of those stochastic dimensions would make the problem much more difficult. Defining a proper mathematical formulation and suitable solution methods would certainly prove to be a significant challenge.

Future work should involve considering how stochastic information could affect both the transportation cost and depot placement. In case of transportation cost, the decisions are mostly operational; therefore, stochastic issues could arise, for example, in the computation of travel costs from uncertainty in traffic conditions, or from uncertainty in demand for pickup and delivery loads. In case of depot placement, the investment decisions are strategic, and therefore uncertainty should be analyzed at a different temporal framework, considering issues such as land cost, rent and accessibility.

Acknowledgments

The authors wish to acknowledge the support of project CONICYT/FONDECYT/REGULAR №1141313, the Complex Engineering Systems Institute (CONICYT - PIA - FB0816) and the Discovery Grant Program of the Canadian Natural Sciences and Engineering Research Council. Finally, the authors thanks Raúl Espinoza for his collaboration in the implementation of the pricing procedures in C++.

References

Ahmadi-Javid, A., & Seddighi, A. H. (2013). A location-routing problem with disruption risk. *Transportation Research Part E: Logistics and Transportation Review, 53*, 63–82.

Archetti, C., & Speranza, M. G. (2008). The split delivery vehicle routing problem: A survey. *The Vehicle Routing Problem: Latest Advances and New Challenges*, 103–122.

Baldacci, R., Bartolini, E., & Mingozzi, A. (2011a). An exact algorithm for the pickup and delivery problem with time windows. *Operations Research, 59*, 414–426.

Baldacci, R., Mingozzi, A., & Wolfler Calvo, R. (2011b). An exact method for the capacitated location-routing problem. *Operations Research, 59*, 1284–1296.

Barreto, S. d. S. (2004). Análise e Modelização de Problemas de localização-distribuição, Departamento de Economia, Gestão e Engenharia Industrial, Universidade de Aveiro. Ph.D. thesis.

Belenguer, J., Benavent, E., Prins, C., Prodhon, C., & Wolfler-Calvo, R. (2011). A branch-and-cut method for the capacitated location-routing problem. *Computers & Operations Research, 38*, 931–941.

Bellman, R. (1958). On a routing problem. *Quarterly of Applied Mathematics, 16*, 87–90.

Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I., & Laporte, G. (2007). Static pickup and delivery problems: a classification scheme and survey. *Top, 15*(1), 1–31.

Berbeglia, G., Cordeau, J.-F., & Laporte, G. (2010). Dynamic pickup and delivery problems. *European Journal of Operational Research, 202*(1), 8–15.

Berger, R., Coullard, C., & Daskin, M. (2007). Location-routing problems with distance constraints. *Transportation Science, 41*, 29–43.

Boudahri, F., Aggoune-Mtalaa, W., Bennekrouf, M., & Sari, Z. (2013). Application of a clustering based location-routing model to a real agri-food supply chain redesign. In *Advanced methods for computational collective intelligence* (pp. 323–331). Springer.

Çetiner, S., Sepil, C., & Süral, H. (2010). Hubbing and routing in postal delivery systems. *Annals of Operations Research, 181*(1), 109–124.

Chan, A. W., & Hearn, D. W. (1977). A rectilinear distance round-trip location problem. *Transportation Science, 11*(2), 107–123.

Chou, Y.-C., Chen, Y.-H., & Chen, H.-M. (2014). Pickup and delivery routing with hub transshipment across flexible time periods for improving dual objectives on workload and waiting time. *Transportation Research Part E: Logistics and Transportation Review, 61*, 98–114.

Christofides, N., & Eilon, S. (1969). Expected distances in distribution problems. *Journal of the Operational Research Society, 20*, 437–443.

Contardo, C., Cordeau, J.-F., & Gendron, B. (2013). An exact algorithm based on cut-and-column generation for the capacitated location-routing problem. *INFORMS Journal on Computing, 26*(1), 88–102.

Contardo, C., Cordeau, J.-F., & Gendron, B. (2014). A grasp+ ilp-based metaheuristic for the capacitated location-routing problem. *Journal of Heuristics, 20*(1), 1–38.

Cordeau, J.-F., & Laporte, G. (2003). The dial-a-ride problem (DARP): Variants, modeling issues and algorithms. *4OR, 1*, 89–101.

Cordeau, J.-F., & Laporte, G. (2007). The dial-a-ride problem: models and algoritms. *Annals of Operations Research, 153*, 29–46.

Derbel, H., Jarboui, B., Hanafi, S., & Chabchoub, H. (2010). An iterated local search for solving a location-routing problem. *Electronic Notes in Discrete Mathematics, 36*, 875–882.

Derbel, H., Jarboui, B., Hanafi, S., & Chabchoub, H. (2012). Genetic algorithm with iterated local search for solving a location-routing problem. *Expert Systems with Applications, 39*(3), 2865–2871.

Desrochers, M., Desrosiers, J., & Solomon, M. (1992). A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research, 40*, 342–354.

Desrosiers, J., Dumas, Y., & Soumis, F. (1986). A dynamic programming solution of the large-scale single-vehicle dial-a-ride problem with time windows. *American Journal of Mathematical and Management Sciences, 6*, 301–325.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik, 1*, 269–271.

Drexl, M., & Schneider, M. (2015). A survey of variants and extensions of the location-routing problem. *European Journal of Operational Research, 241*(2), 283–308.

Drezner, Z., & Wesolowsky, G. (1982). A trajectory approach to the round-trip location problem. *Transportation Science, 16*(1), 56–66.

Dridi, I. H., Kammarti, R., Borne, P., & Ksouri, M. (2011). Multi-objective optimization for the dynamic multi-pickup and delivery problem with time windows. arXiv preprint arXiv:1101.3396.

Dumas, Y., Desrosiers, J., & Soumis, F. (1991). The pickup and delivery problem with time problem. *European Journal of Operational Research, 54*, 7–22.

Ghiani, G., Laporte, G., & Musmanno, R. (2013). Introduction to logistics systems management. John Wiley & Sons.

Hemmelmayr, V. C., Cordeau, J.-F., & Crainic, T. G. (2012). An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. *Computers & Operations Research, 39*(12), 3215–3228.

Karaoglan, I., Altiparmak, F., Kara, I., & Dengiz, B. (2011). A branch and cut algorithm for the location-routing problem with simultaneous pickup and delivery. *European Journal of Operational Research, 211*(2), 318–332.

Karaoglan, I., Altiparmak, F., Kara, I., & Dengiz, B. (2012). The location-routing problem with simultaneous pickup and delivery: Formulations and a heuristic approach. *Omega, 40*, 465–477.

Kolen, A. (1985). The round-trip p-center and covering problem on a tree. *Transportation Science, 19*(3), 222–234.

Laporte, G., Nobert, Y., & Taillefer, S. (1988). Solving a family of multi-depot vehicle routing and location-routing problems. *Transportation Science, 22*, 161–172.

Manzour-al Ajdad, S., Torabi, S. A., & Salhi, S. (2012). A hierarchical algorithm for the planar single-facility location routing problem. *Computers & Operations Research, 39*(2), 461–470.

Mitrović-Minić, S., & Laporte, G. (2004). Waiting strategies for the dynamic pickup and deliery problem with time windows. *Transportation Research Part B: Methodological, 38*, 635–655.

Nagy, G., & Salhi, S. (2007). Location-routing: issues, models and methods. *European Journal of Operational Research, 177*, 649–672.

Parragh, S., Doerner, K., & Hartl, R. (2008). A survey on pickup and delivery problems. *Journal für Betriebswirtschaft, 58*, 21–51.

Prodhon, C., & Prins, C. (2014). A survey of recent research on location-routing problems. *European Journal of Operational Research, 238*, 1–17.

Rieck, J., Ehrenberg, C., & Zimmermann, J. (2014). Many-to-many location-routing with inter-hub transport and multi-commodity pickup-and-delivery. *European Journal of Operational Research, 236*(3), 863–878.

Ropke, S., & Cordeau, J.-F. (2009). Branch and cut and price for the pickup and delivery problem with time windows. *Transportation Science, 43*, 267–286.

Savelsbergh, M., & Sol, M. (1995a). The general pickup and delivery problem. *Transportation Science, 29*, 17–29.

Savelsbergh, M. W., & Sol, M. (1995b). The general pickup and delivery problem. *Transportation Science, 29*(1), 17–29.

Vonolfen, S., & Affenzeller, M. (2016). Distribution of waiting time for dynamic pickup and delivery problems. *Annals of Operations Research, 236*(2), 359–382.

Wasner, M., & Zäpfel, G. (2004). An integrated multi-depot hub-location vehicle routing model for network planning of parcel service. *International Journal of Production Economics, 90*(3), 403–419.

Webb, M. (1968). Cost functions in the location of depot for multiple delivery journeys. *Operational Research Quarterly, 19*, 311–320.