A column generation based heuristic for the multicommodity-ring vehicle routing problem

Paolo Gianessi*a, Laurent Alfantari, Lucas Létocarta, Roberto Wolfler Calvoa

aLIPN, CNRS, (UMR7030), Université Paris 13, Sorbonne Paris Cité, 99, avenue Jean-Baptiste Clément, 93430 Villetaneuse, France
bESSEC Business School, 1, avenue Bernard Hirsch, BP105 95021, Cergy Pontoise, France

Abstract

We study a new routing problem arising in City Logistics. Given a ring connecting a set of urban distribution centers (UDCs) in the outskirts of a city, the problem consists in delivering goods from virtual gates located outside the city to the customers inside of it. Goods are transported from a gate to a UDC, then either go to another UDC before being delivered to customers or are directly shipped from the first UDC. The reverse process occurs for pick-up. Routes are performed by electric vans and may be open. The objective is to find a set of routes that visit each customer and to determine ring and gates-UDC flows so that the total transportation and routing cost is minimized. We solve this problem using a column generation-based heuristic, which is tested over a set of benchmark instances issued from a more strategic location-routing problem.

© 2016 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

Peer-review under responsibility of the organising committee of the 9th International Conference on City Logistics

Keywords: Single-Tier Freight Distribution Systems; Two-Echelon Vehicle Routing Problems; Column Generation; Heuristic Algorithms.

1. Introduction

The Multi-commodity Ring Vehicle Routing Problem (MRVRP) studied in this paper can be considered as belonging to the family of the Multi-level VRPs. Differently from canonical VRPs where the vehicles that visit the final customers start from a central depot, in Multi-level VRPs goods are dispatched to intermediate depots before
reaching their final destination. The motivation to study such complex distribution systems comes mainly from
time. The MRVRP is derived of the so-called Multi-commodity Ring Location Routing Problem (MRLRP) studied in
the framework of a research project called MODUM (see recent publication Gianessi et al. (2015)). The MRLRP is a
strategic planning problem with a high degree of difficulty, due to several decision levels. It consists in locating
Urban Distribution Centers (UDCs) around the city and connect them through a ring over which massive
transportation flows circulate. These massive flows, that may use shuttles for example, are supposed to make it less
costly to use the ring to go to the other side of the city. To estimate the cost of this system, not only the construction
cost of UDCs and ring are considered, but also the transportation cost and the routing cost (both pick-up and delivery
are considered).

In the MRVRP, we skip the strategic location issue and consider only the transportation and routing aspects, i.e.
the ring and distribution centers are already installed. All other aspects of the problem remain the same in every
respect. We briefly describe them here; more details are given in the next section. Goods to be delivered arrive to a
first UDC from gates, are possibly transported along the ring, and finally shipped using electric vans that can get into
the city center; the reverse process occurs for pick-up. No time dependence is considered. Each delivery or pick-up
demand is characterized by a quantity of goods, a customer and a gate. The attribute \textit{multicommodity} in the problem
name refers solely to the different gates. The retail shipments performed by electric vans have both maximum route
length and maximum load limits. The fleet of vehicles is shared among UDCs and Self-service Parking Lots (SPLs)
that are located inside the city. Service routes can be open, i.e. the start node of a route is not necessarily the end
node. Hence a rebalancing policy is imposed to simplify repositioning. The objective is to ship every demand
(delivery or pick-up) from its source to its destination, in such a way as to minimize the overall routing and flow
transportation costs, while respecting UDC and ring arc capacities and the rebalancing constraints. Here, the term
\textit{flow transportation} refers to the circulation of goods from the gates to the ring (using trucks) and along the ring
(possibly using larger trucks or railway shuttles), whereas \textit{routing} refers to the delivery of goods from UDCs to
customers, or to pick-up from customers to UDCs, using electric vans.

This problem is not far from another Multi-level VRP, the \textit{Two-Echelon Vehicle Routing Problem (2E-VRP)}, in
which goods are initially stored at a central warehouse, from where they are delivered to secondary-level logistical
platforms or satellites which correspond to our UDCs. After being consolidated in second-level vehicles, products
can finally be shipped to customers. Split delivery are forbidden at the second level, but allowed at the first level,
therefore a UDC can receive the merchandise it has to deliver from many first-level vehicles. UDCs have a capacity
that bounds the first-level deliveries. Second-level vehicles can only perform one service route and must return at the
depot from where they started. In more general versions, other features can be taken into account, like multiple
warehouses, possibility to deliver customers via first-level vehicles, possibility that second level vehicles perform
more than one service trip, time dependent travel times, customer time windows, synchronization between first- and
second-level. No ring structure has been studied so far for VRP, to our knowledge. The paper Crainic et al. (2009)
studies a widely generalized version of a two-tier distribution structure in which many of these aspects are
considered. Several heuristic approaches to the 2E-VRP can be found in the literature. In Crainic et al. (2008), a set
of two-phase heuristics are proposed. The second-level subproblem is solved as a \textit{Multi-Depot VRP (MDVRP)}, or
alternatively as a set of small VRPs after a clustering of the customers in order to assign them to UDCs. Then, the
first level subproblem is solved as a \text{Capacitated VRP}. In the second phase, a series of heuristics are used to
improve the solution. In Crainic et al. (2011b), the problem is approached in a similar way. A greedy initial
clustering heuristic is used to decompose the problem in as many Capacitated VRPs as the number of UDCs plus one,
i.e. the first-level problem. Then, a \textit{local search (LS)} step changes the customer-UDC assignment so as to improve
the solution. Finally, a multi-start phase is applied for a given number of iterations: the current best solution is
perturbed according to \textit{savings-inspired} criteria, yielding either an infeasible solution, which is then repaired, or a
feasible one. In the latter case, if the quality is promising, the solution in further improved by means of the customer-
UDC assignment improvement LS tool. To mention other heuristic algorithms, we refer the reader for instance to the
\textit{Greedy Randomized Adaptive Search Procedure (GRASP)} with path-relinking of Crainic et al. (2011a) or the
\textit{Adaptive Large Neighborhood Search (ALNS)} proposed in Cordeau et al. (2011).
The literature of exact methods is more scarce. Among the most recent works, the exact algorithm presented in Baldacci et al. (2013) decomposes the 2E-VRP into a set of MDVRP with additional constraints. Valid lower bounds are provided which allow to restrict the search. The algorithm achieves the best computational results to date. We also mention the Branch&Cut approach of Jepsen et al. (2013), the flow model and valid inequalities for the 2E-VRP presented in Gonzalez-Feliu et al. (2008), and the study on valid inequalities for the 2E-VRP of Masoero et al. (2009). Finally, we refer the reader to Gonzalez-Feliu (2011) for a recent survey on two-level distribution systems.

The paper is structured as follows. Section 2 provides the problem statement of the MRVRP and a mixed-integer extended formulation which associates a binary variable with each possible route. Section 3 outlines the Column-Generation-based approach to the MRVRP and the Dynamic Programming procedure for the Pricing Problem. Section 4 describes computational experiments on various instances issued from benchmark MRLRP instances of Gianessi et al. (2015). Finally, section 5 concludes the paper.

2. Problem statement and extended formulation

2.1. Notation

The set of UDCs that compose the ring is noted \( U = \{1 \ldots N\} \) where UDCs are indexed according to their order on the ring. Each distribution center \( u \in U \) is characterized by a cross-docking capacity \( Q_u \). The quantity of goods sent on a link between two consecutive UDCs \((u, u+1)\) on the ring cannot exceed a capacity \( q_{u,u+1} \). We denote as \( A_u = \{(u, u+1), (u, u-1) : u \in U\} \) the set of \( 2|U| \) ring links.

In this problem, we only consider exchanges of goods between the city and other cities, and exclude exchanges within the city. The goods arrive to the city or leave the city via a set of gates denoted by \( K \). Gates are nodes of the transportation network outside the city such that all trucks that arrive to or leave the city necessarily transit through one of them (e.g. crossings of highways). Each gate can send (resp. receive) goods to (resp. from) no more than \( B \) UDCs. A delivery (resp. pick-up) demand \( i \) is composed of a quantity of goods \( q_i \) to be delivered to (resp. collected from) a customer in the city from (resp. to) a gate \( k_i \in K \). We denote by \( D \) and \( P \) the sets of delivery and pick-up demands. Inside the city, a set \( L \) of Self-service Parking Lots (SPLs) can be used to park the electric vans. For delivery, goods are transported from a gate to an UDC \( u \in U \), then to another UDC \( v \in U \) via the ring or stay at the same UDC \( v = u \), and are shipped to customers from UDC \( v \) with delivery routes using electric vans. These routes are open; they necessarily start at some UDC but can end at another UDC or a SPL. The set of all possible delivery routes is noted \( \mathcal{R}^d \). Each route delivers a subset of demands and respects the vehicle capacity \( Q \) and a maximum trip length \( M \). For pick-up it is the reverse process. A pick-up route starts at an UDC or a SPL, collects the demands of a subset of customers and arrives at an UDC. Then the load goes to a gate to leave the city, either directly from this UDC or from a second UDC on the ring. The set of all feasible pick-up routes is denoted by \( \mathcal{R}^p \), and we note \( \mathcal{R} = \mathcal{R}^d \cup \mathcal{R}^p \) the whole set of delivery or pick-up routes. The total load of a route \( r \in \mathcal{R} \) coming from or arriving to gate \( k \in K \) is denoted by \( q_k(r) \). Note that repositioning constraints impose that the number of vehicles arriving at each UDC and each SPL at the end of the day is approximately the same number as at the beginning of the day, more precisely it should remain within an interval \([-\delta^- \ldots \delta^+]\) w.r.t. the initial number, for \( h \in U \cup L \). For modeling the routing costs at the second-level, we need to define a routing graph \( G^d = (U \cup L \cup D, E^d) \) for delivery, and \( G^p = (U \cup L \cup P, E^p) \) for pick-up, where edge sets \( E^d \) and \( E^p \) represent the possible connections between nodes. The routing cost of going from a node \( i \) to a node \( j \) with an electric van is noted \( c_{ij} \). Finally, we introduce some binary coefficients to characterize a route \( r \in \mathcal{R} \):

- \( a_i^r = 1 \iff \) demand \( i \) is served by route \( r \);
- \( e_i^r = 1 \) (resp. \( e_i^{-} = 1 \)) \iff \) route \( r \in \mathcal{R} \) ends (resp. starts) at \( h \in U \cup L \);
- \( b_i^r = 1 \iff \) node \( j \) is visited just after node \( i \) in route \( r \);

and we note the cost of a route \( r \in \mathcal{R}^d \) (resp. \( \mathcal{R}^p \)) as \( c(r) = \sum_{(i,j) \in E^d} b_i^r c_{ij} \) (resp. \( \sum_{(i,j) \in E^p} b_i^r c_{ij} \)).
2.2. Decision variables

The decision variables are the following:

1. binary service variables \(x_{ku} = 1 \iff \text{gate } k \text{ exchanges goods with } u\);
2. binary second-level routing variables \(x_r = 1 \iff \text{route } r \in \mathcal{R} \text{ is selected};\)
3. first-level flow variables, which are all nonnegative and continuous:
   - \(\varphi_{ku} = \text{flow from gate } k \text{ to site } u \text{ (k-outflow)}\);
   - \(\varphi_{uk} = \text{flow from site } u \text{ to gate } k \text{ (k-inflow)}\);
   - \(\varphi_{uk}^{de} = k\text{-outflow on the ring arc } (u, u + 1)\);
   - \(\varphi_{uk}^{pk} = k\text{-inflow on the ring arc } (u, u + 1)\);
   - \(\phi_{ku} = \text{upper bound on the capacity occupied at } u \in U \text{ due to deliveries of } k \in K\);
   - \(\phi_{uk} = \text{upper bound on the capacity occupied at } u \in U \text{ due to pick-ups of } k \in K\).

\(k\)-outflows and \(k\)-inflows are flows of commodity \(k\) for, respectively, delivery and pick-up purposes. They are identified by the \(d\) and \(p\) superscripts.

2.3. MILP extended formulation

The proposed model for the MRVRP is shown in Fig. 1.

The objective function (1) takes into account the routing costs and the costs to transport flows of goods between gates and UDCs and along the ring. Constraints (2) and (3) ensure that for each \(k \in K\) the total \(k\)-outflow and \(k\)-inflow sum up to the overall delivery and pick-up demands in which \(k\) is involved. However, the flows that \(k\) exchanges with each \(u \in U\) is null if \(x_{ku} = 0\), as imposed by (4), and the number of UDCs a gate \(k\) can address is bounded to \(B\) by (5). Constraints (6) and (7) are flow balance equations on each \(u \in U\) and for the \(k\)-inflows and \(k\)-outflows of each commodity. Relations (8) assure that each demand is served by exactly one route \(r \in \mathcal{R}\).

Constraints (9) are needed for rebalancing purposes. Constraints (10) impose the capacity bound of ring arcs, and the same do relations (11)–(13) w.r.t. the capacity of UDCs.

3. A Column Generation-based heuristic for the MRVRP

In this section we present a Column Generation approach based on the MILP model \(\mathcal{M}^{MRVRP}\). This model is in the form of Gilmore and Gomory (see Gilmore and Gomory (1963)), as it contains a class of integer variables, the route variables \(x_r; r \in \mathcal{R}\), whose number \(|\mathcal{R}| = O(|P| + |D|)|\) is exponential in the size of the problem instance. Therefore, it is hard even to find a solution to the LP relaxation of the integer problem. Column Generation (CG) approaches are known to be useful to compute the value of such LP relaxations. The original MILP model with the full column set \(\mathcal{R}\) is called the Master Problem (MP). At each iteration of CG, we have a Restricted Master Problem (RMP) which is the original relaxed model restricted to a limited subset \(\mathcal{R}\) of columns, that can thus be solved by the Simplex method. The variable (route \(r \in \mathcal{R}\)) with minimum reduced cost that enters the basis is computed implicitly, i.e. without enumerating all variables of \(\mathcal{R} \setminus \mathcal{R}\). This can be done by solving a so-called Pricing Problem (PP) based on the values of the dual variables associated with the optimal solution of the RMP. This Pricing Problem, which we will later describe in detail, outputs the least reduced cost variable, hopefully in reasonable time. The column associated with this variable, or a subset of columns with negative reduced cost, is added to the RMP which is solved again. This process is reiterated until no column with negative reduced cost is found by the PP at some iteration. This guarantees to have solved to optimality the LP relaxation of the Master Problem.

Column Generation represents one of the most investigated techniques of the last decades for large-size problems. The reader can refer to: Vanderbeck and Wolsey (2009), Lübbecke and Desrosiers (2002) or Feillet (2010), to have an exhaustive insight to the subject; du Merle et al. (1997), to be introduced to some known CG issues; Létocart et al. (2010), for an example of variable selection strategy.

In this paper we use CG to yield a heuristic algorithm for the MRVRP. Solving the RMP does not require any particular technique, as the model \(\mathcal{M}^{MRVRP}\) has a polynomial number of constraints and apart from \(x_r\) and \(x_{ku}\).
variables it has continuous variables only. Therefore, the description that follows of the implemented CG framework will revolve around the solving of the Pricing Problem.

\[(M^{MRVRP})\]

\[
\begin{align*}
\min & \sum_{r \in R} c(r)x_r + \sum_{k \in K} (c_{ku}q_{ku} + c_{uk}q_{uk}) + \sum_{(u,v) \in A_U} c_{uv}(\phi_{uv}^k + \phi_{uv}^d) \\
\text{s.t.} & \sum_{u \in U} \phi_{ku} = \sum_{i \in D_k} q_i \quad \forall k \in K \quad (2) \\
& \sum_{u \in U} \phi_{uk} = \sum_{i \in F_k} q_i \quad \forall k \in K \quad (3) \\
& \phi_{ku} + \phi_{uk} \leq \chi_{ku} \sum_{i \in P_k \cup D_k} q_i \quad \forall k \in K, u \in U \quad (4) \\
& \sum_{u \in U} \chi_{ku} \leq B \quad \forall k \in K \quad (5) \\
& \phi_{ku} + \phi_{u-1,u}^d + \phi_{u+1,u}^d = \phi_{u,u+1}^d + \phi_{u,u-1}^d + \sum_{r \in R^d} e_{r}^{-u}q_r(r)x_r \quad \forall k \in K, u \in U \quad (6) \\
& \sum_{r \in R^P} e_{r}^{+u}q_r(r)x_r + \phi_{u-1,u}^p + \phi_{u+1,u}^p = \phi_{u,u+1}^p + \phi_{u,u-1}^p + \phi_{uk} \quad \forall k \in K, u \in U \quad (7) \\
& \sum_{r \in R} a_r^i x_r = 1 \quad \forall i \in P \cup D \quad (8) \\
& -\delta^h \leq \sum_{r \in R} e_{r}^{+h}x_r - \sum_{r \in R} e_{r}^{-h}x_r \leq \delta^+ \quad \forall h \in U \cup L \quad (9) \\
& \sum_{k \in K} (\phi_{uv}^k + \phi_{uv}^p) \leq q_{uv} \quad \forall (u,v) \in A_U \quad (10) \\
& \sum_{k \in K} (\phi_{ku} + \phi_{ku} + \phi_{uk} + \phi_{uk}) \leq Q_u \quad \forall u \in U \quad (11) \\
& \phi_{ku} \geq \sum_{r \in R^d} e_{r}^{-u}q_r(r)x_r - \phi_{ku} \quad \forall k \in K, u \in U \quad (12) \\
& \phi_{uk} \geq \sum_{r \in R^P} e_{r}^{+u}q_r(r)x_r - \phi_{uk} \quad \forall k \in K, u \in U \quad (13) \\
& x_r, \chi_{ku} \in \{0,1\}, \phi_{ku}, \phi_{uk}, \phi_{ku}, \phi_{uk}, \phi_{uv}^d, \phi_{uv}^p \geq 0 \quad \forall k \in K, u \in U, r \in R, uv \in A_U \quad (14)
\end{align*}
\]

Fig. 1. The proposed MILP model for the MRVRP.

### 3.1. Computing the reduced cost of a route

The first step is to determine the reduced cost of route variables, before deciding how to solve the PP. In the following we consider a delivery route \( r \in R^d \). We omit to explain the reduced cost of a pick-up route, as it is similar. The reduced cost of the variable \( x_r \) associated with a route \( r \in R^d \) is:
\[ \overline{c}_r = c(r) - \sum_{t \in \mathcal{D}} a_t^i \beta_i^r - \left( \sum_{k \in \mathcal{K}, u \in \mathcal{U}} (a_{ku}^d + y_{ku}^d) q_k(r) e_r^{-u} \right) - \sum_{h \in \mathcal{H}, l \in \mathcal{L}} (e_r^{-h} - e_r^{+h})(\theta_h^* + \theta_h^\circ) \]

Terms \( \beta_i^r, y_{ku}^d, a_{ku}^d, \theta_h^* \) and \( \theta_h^\circ \) are the values (in the current solution of the RMP) of the dual variables associated to the constraints in which \( x_r \) appears, namely: (8), (13), (6), and the two families of inequalities of (9). Each of such variables is multiplied by the coefficient of \( x_r \) in the related constraint and subtracted from \( c(r) \).

Finding a route with minimum reduced cost amounts to solve an Elementary Shortest Path Problem with Resource Constraints (ESPPRC) (because of the vehicle capacity and maximum length constraints) in a graph constructed from \( G^d \), where weights on arcs are simply modified as follows. The weight of arc \((i, j) \in E^d \) is set to \( c_{ij} - \beta_j^* - (a_{ij}^{d_u} + y_{ij}^{d_u}) q_j \) with \( k_j \) the commodity (gate) of demand \( j \in \mathcal{D} \), and \( u \) the start of the delivery route. The weight of arc \((u, j) \) is \( c_{uj} + (\theta_u^* + \theta_u^\circ) - \beta_j^* - (a_{uj}^{d_u} + y_{uj}^{d_u}) q_j \). If \( h \in U \cup L \) is the end of the route, the weight of arc \((i, h) \) is \( c_{ih} - (\theta_h^* + \theta_h^\circ) \). Then an arc with a zero cost connects each \( h \in U \cup L \) to a dummy sink node, to compute shortest paths. We invert indices \( u \) and \( h \) for a pick-up route as it starts at an UDC or an SPL \( h \in U \cup L \) and necessarily ends at some UDC \( u \in U \). Since the reduced cost of a delivery route \( r \in \mathcal{R}^d \) depends on its starting UDC \( u \in U \), we solve \( |U| = N \) Pricing Problems for delivery. Given \( u \in U \), the PP associated with \( u \) is an ESPPRC looking for a minimum-weight (constrained) shortest path from \( u \) to the dummy sink node, where weights are computed as described above. For pick-up, the process is almost the same but solution routes are sought for backward, i.e. from the fixed UDC endpoint \( u \) towards the possible starting points in \( U \cup L \). A subset of \( N_{PP} \) most negative reduced cost paths is added to the RMP. Hence, at each CG iteration we add up to \( 2N_{PP} |U| \) new route variables to the MP.

3.2. Solving the Pricing Problem as an ESPPRC

The ESPPRC on an oriented graph \( G = (V, A) \) consists in finding the shortest path from a source node to a destination node while taking into account the consumption of a set \( \{1 \ldots T\} \) of resources. With each resource \( t \in \{1 \ldots T\} \) are associated both a consumption \( c_{ij}^t \) for each arc \( ij \) and an interval \([a_t^i, b_t^j]\) of allowed values for the consumption level at each node \( v_t \). Triangle inequality is assumed to hold for consumption terms \( c_{ij}^t \), as it holds in the case of MRVRP. A path from the source node to the destination node is feasible if the level of each resource fits the corresponding interval for each of the visited nodes, or can be adapted to it. The time resource is the easiest to understand, the interval and the consumption level being, respectively, a time window and an arrival moment, which can be delayed up to the beginning of the window if needed.

The ESPPRC is NP-hard in the strong sense (see Dror (1994)). The best-known algorithm to solve ESPPRC on instances with positive or negative arc costs is the one presented in Feillet et al. (2004). This algorithm is based on Dynamic Programming (DP) and implicitly enumerates all the possible paths from the source to each node \( v_t \) and associates a state, most often represented by means of a label \( l \), with each of such paths, in order to keep track of:

- The consumption vector \( D_l = \{D_{1l}^1 \ldots D_{1l}^T\} \), with \( D_{tl}^i \) the consumption of resource \( t \) on \( l \);
- The cost \( C(D_l) \) of the path;
- The vector \( U_l = \{U_{1l}^1 \ldots U_{1l}^n\} \) and the number \( \psi_l \in \{1 \ldots n\} \) of unreachable nodes, \( n \equiv |V| \). \( U_{lj}^j = 1 \) denotes that \( j \) is unreachable for label \( l \) associated with a path, i.e. that either it has already been visited by the path, or its insertion as the next node in the path would violate at least one of the resource intervals.

The aim of the unreachable nodes vector \( U \) is twofold: on one hand it ensures to generate only elementary paths, while on the other hand it helps preventing combinatorial explosion. To achieve this second purpose, the introduction of a dominance rule is needed. Given two labels \( l, l' \) of a same node \( i, l \) is said to dominate \( l' \) if:

1. \((\forall t \in \{1 \ldots T\}) D_{tl}^i \leq D_{tl'}^i \quad 2. C(D_l) \leq C(D_{l'}) \quad 3. (\forall j \in V) U_{lj}^j \leq U_{l'}^j \quad 4. l \neq l' \)
Condition 4 means that at least one of the state variables of $l$ must be strictly less than that of $l'$. The redundant label variable $\psi_i$ can help speed up the check of dominance condition 3. Dominated labels are discarded, since only nondominated partial paths can lead to optimal solutions, as the authors of Feillet et al. (2004) have shown.

The above dominance check can be relaxed by simplifying or removing some of the conditions 1-3, for instance by removing condition 3, or by replacing it with $\psi_i \leq \psi_{i'}$, in order to further speed up the DP algorithm. However, the minor accuracy in the check leads to dismiss labels which would not be dominated by a full check, i.e. to possibly have subpaths of optimal paths discarded. Hence, relaxing the dominance rule results in a heuristic DP-based algorithm for ESPPRC. In a Column Generation framework where the PP is an ESPPRC, using such a relaxed dominance rule can prevent some negative reduced cost variable to be found, ultimately leading to nonoptimal solutions of the (R)MP.

The ESPPRC subproblem of the MRVRP accounts for two resources: load and length. In our implementation of the DP algorithm for ESPPRC, the condition 3 of the dominance check is replaced by the condition on the $\psi_i$ variables only, in order to allow a faster solving of the subproblem. When no more negative reduced cost variables can be found, a full dominance check is performed, to ensure the convergence of the CG to the optimal solution of the relaxed MP.

### 3.3. A more in-depth view of the Dynamic Programming Algorithm to solve the Pricing Problem

#### 3.3.1. Completion bound

In order to further restrain the combinatorial explosion, a completion bound method is used. When a new label is created, a lower bound on the cost to arrive to any of the possible ending points is added to its reduced cost: if the result is nonnegative, the label is discarded. The lower bound is given by a so-called q-path and is computed in a preprocessing phase by means of Dynamic Programming. q-paths, which are based on a state-space relaxation, have been widely used in the literature to compute dual bounds in the context of exact algorithms for the CVRP, like for instance in Christofides et al. (1981), which offers a detailed discussion on the subject.

A q-path for the CVRP is a least cost path from the central depot to a customer $i$, with a given total load $q(w) \in W = \{q \leq Q : (\exists c \subseteq C) \sum_{i \in c} q_i = q\}$, where $C$ is the set of customers. Although not necessarily elementary, a q-path is built in such a way as to avoid two-loops, which occur when one customer is visited twice and only one other customer is visited in between. We denote such a q-path as $f_w(i)$, and its cost as $d_w(i)$. $W$ is the ordered set of all possible load values for a vehicle, and $w \in \{1 \ldots |W|\}$. Note that if customer demands are integer, then $|W| \leq Q$ and we have no more than $Q|C|$ q-paths to keep track of.

Suppose we have computed in a preprocessing phase the q-paths associated with each customer and each possible (feasible) load $q(w), w \in W$. Given a label $l$ associated with a partial path, let $i$ be its last visited customer, and $d$ and $q$ its cost and load. If routing costs are symmetric, a lower bound to the cost to complete the path associated with $l$ is given by the least cost q-path $f_w(i)$ s.t. $q(w) \leq Q - q + q_i$. Let:

$$w' = \arg \min_{q(w):q(w) \leq Q - q + q_i} d_w(i)$$

If $d + d_w(i) \geq 0$, label $l$ is discarded as it cannot produce a negative reduced cost path.

Another possible implementation of q-paths, which we used, defines $q(w) \equiv w$ as the number of clients visited so far in the route, with $w \in W = \{1 \ldots \bar{n}\}$, $\bar{n}$ being the maximum number of customers that can be visited in the same route. This allows a further reduction in the number of q-paths per customer.

The q-paths completion bound method is used only when the Pricing algorithm is invoked with a strong dominance level, i.e. when a full dominance check is performed.

#### 3.3.2. ng-paths

Another technique that we have used is offered by ng-paths. An ng-path is a partial path $\varrho$ associated with a set $\Pi(\varrho)$ of nodes that would make it lose its property when added to it as next node. An ng-path is not necessarily simple, whereas an elementary path is always an ng-path. By replacing $\Pi_l$ with $\Pi(\varrho)$, the DP algorithm generates
ng-paths instead of elementary paths, though a small effort runtime tuning of the algorithm can make it generate ng-path which are elementary. The effect is to dramatically reduce the number of partial paths from one of the sources to a customer node $i$ and therefore to further limit the combinatorics in the label generation of the DP algorithm.

ng-paths have been introduced in Baldacci et al. (2011) in the context of an exact method to solve both the CVRP and the VRP with Time Windows (VRPTW). They generalize $q$-paths in that their definition allows to forbid $n$-loops, with $n \geq 2$. An ng-path is defined recursively as follows. Let us define a neighborhood $N_i \subseteq C$ for each customer $i \in C$. Now let $q$ be a path, not necessarily simple, that starts from one of the starting points and visits some clients. Let:

- $\sigma(q)$ denote the last customer visited by $q$,
- $\pi(q)$ be the subpath of $q$ up to the predecessor of $\sigma(q)$, and
- $\Pi(q)$ be the set of all the customers in $q$ (with the exception of $\sigma(q)$) that appear in the neighborhood of all the following customers.

Then $q$ is said to be an ng-path if $\pi(q)$ is an ng-path and $\sigma(q) \notin \Pi(\pi(q))$.

Let us make some important observations. Given an ng-path $q$, $\Pi(q)$ is the set of nodes that would make $q$ lose its property when added to it as next visited customer. It is therefore, in a sense, a set of forbidden nodes. Even more important, since $\Pi(q)$ can contain only nodes in $\pi(q)$, an elementary path is an ng-path by definition.

A small example can help understand how ng-paths work. Let $q = (u, 1, 2, 3, 4, 1)$: we have $\sigma(q) = 1$ and $\pi(q) = (u, 1, 2, 3, 4)$. Suppose that $N_1 = \{3, 4\}$, $N_2 = \{1, 5\}$, $N_3 = \{1, 4\}$ and $N_4 = \{2, 3\}$. We will have:

$$1 \notin N_2 \cap N_3 \cap N_4 \cap N_1, \quad 2 \notin N_3 \cap N_4 \cap N_1, \quad 3 \in N_4 \cap N_1, \quad 4 \in N_1 \Rightarrow \Pi(\pi(q)) = \{3, 4\}$$

and since $\pi(q)$ is elementary – hence an ng-path, and $1 \notin \Pi(\pi(q))$, $q$ is an ng-path.

The example also shows that although a simple path is always an ng-path, the opposite is generally not true, unless ($\forall i \in C$) $N_i = C$. However, it is easy to see that by suitably choosing the node neighborhoods $N_i$, ng-paths offer a very good approximation of elementariness.

We have used ng-paths precisely to exploit this feature. In the DP algorithm to solve the ESPPRC, the state of a path $q$ associated with a label $l$ is given by the load $q$, the cost $d$, and the full vector of unreachable nodes: this means that for each pair $(q, d)$ we can theoretically have $O(2^{|C|})$ labels on a node. By replacing $U_i$ with $\Pi(q) \subseteq N_{\sigma(q)}$, the same DP algorithm generates ng-paths, hence yielding Pareto-optimal ng-paths. The combinatorics is significantly reduced, as for each couple $(q, d)$ we now have no more than $2^{|N_{\sigma(q)}|}$ labels. Of course, the algorithm is prone to generate non-elementary ng-paths and is therefore no more valid for the ESPPRC, but a fine tuning of the neighborhood sets during its execution can hopefully make it converge to a set of optimal elementary paths.

One could wonder whether this strategy leads to a heuristic Pricing Problem, which actually is not the case. Indeed, by using the neighborhood sets and ng-paths instead of the unreachable customers vector we are not relaxing the dominance check: we are performing a full check on the paths of a state-space relaxation. Hence, the resulting paths are not suboptimal: they are optimal w.r.t. a relaxed problem. Therefore, when the output path set is made of elementary paths only (possibly after correcting neighborhood at runtime), we have the optimal solution of the ESPPRC.

3.4. A Column Generation-based heuristic algorithm

The Column Generation procedure previously described gives rise to the CG-based heuristic algorithm CGHEUR. Basically, it consists of:

1. Launching the CG process in order to determine the optimal solution of the linear relaxation of $\mathcal{M}_{MRVRP}$. In order to do so, the model is initialized with the set $\mathcal{R}_{oc}$ of all feasible one-customer routes:
This is required to avoid the infeasibility of the LP optimization in the initial stages of the CG, when only few columns have been generated and the RMP may lack routes to satisfy constraints (8);

2. Restoring the integrality constraint and solving the last RMP via Branch&Bound.

The results obtained with CGHEUR are presented in the next section.

4. Computational experiments

4.1. Instance set

The MRVRP is, to the best of our knowledge, a new problem and no previously generated instances exist. The MRVRP instances that we used to conduct our computational experiments are therefore derived from a subset of the MRLRP instances of Gianessi et al. (2015) and their solutions. Given a MRLRP instance and an optimal solution to it, a MRVRP instance is easily obtained by taking the ring of as the ring of the instance. This has two advantages. First, it ensures that is feasible; second, it yields the optimal solution of as a sub-solution of . This allows to evaluate the quality of a solution returned by a heuristic algorithm for the MRVRP as is the case with CGHEUR. More generally, if is a feasible solution to , the derived instance is assured to be feasible, and the value of the derived solution of is an upper bound to that of the optimal solution of .

We considered a set of 30 MRLRP instances of Gianessi et al. (2015) of various sizes. In this paper, 8 collections of instances are defined: collections galwc01-03 of small-sized instances, galwc04-05 of medium-sized instances, and galwc06-08 of large-sized instances. The instances in a collection are divided into five scenarios: the scenario 0, which features an initial size of sets , , , , and , and four more scenarios (denoted as 1 to 4) in which, one at a time, each of the aforementioned sets is enlarged with additional elements. and have always the same size. Moreover, each scenario accounts for four instances which differ in terms of ring construction and transportation costs. We took the instances, i.e. those with the lowest ring costs, of each of the five scenarios (0...4) of the collections galwc01 to galwc06, thus obtaining the MRVRP instances p01-p03 to p06-4. The authors provide optimal solutions to the instances of collections galwc01-03. As said, this yields an optimal solution to the generated MRVRP instances.

4.2. Analysis of numerical results

We evaluate the quality of the final solution of CGHEUR by comparing its value, denoted as , to the value of the optimal solution when it is available, as it is the case of the small-sized instances p01-p03. The availability of the optimal solution allows to evaluate also the quality of the MILP formulation by comparing to , the linear lower bound at the root node, i.e. of the value of the fractional optimal solution of the last RMP. When the optimal solution is not known a priori, the quality of can be evaluated by comparing to .

The tests have been run on an Intel Core i7-4770 3.4 GHz machine with 7.76 GB RAM, with a time limit of 3600s for instances p01 to p05, and of 7200s for p06 instances. The execution of CGHEUR is never injected with any optimal solution or previously computed upper bound. The parameter is always set to 50.

Table 1 is the key to the results of our tests, which are presented in Tables 2 and 3. The results on the small-sized instances p01-p03, which feature up to 20 delivery and pick-up clients, show the effectiveness of the DP engine for the Pricing Problem, and in general of the Column Generation algorithm, as the gap between the optimal solution and the linear lower bound at the root node is in most of the cases under 2%, with a very good average value of 1.13%. Moreover, the final route set delivered by the Column Generation, which form the last Restricted Master Problem, is very near to the set of optimal columns, as the average gap between the
solution of CGHEUR and the optimal solution is 0.20%. The total running time is in most of the cases under 10 seconds, even though this is not surprising if we consider that the Branch&Bound tree has on average no more than 2300 nodes.

This is not the case for the medium-sized and large-size instances, for which the number #BB of nodes in the Branch&Bound tree grows up to some tens of thousands of nodes, and up to some millions for p06 instances. These instances prove therefore to be computationally challenging, as one could expect since the number of both delivery and pick-up clients grows up to 25-40 for p04-p05 instances and to 50-80 for p06 instances. However, even in these conditions CGHEUR performs very well.

Table 1. Key of the results table.

<table>
<thead>
<tr>
<th>notation</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance</td>
<td>instance identifier</td>
</tr>
<tr>
<td></td>
<td>number of gates, UDCs, SPLs, delivery demands, pick-up demands</td>
</tr>
<tr>
<td></td>
<td>load and length limits on routes</td>
</tr>
<tr>
<td></td>
<td>maximum number of UDC that can be used by a gate</td>
</tr>
<tr>
<td></td>
<td>value of the optimal solution, s*</td>
</tr>
<tr>
<td></td>
<td>value of the fractional optimal solution of the last RMP</td>
</tr>
<tr>
<td></td>
<td>gap of the final solution of CGHEUR w.r.t the optimal solution</td>
</tr>
<tr>
<td></td>
<td>gap of the optimal solution w.r.t. the fractional optimal solution of the last RMP</td>
</tr>
<tr>
<td></td>
<td>gap of the final solution of CGHEUR w.r.t the fractional optimal solution of the last RMP</td>
</tr>
<tr>
<td></td>
<td>total running time of CGHEUR, or gap still to close (in brackets) if the time limit has been exceeded</td>
</tr>
<tr>
<td></td>
<td>time to determine the final solution s*</td>
</tr>
<tr>
<td></td>
<td>number of nodes in the Branch&amp;Bound tree</td>
</tr>
</tbody>
</table>

Table 2. Results of CGHEUR on small-sized instances.

| Instance | | | | | | | | | | |
|----------|--------|---|---|---|---|---|---|---|---|
| p01-0    | 5      | 3 | 5 | 15 | 70 | 50 | 2 | 73562.5 | 73562.5 | 71800.2 | 0.00 | 2.37 | 7.9  | 0.8  | 4404 |
| p01-1    | 5      | 3 | 5 | 20 | 70 | 50 | 2 | 11563.4 | 11563.4 | 114469.0 | 0.00 | 1.02 | 2.1  | 1.7  | 181  |
| p01-2    | 5      | 3 | 10 | 15 | 70 | 50 | 2 | 74360.9 | 74360.9 | 73756.1 | 0.00 | 0.82 | 1.8  | 1.3  | 304  |
| p01-3    | 10     | 3 | 5 | 15 | 70 | 50 | 2 | 72779.7 | 72779.7 | 72185.8 | 0.00 | 0.82 | 2.3  | 1.1  | 496  |
| p01-4    | 5      | 3 | 5 | 15 | 70 | 50 | 2 | 71221.0 | 71188.0 | 69128.7 | 0.05 | 2.98 | 44.1 | 26.5 | 19433 |
| p02-0    | 5      | 3 | 5 | 15 | 70 | 50 | 2 | 70571.2 | 70313.2 | 69024.3 | 0.37 | 1.87 | 7.1  | 0.5  | 4557 |
| p02-1    | 5      | 3 | 5 | 20 | 70 | 50 | 2 | 105226.0 | 104832.0 | 103123.0 | 0.38 | 1.66 | 3.1  | 1.3  | 1050 |
| p02-2    | 5      | 3 | 10 | 15 | 70 | 50 | 2 | 65164.7 | 65029.3 | 64550.3 | 0.21 | 0.74 | 1.8  | 1.2  | 579  |
| p02-3    | 10     | 3 | 5 | 15 | 70 | 50 | 2 | 68669.8 | 68669.8 | 68140.0 | 0.00 | 0.78 | 1.5  | 0.5  | 439  |
| p02-4    | 5      | 3 | 5 | 15 | 70 | 50 | 2 | 64692.7 | 64203.5 | 63143.7 | 0.76 | 1.68 | 3.6  | 0.6  | 1918 |
| p03-0    | 5      | 3 | 5 | 15 | 150 | 65 | 2 | 66931.4 | 66931.4 | 66931.4 | 0.00 | 0.00 | 0.1  | 0.1  | 1    |
| p03-1    | 5      | 3 | 5 | 20 | 150 | 65 | 2 | 96495.8 | 95423.4 | 93691.8 | 1.12 | 1.85 | 4.1  | 1.2  | 801  |
| p03-2    | 5      | 3 | 10 | 15 | 150 | 65 | 2 | 64311.0 | 64311.0 | 64311.0 | 0.00 | 0.00 | 0.3  | 0.3  | 1    |
| p03-3    | 10     | 3 | 5 | 15 | 150 | 65 | 2 | 59398.1 | 59355.1 | 59143.4 | 0.07 | 0.36 | 0.3  | 0.1  | 9    |
| p03-4    | 5      | 3 | 5 | 15 | 150 | 65 | 2 | 67168.8 | 67168.8 | 67153.2 | 0.00 | 0.02 | 0.5  | 0.1  | 3    |
| average  | | | | | | | | | | 0.20 | 1.13 | 5.4  | 2.5  | 2272 |
Table 3. Results of CGHEUR on medium- and large-sized instances.

| Instance | |K| | |U| | |L| | |D| = |P| |q| |M| |B| |z_{LU}| |z_{RL}| |%_{RL}| |T/(%)| |t| |#BB|
|----------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| p04-0    | 5      | 3      | 5      | 25     | 70     | 60     | 2      | 95813.7 | 94553.0 | 1.33    | 3.8    | 1.8    | 829    |
| p04-1    | 5      | 3      | 5      | 40     | 70     | 60     | 2      | 188120.4 | 182656.0 | 2.99    | 597.8  | 30.7   | 221810.0 |
| p04-2    | 5      | 3      | 10     | 25     | 70     | 60     | 2      | 93681.6  | 92822.5  | 0.93    | 2.6    | 2.2    | 348    |
| p04-3    | 10     | 4      | 5      | 25     | 70     | 60     | 2      | 91045.0  | 89672.6  | 1.53    | 6.8    | 4.0    | 1264   |
| p04-4    | 5      | 3      | 5      | 25     | 70     | 60     | 2      | 93804.8  | 92571.9  | 1.33    | 7.9    | 1.6    | 2569   |
| p05-0    | 5      | 3      | 5      | 25     | 150    | 65     | 2      | 81867.0  | 79788.8  | 2.60    | 119.1  | 22.0   | 15485  |
| p05-1    | 5      | 3      | 5      | 40     | 150    | 65     | 2      | 154343.5 | 152528.0 | 1.19    | 306.0  | 179.0  | 7693   |
| p05-2    | 5      | 3      | 10     | 25     | 150    | 65     | 2      | 83928.1  | 82038.1  | 2.30    | 90.6   | 11.3   | 10636  |
| p05-3    | 10     | 3      | 5      | 25     | 150    | 65     | 2      | 88886.4  | 86512.2  | 2.74    | 402.1  | 111.0  | 41608  |
| p05-4    | 5      | 3      | 5      | 25     | 150    | 65     | 2      | 78175.3  | 76598.5  | 2.06    | 25.4   | 3.9    | 3007   |
| average  |       |        |        |        |        |        |        | 1.90     | 156.2    | 36.8    | 30525  |
| p06-0    | 5      | 3      | 10     | 50     | 70     | 50     | 2      | 190864.0 | 187840.0 | 1.61    | 0.48%  | 8.3    | 275929  |
| p06-1    | 5      | 3      | 10     | 80     | 70     | 50     | 2      | 362547.0 | 354465.0 | 2.28    | 1.65%  | 7112.0 | 1252038 |
| p06-2    | 5      | 3      | 15     | 50     | 70     | 50     | 2      | 200318.0 | 196461.0 | 1.96    | 1.31%  | 1970.0 | 1951381 |
| p06-3    | 10     | 3      | 10     | 50     | 70     | 50     | 2      | 187289.0 | 185704.0 | 0.85    | 810.5  | 118.0  | 202057  |
| p06-4    | 5      | 3      | 10     | 50     | 70     | 50     | 2      | 188581.0 | 183757.0 | 2.63    | 4645.5 | 384.0  | 1193773|
| average  |       |        |        |        |        |        |        | 1.87     | 5411.2   | 1918.5  | 1471708 |

Table 3, which reports the results on such instances, only features the gap $\%_{RL}$ as the optimal solution is not available. Such gap is on average 1.9% after one hour for instances with up to 40 pick-up and delivery demands, and less than 1.9% after 7200s for instances with $|D| = |P|$ between 50 and 80. This shows once more the effectiveness of the lower bound at the root node and ultimately of the MILP model $\mathcal{M}_{MRVRP}$. Note that the running time $T$ is in general very good w.r.t. the problem size, even though in some cases the size of the Branch&Bound tree does not allow to close the gap within the time limit. However, for such cases (and in general for most of the instances), we note that in spite of some convergence difficulty, the time $t$ required by CGHEUR to find its best solution is much lower than $T$. This suggests that the time limit could be tightened even more without compromising –in most of the cases– the solution delivered by CGHEUR, thus increasing its average ratio between solution quality and runtime.

5. Conclusion

We presented the Multicommodity-Ring Vehicle Routing Problem (MRVRP), a problem that belongs to the family of Multi-level VRPs. The MRVRP is, as far as we are aware of, a new problem, which arises in City Logistics and concerns a freight distribution system based on a ring of Urban Distribution Centers (UDCs). In order to achieve environmental sustainability purposes, electric vans are used to visit final customers and shipment routes may be open so as to reduce empty miles. Unlike the Multicommodity-Ring Location Routing Problem (MRLRP) proposed in Gianessi et al. (2015) that tackles the strategic planning of the same distribution system, the MRVRP considers that the UDCs and the ring have already been installed and focuses on the tactical-operational decision-making aspects.

In this work, we proposed an extended formulation for the MRVRP and solved it with a heuristic algorithm, CGHEUR, based on Column Generation, in which the Pricing Problem is an Elementary Shortest Path Problem with Resource Constraints (ESPPRC) solved by means of Dynamic Programming. After solving the LP relaxation of the root node, CGHEUR finds the integer solution of the last Restricted Master Problem by Branch&Bound. The algorithm has been tested over a set of 30 instances issued from as many MRLRP benchmark instances.

We believe that the achieved results are very promising and encourage further developments of the Column Generation approach by introducing branching rules in a Branch&Price framework.

References


