The splittable pickup and delivery problem with reloads

H. L. M. Kerivin, M. Lacroix, A. R. Mahjoub*† and A. Quilliot

LIMOS, CNRS UMR 6158,
Université Blaise Pascal - Clermont-Ferrand II
Complexe Scientifique des Cézeaux, 63177 Aubière, Cedex - France
E-mail:{herve.kerivin,ridha.mahjoub}@math.univ-bpclermont.fr
E-mail:{mathieu.lacroix,alain.quilliot}@isima.fr
*Corresponding author
† current address : LAMSADE,CNRS UMR 7024,
Université de Paris-Dauphine
Place du Maréchal de Lattre de Tassigny, 75775 PARIS Cedex 16.
E-mail: mahjoub@lamsade.dauphine.fr

Abstract: In this paper, we consider a variant of the pickup and delivery problem where any demand may be dropped off elsewhere than at its destination, picked up later by the same or another vehicle, and so on until it has reached its destination. We discuss the complexity of this problem and present two mixed-integer linear programming formulations based on a space-time graph. We describe some valid inequalities for the problem along with separation routines. Based on these results, we develop a branch-and-cut algorithm for the problem and present some computational results.

Keywords: Transportation problem; mixed-integer linear program; multicommodity flow; metric inequalities; branch-and-cut.

Reference to this paper should be made as follows: Kerivin, H.L.M., Lacroix M., Mahjoub R. and Quilliot A. (2007) 'The splittable pickup and delivery problem with reloads', *European J. Industrial Engineering*, Vol. x, No. x, pp. x

Biographical notes: Hervé L. M. Kerivin received his PhD in combinatorial optimization from the University Blaise Pascal, Clermont-Ferrand, France, in November 2000. After finishing the PhD, he held a research staff member position at France Telecom Research and Development, Paris, France, for almost two years. In September 2002, he joined the Institute for Mathematics and its Applications (IMA) at the University of Minnesota, Minneapolis, as a two-year postdoctoral associate. Since September 2004, he has been with the Mathematics and Computer Science Department at the University Blaise Pascal as an assistant professor. His major research area is combinatorial optimization, and essentially consists of using polyhedral combinatorics to devise efficient algorithms for solving large-scale optimization problems.
Mathieu Lacroix is a PhD student at the University Blaise Pascal, Clermont-Ferrand, France. His research interests include combinatorial optimization and polyhedral theory with a special emphasize to transportation problems.
A. Ridha Mahjoub is a full professor of operations research and combinatorial optimization at the University Blaise Pascal, Clermont-Ferrand, France. He earned

his thèse d'Etat in operations research from University of Grenoble, France. His research areas include the theory and applications of polyhedral approaches for solving large NP-hard combinatorial optimization problems, integer programming as well as complexity and graph theory. In particular, he is interested in applications to network design.
Alain Quilliot is professor in computer sciences at the University Blaise Pascal of Clermont-Ferrand, France. He is the head of the LIMOS laboratory CNRS, dedicated to computer sciences, modelling and optimization. His own research is mainly devoted to operations research, combinatorial theory, game theory and artificial intelligence.

1 Introduction

Transportation problems mainly consist of carrying some demands (e.g., products, goods, people) from their origin to their destination through a given network. Because of important economic and ecologic stakes, operations research practitioners have been giving more and more attention to these problems. Many variants have then been introduced in order to match as close as possible the real-world applications. One of these variants allows demands to be unloaded and then reloaded in order to get better routes for the vehicles and reduce the total cost. In this paper, we consider such a variant of the transportation problem, called the splittable pickup and delivery problem with reloads.

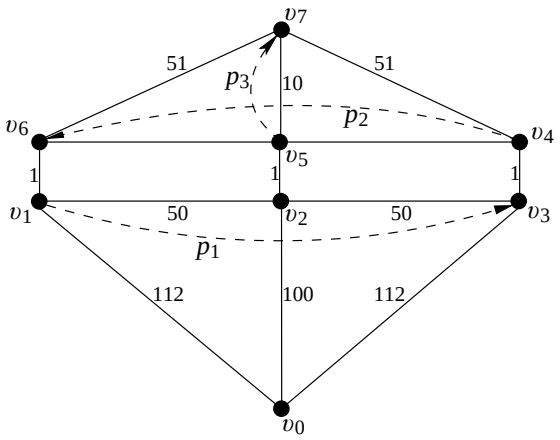
This problem can be stated as follows. Consider a network specified by a set of nodes that are connected with each other by links. Let v_0 be a distinguished node which represents the *depot*. The other nodes are called *stops*. Suppose also that a set of demands is given, each demand being specified by an origin stop, a destination stop and a volume. The demands need to be carried through the network from their origin to their destination using vehicles of a given fleet. The vehicles all have the same transportation capacity and are available at the depot.

Every demand can be unloaded (fully or partially) at any intermediate stop (i.e., a stop different from its destination one), and can then be picked up later by the same or another vehicle. This unloading/picking-up process, called a *reload*, can be repeated several times for a demand until its destination stop is reached. Moreover, every demand can be splitted onto different routes and can be carried by several vehicles. Also there is no restriction on the routes of the vehicles. They can pass by any node or link as many times as necessary.

With each link of the network is associated a cost that corresponds to what must be paid by a vehicle to use the link once. We suppose that the costs satisfy the triangle inequalities and that the reload costs and time are neglected. The *Splittable Pickup and Delivery Problem with Reloads* (SPDPR) then consists of finding the vehicle routes so that all the demands are carried to their destination, a vehicle is never overloaded, and the total cost is minimum.

This problem arises in a wide variety of freight and passenger transportation systems. For instance, this is the case in postal services as described by Grünert and Sebastian (2000). They consider a problem encountered by the Deutch Post AG in its transportation chain which can be presented as follows. Mail is first collected from customers and mailboxes and is then brought to the nearest Letter Mail Center (LMC). After being sorted, mail is

Figure 1 A reload with one vehicle

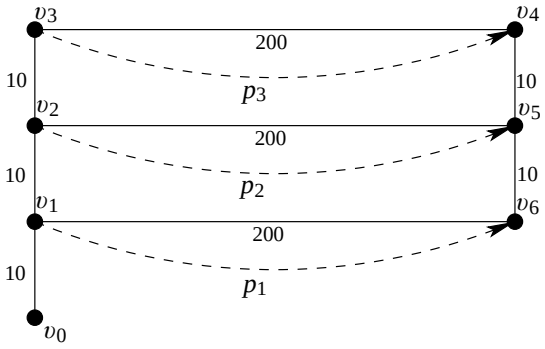


sent from its origin LMCs to its destination ones. This transportation is performed by cars, trucks, aircrafts or railways. Finally, after another sorting stage, mail is delivered to the given addresses by postmen. Here, several optimization problems may be considered. In fact, one has to deal with a capacitated vehicle routing problem (introduced by Dantzig and Ramser (1959)), a ground problem which is similar to the SPDPR and an arc routing problem (e.g., Chinese postman problem (Gendreau and Laporte, 1995)).

The SPDPR can be seen as a relaxation of the standard Pickup and Delivery Problem (PDP). In fact in the PDP, reloads are not allowed, only unsplittable flows for the demands are permitted and vehicle routes have to be vertex-disjoint simple walks. (See (Savelsberg and Sol, 1995) for a thorough description of the PDP.) As illustrated by the two following examples, considering the SPDPR instead of the PDP may lead to important savings. The first example shows that allowing reloads tends to bypass the limited vehicle capacity whereas in the second one, transporting a demand onto different paths permits to reduce the non-used transportation capacity of the vehicle. In both examples, each edge represents two opposite links and even if all links are not shown, we consider a fully-meshed network. The numbers correspond to the costs associated with the links and we can deduce the cost of the missing links by computing the shortest paths between their endnodes with the costs provided in the figures. The demands are represented by dashed arcs and there is only one vehicle available at depot v_0.

We first consider the network given in Figure 1. Suppose that we have to carry three demands p_1, p_2 and p_3. The volume of each demand is equal to the transportation capacity of the vehicle. An optimal solution of the PDP consists of first making the vehicle go to v_5 where it loads p_3 and bring it to its destination v_7. Then the vehicle goes to v_4 where it loads p_2, conveys it to v_6 through (v_4, v_6) and then passes by (v_6, v_1) and arrives at v_1 to start the transportation of p_1. Once it has unloaded p_1 at its destination v_3, it goes directly back to the depot. The associated cost is then 475. When reloads are allowed, an optimal solution can be as follows. The vehicle starts its route by going to v_1 where it loads p_1. It carries it from v_1 to v_3 through (v_1, v_3) and then goes to v_4 to start the transportation

Figure 2 Demand carried on different paths



of p_2. It then arrives at stop v_5 where it unloads p_2 and loads p_3 instead. It then carries p_3 from v_5 to v_7 and goes back to v_5 using arc (v_7, v_5). It continues its trip by reloading p_2 and conveying it to v_6, and ends by going back to v_0. The associated cost is then 446.

In the second example shown in Figure 2, we consider a vehicle with a transportation capacity equal to 3 as well as three demands, each one having a volume equal to 2. In an optimal solution of the PDP, the vehicle first goes to v_1 where it loads p_1. It conveys p_1 from v_1 to v_6 and then goes to v_2 to load p_2. It carries p_2 to its destination and then goes to v_3 to transport the last demand. It loads p_3 at v_3, goes to v_4 and terminates at the depot. The associated cost is equal to 1260. For the SPDPR, we obtain a cheaper solution. The vehicle starts by passing by v_1 where it loads p_1. It then goes to v_2 where it takes one unit of p_2. At this point, the vehicle is full-loaded. It then goes to v_5 where it unloads the unit of p_2 and goes to v_6 where it unloads the two units of p_1. It then returns to v_2 to achieve the load of p_2 and pass by v_3 to also load p_3. It then conveys p_3 from v_3 to v_4 and goes to v_5 to end the transportation of p_2. The vehicle then returns to the depot. The associated cost is 880.

Although the SPDPR is a new problem, it is close to other optimization problems. In what follows, we present some problems that are similar to the SPDPR.

Oertel (2000) studied a variant of the PDP where reloads are permitted on certain nodes called *hubs*. This latter implies that vehicle routes are no more vertex-disjoint simple walks. Indeed, when a demand is unloaded at a hub, it is then reloaded by the same vehicle (in this case, its trip is not a simple walk) or by another one (the two routes are no more vertex-disjoint.) Oertel used an auxiliary graph in which every hub is splitted into several vertices to ensure that vehicle routes are vertex-disjoint simple walks. He developed a mixed-integer linear programming model based on this auxiliary graph. He also considered time windows constraints. The model was then solved using a tabu-search algorithm. Instances with up to 40 demands were solved efficiently with this method.

Grünert and Sebastian (2000) modeled the ground transportation problem which arises in the transportation chain of the Deutch Post AG that was previously described. This problem is known as the Vehicle and Request Flow Network Design Problem (VRFNDP). This

problem is closely related to the SPDPR. However, the reloads can only be made on certain nodes (i.e., hubs). Moreover, each time a vehicle passes by a hub, it automatically unloads all the carried demand. This demand is then unavailable for some time that corresponds to the time needed to sort the demand. Finally, strong time windows constraints are taken into account. Grünert and Sebastian considered a discrete model based on time periods. They used a space-time graph by creating two nodes, one for the pickup action and the other for the delivery one, for all physical locations at each period. They gave a mixed-integer linear programming formulation which is based on two types of commodity flows: the vehicles and demands. They proposed a modelization for this problem. The VRFNDP differs from the SPDPR for several reasons. The first problem imposes that each vehicle has to be unloaded whenever it reaches a hub and considers sorting time and cost at each hub. It also considers different vehicle types, each one with specific depot and capacity. It also takes into account driving costs that are proportionnal to the carried load and storage cost at any hub.

The SPDPR can also be considered as a constrained version of the Minimum-Cost Capacity Installation for Multicommodity network flows (MCCI) studied by Bienstock et al. (1995). In fact, transportation capacity of vehicles can always be brought back to one (by dividing any demand volume by this capacity) since all the vehicles have the same capacity. The SPDPR is then a version of the MCCI in which the chosen capacities on arcs have to respect additional constraints in order to form feasible routes for the vehicles.

In the next section, we prove that the SPDPR is NP-hard. We then give in Section 3 two mixed-integer linear programming formulations for the problem which are based on an auxiliary graph. In section 4, we present some valid inequalities which may be added in order to strengthen the associated linear relaxations and we study their separation problems in Section 5. We finally describe a branch-and-cut algorithm for solving these two models and present some computational results.

The rest of this section is devoted to more definitions and notation. The graphs we consider are finite, directed, loopless and connected. Let $G = (V, A)$ be a graph where V corresponds to the set of vertices and A to the set of arcs. We denote by n its number of vertices, that is, $n = |V|$. An arc from vertex u to vertex v will be denoted by (u, v) and we will call v its head and u its tail. For $W \subset V$ with $W \neq \emptyset$, we set $\overline{W} = V \setminus W$ and we denote by $\delta^+(W)$ (resp. $\delta^-(W)$) the set of arcs having their tail (resp. head) in W and their head (resp. tail) in $\overline{W}$. The union of these two arc sets is denoted by $\delta(W) = \delta^-(W) \cup \delta^+(W)$. If $W = \{u\}$, we will write $\delta^+(u)$ (resp. $\delta^-(u)$, $\delta(u)$) instead of $\delta^+(\{u\})$ (resp. $\delta^-(\{u\})$, $\delta(\{u\})$). If X is a set, $z \in \mathbb{R}^X$ a vector indexed on the elements of X and $X' \subseteq X$ a subset of X, then we write $z(X')$ for $\sum_{x \in X'} z(x)$.

2 Complexity of the SPDPR

Some notation and a more formal definition of the SPDPR are required at this point. To represent the network, we consider a directed graph $G = (V, A)$, called *initial graph*. Remark that V contains the vertex v_0 which represents the depot of the vehicles. For each arc $a \in A$, let $c_a \in \mathbb{R}_+$ be the cost for a vehicle to use the associated link, and let $l_a \in \mathbb{Z}_+$

the time a vehicle needs to go through a. We suppose that the costs satisfy the triangle inequalities, that is, for all $(i, j), (j, k), (i, k) \in A$, $c_{(i,j)} + c_{(j,k)} \geq c_{(i,k)}$. We denote by F the fleet of vehicles and by $B \in \mathbb{Z}_+$ their transportation capacity. Let P be the set of demands that have to be carried through the network. With every demand $p \in P$, we associate a triplet (o^p, d^p, q^p) where $o^p \in V \setminus \{v_0\}$ represents its origin stop, $d^p \in V \setminus \{v_0, o^p\}$ its destination stop and $q^p \in \mathbb{Q}_+$ its volume.

Even though the SPDPR is a relaxation of the PDP, it remains an NP-hard problem as stated in the following proposition.

Proposition 1 *The splittable pickup and delivery problem with reloads is NP-hard, even if the initial graph is complete, the costs satisfy the triangle inequalities and the fleet is composed of only one vehicle.*

Proof: To prove the statement, we use a reduction from the Hamiltonian circuit problem in a directed graph. Given a directed graph $G = (V, A)$ with $V = \{v_1, \ldots, v_n\}$ (so $|V| = n$), we construct an instance of the SPDPR as follows. Let $G' = (V', A')$ be the complete directed graph induced by $V' = V \cup \{v_0\}$, where v_0 serves as the depot. We set $P = \{(v_1, v_i, 1) \; ; \; i = 2, .., n\} \cup \{(v_i, v_1, 1) \; ; \; i = 2, .., n\}$. There is only one vehicle with a transportation capacity equal to the sum of the volumes of the demands. The costs on the arcs of G' are defined as follows. For all arc $(i, j) \in A' \setminus \delta(v_0)$, we set $c_{(i,j)} = 1$ if $(i, j) \in A$ and $c_{(i,j)}$ is equal to the shortest path (in terms of hops) in G from i to j if $(i, j) \notin A$. For any arc $(i, j) \in \delta(v_0)$, we set $c_{(i,j)} = n$. It is clear that in this case, the costs of A' satisfy the triangle inequalities and for any arc $a \in A' \setminus A$, $c_a > 1$. We now prove that the instance (G', P) of the SPDPR has a solution of cost less than or equal to $3n$ if and only if G contains a Hamiltonian circuit.

Given a Hamiltonian circuit of G, we can obtain a solution to the SPDPR. We can construct a circuit in G' by adding to the Hamiltonian circuit of G the two arcs (v_0, v_1) and (v_1, v_0). This circuit in G' can be considered as the route of the vehicle. Moreover, it obviously implies a feasible solution to the SPDPR since the capacity of the vehicle is equal to the whole demand volume. The cost associated with this solution corresponds to the sum of the costs of the arcs of the circuit in G', that is, $3n$.

To prove the reverse, assume that we have a solution to the SPDPR having a cost less than or equal to $3n$. Without loss of generality, we suppose that there is no reloads in the solution and all the demands are carried on a single path. (This assumption can be done because of the property of the vehicle capacity.) From the definition of the demand set P, the vehicle has to pass by the vertices $v_2, \ldots, v_n$ at least once and by v_1 at least twice. Therefore, the vehicle starts its route from v_0, goes through a walk containing a circuit spanning the vertices of V and having at least n arcs of $A' \setminus \delta(v_0)$, and comes back to v_0. Since the vehicle uses two arcs incident with v_0, the cost of the path in $A' \setminus \delta(v_0)$ cannot exceed n. Moreover, all the arcs not in A have costs greater than 1. Therefore, the solution to the SPDPR exists only if the path in $A' \setminus \delta(v_0)$ is a circuit passing by each vertex of V exactly once and using only arcs of cost 1, that is, there exists a Hamiltonian circuit in G. □

On the other hand, if there is only one demand, the SPDPR can then be solved in polynomial time since it can be reduced to a sequence of shortest paths.

One should also remark that if a bound on the time, say T, is given, knowing if the problem contains a solution of total time less than or equal to T, that is the feasibility

problem of the SPDPR with completion time limit, is also NP-hard. The proof is similar to the previous one. The only difference is that costs are replaced by duration times. (Remark that since we only consider one vehicle in the proof, the total duration time of the solution is equal to the sum of the duration time of the arcs used by the vehicle.)

3 Mixed-integer linear programming formulations

Any solution to the SPDPR has to satisfy what we call *precedence conditions*. In fact, because of the reload policy, part of a demand $p \in P$ may be dropped off at a stop $v \in V \setminus \{v_0\}$ that is different from its destination d^p and then picked up later. Therefore, the vehicle that carries this part of p on the leg started at v has to pass by v once the part of the demand is arrived at v, that is, after the leg ended at v is completed. To handle the precedence conditions, we consider an auxiliary directed graph that is based on a space-time graph. (Similar space-time graphs can be found in (Ahuja et al., 1993) and in (Grunert and Sebastian, 2000).) In order to have a finite space-time graph, we then need to be given a completion time limit $T \in \mathbb{Z}_+$ that corresponds to the latest time any demand of P arrives at its destination stop. (Remark that T can be as big as necessary to keep the initial solution space.)

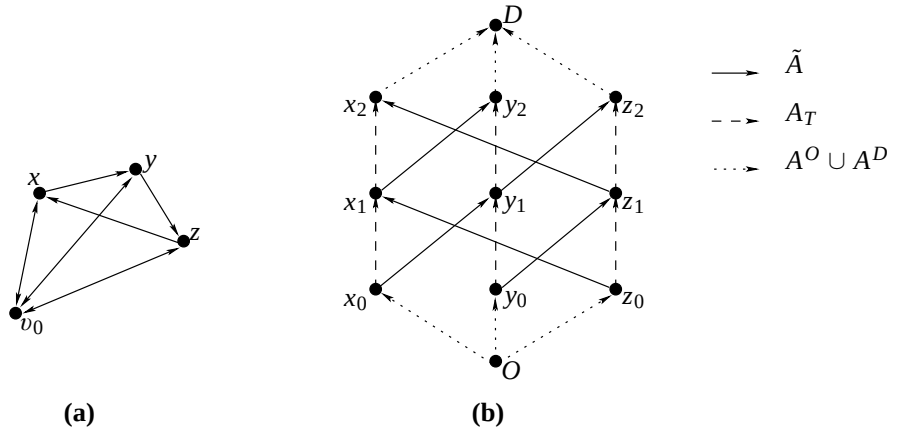
The time component is introduced because of the reloads in order to ensure that all the precedence conditions are satisfied. If reloads are not permitted, any node of the network would be visited exactly once. In fact without loss of generality, we can consider a complete directed graph, a cost function satisfying the triangle inequalities and only vertices that are incident with a demand. In this case, the problem can be formulated on the initial graph by associating with every vertex a time variable which corresponds to the departure time of the vehicle from the vertex. The remaining precedence conditions, which impose that the origin of a demand is visited before its destination by the vehicle, can then be formulated by forcing the departure time, associated with the destination of any demand, to be greater than the one associated with the origin of the demand.

3.1 Auxiliary graph

This subsection is devoted to the construction of an auxiliary graph, denoted by $G' = (V', A')$, from the initial graph G. Since no demand is incident with the depot, no reload is allowed at the depot and the cost function satisfies the triangle inequalities, it is straightforward to see that the vehicles do not pass by the depot unless when they start or end their trip. So each vehicle passes by v_0 at most twice. This leads us to separate the depot from the other vertices in the construction of the auxiliary graph.

For each vertex $u \in V \setminus \{v_0\}$, we associate $T+1$ vertices $u_0, u_1, \ldots, u_T$ in G'. The vertex u_t represents $u \in V \setminus \{v_0\}$ at time $t \in \{0, \ldots, T\}$. (Note that time is considered discrete.) We denote this vertex set V_{st}. We then consider a first arc set of G', $A_T = \{(u_t, u_{t+1}) \mid u \in V \setminus \{v_0\},\ t \in \{0, \ldots, T-1\}\}$. An arc of A_T corresponds to a vehicle or a demand that stays at a node for one time unit. We also consider a second arc set $\tilde{A} = \{(u_t, w_{t+l_{(u,w)}}) \mid (u, w) \in A \setminus \delta(v_0),\ t \in \{0, \ldots, T - l_{(u,w)}\}\}$. An arc $a = (u_t, w_{t'}) \in \tilde{A}$ corresponds to a vehicle that goes from vertex u at time t to vertex w at time t' with $t' = t + l_{(u,w)}$. The arcs in A_T have a zero cost (since we assume that it does not cost anything to stay at a network node) whereas an arc $a = (u_t, w_{t'}) \in \tilde{A}$ has its cost equal to

Figure 3 Intial graph and its associated auxiliary graph



$c_{(u,w)}$.

We add two additional vertices O and D that represent the origin and the destination of the vehicle routes respectively. These two vertices actually correspond to the depot in the initial graph. Without loss of generality, we force all the vehicles of the fleet to start their routes at time 0 and finish it at time T. (Remark that a vehicle that is not used for transportation can reach D (from O) using only arcs of A_T.) For all vertices $u \in V \setminus \{v_0\}$, we then add arcs (O, u_0) and (u_T, D) both having the same cost and duration time that are equal to those of the arcs (v_0, u) and (u, v_0) in G. Let $A^O = \{(O, u_0) \mid u \in V \setminus \{v_0\}\}$ and $A^D = \{(u_T, D) \mid u \in V \setminus \{v_0\}\}$. To conclude, we have built an auxiliary graph $G' = (V', A')$ with $V' = V_{st} \cup \{O, D\}$ and $A' = A_T \cup \tilde{A} \cup A^O \cup A^D$.

Figure 3 illustrates the construction of the auxiliary graph G' associated with the initial graph $G = (V, A)$ given in Figure 3(a). In this example, we suppose that all the arcs $a \in A$ have a duration time l_a equal to one and T equals two. Starting with an initial graph having 4 vertices and 9 arcs (note that some arcs are bidirected), we end up with an auxiliary graph that has 11 vertices and 18 arcs.

3.2 Multicommodity flow based formulation

In our first formulation, we consider a multicommodity flow to represent the routing of the demands in the network and a flow to model the routes of the vehicles. We then consider two types of variables that are:

- $y \in \mathbb{Z}_+^{|A'|}$ where y_a represents the number of vehicles passing on arc $a \in A'$,
- $x \in \mathbb{R}_+^{|\tilde{A} \cup A_T| \times |P|}$ where x_a^p represents the amount of demand $p \in P$ carried on the arc $a \in \tilde{A} \cup A_T$.

For a vertex u_t with $u \in V \setminus \{v_0\}$ and $t \in \{0, \ldots, T\}$, and a demand $p \in P$, the number $b_{u_t}^p$ defined by

$$b_{u_t}^p = \begin{cases} q^p & \text{if} \quad u = o^p \text{ and } t = 0, \\ -q^p & \text{if} \quad u = d^p \text{ and } t = T, \\ 0 & \text{otherwise,} \end{cases}$$

represents the supply/demand associated with vertex u_t with respect to demand p. We remark that for a given $p \in P$, there are exactly two vertices of V_{st} for which $b_{u_t}^p$ is not null. These two vertices are associated with the origin node of p at time 0 and the destination one at time T. The correctness of the definition of $b_{u_t}^p$ is implied by the equivalence between demand p (or part of it) staying for some time at a node of the network and a path using only arcs of A_T in G'.

The SPDPR can be formulated as a mixed-integer linear program using an arc-vertex based approach (Ahuja et al., 1993) as follows.

$$\min \sum_{a \in A'} c_a y_a$$

s.t.

$$\sum_{a \in A^O} y_a \leq |F|, \tag{1}$$

$$\sum_{a \in \delta^+(v)} y_a - \sum_{a \in \delta^-(v)} y_a = 0 \qquad \forall v \in V_{st}, \tag{2}$$

$$\sum_{a \in \delta^+(v) \setminus A^D} x_a^p - \sum_{a \in \delta^-(v) \setminus A^O} x_a^p = b_v^p \qquad \forall p \in P, \ \forall v \in V_{st}, \tag{3}$$

$$\sum_{p \in P} x_a^p - B y_a \leq 0 \qquad \forall a \in \tilde{A}, \tag{4}$$

$$x_a^p \geq 0 \qquad \forall a \in \tilde{A} \cup A_T, \ \forall p \in P, \tag{5}$$

$$y_a \geq 0 \qquad \forall a \in A', \tag{6}$$

$$y_a \text{ integer} \qquad \forall a \in A', \tag{7}$$

The objective function states that the vehicle-related total cost must be minimized. (We remark that this objective function can be easily extended if we consider costs that are proportional to the demand amounts carried on arcs.) Constraint (1) forces the number of used vehicles to be at most $|F|$. Constraints (2) (resp. (3)) are the flow conservation constraints associated with the vehicles (resp. demands). Constraints (4) impose the amount of the demands carried on an arc of $\tilde{A}$ to be no more than the total capacity of the vehicles passing on this arc. The other constraints are the trivial constraints and the integrity constraints. We remark that constraints (6) are not necessary since constraints constraints (4) and (5) already imply that y is non negative. This model contains $|A'| + |P| \times (|\tilde{A}| + |A_T|)$ variables and $|V_{st}|(|P| + 1) + |A'| + |P| \times (|\tilde{A}| + |A_T|) + |\tilde{A}| + 1$ constraints.

3.3 Metric constraints based formulation

In the formulation given in the previous subsection, we determine the routing of the demands even though the value of the objective function only depends on the vehicle routes and no additional constraints on the demand routing are considered. Therefore, it would be

enough to only focus on determining the vehicle routes that allow a feasible routing of the demands. This can be achieved by considering no other variables than y_a for all $a \in A'$ and replacing constraints (3) and (4) by the so-called metric constraints as described below.

Iri (1971) and Onaga and Kakhuso (1971) independently showed that the metric constraints can be used to check whether or not there exists a feasible multicommodity flow in a graph when demands and arc capacities are given. Their result, known as the Japanese theorem, can be briefly presented as follows. Let $\overline{G} = (\overline{V}, \overline{A})$ be a complete directed graph and $w \in \mathbb{R}_+^{|\overline{A}|}$ (resp. $r \in \mathbb{R}_+^{|\overline{A}|}$) be the capacity (resp. demand) vector indexed on the arcs of $\overline{A}$. The capacity vector w allows the transportation of the demands of r if and only if all the metric constraints

$$(w - r)^T \pi \geq 0 \quad \forall \pi \in Met_n \tag{8}$$

are satisfied where $Met_n = \{\pi \in \mathbb{R}_+^{\overline{A}} \mid \pi_{ik} + \pi_{kj} - \pi_{ij} \geq 0 \; \forall i \neq j \neq k \neq i\}$ is the *metric cone*. Clearly, the number of metric constraints is infinite. However, those associated with extreme rays of the metric cone are sufficient to ensure that the capacity vector w allows the transportation of the demands of r. These constraints are in exponential number. For our problem, the metric cone is the one induced by the complete graph on V_{st}, and capacity and demand vectors are given by the number of vehicles per arc

$$w_a = \begin{cases} +\infty & \text{if } a \in A_T, \\ y_a & \text{if } a \in \tilde{A}, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$r_a = \begin{cases} \frac{q^p}{B} & \text{if } a = (u_0, w_T) \text{ with } u = o^p \text{ and } w = d^p \text{ for some } p \in P, \\ 0 & \text{otherwise,} \end{cases}$$

for all $a \in \overline{A}$. (We remark that we could have considered the whole vehicle (resp. demand) volumes for the capacity (resp. demand) vector as well.) Using metric constraints (8) instead of constraints (3) and (4), we now introduce the following integer linear formulation for the SPDPR:

$$\min\{\sum_{a \in A'} c_a y_a \mid y \text{ satisfies (1), (2), (8), (6), (7)}\} \tag{9}$$

This program contains less variables than the first one (i.e., $|A'|$ versus $|A'| + |\tilde{A} \cup A_T| \times |P|$), but it has an exponential number of constraints whereas the first model contains a polynomial one. As it will be pointed out in Section 5, the exponential number of metric constraints can be tackled in polynomial time.

Furthermore, once we have an optimal solution of (9), determining the routing of the demands (i.e., the variables x_a^p for all $a \in \tilde{A} \cup A_T$ and for all $p \in P$ in the first formulation) can be performed in polynomial time. It is actually nothing but a continuous multicommodity flow problem which is a well-known polynomially-solvable problem (Ahuja et al., 1993).

The two presented models can be easily extended to a variant of the SPDPR where capacity constraints on the demand volume stocked at the same time at any stop of the network are taken into account. In fact, if we denote by e_u the maximum waiting load of

demands allowed at vertex $u \in V$, then we only have to add the constraints $\sum_{p \in P} x_a^p \leq e_u$ for all $a = (u_t, u_{t+1})$ with $t \in \{0, \ldots, T-1\}$, in the multicommodity flow based formulation. To consider such constraints in the metric constraints based model, we only have to modify the vector ω in the metric inequalities. In fact, we replace the infinite capacity associated with the arcs of A_T by $\omega_a = \frac{e_u}{B}$ for all $a = (u_t, u_{t+1})$ with $t \in \{0, \ldots, T-1\}$.

4 Formulation strengthening

We now present four families of constraints we will use in our algorithm to strengthten the linear relaxations. The first one can be used by both models whereas the second and the third ones are based on the flow variables associated with the demands (i.e., variables x) and then cannot be used in the metric constraints based formulation. The last family of constraints is a strengthened formulation of the metric constraints (8), so it is only used in the metric constraint based formulation. The way to handle algorithmically these families of constraints will be discussed in Section 5.

4.1 Cut Constraints

In this part, we introduce constraints known as cut constraints that we will use in our branch-and-cut algorithm to strengthen both linear relaxations of the SPDPR. These constraints have been already considered in various optimization problems. (See (Bienstock et al., 1995), (Barahona, 1996) for instance.) They are based on the notion of cuts in G'. For a vertex subset $W \subset V_{st}$, $W \neq \emptyset$, let us denote by $q[W, V_{st} \setminus W]$ the total volume of the demands of P having their origin vertex in W and their destination one in $V_{st} \setminus W$.

Proposition 2 *Let $W \subset V_{st}$, $W \neq \emptyset$ induce a cut $\delta^+(W)$ so that $\delta^+(W) \cap A_T = \emptyset$. Then the cut constraint*

$$y(\delta^+(W)) \geq \left\lceil \frac{q[W, V_{st} \setminus W]}{B} \right\rceil \tag{10}$$

is valid for the SPDPR.

Proof: The constraint $By(\delta^+(W)) \geq q[W, V_{st} \setminus W]$ is obviously valid for the SPDPR. In fact, it expresses the fact that the whole vehicle capacity of the arcs of $\delta^+(W)$ must exceed the whole demand from W to $V_{st} \setminus W$. Since y is integer, dividing the two members of this inequality by B and rounding-up the right-hand side yields (10). □

We remark that only cuts that do not intersect A_T are considered because capacity constraints do not apply for these arcs. (In the first formulation, there are no constraints (4) associated with the arcs of A_T, and in the second one, we consider an infinite capacity on them.) This comes directly from the fact that any demand (or part of it) can stay without any conditions at any node of the network.

4.2 Extended Arc Residual Capacity Constraints

The extended arc residual capacity constraints are used to strengthen the linear relaxation of the arc-vertex based formulation. They cannot be used for the metric constraints based

formulation since they are defined with respect to variables x. Since they are an extension of the arc residual capacity constraints, we begin this paragraph by presenting the so-called arc residual capacity constraints that were first introduced by Magnanti et al. (1993). These are in fact a special case of the mixed-integer rounding constraints (Nemhauser and Wolsey, 1990).

For a given set $K \subseteq P$, the total volume of the demands $p \in K$ is equal to $q(K) = \sum_{p \in K} q^p$. Let $\gamma_K = \lceil \frac{q(K)}{B} \rceil$ and $r_K = (q(K) \ mod \ B)$. By convention, we set r_K equal to B if $q(K)$ is a multiple of B (i.e., $r_K \in]0, B]$.) We first state that the arc residual capacity constraints are valid for our problem. We omit the proof since it is similar to the one given by Magnanti et al. (1993) when they introduced those inequalities.

Proposition 3 *Let $K \subseteq P$ and $a \in \tilde{A}$. The arc residual capacity constraint*

$$\sum_{p \in K} x_a^p - r_K \ y_a \leq (\gamma_K - 1)(B - r_K) \tag{11}$$

is valid for SPDPR. □

In what follows, we give an example of a violated arc residual capacity constraint. Suppose that we have two demands which have respectively volumes $q^1 = 6$ and $q^2 = 12$, and that the vehicle capacity B is equal to 10. Let $(\overline{y}, \overline{x})$ be a fractional solution satisfying (1)-(6) so that there exists $a' \in \tilde{A}$ with $x_{a'}^p = q^p$ for $p = 1, 2$ and $y_{a'} = 1.8$. There exists an arc residual capacity constraint defined on the arc a' which is violated by $(\overline{y}, \overline{x})$. Indeed, if we consider the demand set $K = P$, we have $\gamma_K = 2$ and $r_K = 8$, and then the constraint implies that we must have $8y_{a'} \geq (6+12) - (2-1)(10-8) = 16$, that is $y_{a'} \geq 2$.

We now introduce the extended arc residual capacity constraints. For this, we first present some notation. For a given $\phi \in \{0, \ldots, T-1\}$, we define $V^\phi = \{u_t \in V_{st} \mid u \in V \setminus \{v_0\}, t = \phi\}$ and $A^\phi = \delta^+(V^\phi) \cap \tilde{A}$. V^ϕ corresponds to the vertices of $V \setminus \{v_0\}$ at time ϕ and A^ϕ corresponds to the arcs of $A \setminus \delta(v_0)$ that begin at date ϕ.

Proposition 4 *Let $K \subseteq P$, $\phi \in \{0, \ldots, T-1\}$ and $X \subseteq A^\phi$. Then the extended arc residual capacity constraint*

$$\sum_{p \in K} \sum_{a \in X} x_a^p - r_K \sum_{a \in X} y_a \leq (\gamma_K - 1)(B - r_K) \tag{12}$$

is valid for SPDPR.

Proof: We first remark that the auxiliary graph has no backward arcs, that is, it does not contain any arc (u, w) with $u \in V^{\phi_1}$, $w \in V^{\phi_2}$ and $\phi_2 \leq \phi_1$. Since $x_a^p \leq q^p$ for all $a \in \tilde{A} \cup A_T$, the constraint $\sum_{a \in \delta^+(V^\phi)} x_a^p \leq q^p$ holds for all $\phi \in \{0, \ldots, T-1\}$ and for all $p \in P$. Let $\phi \in \{0, \ldots, T-1\}$, $K \subseteq P$ and $X \subseteq A^\phi$. We can easily show that constraint (11) is dominated by the sum of the constraints $\sum_{a \in X} x_a^p \leq q^p$ for all $p \in K$ (resp. constraints (4) for all arc in X) if $\sum_{a \in X} y_a \geq \gamma_K$ (resp. $\sum_{a \in X} y_a \leq \gamma_K - 1$.) Therefore, the extended arc residual capacity constraints (12) are valid for the SPDPR. □

4.3 Rounded Metric Constraints

The metric constraints (8) can be strengthened. Indeed, if the metric coefficients π_a of the constraint are rational for all $a \in \tilde{A}$, since w is by definition rational (because of $B \in \mathbb{Z}^+$ and y is integer), $w^T\pi$ is rational and we can then consider the rounded metric constraint

$$w^T\pi \geq \lceil r^T\pi \rceil \tag{13}$$

which is also valid for SPDPR. Similar constraints have been used (Bienstock et al., 1995) for solving the minimum cost capacity installation for multicommodity network flows. We point out that the cut constraints (10) correspond to a particular case of the rounded metric constraints.

5 Separation

One of the most important parts of an efficient branch-and-cut algorithm is the so-called separation problem that can be described as follows. Given a constraint system $Ax \leq b$ based on $\mathbb{R}^n$ and a point x of $\mathbb{R}^n$, the *separation problem* associated with this system consists of deciding whether all the constraints of the system are satisfied by x and if not, of finding a constraint violated by x. Grötschel, Lovász and Schrijver (1985) have shown that if the separation problem of a constraint system $Ax \leq b$ can be solved in polynomial time, then any optimization problem over this system can also be solved in polynomial time, even if the number of constraints is exponential.

In this section, we consider the separation problems for the inequalities presented in the previous section. The solutions we consider in the separation problems are the vectors $\overline{y} \in \mathbb{R}^{|A'|}$ for the metric constraints based formulation and the vectors $(\overline{y}, \overline{x}) \in \mathbb{R}^{|A'|} \times \mathbb{R}^{|\tilde{A} \cup A_T| \times |P|}$ for the multicommodity flow formulation that we obtain after having solved the current linear relaxation.

5.1 Separation of the metric constraints

The separation problem for metric constraints (8) can be solved in polynomial time. In fact, it can be reduced to the following linear program:

$$\begin{aligned} &\min (w - r)^T \pi \\ &\quad \text{s.t.} \\ &\quad \pi \in Met_n, \end{aligned}$$

where Met_n, w and r are defined as in §3.3. Let π^* be an optimal solution of this linear program. If the objective value associated with π^* is negative, then the metric constraint $(w - r)^T \pi^* \geq 0$ is violated by $\overline{y}$. Otherwise, we can assert that $\overline{y}$ satisfies all the metric constraints.

5.2 Separation of the rounded metric constraints

The separation algorithm for the rounded metric constraints (13) is a heuristic based on the one developed by Bienstock et al. (1995). Each time a violated metric (8) is found, we search among the metric coefficients π_a, $a \in \tilde{A}$, the one with the smallest positive value, say π_0, and we then check if all the values $\frac{\pi_a}{\pi_0}$ for all $a \in \tilde{A}$ are integer. If this is the case, then we have found a stronger violated rounded metric constraint; if not, we keep the initial metric constraint (8) we have found. This heuristic is then performed in linear time.

5.3 Separation of the cut constraints

The separation problem associated with the cut constraints (10) is NP-hard in general (Bienstock et al., 1995). Therefore, we have developed heuristics to separate the cut constraints. In what follows, we describe three separation heuristics that are based on previous works ((Bienstock et al., 1995), (Gabrel et al., 1999)). We adapt these heuristics to space-time graph. In the description of those heuristics, W will always represent a vertex subset of V_{st} so that $\delta^+(W) \cap A_T = \emptyset$.

The first one is the so-called n-cut heuristic that was developed by Bienstock et al. (1995) for the MCCI. This heuristic works as follows. For any demand $p \in P$, we check whether there exists a path from o_0^p to d_T^p (these two vertices correspond to the origin o^p at time 0 and the destination d^p at time T of the demand p) in the auxiliary graph G' where we only consider the arc set $A_T \cup \{a \in \tilde{A} \mid \overline{y_a} > 0\}$. Obviously, this can be performed using any search algorithm. If there is no such a path, it is straightforward that there exists a violated constraint (10). This constraint is actually induced by a vertex set W so that $o_0^p \in W$, $d_T^p \in V_{st} \setminus W$ and the vertices of V_{st} that are reachable from o_0^p belong to W. ($\delta^+(W) \cap A_T = \emptyset$ is guaranteed because all the arcs of A_T are considered.) If a path from o_0^p to d_T^p is found for all the demands $p \in P$, we then randomly pick out some vertices of V_{st} to form a set $W$and we check if the cut constraint associated with W is violated or not. Since the complexity of a search algorithm is in $\mathcal{O}(|\tilde{A} \cup A_T|)$ and we need to perform it $|D|$ times, the complexity of this heuristic is in $\mathcal{O}(|D||\tilde{A} \cup A_T|)$.

The second heuristic is based on the so-called n-partition heuristic devised by Bienstock et al. (1995). We only consider the case n=2 since we seek violated cut constraints (10). We start with a randomly chosen demand $p \in P$ and the two vertex subsets of V_{st}, $V_1 = \{o_t^p \mid t = 0, \ldots, T\}$ and $V_2 = \{d_t^p \mid t = 0, \ldots, T\}$. We then iteratively assign the vertices of $V_{st} \setminus (V_1 \cup V_2)$ to either V_1 or V_2 as described below. (At the end of the heuristic, we will consider the cut constraint induced by $W = V_1$.) At each iteration, we randomly select a vertex $u \in V \setminus \{v_0, o^p, d^p\}$ that has not been considered yet. Then, we only focus on the vertex set $V_1 \cup V_2 \cup \{u_t \mid t = 0, \ldots, T\}$ and assign vertices $\{u_t \mid t = 0, \ldots, T\}$ either to V_1 or V_2, the assignment of the vertices of the two latter sets remaining unchanged. The assignment of the vertices $\{u_t \mid t = 0, \ldots, T\}$ is performed in order to obtain new sets V_1 and V_2 so that $\delta^+(V_1) \cap A_T = \emptyset$ and $\overline{y}(\delta^+(V_1) \cap \delta^-(V_2)) - \lceil q[V_1, V_2]/B \rceil$ is minimum. This can be done by enumerating and testing all the possible assignments of vertices $\{v_t \mid t = 0, \ldots, T\}$ since there only exist $T + 2$ feasible ones with respect to the condition $\delta^+(V_1) \cap A_T = \emptyset$. Once all the vertices of $V \setminus \{v_0, o^p, d^p\}$ have been considered, we check whether the cut constraint induced by $W = V_1$ is violated. This heuristic works in $\mathcal{O}(|\tilde{A} \cup A_T|)$ time.

The last heuristic is an extension of the one developed by Gabrel et al. (1999) for the exact solution of multicommodity network optimization problems with general step cost functions. Given a demand $p \in P$, let W be a randomly chosen set so that $o_0^p \in W$ and $d_T^p \in V_{st} \setminus W$. This heuristic consists of iteratively switching vertices (except o_t^p and d_t^p for all $t = 0, \ldots, T$) in between W and $V_{st} \setminus W$ in order to increase the value of the ratio $p(W) = \lceil q\,[W, V_{st} \setminus W]\,/B \rceil\,/\overline{y}(\delta^+(W))$. This process stops when no switching increases the ratio $p(W)$. To find the switching, we compute for each vertex $u \in V \setminus \{v_0, o^p, d^p\}$ that has not been considered yet the value $\alpha(u)$ that corresponds to the assignment of the vertices u_t for all $t = 0, \ldots, T$ to the subsets W and $V_{st} \setminus W$ that maximizes $p(W)$. (The assignment of the other vertices does not change.) Again, there only exist $T + 2$ feasible assignments which respect the condition $\delta^+(W) \cap A_T = \emptyset$. We then only modify the assignment of the vertices $\{u_t \mid t = 0, \ldots, T\}$ associated with the vertex u for which $\alpha(u)$ is maximum. We repeat this process until all the vertices of $V \setminus \{v_0, o^p, d^p\}$ have been considered. At this point, we obtain the new subset W. The computation of this new subset can be performed in $\mathcal{O}(|V||\tilde{A} \cup A_T|)$ time. At the end of the heuristic, we check if the cut constraint induced by the obtained set W is violated. In our experiments, each time the algorithm is applied, up to ten random initial vertex subsets W are tried and the final result is taken to be the best over the 10 locally optimal subsets found. We apply this algorithm for all the demands $p \in P$.

5.4 Separation of the extended arc residual capacity constraints

We present in this section the separation routine we use to separate the extended residual capacity constraints. Since this separation routine is based on the separation algorithm of the arc residual capacity constraints, we first present this algorithm.

The separation problem for the arc residual capacity constraints (11) can be solved in polynomial time. Atamtürk et Rajan (2002) showed that the separation problem on each arc can be solved in $\mathcal{O}(|P|)$. For a given arc $a \in \tilde{A}$, let $D = \{p \in P \mid \overline{x}_a^p > q^p(\overline{y}_a - \lfloor \overline{y}_a \rfloor)\}$. They showed that if a constraint of type (11) associated with arc a is violated by $(\overline{y}, \overline{x})$ for a demand set $K \subseteq P$, then there exists a violated one for the demand set D. Therefore, the separation problem consists of checking for all arc $a \in \tilde{A}$ whether the associated constraint is violated for the demand set D or not. This can be easily performed in linear time.

The separation problem of the extended arc residual capacity constraints (12) can be decomposed into T independent problems, each one defined by a value $\phi \in \{0, \ldots, T-1\}$. The complexity of this separation problem for a given value ϕ has not been stated yet. Nevertheless, if the arc set $X \subseteq A^\phi$ is fixed, the separation problem is the same as the one for the arc residual capacity constraints (11). The difference is that in this case, the set D is defined as $D = \{p \in P \mid \sum_{a \in X} \overline{x}_a^p > q^p(\sum_{a \in X} \overline{y}_a - \lfloor \sum_{a \in X} \overline{y}_a \rfloor)\}$. So, when the arc set X is fixed, the extended arc residual capacity constraints can be separated in $\mathcal{O}(|P|)$.

For every $\phi \in \{0, \ldots, T-1\}$, instead of devising a heuristic to separate all the extended arc residual capacity constraints, we only separate constraints (12) associated with one arc of A^ϕ, two arcs of A^ϕ, all the arcs of A^ϕ but one, and all the arcs of A^ϕ but two. (We can remark that the heuristic also separates the arc residual capacity inequalities which are a particular case of the extended arc residual capacity inequalities when $|X| = 1$.) The complexity of the separation routine is in $\mathcal{O}(|P| \sum_{\phi=0}^{T-1} |A^\phi|(|A^\phi| + 1))$ since there are

$\frac{|A^\phi|(|A^\phi|-1)}{2}$ unordered pairs $a, a' \in A^\phi$ for a given $\phi \in \{0, \ldots, T-1\}$.

6 Computational results

This section is devoted to the experiments we have done to solve the SPDPR, based on the foregoing theorical developments. Our aim is not to evaluate the cost savings (and the additional computational time) implied by considering reloads in the splittable pickup and delivery problem. This was actually addressed briefly in the introduction. Moreover for a variant with time windows, Mitrović-Minić and and Laporte (2006) empirically showed the usefulness of the reloads in demand transportation. Our purpose is more to provide a basic frame for further researches, and give lower bounds to check the efficiency of heuristics developed for the problem. We begin this section by presenting the branch-and-cut algorithm we use to solve the two formulations introduced in Section 3.

For the first formulation, the linear program we start with is composed of the constraints (1)-(6) together with the cut constraints implied by vertices of V, that is,

$$y(\delta^+(\{u_0, u_1, \ldots, u_T\})) \geq \left\lceil \frac{q(K)}{B} \right\rceil \tag{14}$$

if there exists $K \subseteq P$ so that $u = o^p$ for all $p \in K$, and

$$y(\delta^+(V_{st} \setminus \{u_0, u_1, \ldots, u_T\})) \geq \left\lceil \frac{q(K)}{B} \right\rceil \tag{15}$$

if there exists $K \subseteq P$ so that $u = d^p$ for all $p \in K$. For the metric constraints based model, the initial linear program is given by the constraints (1), (2), (6), (14) and (15).

For the multicommodity flow formulation, the optimal solution of the relaxation is feasible for the SPDPR if it is an integral vector. Usually, the solution is not feasible, and thus, in each iteration of the branch-and-cut algoritm, it is necessary to generate further inequalities that are valid for the SPDPR but violated by the current solution. For this, one has to solve the separation problems introduced in Section 5. The separation is first performed on the cut constraints (10) using the three heuristics described on the previous section and if none is found, then we separate the extended arc residual capacity inequalities (12).

For the metric constraints based formulation, the optimal solution of the relaxation has to be an integral vector that satisfies all the metric constraints in order to be a feasible solution for SPDPR. So, at each iteration, we first separate the cut constraints (10). If no such violated inequality is found, then we separate in an exact way the metric constraints (8). If a violated metric constraint is found, the separation problem associated with the rounded metric constraints (13) is then performed in order to strengthen this inequality.

The branch-and-cut algorithm was implemented in C++, using COIN-BCP (Lougee-Heimer, 2003) to manage the branch-and-cut tree and Cplex 9.1 (ILOG CPLEX, 2003) as a LP-solver for all the linear programs except the linear relaxation of the second formulation that is solved using COIN-CLP (Lougee-Heimer, 2003). It was tested on a Pentium IV 3.2 Ghz with 1GB of RAM, running under Linux. We fixed the maximum CPU time to 5 hours.

For our tests, we considered random instances. These are induced by complete directed graphs which come from the TSP Library (Reinelt, 1991). The arc costs are equal to rounded-up Euclidian distances. For each graph, three different demand sets of sizes 5, 10 and 15 are randomly generated so that the volume of each demand is between $B/4$ and $3B/4$. (Remember that B denotes the transportation capacity of the vehicles.)

The instances are created in order to be feasible with respect to a given completion time limit T. To do so, we set the duration times to 1 for all the arcs. Two sizes for the fleet are considered: $|F| = \frac{|P|}{\lfloor (T-1)/2 \rfloor}$ and $|F| = \frac{|P|}{\lfloor (T-1)/2 \rfloor} + 2$. Both values ensure that we have enough vehicles to carry all the demands. In fact, each vehicle can carry at least $\lfloor \frac{T-1}{2} \rfloor$ demands. Indeed, each vehicle starts from the depot and goes directly to the origin of the first demand it will be carrying. For each demand p of the $\lfloor \frac{T-1}{2} \rfloor$ demands, the vehicle only carries it on the arc (o^p, d^p). Once the vehicle has reached the destination of the demand p, it goes to the origin of the next demand p' through arc $(d^p, o^{p'})$. The vehicle ends its route at the depot. (Remark that this solution is feasible since the volume of each demand is less than the transportation capacity of the vehicles, the graph is complete and the duration times are all equal to 1.) All considered instances are then feasible. For all the test problems, the completion time limit T is fixed to 7.

In the following tables, the entries are:

- $|V|$: the number of vertices of the initial graph,
- $|P|$: the number of demands,
- $|F|$: the number of vehicles,
- NC : the number of generated violated cut constraints (10),
- NRM : the number of generated violated rounded metric constraints (13),
- NM : the number of generated violated metric constraints (8),
- NE : the number of generated violated extended arc residual capacity constraints (12),
- o/p : the number of problems solved to optimality over the number of instances tested,
- Gap1 : the relative error between the best upper bound (the optimal value if the problem has been solved to optimality) and the lower bound obtained before considering cutting planes that strengthen the linear relaxation,
- Gap2 : the relative error between the best upper bound and the lower bound achieved by the cutting plane phase at the root node (before branching),
- CPU : the total CPU time in seconds.

Each line of the tables reports the average results obtained for three instances, all of them having the same number of vertices, demands and vehicles. The three instances only differ by the coordinates of vertices, and for every demand, by its origin, destination and volume. Both formulations were run on the same test problems.

Each test instance was first transformed into an instance in the auxiliary graph as described in Section 3. In consequence, an initial graph having n vertices gives rise to an auxiliary graph with $6n+2$ vertices and $5n^2+2n$ arcs. This transformation leads to mixed-integer linear programs having either $5n^2 + 2n$ variables for the metric constraints based formulation or $(|P| + 1) \times 5n^2 + 2n$ variables for the multicommodity flow model.

Table 1 Results obtained with the multicommodity flow formulation

|V|	|P|	|F|	NC	NE	o/p	Gap1	Gap2	CPU
6	5	2	17.33	733.00	3/3	17.26	1.69	1.02
6	5	4	24.00	988.00	3/3	18.81	2.09	1.36
6	10	4	85.33	2261.66	3/3	13.82	1.64	42.86
6	10	6	96.00	3222.00	3/3	14.29	1.59	47.23
6	15	5	238.66	7103.00	3/3	13.35	4.16	362.11
6	15	7	242.00	7062.33	3/3	15.17	4.74	656.95
7	5	2	104.33	4977.66	3/3	33.24	20.72	64.05
7	5	4	38.00	2061.33	3/3	15.46	3.21	4.81
7	10	4	221.66	6582.00	3/3	17.17	5.44	208.24
7	10	6	84.00	3602.33	3/3	12.57	0.09	36.05
7	15	5	965.66	17662.66	3/3	17.59	8.75	5174.03
7	15	7	1292.66	17883.33	3/3	16.87	9.12	7313.39
8	5	2	103.00	3703.33	3/3	19.80	6.99	34.74
8	5	4	87.33	4960.66	3/3	20.03	6.49	33.66
8	10	4	344.33	9581.00	3/3	17.73	4.09	673.36
8	10	6	278.33	8175.66	3/3	16.75	3.83	647.17
8	15	5	1432.33	17653.33	2/3	18.24	5.79	9162.24
8	15	7	1447.00	20645.66	2/3	21.32	8.48	13587.84
9	5	2	39.00	5498.66	3/3	13.35	2.99	27.52
9	5	4	79.00	6973.33	3/3	16.93	4.05	62.16
9	10	4	550.66	12309.33	3/3	18.34	3.77	2179.85
9	10	6	785.33	17228.00	3/3	13.68	3.94	4952.95
9	15	5	1056.66	17126.66	0/3	32.71	25.16	18000.00
9	15	7	1447.66	19059.66	0/3	32.40	22.42	18000.00
10	5	2	146.33	8985.00	3/3	16.71	7.59	152.38
10	5	4	149.33	7220.66	3/3	16.77	7.96	562.95
10	10	4	2125.33	23081.00	1/3	26.10	11.99	13025.04
10	10	6	1571.66	29175.66	2/3	20.89	10.92	14093.96
10	15	5	1681.33	22241.00	0/3	38.57	28.37	18000.00
10	15	7	1907.33	27872.00	0/3	26.10	15.09	18000.00

Table 2 Results obtained with the metric constraints based formulation

$\|V\|$	$\|P\|$	$\|F\|$	NC	NRM	NM	o/p	Gap1	Gap2	CPU
6	5	2	17.66	0.00	0.00	3/3	17.26	1.82	0.24
6	5	4	34.33	0.66	0.00	3/3	18.81	2.76	0.96
6	10	4	76.00	4.33	0.00	3/3	13.82	1.70	3.95
6	10	6	106.00	10.33	0.00	3/3	14.29	2.02	4.88
6	15	5	213.33	29.33	0.00	3/3	13.35	4.39	26.74
6	15	7	271.33	60.00	0.00	3/3	15.17	5.21	51.99
7	5	2	304.33	1.66	0.00	3/3	33.24	20.89	36.39
7	5	4	61.33	0.00	0.00	3/3	15.46	3.21	2.85
7	10	4	284.66	23.00	0.00	3/3	17.17	5.49	29.78
7	10	6	60.00	4.00	0.00	3/3	12.57	0.19	1.43
7	15	5	1580.00	179.00	0.33	3/3	17.59	9.17	1620.19
7	15	7	1785.00	166.33	0.00	3/3	16.87	9.44	2176.62
8	5	2	121.33	1.00	0.00	3/3	19.80	7.00	6.90
8	5	4	187.66	0.66	0.00	3/3	20.03	6.50	16.69
8	10	4	440.00	21.33	0.00	3/3	17.73	4.13	120.67
8	10	6	436.33	31.66	0.00	3/3	16.75	4.43	129.39
8	15	5	3110.66	210.33	0.33	3/3	17.89	6.39	3073.67
8	15	7	1719.33	75.33	0.00	3/3	16.80	3.70	846.76
9	5	2	155.66	1.66	0.00	3/3	13.35	2.99	15.55
9	5	4	218.33	9.33	0.00	3/3	16.93	4.06	46.83
9	10	4	388.33	15.00	0.00	3/3	18.34	3.64	108.95
9	10	6	1072.00	31.00	0.00	3/3	13.68	3.94	2125.72
9	15	5	3488.00	364.00	0.66	1/3	16.90	8.24	14947.41
9	15	7	3098.33	226.33	1.66	3/3	14.60	2.43	5980.95
10	5	2	173.00	2.66	0.00	3/3	16.71	7.63	79.13
10	5	4	304.00	7.33	0.00	3/3	16.77	8.18	224.40
10	10	4	2582.66	50.33	0.00	3/3	20.38	4.85	6580.89
10	10	6	2314.66	71.00	0.00	3/3	16.52	5.94	6329.88
10	15	5	6105.66	151.33	0.00	1/3	20.83	8.04	14565.66
10	15	7	5038.66	228.00	0.33	0/3	26.30	16.01	18000.00

Table 1 gives the results obtained using the multicommodity flow model. We remark that almost all the instances with 5 or 10 demands have been solved to optimality except 3 of them. For the instances with 15 demands, our branch-and-cut algorithm could solve problems with up to 7 vertices. However, for bigger instances, the problem appeared harder to solve. In fact, only two thirds of the instances having 8 vertices (and 15 demands) could be solved. Moreover, no instances with 9 vertices and more were solved.

We can also remark that our branch-and-cut algorithm generated a lot of cut constraints (10) (up to 2125) and a huge number of extended arc residual capacity inequalities (12) (up to 29175). We can also observe that Gap2 is quite small for most of the instances. Moreover, this gap is much smaller than Gap1, which shows that adding the violated inequalities significantly strengthened the linear relaxation.

Table 2 reports the results obtained using the metric constraints based formulation. We remark that for instances with 9 vertices or less, all problems have been solved to optimality

except 2 of them. We can also note that for all the test instances with up to 10 vertices and 10 demands, we could get the optimal solution. However, for 10 vertices and 15 demands, only one instance over six has been solved. This can be explained by the fact that the problem becomes harder when the size of the problem increases. Moreover, we believe that the most decisive parameter is the number of demands. Indeed, as it can be seen in Table 2, the instances with 15 demands needed at least twice the time required for solving the same instances but with only 10 demands. One should also note that the rounded metric constraints (13) play an important role in the resolution of all the instances.

However, the algorithm does not generate a lot of metric inequalities (8). This is because in most of the cases, a violated metric inequality is extended to a rounded metric one using our separation algorithm. This shows at the same time that the separation heuristic we have for the rounded metric inequalities (13) is quite efficient. We also notice that the algorithm generates an important number of cut inequalities (10). Finally, as for the multicommodity flow formulation, Gap2 is much smaller than Gap1. The rounded metric constraints (13) and the cut constraints (10) are thus very effective in the improvement of the lower bound.

Comparing the results obtained using both formulations, one can see that more instances are solved to optimality using the metric formulation. Moreover, the time is around 5 times less than the one obtained by the multicommodity flow formulation. However, this latter provides a lower gap. For some instances like those with 9 vertices, 15 demands and 7 vehicles, Gap2 obtained with the multicommodity flow formulation is higher than that related to the metric one. However, this can be explained by the fact that the upper bound obtained with the first formulation is quite large.

7 Concluding remarks

In this paper we have introduced a new NP-hard problem called the splittable pickup and delivery problem with reloads. We have proposed two mixed-integer linear programming formulations for the problem and studied a cutting-plane approach for solving them. We have identified some valid inequalities. In particular, we have introduced a new family of constraints which generalizes the so-called arc residual capacity inequalities. As it appears from the presented computational study, these inequalities are useful for solving the multicommodity flow formulation.

Our models can be easily extended to take into account additional constraints such as time windows and capacity limits on waiting loads. We can also consider different capacities for the vehicles. To do so, one has to consider binary variables with each vehicle (instead of integer variables associated with all the vehicles). Some extra work would be then necessary to extend the inequalities considered in Section 4 to a heterogeneous vehicle fleet. In this case, different depots for the vehicles can also be considered.

Our approach can be seen as a prospective work in order to develop an efficient algorithm for solving that problem. It would be interesting to identify further valid inequalities to strengthen more both formulations. Moreover, other approaches merit to be tried for solving this new problem. In particular, one can consider a column generation approach based on an arc-path formulation for the problem. Another approach which might be interesting is a Benders' decomposition applied to the metric model. Finally, an interesting question

would be to develop a model for the problem which does not use time indexation. All these questions are our line for future research work.

Acknowledgement

The authors would like to thank Francisco Barahona for helpful discussions in developing some ideas of the paper. They also wish to thank the anonymous referees for several suggestions that led to improvements in the paper.

References and Notes

1 Ahuja, R., Magnanti, T. and Orlin, J. (1993) 'Networks flows : theory, algorithms and applications', *Englewood Cliffs, NJ : Prentice Hall.*

2 Atamtürk, A. and Rajan, D. (2000) 'On splittable and unsplittable flow capacitated network design arc-set polyhedra', *Mathematical Programming*, Vol. 92, pp.315–3332.

3 Barahona, F. (1996) 'Network design using cut inequalities', *SIAM Journal on Optimization*, Vol. 6, pp.823–837.

4 Bienstock, D., Chopra, S., Günlünk, O. and Tsai, C. (1995) 'Minimum cost capacity installation for multicommodity network flows', *Mathematical Programming*, Vol. 81, pp.177–199.

5 Dantzig, G.B. and Ramser, R.H. (1959) 'The Truck Dispatching Problem', *Management Science*, Vol. 6, pp.80–91.

6 Gabrel, V., Knippel, A. and Minoux, M. (1999) 'Exact solution of multicommodity network optimization problems with general step cost functions', *Operation Research Letter*, Vol. 25, pp.15–23.

7 Gendrau, M. and Laporte, G. (1995) 'Arc routing problems, part I : The Chinese Postman Problem', *Operations Research*, Vol. 43, pp.231–242.

8 Grötschel, M., Lovász, L. and Schrijver, A. (1985) 'Geometric algorithms and combinatorial optimization', *Springer-Verlag*.

9 Grünert, T. and Sebastian, H.-J. (2000) 'Planning models for long-haul operations of postal and express shipment companies', *European Journal of Operational Research*, Vol. 122, pp.289–309.

10 ILOG CPLEX. (2003) 'Reference manual', *http://www.ilog.com/products/cplex*.

11 Iri, M. (1971) 'On an extension of the maximum-flow minimum-cut theorem to multicommodity flows', *J. Operations Research Soc. of Japan*, Vol. 13, No. 3.

12 Lougee-Heimer, R. (2003) 'The Common Optimization INterface for Operations Research', *IBM Journal of Research and Development*, Vol. 47, No. 1, pp.57–66.

13 Magnanti, T., Mirchandani, P. and Vachani, R. (1993) 'The convex hull of two core capacitated network loading problems', *Mathematical Programming*, Vol. 60, pp.233–250.

14 Mitrović-Minić, S., and Laporte G. (2006) 'The pickup and delivery problem with time windows and transshipment', Information Systems and Operation Research, Vol. 44, No. 3, pp.217–227.

15 Nemhauser, G. L. and Wolsey, L. A. (1990) 'A recursive procedure to generate all cuts for 0-1 mixed integer programs', *Mathematical Programming*, Vol. 46, pp.379–390.

16 Oertel, P. (2000) 'Routing with reloads', *PhD Thesis*, Köln.

17 Onaga, K. and Kakhuso, O. (1971) 'On feasibility conditions of multicommodity flows networks', *Transactions on circuit theory*, Vol. 4, pp.425–429.

18 Reinelt, G. (1991) 'TSPLIB - A Traveling Salesman Problem Library', *ORSA Journal on Computing*, Vol. 3, pp.376–384.

19 Savelsberg, M. W. P. and Sol, M. (1995) 'The general pickup and delivery problem', *Transportation Science*, Vol. 29, pp.17–29.