Noname manuscript No.
(will be inserted by the editor)

Models for the single-vehicle preemptive pickup and delivery problem

H. L. M. Kerivin · M. Lacroix · A. R. Mahjoub

Received: date / Accepted: date

Abstract In this paper, we study a variant of the well-known single-vehicle pickup and delivery problem where the demands can be unloaded/reloaded at any node. By proving new complexity results, we give the minimum information which is necessary to represent feasible solutions. Using this, we present integer linear programs for both the unitary and the general versions. We then show that the associated linear relaxations are polynomial-time solvable and present some computational results.

Keywords pickup and delivery · reloads · minimal representations · complexity · integer linear programs · separation problems

1 Introduction

The *Single-vehicle Pickup and Delivery Problem (SPDP)* is a well-studied problem which consists of constructing a route for a limited-capacity vehicle in order to satisfy transportation demands defined by paired pickup and delivery locations, see Parragh et al. (2008); Berbeglia et al. (2007); Savelsberg and Sol (1995) for surveys of the SPDP and close-related problems. In this paper, we are interested in a variant of the SPDP where no time-windows are considered

H. L. M. Kerivin
LIMOS, CNRS UMR 6158, Université Blaise-Pascal - Clermont-Ferrand II,
Complexe Scientifique des Cézeaux, 63173 Aubière Cedex, France
E-mail: kerivin@clemson.edu
Present address: Department of Mathematical Sciences, Clemson University, O-326 Martin Hall, Clemson, SC 29634, USA

A. R. Mahjoub · M. Lacroix
LIMOS, CNRS UMR 6158, Université Blaise-Pascal - Clermont-Ferrand II,
Complexe Scientifique des Cézeaux, 63173 Aubière Cedex, France
E-mail: {lacroix,mahjoub}@lamsade.dauphine.fr
Present address: Université Paris Dauphine, LAMSADE, CNRS UMR 7024, Place du Maréchal de Lattre de Tassigny, 75775 PARIS Cedex 16, France

and reloads are allowed anywhere during the transportation of the demands in order to get a better route for the vehicle. We will call this problem the *Single-vehicle Preemptive Pickup and Delivery Problem (SPPDP)*. Following the graph terminology and notation in Schrijver (2002), the SPPDP can be stated in terms of graphs as follows.

Let $D = (V, A)$ be a simple and strongly connected digraph, hereafter called the *initial digraph*. (Remark that D may not be complete.) Let $v_0 \in V$ be a distinguished node corresponding to the *depot*. We also consider one vehicle having a given transportation capacity $Q \in \mathbb{R}_+$, and a non-empty set P of demands. Each demand $p \in P$ is specified by an arc (o^p, d^p), where $o^p \in V$ corresponds to its origin node and $d^p \in V \setminus \{o^p\}$ to its destination node, and a volume $q^p \in]0; Q]$. The digraph $\Phi = (V, P)$ is called the *demand digraph*. We suppose that a demand cannot be split and cannot go through the same node more than once. This implies that any demand is carried on a path. However, before reaching its destination node, a demand can be fully unloaded at any node and then picked up later by the vehicle. This unloading/picking-up process, called a *reload*, can be repeated several times and we consider restrictions neither on the storage volume nor on the number of reloads at any node. The vehicle closed walk starts at the depot and we suppose that no arc can appear more than once in the vehicle closed walk. We associate with each arc $a \in A$ of the digraph a cost $c_a \in \mathbb{R}_+$ which corresponds to what must be paid by the vehicle to use this arc, and we consider no reload costs. Hence, the SPPDP consists of finding the vehicle closed walk and the demand paths so that the vehicle carries every demand from its origin node to its destination node, the demand paths may contain some reloads, the vehicle is never overloaded, and the cost of the vehicle closed walk is minimum.

The fact of considering reloads in the SPDP considerably modified the structure of its solutions. First of all, even if the digraph is complete, any node is incident to at most one demand, no demand is incident to the depot and the cost function satisfies the triangle inequalities, the vehicle closed walk is no more totally defined by its set of arcs. In fact, at any node where a reload occurs, the vehicle has to go through this node at least twice, which implies that the vehicle closed walk does not correspond to a circuit (that is, a closed walk composed of nodes traversed exactly once) as it is the case for the SPDP. Moreover, reloads make necessary to differentiate between the vehicle closed walk and the demand paths, that is, a demand path may not coincide with part of the vehicle closed walk. Furthermore, it is not possible to restrict the set of nodes to the depot and the origin/destination nodes since reloads may take place anywhere in the digraph.

Despite its additional complexity, it is worth considering the SPPDP because of the important savings that may be obtained with the reloads. The following example illustrates this cost reduction between the SPDP and the SPPDP. Consider the graph given in Figure 1 where each edge represents two opposite arcs, the weights on the edges correspond to the costs associated with the arcs, and the dashed arcs represent the three demands p_1, p_2 and p_3. For the sake of clarity, we have only depicted the arcs which allow us to present

optimal solutions even though the digraph we consider is complete. (The costs of the non-depicted edges can be easily obtained by computing the shortest paths between their extremities.) The volume of each demand is supposed to be equal to the transportation capacity of the vehicle. An optimal solution to the SPDP consists of passing by the nodes $v_0, v_5, v_7, v_4, v_6, v_1, v_3, v_0$ in that order for a cost of 475. An optimal solution to the SPPDP is the ordered sequence of nodes $v_0, v_1, v_3, v_4, v_5, v_7, v_5, v_6, v_0$ in which a reload for demand p_2 occurs at node v_5. The cost of this solution is 446 which corresponds to a saving of more than 6%.

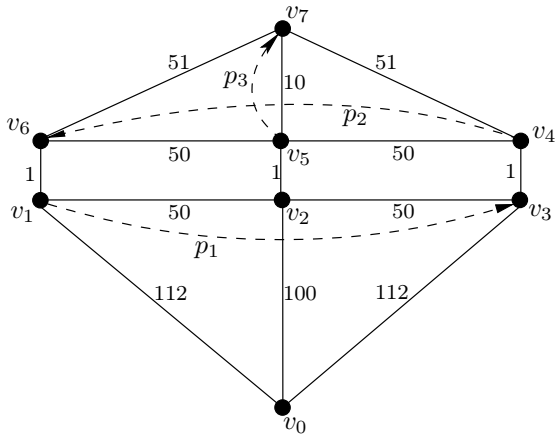

Fig. 1 Example of savings thanks to reloads

Clearly, the SPPDP is NP-hard. Indeed, consider an instance of the SPPDP where the overall volume of demands is lower than the transportation capacity of the vehicle, the digraph is complete, each node different from the depot is the origin or the destination of exactly one demand and the costs are positive, symetric and satisfy the triangle inequalities. Under these assumptions, it is straightforward to see that every optimal solution to the SPPDP is such that no reloads occur and the vehicle closed walk corresponds to a Hamiltonian circuit traversing the origin of every demand before its destination. This problem is nothing but the *pickup and delivery traveling salesman problem* which has been shown to be NP-hard (Renaud et al. 2002).

In this paper, we are interested in integer linear programming formulations for the SPPDP based on minimal representations of feasible solutions. In Section 2, we give some literature overview. In Section 3, we first consider the complexity of the demand-paths checking problem and show that this problem is NP-hard, which implies that the set of arcs associated with the demand paths are necessary in minimal representations. We then look at the vehicle-sequence checking problem, which permits to distinguish the unitary version

of the SPPDP (i.e., when the vehicle cannot carry more than one demand at a time) from the general one. We actually prove that for the unitary case, the order of the arcs of the vehicle closed walk can be removed from a minimal representation, which is not the case in general. The obtained minimal representations of the solutions are used in Sections 4 and 5 to give integer linear programming formulations for the unitary and general versions of the SPPDP. We also prove that the associated linear relaxations are polynomial-time solvable, and present some computational results. Finally, Section 6 gives some concluding remarks.

The rest of this section is devoted to more graph definitions and notation. Given a walk $P = (a_1, \ldots, a_k)$ and two distinct arcs $a_i, a_j \in P$ with $i < j$, we say that a_i is traversed before a_j in P and we write $a_i \prec_P a_j$. We say that a walk C *respects* a path P if for any arc a of P that belongs to C, the arcs traversed before a in P are also traversed before a in C. Note that if C respects P, two adjacent arcs of P are not necessarily adjacent in C, and an arc of P may not appear in C. If C respects all the paths of a given set K, then C respects K. (We remark that it is possible for two vertices to appear before each other with respect to a path set K.) Furthermore, when a graph notation is used without specifying as subscript the graph on which it is applied, it is applied on the initial digraph D.

2 Literature overview

Even though reloads have already been considered in transportation problems for a few years, no previous work seems to have specifically dealt with the SPPDP. In the literature, reloads are actually involved in more general problems or particular cases of our problem.

The Preemptive Stacker Crane Problem (PSCP) (Atallah and Kosaraju 1988) is probably the closest problem to ours which has already been studied. The main difference with the SPPDP lies in the fact that the vehicle cannot carry more than one demand at a time. Some other minor differences exist: the vehicle may pass several times per arc, and as the costs are symetric, the initial graph is undirected. The PSCP can then be considered as a particular case of the SPPDP. However, we are not aware of any integer linear programming formulation dedicated to the PSCP. In fact, only few works have been published on this problem. Atallah and Kosaraju (1988) study the problem when the graph is a line or a circle. They develop an exact algorithm which runs in $\mathcal{O}(k+n)$. Frederickson and Guan (1992) show that the PSCP is polynomial-time solvable if the graph is a tree. They present two algorithms running in $\mathcal{O}(k+qn)$ and $\mathcal{O}(k+nlog(n))$ where k denotes the number of demands, n the number of nodes, and q is less than or equal to $\min\{k, n\}$ and corresponds to the number of non-trivial components in a related digraph. A survey of the complexity of this problem and other close ones can be found in Anily et al.

(2006).

Among the more general versions, we can first mention the Pickup and Delivery Problem with Transfers (PDPT) (Cortés et al. 2005), the Pickup and Delivery Problem with Time-Windows and Transshipments (PDPTWT) (Mitrović-Minić and Laporte 2006) and the Pickup and Delivery Problem with Reloads (RPDP) (Oertel 2000). These problems actually correspond to the same one which is the pickup and delivery in which demands may be fully unloaded and picked up later on certain specific nodes, called *hubs*. Moreover, reloads may be transshipments since the vehicle unloading a demand may be different from the one picking it up. The problem also differs from ours since it takes into account time-windows constraints.

Mitrović-Minić and Laporte (2006) give a two-phase heuristic to approximately solve the PDPTWT. They first construct an initial solution using multi-start cheapest insertion procedure. The best solution is used as the initial one. This solution is then improved by successively removing and reinserting every demand. In both phases, a demand may be inserted with one or no reload. This choice is made by considering all possible insertions and choosing the best one. The insertion of a demand p carried from o^p to d^p with a reload at node v is represented by two demands having o^p, v as origins and v, d^p as destinations respectively. Time-windows of both demands are chosen to ensure that p is carried within its time-window and the first part of the demand path (corresponding to the path from o^p to v) is made before the second one (from v to d^p). The experimental results they obtain show that allowing transshipments is very useful to reduce total travel distance.

To model the RPDP, Oertel (2000) creates an auxiliary graph by considering multiple copies per hub. In fact, every hub is split into two nodes for every demand. This transformation ensures that every vehicle closed walk now corresponds to a circuit. Using this new graph, Oertel then gives a mixed-integer formulation for the problem. He then solves this latter using a tabu-search algorithm by reinserting every demand in the same way as in (Mitrović-Minić and Laporte 2006). Instances with about seventy demands and a hub are solved with this heuristic.

Cortés et al. (2005) also consider for the PDPT an auxiliary graph with multiple copies per hub. The number of copies is equal to twice the maximum number of times a vehicle can make a reload at a same node. (This limit is arbitrary fixed by the user.) The given formulation is then an arc-node formulation. They also present a solution method based on Benders decomposition. This method is then applied to solve exactly instances up to six demands, two vehicles and a hub.

We can also mention *The Vehicle and Request Flow Network Design Problem (VRFNDP)* (Grünert and Sebastian 2000) which arises in the context of ground transportation problem for postal deliveries. In this problem, nodes correspond to letter mail centers and demands to mail packages. Several vehicles and tight time-windows constraints are considered. Moreover, each time a vehicle reaches a node, the mail carried inside the vehicle has to be unloaded

for a sorting stage. Grünert and Sebastian (2000) consider a discrete model based on time periods. They use 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 give a mixed-integer linear programming formulation which is based on two types of commodity flows: the vehicle and the demands. They yet give neither solution method nor computational results for this problem.

The splittable pickup and delivery problem with reloads (SPDPR) (Kerivin et al. 2008) is an extension of the SPPDP when several vehicles are available to carry the demands, transshipments are permitted and demands may be carried on several paths. The authors give a model for the problem using a space-time graph similar to the one given by Grünert and Sebastian (2000) for the VRFNDP. They provide two mixed linear programming formulations based on that model and develop a branch-and-cut algorithm for each formulation. Instances with up to 10 vertices and 15 demands are solved to optimality.

As the SPPDP can be seen as a particular case of different more general problems such as the SPPDP, the RPDP or the PDPT, the models and solution methods which have been developed for those problems could naturally be used for the SPPDP. However, the models developed for the RPDP and the PDPT take advantages of the small number of hubs in the graph. If such approach is used for modeling the SPPDP, every node will have to be replaced by multiple copies, leading to a very large auxiliary digraph. Similarly, since there is no time-windows constraints and no time limit is given for the vehicle closed walk duration, the space-time graph that one should construct for modeling the SPPDP will be much bigger than the initial digraph. Consequently, none of these approaches can be used for the SPPDP for devising exact efficient algorithms. For this, it is necessary to devise models for the SPPDP which are specifically adapted. This is primordial if a polyhedral approach is chosen to solve the problem. In fact, as mentioned by Queyranne and Schulz (1994), "the success of this approach depends highly on the choice of variables which is typically the first question addressed in formulating a model". The set of variables, considered in an integer linear programming formulation, actually corresponds to a representation of some kind of information which may induce several non-necessary feasible solutions. A *tractable representation* must contain enough information to assert in polynomial time whether or not a feasible solution can be obtained from it. A *(inclusionwise) minimal representation* is a tractable one from which no information can be removed without losing the polynomial tractability. A minimal representation then always implies the minimum number of distinct solutions with respect to the given manner a representation is transcribed into variables. Therefore, dealing with this reduced number of possible solutions in a method based on implicit enumeration (e.g., branch-and-bound, branch-and-cut) allows to limit the combinatorial explosion.

3 Minimal representations of the solutions to the SPPDP

This section is devoted to the problem of determining minimal representations of the solutions to the SPPDP. The common way of representing a solution to a transportation problem consists of specifying the sequences of arcs of the vehicle closed walk and the demand paths. Furthermore, it is well-known that for the classical single-vehicle pickup and delivery problem, a solution can be represented by only considering the set of arcs traversed by the vehicle (Savelsberg and Sol 1995). This representation of a solution to the SPDP is mainly based on the fact that every vertex is visited exactly once. This property clearly does not hold for the SPPDP and then, such a representation cannot be straightforwardly proved tractable in our case. Therefore, a natural question to address is whether or not there exists a (minimal) representation of the solutions to the SPPDP where some information on the vehicle closed walk and the demand paths may be discarded.

Clearly, it is sufficient to represent any demand path by its set of arcs instead of its sequence. Moreover, it is trivial that we cannot put aside the whole information associated with the vehicle closed walk. Consequently, among the possible representations we need to investigate, we consider the two defined by the following decision problems. The first one, called *Demand-Paths Checking Problem (DPCP)*, consists of deciding if there exists a feasible solution to the SPPDP when the only information we have is the sequence of arcs of the vehicle closed walk. In the second problem, called *Vehicle-Sequence Checking Problem (VSCP)*, we are given the sets of arcs of the vehicle closed walk and the demand paths, and we check whether there exists an order on all the arcs of the vehicle closed walk which induces a feasible solution to the SPPDP.

For both checking problems, we consider the set of demands P we defined in the introduction, and the arc set $A' \subseteq A$ of the vehicle closed walk. This set clearly induces an Eulerian digraph $D' = (V', A')$ where V' is the set of vertices of V covered by A'. Since v_0 is the starting (and ending) vertex of the vehicle closed walk, it is obvious that v_0 belongs to V'. Moreover, as the vehicle passes exactly once by every arc of A', at most Q units of the demands can be carried on any arc of A'.

3.1 The demand-paths checking problem

Given an Eulerian closed walk C of D', the DPCP seeks to answer the following question. Does there exist a set $K = \{K_p : p \in P\}$ of demand paths of D' so that for every demand $p \in P$, the arcs of the $o^p d^p$-path K_p are traversed in the same order as in C, and for every arc $a \in A'$, the whole demand volume carried on a does not exceed the vehicle capacity Q? The DPCP is NP-complete as shown in the following theorem. The proof is given in the appendix.

Theorem 1 *The DPCP is NP-complete even for the unitary case, and if no vertex of D' appears more than twice in the Eulerian closed walk C and v_0 appears only as the starting and ending vertex of C.* □

As it can be seen, for the unitary case, the DPCP is nothing but an *Arc-Disjoint Path Problem (ADPP)* where the demand paths must fulfill some precedence constraints induced by C. The proof given in the appendix shows that the precedence conditions can be ignored, leading to an ADPP by transforming D' to a specific acyclic digraph called *Quasi-topological digraph*. The NP-completeness of the DPCP is then settled by proving that the ADPP remains NP-complete in quasi-topological digraphs.

Theorem 1 implies that, in any tractable representation of a solution to the SPPDP, no information relative to the arc sets of the demand paths can be put aside. In the next subsection, we look at the VSCP which aims to decide whether or not the order on the arc set of the vehicle closed walk can be dropped in a tractable representation.

3.2 The vehicle-sequence checking problem

Given the demand path set $K = \{K_1, K_2, \ldots, K_p\}$, the VSCP consists of determining whether or not there exists an Eulerian closed walk C of D', starting at v_0, which traverses the arcs of D' in the same order as in the demand paths. The complexity of the VSCP has been previously stated in Kerivin et al. (2010) where the VSCP is referred to as the so-called *Eulerian Closed Walk with Precedence Path Constraints Problem*.

Theorem 2 (Kerivin et al. 2010) *The VSCP is NP-complete.* □

Kerivin et al. (2010) also present a polynomial-time algorithm to solve the VSCP when the set K is composed of arc-disjoint paths. They give necessary and sufficient conditions for the existence of a feasible solution in this case. These conditions are based on the concept of impregnable Eulerian subgraphs which can be defined as follows. Let $\tilde{D} = (\tilde{V}, \tilde{A})$ be an Eulerian subgraph of D'. A vertex v of $\tilde{V}$ is called $\tilde{D}$-*impregnable* with respect to K if for every arc a of $\delta^{\text{out}}_{\tilde{D}}(v)$, there exists an arc a' of $\delta^{\text{in}}_{\tilde{D}}(v)$ so that

(i) $a' \prec_{K_i} a$ for some path K_i of K, if $v = v_0$
(ii) either $a' \prec_{K_i} a$ for some path K_i of K or v is incident with no arc of $A' \setminus \tilde{A}$, if $v \neq v_0$.

The subgraph $\tilde{D}$ is then called *impregnable* if every vertex of $\tilde{D}$ is $\tilde{D}$-impregnable. This new concept is the keystone of our argument towards the polynomial-time solvable case. In fact, an impregnable Eulerian subgraph corresponds to a component of the digraph D' that cannot be traversed by the vehicle according to the order on the arcs specified by the demand paths.

Theorem 3 Kerivin et al. (2010) *The VSCP admits a feasible solution when the set K is composed of arc-disjoint paths if and only if D' does not contain any impregnable Eulerian subgraph with respect to K. Moreover, checking whether or not D' contains an impregnable Eulerian subgraph can be done in polynomial time.* □

Theorem 3 can be used to deduce a particular case of the SPPDP for which the VSCP can be solved in polynomial time. This particular case occurs when the demand paths are arc-disjoint and corresponds to the unitary version of the SPPDP, that is, when the vehicle can carry at most one demand at a time. It follows that for the unitary SPPDP, the information relative to the order on the arc set of the vehicle closed walk is not necessary in a tractable representation of solutions to the SPPDP. It can be recovered in polynomial time from the conditions of Theorem 3. However, for the SPPDP, this information has to be taken into account in a tractable representation. These results provide the structure of the remainder of the paper. In fact, the next section is devoted to giving an integer linear programming formulation of the unitary SPPDP using a minimum representation, whereas Section 5 focuses on the general case.

4 The unitary SPPDP

4.1 Formulation for the unitary SPPDP

We present in this section an integer linear programming formulation for the unitary SPPDP. In this variant of the SPPDP, the vehicle can carry only one demand at the same time. Since we have supposed that a demand transportation cannot be split onto several paths, the unitary SPPDP occurs when $q^p + q^{p'} > Q$ for all distinct p and p' of the set P of demands. An instance of the unitary SPPDP then consists of an initial digraph $D = (V, A)$, a specific vertex v_0 of V called the depot, and a demand digraph $\Phi = (V, P)$.

According to the developments of Section 3, a minimal representation of solutions to the unitary SPPDP can be defined by only considering the sets of arcs of the vehicle closed walk and demand paths. We then introduce the following two sets of variables, the first one corresponding to the arcs of the demand paths, and the second one representing which arcs belong to the vehicle closed walk. Let $x \in \{0,1\}^{A \times P}$ be so that

$$x_a^p = \begin{cases} 1 & \text{if the demand } p \text{ is carried on the arc } a, \\ 0 & \text{otherwise,} \end{cases}$$

for all arcs $a \in A$ and for all demands $p \in P$, and let $y \in \{0,1\}^A$ be so that

$$y_a = \begin{cases} 1 & \text{if the vehicle traverses the arc } a, \\ 0 & \text{otherwise,} \end{cases}$$

for all arcs $a \in A$. Let $S_U(D, v_0, \Phi)$ denote the set of vectors (x, y) associated with the feasible solutions to the unitary SPPDP. A vector (x, y) of

$S_U(D, v_0, \Phi)$ satisfies the following inequalities

$$\sum_{a \in \delta^{\text{out}}(W)} y_a - y_{a'} \geq 0 \qquad \forall\, W \subseteq V \text{ with } v_0 \in W, \forall\, a' \in A[\overline{W}], \quad (1)$$

$$\sum_{a \in \delta^{\text{out}}(v)} y_a - \sum_{a \in \delta^{\text{in}}(v)} y_a = 0 \qquad \forall\, v \in V, \quad (2)$$

$$\sum_{a \in \delta^{\text{out}}(v)} x_a^p - \sum_{a \in \delta^{\text{in}}(v)} x_a^p = b_v^p \qquad \forall\, p \in P,\ \forall\, v \in V, \quad (3)$$

$$\sum_{a \in \delta^{\text{out}}(W)} x_a^p - \sum_{a \in \delta^{\text{out}}(v)} x_a^p \geq 0 \qquad \begin{array}{l} \forall\, p \in P, \forall\, W \subseteq V \text{ with } o^p, d^p \in W, \\ \forall\, v \in \overline{W}, \end{array} \quad (4)$$

$$\sum_{a \in \delta^{\text{out}}(v)} x_a^p \leq 1 \qquad \forall\, p \in P,\ \forall\, v \in V, \quad (5)$$

$$y_a - \sum_{p \in P} x_a^p \geq 0 \qquad \forall\, a \in A, \quad (6)$$

where the number b_v^p defined by

$$b_v^p = \begin{cases} 1 & \text{if } v = o^p, \\ -1 & \text{if } v = d^p, \\ 0 & \text{otherwise,} \end{cases}$$

represents the supply/demand associated with vertex $v \in V$ with respect to demand $p \in P$. In fact, constraints (1) and (2) imply that the arc set $A_y = \{a \in A : y_a = 1\}$, corresponding to the vehicle closed walk, induces an Eulerian digraph passing by v_0. Constraints (1), hereafter called *connectivity constraints*, actually ensure that v_0 is incident with at least one arc of A_y and that the induced digraph is weakly connected. Constraints (2) are the *flow-conservation constraints* and enforce the number of arcs entering any vertex to be equal to the number of arcs leaving this vertex. Constraints (3) are the *flow-conservation constraints* associated with the demands. Constraints (4) are the *connectivity-demand inequalities* and ensure that the set of arcs traversed by any demand leads to a weakly connected digraph. The *circuit constraints* (5) prevent the demands from passing more than once per vertex and then garantee, with constraints (3) and (4), that every demand is carried on a path from its origin to its destination. Constraints (6) are the *capacity constraints*. They impose that at most one demand is carried at the same time on an arc traversed by the vehicle.

Let S be the set of binary vectors (x, y) which satisfy (1)-(6). Theorem 3 implies that $S_U(D, v_0, \Phi) \subsetneq S$ since constraints (1)-(6) do not prevent the Eulerian digraph induced by A_y from containing an impregnable Eulerian subgraph with respect to the demand paths associated with variables x. We then need further inequalities to formulate the problem. In what follows, we describe a new class of valid inequalities.

Proposition 1 *Let $W \neq \emptyset$ be a proper vertex subset of V so that $v_0 \in W$, $A_\Phi[\overline{W}] \neq \emptyset$ and $\delta_\Phi(W) = \emptyset$. Then, the inequality*

$$\sum_{a \in \delta^{\text{out}}(W)} y_a - \sum_{p \in A_\Phi[\overline{W}]} \sum_{a \in \delta^{\text{out}}(W)} x_a^p \geq 1 \tag{7}$$

is valid for the unitary SPPDP.

Proof Assume that we are given a feasible solution (x, y) to the unitary SPPDP that violates an inequality of type (8), that is,

$$\sum_{a \in \delta^{\text{out}}(W)} y_a - \sum_{p \in A_\Phi[\overline{W}]} \sum_{a \in \delta^{\text{out}}(W)} x_a^p \leq 0.$$

From the feasibility of (x, y), we know that $y_a - x_a(A_\Phi[\overline{W}]) \geq 0$ for all $a \in A$ and then, we deduce that $y_a = x_a(A_\Phi[\overline{W}])$ for all $a \in \delta^{\text{out}}(W)$. Consequently, on any arc a leaving W with $y_a = 1$, the vehicle carries a demand having both its origin and its destination in $\overline{W}$. Since the depot v_0 belongs to W and $A_\Phi[\overline{W}] \neq \emptyset$, this implies that there is no arc of $\delta^{\text{out}}(W)$ to first reach the origin of a demand of $A_\Phi[\overline{W}]$. Therefore, (x, y) could not be a vector of $S_U(D, v_0, \Phi)$. □

Inequalities of type (7) will be called *vulnerability constraints*. In the following, we introduce a weaker version of inequalities (7). Let $W \neq \emptyset$ be a proper vertex subset of V so that $v_0 \in W$, $A_\Phi[\overline{W}] \neq \emptyset$ and $\delta_\Phi(W) = \emptyset$. Consider the inequality

$$\sum_{a \in \delta^{\text{out}}(W)} y_a - \sum_{p \in A_\Phi[\overline{W}]} \sum_{a \in \delta^{\text{out}}(W)} x_a^p + M \sum_{p \in A_\Phi[W]} \sum_{a \in \delta^{\text{out}}(W)} x_a^p \geq 1, \tag{8}$$

where M denotes a sufficiently large constant. It is not hard to see that inequality (8) is also valid for the unitary SPPDP. Indeed, every inequality (8) associated with a vertex subset W is nothing but a linear combination of the vulnerability constraint (7) associated with W and some trivial constraints $x_a^p \geq 0$. The value M may be any non-negative value. Inequalities of type (8) will be called *relaxed vulnerability constraints*. As it will be seen later, inequalities (8) can be separated in polynomial time for specific values of M.

We now prove that the relaxed vulnerability constraints (8) are sufficient to reduce the set S to $S_U(D, v_0, \Phi)$.

Proposition 2 *Any binary vector of S satisfying constraints* (8) *induces a feasible solution to the unitary SPPDP.*

Proof Let (x, y) be a vector of S, D_y be the Eulerian digraph induced by the arc set A_y of the vehicle closed walk, and $K = \{K_1, \ldots, K_p\}$ be the set of demand paths induced by variables x. Due to the unitary property, the paths of K are pairwise arc-disjoint. As previously mentioned, (x, y) induces a feasible solution to the unitary SPPDP if and only if the Eulerian digraph

D_y does not contain an impregnable Eulerian subgraph with respect to K. We then prove that the existence of an impregnable Eulerian subgraph, say $D' = (V', A')$, of D_y with respect to K implies that (x, y) violates a relaxed vulnerability constraint (8).

Consider the vertex subsets $V_1 = \{v \in V' : v = v_0 \text{ or } \delta_{D_y}(v) \setminus A' \neq \emptyset\}$ and $V_2 = V' \setminus V_1$. Let $W = V \setminus V_2$. We remark that $v_0 \in W$ and V_2 is composed of all the vertices of $V' \setminus \{v_0\}$ that are not incident with arcs of $A_y \setminus A'$. Clearly, we have $V_1 \subseteq W$ and

$$\delta^{\text{out}}_{D_y}(W) \subseteq \bigcup_{v \in V_1} \delta^{\text{out}}_{D'}(v). \tag{9}$$

We claim that any demand that is carried on an arc of A' has its origin and its destination in V_2. Consider a demand $p \in P$ carried on the path $K_p = (a_1, a_2, \ldots, a_q)$ with $q \geq 1$ so that at least one of the arcs of K_p belongs to A'. Suppose that $o^p \notin V_2$ and let a_j, $j \in \{1, 2, \ldots, q\}$, be the first arc of K_p in A'. If $j = 1$, then it is obvious that $o^p \in V_1$. Therefore, since the demand paths are pairwise arc-disjoint, the arc a_1 has no predecessor which means that o^p is not D'-impregnable. If $j \in \{2, 3, \ldots, q\}$, then the arc $a_{j-1} = (u, v)$ is in $A_y \setminus A'$ and $v \in V_1$. Thus, the arc a_j has no predecessor in A' which implies that v is not D'-impregnable. Using similar arguments, we can prove that $d^p \in V_2$.

From the definition of impregnable Eulerian subgraphs, we know that D' contains at least one arc which is traversed by a demand of P. The previous claim directly implies that there exists at least one demand whose origin and destination are in V_2. Moreover, since vertices of V_2 are only incident with arcs of A', there cannot exist a demand having exactly one extremity in V_2. As $V_2 = V \setminus W$, we also have $A_\Phi[\overline{W}] \neq \emptyset$ and $\delta_\Phi(W) = \emptyset$.

Let K' be the path subset of K induced by the demands of $A_\Phi[\overline{W}]$. The vertex set V_1 is thus composed of the vertices v of V' so that for every arc a of $\delta^{\text{out}}_{D'}(v)$, there exists an arc a' of $\delta^{\text{in}}_{D'}(v)$ with $a' \prec_{K_i} a$ for some path K_i of K'. From (9), we then obtain that $y_a - x_a(A_\Phi[\overline{W}]) = 0$ for all arcs a of $\delta^{\text{out}}_D(W)$. Furthermore, since the demand paths of K are pairwise arc-disjoint, it is straightforward to see that $x^p_a = 0$ for all arcs a of $\delta^{\text{out}}_D(W)$ and for all demands p of $A_\Phi[W]$. Consequently, the vulnerability constraint (8) associated with W is violated by (x, y). □

Using Propositions 1 and 2, we can now define the set of the feasible solutions to the unitary SPPDP.

Theorem 4 *The set* $\{(x, y) \in \{0, 1\}^{A \times P} \times \{0, 1\}^A : (x, y)$ *satisfies* (1)−(6), (8)$\}$ *corresponds to the set* $S_U(D, v_0, \Phi)$ *of the feasible solutions to the unitary SPPDP.* □

The previous theorem allows to formulate the unitary SPPDP as the following integer linear program

$$\min\{c^T y \mid (x, y) \in \text{conv}(S_U(D, v_0, \Phi))\},$$

hereafter denoted by P_U.

We also point out that the connectivity constraints (1) are no more necessary in the formulation P_U if arc cost vector c is positive since the digraph induced by A_y is weakly connected by (8). We are now interested in the complexity of solving the linear relaxation of P_U. Before stating it, we consider the separation problem for inequalities (8). We then show that inequalities (8) can be separated in polynomial time for any $M \geq \max_{a\in A, p\in P}\{\frac{1}{x_a^p}\}$. In consequence, the value of M depends on the precision used for coding rational values.

Proposition 3 *Suppose that* $M \geq \max_{a\in A, p\in P}\{\frac{1}{x_a^p}\}$. *Let* $(\bar{x}, \bar{y})$ *be a vector of* $\mathbb{R}_+^{A\times P} \times \mathbb{R}^A$ *satisfying constraints* (4) *and* (6). *The separation problem for the relaxed vulnerability constraints* (8) *with respect to* $(\bar{x}, \bar{y})$ *can be solved in polynomial time.*

Proof We show that the separation problem associated with the relaxed vulnerability constraints reduces to a polynomial number of computations of minimum cut in an auxiliary digraph. Let $\hat{D} = (\hat{V}, \hat{A})$ be the digraph obtained from $D = (V, A)$ by contracting the vertices o^p and d^p into a vertex v^p for every demand $p \in P$. Let $\hat{w} \in \mathbb{R}^{\hat{A}}$ be the vector associated with the arcs of $\hat{A}$ so that for all $(u, v) \in \hat{A}$, the value $\hat{w}_{(u,v)}$ corresponds to the sum of the values $y_a - x_a(P)$ associated with the arcs $a \in A$ for which the contraction of the vertices o^p and d^p into v^p for all $p \in P$ transforms a into (u, v) in $\hat{D}$.

For all demands $p \in P$, we denote by V^p the vertices of the digraph covered by the arc set $\{a \in A : x_a^p > 0\}$. We then define, for every demand $p \in P$, the arc set

$$B^p = \{(v^p, v) : v \in V^p \setminus \{o^p, d^p\}\}.$$

Consider the digraph $\tilde{D} = (\hat{V}, \tilde{A})$ obtained by adding the arcs of B^p in $\hat{D}$ for all $p \in P$. We then have $\tilde{A} = (\cup_{p\in P} B^p) \cup \hat{A}$. Let $\tilde{w} \in \mathbb{R}^{\tilde{A}}$ be the vector associated with the arcs of $\tilde{D}$ so that

$$\tilde{w}_a = \begin{cases} +\infty \text{ if } a \in \cup_{p\in P} B^p, \\ \hat{w}_a \quad \text{otherwise,} \end{cases} \qquad \forall\ a \in \tilde{A}.$$

Consider a demand of P, say $\bar{p}$. We now show that a relaxed vulnerability constraint (8) associated with a vertex subset W so that $\bar{p} \in A_\Phi[\overline{W}]$ is violated if and only if there exists a $v_0 v^{\bar{p}}$-cut whose weight is less than 1 in the digraph $\tilde{D}$ with weights $\tilde{w}$.

Suppose that the vector $(\bar{x}, \bar{y})$ violates the vulnerability constraint associated with a vertex subset W so that $\bar{p}$ has both extremities in $\overline{W}$. We then have

$$\sum_{a\in\delta^{\text{out}}(W)} \bar{y}_a - \sum_{p\in A_\Phi[\overline{W}]} \sum_{a\in\delta^{\text{out}}(W)} \bar{x}_a^p + M \sum_{p\in A_\Phi[W]} \sum_{a\in\delta^{\text{out}}(W)} \bar{x}_a^p < 1.$$

Since $\bar{x}$ is non-negative and due to the hypothesis made on the value of M, this means that, for every demand $p \in A_\Phi[W]$, $\bar{x}_a^p = 0$ for all arcs $a \in \delta^{\text{out}}(W)$. Let W' be the vertex subset obtained from W by contracting the origin o^p and

the destination d^p into a vertex v^p for every demand $p \in P$. It is clear that $v_0 \in W'$ and $v^{\bar{p}} \in \overline{W}'$. Since $\delta_\Phi(W) = \emptyset$, by definition of $\hat{w}$, we have

$$\hat{w}(\delta^{\text{out}}(W')) = \sum_{a \in \delta^{\text{out}}(W)} \bar{y}_a - \sum_{p \in A_\Phi[\overline{W}]} \sum_{a \in \delta^{\text{out}}(W)} \bar{x}_a^p.$$

Consider a demand p of $A_\Phi[W]$. Since $\bar{x}$ satisfies constraints (4), the digraph induced by $\{a \in A : x_a^p > 0\}$ is weakly connected and no arc of this digraph belongs to $\delta^{\text{out}}(W)$. We then have $V^p \subseteq W$. Consequently, there does not exist an arc of B^p, $p \in P$, belonging to $\delta^{\text{out}}(W')$. We then deduce that $\hat{w}(\delta^{\text{out}}(W')) = \tilde{w}(\delta^{\text{out}}(W')) < 1$. The cut associated with W' has a weight less than 1 in the digraph $\tilde{D}$. Finally, as $v_0 \in W'$ and $v^{\bar{p}} \in \overline{W}'$, there exists a $v_0 v^{\bar{p}}$-cut whose weight is less than 1 in $\tilde{D}$ with weights $\tilde{w}$. The proof of the converse is similar.

Looking for a violated vulnerability constraint associated with a vertex subset W so that $\bar{p}$ belongs to $A_\Phi[\overline{W}]$ then reduces to find a minimum $v_0 v^{\bar{p}}$-cut in $\tilde{D}$ with weights $\tilde{w}$. Since $\bar{p}$ is any demand of P such that $v_0 \notin \{o^p, d^p\}$, this means that the separation problem associated with the relaxed vulnerability constraints (8) reduces to the computation of at most $|P|$ minimum cuts in the digraph $\tilde{D}$ with weights $\tilde{w}$. Moreover, as $(\bar{x}, \bar{y})$ satisfies the capacity constraints (6) and the weight vector $\tilde{w}$ is non-negative, the computation of every minimum cut can be performed in polynomial time. Since the construction of $\tilde{D}$ from D can also be performed in polynomial time, it follows that the separation problem associated with the relaxed vulnerability constraints (8) is polynomial-time solvable. □

Theorem 5 *The linear relaxation of P_U can be solved in polynomial time.*

Proof Since P_U contains a polynomial number of variables, and constraints (2),(3),(5),(6) appear in a polynomial number in P_U, the complexity of solving its linear relaxation only depends on the separation problems associated with inequalities (1), (4) and (8) for any vector $(\bar{x}, \bar{y}) \in [0,1]^{A \times P} \times [0,1]^A$ satisfying (2),(3),(5) and (6).

Separating (1) can be reduced to $|A|$ minimum $v_0 v^{a'}$-cuts, where $v^{a'}$ is the vertex obtained from the contraction of a' and the arc weight function is given by $\bar{y} \geq 0$. The separation problem of constraints (4) associated with any demand $p \in P$ reduces to the computation of $|V|$ $v^p v$-minimum cuts, where v^p is the vertex obtained by contracting o^p and d^p and v is any vertex of $V \setminus \{v^p\}$, the arc weight function being given by $\bar{x} \geq 0$. Therefore, inequalities (1) and (4) are polynomial-time separable.

Suppose now that $(\bar{x}, \bar{y})$ satisfies constraints (4). Since $(\bar{x}, \bar{y})$ also satisfies inequalities (6) and $\bar{x} \geq 0$, it follows, from Proposition 3, that the separation problem for inequalities (8) can be also solved in polynomial time. □

From the previous results, it is clear that by replacing inequalities (8) by inequalities (7) in the formulation P_U of the unitary SPPDP given by Theorem 4, one gets a strengthened formulation of the problem which might be

a stronger formulation for a cutting plane based algorithm. However, we do not have an exact efficient algorithm for separating inequalities (7). We even conjecture that the separation problem for these inequalities is NP-complete. For this, we will use formulation P_U in our branch-and-cut algorithm. However, we can remark that if a relaxed vulnerability constraint (8) associated with a vertex subset W is violated by a vector (x, y) so that x is non-negative, then, (x, y) also violates the vulnerability constraint (7) associated with W. Therefore, during our branch-and-cut algorithm, once a violated relaxed vulnerability constraint (8), associated with a vertex subset W has been found during the separation procedure, the constraint we add in the current linear program is the vulnerability constraint (7) associated with W in order to strenghten the linear relaxation. In consequence, the big constant M does not appear in the constraints during the algorithm. One can also remark that the connectivity-demand inequalities (4) are not necessary in the formulation since no cost is associated with the demands. However, they are necessary from an optimization point of view in order to exactly solve the separation problem associated with the relaxed vulnerability constraints (8).

In Lacroix (2009), it has been shown that, if the digraph D is complete, the origins and destinations of the demands are distinct nodes of $V \setminus \{v_0\}$ and the costs satisfy the triangle inequalities, allowing the vehicle to pass by any arc more than once does not change the optimal value of the unitary SPPDP. This result has an interesting application for the PSCP. Indeed, we can suppose, without loss of generality, that any instance of the PSCP satisfies the three previous assumptions. These latter can be fulfilled by considering copies of nodes and by computing shortest paths between the two extremities of each edge. Furthermore, as the vehicle may pass several times per edge in the PSCP, we can replace each edge by two opposite arcs having the same cost as the edge. Therefore, this instance is also an instance of the unitary SPPDP and the only difference between these two problems is that the PSCP gives the possibility for the vehicle to pass by every arc more than once whereas the unitary SPPDP does not. However, the result given in Lacroix (2009) implies that this possibility does not change the optimal value, which means that we can restrict our attention to solutions in which every arc is traversed by the vehicle at most once. Thus, the two problems are equivalent from an optimization point of view and the formulation P_U, devised for the unitary SPPDP, is then valid for the PSCP providing, to the best of our knowledge, the first one dedicated to this problem.

4.2 Experimental results

In this section, we present a branch-and-cut algorithm for the unitary SPPDP. Our aim is to determine if the formulation P_U has an interest from an optimization point of view. For this, the algorithm does not consider constraints in addition to the vulnerability constraints (7) and those belonging to P_U.

Indeed, we think that an efficient branch-and-cut must be based on a deep polyhedral study and such work is clearly beyond the scope of this paper. Furthermore, as in the tested instances, arc costs are positive, we do not consider the connectivity constraints (1) since these latter are not necessary.

To start the optimization, we consider the following program, called P_0, given by the inequalities of P_U that appear in polynomial number, that is,

$$P_0 = \{\min c^T y \mid (x, y) \in [0,1]^{A \times P} \times [0,1]^A : (x, y) \text{ satisfies } (2), (3), (5), (6)\}.$$

An important task for the branch-and-cut algorithm is to determine whether or not an optimal solution of the relaxation of P_U is feasible. An optimal solution $(\bar{x}, \bar{y})$ is feasible for the unitary SPPDP if it is an integer vector that satisfies the connectivity-demand constraints (4) and the relaxed vulnerability constraints (8).

If an optimal solution $(\bar{x}, \bar{y})$ of the linear relaxation is not feasible, the branch-and-cut algorithm generates further valid inequalities that are violated by $(\bar{x}, \bar{y})$. The separation of valid inequalities is performed in the following order:

- connectivity-demand inequalities,
- relaxed vulnerability inequalities.

We remark that all inequalities are global (*i.e.*, valid in all the branch-and-cut tree) and several constraints may be added at each iteration. Moreover, we go to the next class of inequalities only if we do not find any violated inequalities in the current class.

To separate the connectivity-demand inequalities, we use the algorithm described in the proof of Theorem 5. We separate the relaxed vulnerability constraints using the algorithm described in the proof of Proposition 3. We remark that the algorithm separates inequalities (8) in an exact way because it is performed only if no violated connectivity-demand constraint is found. Moreover, as noted before, each time a violated relaxed vulnerability constraint (8) associated with a vertex subset W is found, we add the violated vulnerability constraint (7) induced by W instead, in order to enforce the linear relaxation.

To store the generated inequalities, we created a pool whose size increases dynamically. All the generated inequalities are put in the pool and are dynamic, *i.e.*, they are removed from the current LP when they are not active. We first separate inequalities from the pool. If all inequalities in the pool are satisfied by the current LP-solution, we separate the classes of inequalities in the order given above.

The branch-and-cut algorithm has been implemented in C++, using CBC (Lougee-Heimer 2003) to manage the branch-and-cut tree, CLP (Lougee-Heimer 2003) as LP-solver and BGL (Siek et al. 2000) for the maximum-flow algorithm used in the separation algorithms. It was tested on processor 2.5 GHz with 6 Gb RAM, running under Linux. We fixed the maximum CPU time limit to 2 hours.

Results are presented here for randomly generated instances. The initial digraphs in the instances are complete digraphs with positive arc costs that

come from the asymetric instances of the TSP Library (Reinelt 1991). We consider digraphs with up to 101 nodes. The origins and destinations of the demands are randomly chosen so that each vertex different from the depot is incident to at most one demand and no demand is incident to the depot.

In Table 1, the entries are :

- $|V|$: the number of nodes in the initial digraph,
- $|P|$: the number of demands,
- o/t : the number of instances solved to optimality over the number of tested instances,
- N_{CD} : the number of generated connectivity-demand inequalities,
- N_{Vuln} : the number of generated vulnerability inequalities,
- N_T : the number of generated nodes in the branch-and-cut tree,
- Gap : the gap between the best upper bound and the lower bound obtained at the root node before branching,
- T_S : the time consumed in the separation algorithms in seconds,
- TT : the total time in seconds.

Table 1 summarizes results for the unitary SPPDP. Each line reports the average results obtained for five instances, all of them having the same number of nodes and demands, and the same arc costs. The five instances only differ by the origins and destinations of the demands. We remark that all the instances except 6 could be solved to optimality within the time limit. Two instances with 81 nodes and 15 and 25 demands respectively could not be solved to optimality within two hours. Moreover, 4 instances with 101 nodes and 10, 30, 35 and 35 demands respectively could neither be solved to optimality. The average number of generated connectivity-demand constraints is very small. Except for instances with 81 nodes and 5 demands, no more than 30 connectivity-demands constraints have been generated within the algorithm. However, a significant number of vulnerability constraints have been generated. We remark that the average gap between the best upper bound and the lower bound obtained before branching is very small. In fact, it does not exceed 1% for all the tested instances. Moreover, the number of nodes in the branch-and-cut tree is very small. Except for the instances with 81 nodes and 5 demands and instances with 101 nodes and 35 demands, this number does not exceed 6. We can also note that the time consumed for the separation of inequalities (4) and (8) is rather small with respect to the number of generated inequalities.

These experimental results show that the linear relaxation of P_U is really tight when every found violated relaxed vulnerability constraint (8) is replaced by the vulnerability constraint (7) induced by the same vertex subset. Indeed, we have a small number of nodes in the branch-and-cut tree and a small value of the gap. Finally, many instances could be solved without branching. However, the total time is quite long. We can remark, for example, that it needs more than 4000 seconds in the average to solve instances with 101 vertices and 50 demands whereas the branch-and-cut tree only contains few nodes and the number of generated constraints is 8 in the average. In fact, almost all the

$\|V\|$	$\|P\|$	o/t	N_{CD}	N_{Vuln}	N_T	Gap	T_S	TT
31	5	5/5	0.00	20.00	0.00	0.00	0.02	0.44
31	10	5/5	0.00	15.60	0.60	0.01	0.03	1.13
31	15	5/5	0.00	9.20	2.40	0.15	0.08	2.21
41	5	5/5	12.00	55.60	5.60	0.01	0.12	5.71
41	10	5/5	30.60	125.80	0.00	0.00	0.99	71.30
41	15	5/5	2.40	52.00	0.20	0.00	0.51	34.22
41	20	5/5	0.00	4.20	0.60	0.00	0.09	8.89
51	5	5/5	0.00	20.80	0.00	0.00	0.07	2.05
51	10	5/5	0.00	11.40	0.60	0.00	0.11	6.55
51	15	5/5	0.00	97.40	2.20	0.02	1.12	27.94
51	20	5/5	0.00	33.60	2.40	0.15	0.79	41.60
51	25	5/5	0.00	4.40	0.00	0.00	0.25	36.30
61	5	5/5	0.00	43.40	0.00	0.00	0.32	7.97
61	10	5/5	0.00	19.20	0.00	0.00	0.21	14.68
61	15	5/5	0.00	205.00	0.20	0.00	4.05	161.83
61	20	5/5	0.00	14.80	0.60	0.00	0.60	59.28
61	25	5/5	0.00	44.20	2.00	0.00	2.06	170.57
61	30	5/5	0.00	5.00	0.40	0.00	0.38	127.52
71	5	5/5	0.00	1.40	0.00	0.25	0.00	0.55
71	10	5/5	0.00	9.60	0.80	0.00	0.10	86.87
71	15	5/5	0.00	10.60	0.00	0.04	0.30	32.50
71	20	5/5	0.00	4.20	1.20	0.05	0.22	417.93
71	25	5/5	0.00	5.80	0.80	0.11	0.24	274.82
71	30	5/5	0.00	13.00	1.00	0.06	1.23	686.09
71	35	5/5	0.00	8.20	0.40	0.02	0.86	259.59
81	5	5/5	123.00	1140.20	421.00	0.23	103.06	1769.63
81	10	5/5	0.00	121.80	0.40	0.00	2.65	176.96
81	15	4/5	0.00	276.00	0.00	0.42	18.75	1561.71
81	20	5/5	0.00	219.80	1.40	0.00	13.01	1458.39
81	25	4/5	0.00	112.20	1.20	0.92	7.66	1931.71
81	30	5/5	0.00	16.00	2.40	0.00	1.86	845.37
81	35	5/5	0.00	9.60	2.80	0.00	1.34	1149.23
81	40	5/5	0.00	13.40	1.80	0.00	2.99	1404.16
91	5	5/5	0.00	152.40	0.80	0.01	1.58	72.18
91	10	5/5	0.00	466.40	2.60	0.01	13.60	518.38
91	15	5/5	0.00	511.80	1.40	0.05	18.65	857.56
91	20	5/5	0.00	405.80	0.00	0.00	22.32	955.36
91	25	5/5	0.00	139.00	1.80	0.00	9.70	1030.19
91	30	5/5	0.00	307.60	1.60	0.00	24.77	2086.76
91	35	5/5	0.00	147.80	4.20	0.00	16.10	2015.45
91	40	5/5	0.00	26.40	4.00	0.00	5.48	2431.55
91	45	5/5	0.00	8.20	1.80	0.00	4.84	3130.44
101	5	5/5	0.00	2.20	0.20	0.00	0.01	2.35
101	10	4/5	15.20	88.40	0.40	0.15	1.88	1457.59
101	15	5/5	0.00	2.00	0.20	0.00	0.08	428.35
101	20	5/5	0.00	3.80	0.80	0.00	0.27	510.40
101	25	5/5	0.00	25.80	0.00	0.05	3.24	3823.82
101	30	4/5	0.00	1.40	0.20	0.04	0.11	3228.09
101	35	3/5	0.00	31.80	14.60	0.43	10.92	3563.17
101	40	5/5	0.00	21.60	0.60	0.00	3.54	2268.91
101	45	5/5	0.00	16.20	1.00	0.00	6.78	2857.48
101	50	5/5	0.00	8.00	1.20	0.00	4.82	4075.84

Table 1 Results obtained for the unitary SPPDP

time is spent in solving the initial linear program P_0. Even if the number of variables and constraints of P_0 is polynomial, it becomes difficult to efficiently solve it when the size of the instances increases.

5 The SPPDP

5.1 Formulation for the SPPDP

We now focus on the general version of the SPPDP, that is, the only assumption we consider for the demand volumes is that each of them is smaller than or equal to the vehicle transportation capacity. In Section 3, we showed that a minimal representation of the solutions to the SPPDP consists of the sequence of arcs of the vehicle closed walk and the arc sets associated with the demand paths. For the unitary SPPDP, we presented in Section 4 an ILP formulation only based on the sets of arcs for the vehicle closed walk and the demand paths. In this section, we extend this formulation by adapting the inequalities (1)-(6) to arbitrary demand volumes and transportation capacity, and by adding some variables and constraints to order the arcs of the vehicle closed walk, and then to obtain a sequence of arcs.

We still consider the variable sets $x \in \{0,1\}^{A\times P}$ and $y \in \{0,1\}^{A}$. Obviously, the constraints (1)-(5) remain valid whatever the demand volumes and transportation capacity are. Furthermore, since we now consider the order on the arcs traversed by the vehicle, we can easily check if the vehicle closed walk traverses the arcs in the same order as in the demand paths. Thus, inequalities similar to constraints (7) and (8) are no more necessary in the formulation. Therefore, the only constraints that need to be adjusted are the capacity constraints (6) that are generalized to the following

$$Qy_a - \sum_{p\in P} q^p x_a^p \geq 0, \tag{10}$$

for all $a \in A$.

We now focus on modeling the sequence of arcs of the vehicle closed walk. A natural way to represent the order on this arc set is by considering the following linear ordering variables (Queyranne and Schulz 1994)

$$\eta_{aa'} = \begin{cases} 1 & \text{if } a \text{ precedes } a' \text{ in the vehicle closed walk,} \\ 0 & \text{otherwise,} \end{cases}$$

for every pair of distinct arcs $a, a' \in A$. Since the vector η defines a linear order on the arcs traversed by the vehicle, that is, on the arc set $A_y = \{a \in A : y_a = 1\}$, the variable set (y, η) can be seen as inducing a partial linear order on A. This concept of partial linear order was first introduced by Sirdey

and Kerivin (2007). Therefore, (y, η) has to satisfy the following constraints

$$y_a + y_{a'} - \eta_{aa'} - \eta_{a'a} \leq 1 \qquad \forall\ a, a' \in A,\ a \neq a', \tag{11}$$

$$\eta_{aa'} + \eta_{a'a} - y_a \leq 0 \qquad \forall\ a, a' \in A\ a \neq a', \tag{12}$$

$$\eta_{aa'} + \eta_{a'a''} - \eta_{aa''} - y_{a'} \leq 0 \qquad \forall\ a, a', a'' \in A,\ a \neq a' \neq a'' \neq a, \tag{13}$$

which were given for the so-called partial linear ordering polytope (Sirdey and Kerivin 2007).

At this point, the linear order defined by η may not correspond to a closed walk. We then have to add new constraints to enforce an alternate sequence of leaving and entering arcs at every vertex. Moreover, we also have to impose that, for every vertex $v \in V \setminus \{v_0\}$ (vertex v_0, respectively), the first arc incident with it is an entering (leaving, respectively) arc. This can be achieved by introducing the following equations

$$\sum_{a \in \delta^{\text{out}}(v) \setminus \{a'\}} \eta_{aa'} - \sum_{a \in \delta^{\text{in}}(v)} \eta_{aa'} + y_{a'} = 0 \tag{14}$$

for all vertices $v \in V \setminus \{v_0\}$ and for all $a' \in \delta^{\text{out}}(v)$. Similar constraints also hold for the depot v_0 as follows

$$\sum_{a \in \delta^{\text{out}}(v_0) \setminus \{a'\}} \eta_{aa'} - \sum_{a \in \delta^{\text{in}}(v_0)} \eta_{aa'} = 0, \tag{15}$$

for all $a' \in \delta^{\text{out}}(v_0)$. Constraints (14) are called *alternate constraints*, whereas constraints (15) are called *depot alternate constraints*. Constraints (14) express the fact that the tail of an arc $a \in A_y$, if it is not v_0, has been entered one more time than left before a is considered in the sequence. In a similar way, constraints (15) express the fact that the depot v_0 has been entered as many time as left before an arc leaving v_0 is considered in the sequence. We remark that constraints (14) and (15) are defined for leaving arcs. It is clear that these constraints could then be replaced by similar inequalities that would be defined for entering arcs (Lacroix 2009).

Let S_{CW} denote the set of binary vectors (y, η) that induce closed walks in D starting at v_0. We have the following result

Proposition 4 *The set S_{CW} is given by*

$$S_{CW} = \{(y, \eta) \in \{0,1\}^A \times \{0,1\}^{\binom{A}{2}} : (y, \eta) \text{ satisfies } (2), (11)\text{-}(15)\}.$$

Proof Consider a closed walk C of D starting at v_0. Let (y, η) be the vector so that $y_a = 1$ if and only if $a \in C$ and $\eta_{aa'} = 1$ if and only if a appears before a' in C. It is straightforward to see that (y, η) satisfies constraints (2),(11)-(15).

We now prove that a binary vector (y, η) that satisfies (2),(11)-(15), induces a closed walk in D starting at v_0. As previously mentioned, η corresponds to a linear order on the arc set A_y. Constraints (14) imply that for every vertex $v \in V \setminus \{v_0\}$, the first arc in the order incident with it is an entering one. We then deduce that the first arc of the linear order, say a_1, leaves v_0. Because of

constraints (14) and (15), the second arc in the order leaves the head of a_1. By repeating this argument, we clearly deduce that (y, η) corresponds to a walk starting at v_0. Moreover, since y satisfies constraints (2), this walk is then a closed walk. This latter result ends the proof. □

We remark that the constraints (2) associated with the vertices $v \in V \setminus \{v_0\}$ are not necessary to describe the set S_{CW}. The one associated with the depot v_0 only needs to be taken into account. Similarly, the constraints (1) do not appear in the description of S_{CW} since we now consider a linear order η on A_y. In fact, this latter ensures that the digraph induced by A_y is weakly connected.

To obtain the set of the feasible solutions to the SPPDP, that we will hereafter denote by $S_G(D, v_0, \Phi)$, we finally need to combine the closed walk and the demand paths in such a way that the closed walk respects all the demand paths. To do so, we introduce the *demand precedence constraints*

$$x_a^p + x_{a'}^p - \eta_{aa'} \leq 1 \tag{16}$$

for all $p \in P$, for all $v \in V \setminus \{o^p, d^p\}$, for all $a \in \delta^{\text{in}}(v)$ and for all $a' \in \delta^{\text{out}}(v)$. These inequalities allow every demand p to be carried from a vertex $v \in V \setminus \{o^p, d^p\}$ through a leaving arc $a' \in \delta^{\text{out}}(v)$ if and only if the arc $a \in \delta^{\text{in}}(v)$ used for the transportation of the demand appears before a' in the closed walk. Clearly, inequalities (16) are valid for the SPPDP. Moreover, from constraints (5), the demands are carried on paths, that is, they can go through every vertex at most once. Therefore, constraints (16) suffice to ensure that the closed walk respects all the demand paths. We can now define the set $S_G(D, v_0, \Phi)$ of the binary vectors (x, y, η) associated with the feasible solutions to the general SPPDP. We remark that the connectivity-demand inequalities (4) are not necessary in the formulation since constraints (16) already ensure that the set of arcs traversed by each demand corresponds to a walk. We then remove these inequalities in the formulation of the general case. Using all the previous results, a description of $S_G(D, v_0, \Phi)$ is given by the next theorem

Theorem 6 *The set $S_G(D, v_0, \Phi)$ of the feasible solutions to the SPPDP is $\{(x, y, \eta) \in \{0,1\}^{A \times P} \times \{0,1\}^A \times \{0,1\}^{\binom{A}{2}} : (x, y, \eta)$ satisfies (2), (3), (5), (10), (11)-(16)$\}$.* □

Using the previous theorem, we can now formulate the general SPPDP as the following integer linear program P_G

$$\min\{c^T y \mid (x, y, \eta) \in \text{conv}(S_G(D, v_0, \Phi))\}.$$

The formulation P_G contains a polynomial number of variables and constraints which leads to the following theorem

Theorem 7 *The linear relaxation of P_G can be solved in polynomial time.* □

5.2 Experimental results

In this section, we present the experimental results obtained by solving the SPPDP using a branch-and-bound algorithm. Since the formulation P_G contains a polynomial number of variables and constraints, we entirely put the formulation in the commercial interactive solver CPLEX 11.2 (Ilog, Inc. 2003). The algorithm was tested on processor 2.5 GHz with 6 Gb RAM, running under Linux. We fixed the maximum CPU time to 3 hours.

Results are presented for randomly generated instances. In these instances, the density of the initial digraph, which is the ratio of the number of arcs over the number of nodes, is a parameter of the instances. The digraphs are obtained from random graphs, generated using RUDY (Rinaldi 1996), by replacing each edge by two opposite arcs having the same cost as the edge. The origin, destination and volume of each demand are randomly chosen. The capacity of the vehicle is a random number that is greater than or equal to the volume of each demand.

Table 2 reports the running CPU time necessary to solve the instances with the branch-and-bound algorithm and the gap between the best lower and upper bounds. The entries of Table 2 different from those of Table 1 are the following:

- dens : the density in percentage of the initial digraph. It is a scalar in the interval [0,100]. The number of arcs of the digraph is given by the closest integer to $n * (n-1) * \text{dens}/100$,
- Gap2 : the gap between the best upper bound and the best lower bound,

We remark that for instances with less than 10 nodes, each instance could be solved to optimality within the time limit. Also, instances of 10 (11, 13, respectively) nodes and a density of 40 (30, 20, respectively) could be solved to optimality. However, many instances could not be solved to optimality within the time limit, in particular those with at least 12 nodes, a number of demands greater than or equal to 5 and a density greater than or equal to 30. Moreover, the algorithm could not find any feasible solution for some instances, as it is the case for instances with 12 nodes and a density 40. Finally, we remark that the gap between the best lower and upper bounds when the instances are not solved to optimality is quite big. This can be explained by the fact that, although the formulation P_G contains a polynomial number of binary variables and constraints, these numbers fastly grow with respect to the size of the instance. These results show the necessity of devising an algorithm dedicated to the SPPDP taking into account the specificities of the formulation P_G. For instance, although the transity constraints (13) are in polynomial number, it would be more appropriate to not add all of them in the initial linear relaxation in order to not uselessly increase the size of the linear program. Moreover, further valid inequalities are necessary in order to strenghten the linear relaxation. For this, a deep investigation of the polyhedral structure of the problem is necessary.

$\|V\|$	dens	$\|P\|$	Gap2	TT
7	40	3	0.00	0.08
7	40	5	0.00	1.83
7	50	3	0.00	24.20
7	50	5	0.00	23.21
8	40	3	0.00	0.38
8	40	5	0.00	0.79
8	50	3	0.00	66.85
8	50	5	0.00	36.93
9	40	3	0.00	0.39
9	40	5	0.00	1.49
9	50	3	0.00	26.06
9	50	5	0.00	5365.54
10	40	3	0.00	7152.56
10	40	5	0.00	10800.00
10	50	3	41.34	10800.00
10	50	5	32.90	10800.00
11	30	3	0.00	735.68
11	30	5	0.00	3620.89
11	30	7	0.00	2855.30
11	40	3	14.32	10800.00
11	40	5	13.68	10800.00
12	30	3	0.00	5961.10
12	30	5	8.42	10800.00
12	30	7	47.25	10800.00
12	40	3	0.00	5961.10
12	40	5	-	10800.00
12	40	7	-	10800.00
13	20	3	0.00	2370.24
13	20	5	0.00	1564.48
13	20	7	0.00	2456.15
14	20	3	0.00	3.71
14	20	5	0.00	817.24
14	20	7	18.27	10800.00
14	20	10	18.24	10800.00

Table 2 Results obtained for the SPPDP

6 Concluding remarks

In this paper, we have introduced a new NP-hard problem which is an extension of the single-vehicle pickup and delivery problem where every demand can be carried using reloads. Based on some new complexity results, we have defined the minimal representations of the solutions to this problem for both the unitary and the general cases. We have then focused on the unitary SPPDP since we have proven that, in this case, only the arc sets associated with the vehicle closed walk and the demand paths are necessary to fully represent the feasible solutions. We have thus given an integer linear programming formulation whose linear relaxation is polynomial-time solvable and we have presented some experimental results obtained using a branch-and-cut algorithm. We have also pointed out that this formulation is also valid for the preemptive stacker crane problem for which, to the best of our knowledge, there did not exist

a dedicated formulation yet. Next, we have extended this formulation to the general SPPDP by considering variables representing a linear order on the arcs of the vehicle closed walk. Once again, we have obtained a polynomial-time solvable linear relaxation. We have finally reported some experimental results obtained by solving the formulation with a branch-and-bound algorithm.

The results obtained for the unitary SPPDP show the interest of the formulation given in this paper. Indeed, we could see that almost all instances were solved to optimality without branching. In all cases, the gap between the best upper bound and the lower bound obtained before branching is very small, so as the number of nodes in the branch-and-bound tree. However, the major weakness of our algorithm is the time spent for solving the initial linear relaxation. A way to bypass this problem is to decrease the size of the initial linear relaxation. As we can see, our formulation describes the demand paths using an arc-node approach. It would really be interesting to use a path formulation instead. The approach would use an exponential number of variables for the demand paths in the formulation but it could be solved using a branch-and-cut-and-price algorithm.

Moreover, in order to better understand the structure of the solutions of the unitary SPPDP and improve its resolution, it would be of big interest to study the polytope of the solutions of this problem. This study would also allow us to identify facet defining inequalities that may strengthen the linear relaxation and speed up the resolution of the problem.

The results obtained for the SPPDP also indicate that a deep study is necessary to obtain an efficient branch-and-cut algorithm for this problem. These investigations form the guidelines of our future work.

Acknowledgements We would like to thank the anonymous referees for their valuable comments that permitted us to considerably improve the paper.

References

Anily S, Gendreau M, Laporte G (2006) The preemptive swapping problem on a tree. preprint

Atallah M, Kosaraju S (1988) Efficient Solutions to Some Transportation Problems with Applications to Minimizing Robot Arm Travel. SIAM Journal on Computing 17:849–869

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

Cortés C E, Matamala M, Contardo C (2005) The Pickup and Delivery Problem with Transfers: Formulation and Solution Approaches. In: VII French - Latin American Congress on Applied Mathematics, Springer

Frederickson G, Guan D (1992) Preemptive Ensemble Motion Planning on a Tree. SIAM Journal on Computing 21:1130–1152

Grünert T, Sebastian H (2000) Planning models for long-haul operations of postal and express shipment companies. European Journal of Operational Research 122(2):289–309

Ilog, Inc (2003) Solver cplex. http://www.ilog.fr/products/cplex/

Kerivin H L M, Lacroix M, Mahjoub A R, Quilliot A (2008) The splittable pickup and delivery problem with reloads. European Journal of Industrial Engineering 2(2):112–133

Kerivin H L M, Lacroix M, Mahjoub A R (2010) On the complexity of the Eulerian closed walk with precedence path constraints problem. Electronic Notes on Discrete Mathematics (36):899–906

Lacroix M (2009) Le problème de ramassage et livraison préemptif : complexité, modèles et polyèdres. PhD thesis, Université Blaise-Pascal, Clermont-II, France

Lougee-Heimer R (2003) The common optimization interface for operations research. IBM Journal of Research and Development 47(1):57–66

Mitrović-Minić S, Laporte G (2006) The pickup and delivery problem with time windows and transshipment. INFOR 44:217–227

Oertel P (2000) Routing with Reloads. Doktorarbeit, Universität zu Köln

Parragh S, Doerner K, Hartl R (2008) A survey on pickup and delivery problems. Journal für Betriebswirtschaft 58(1):21–51

Queyranne M, Schulz A (1994) Polyhedral approaches to machine scheduling. Tech. rep.

Reinelt G (1991) TSPLIB - A Traveling Salesman Problem Library. ORSA Journal on Computing 3(4):376–384

Renaud J, Boctor F F, Laporte G (2002) Perturbation heuristics for the pickup and delivery traveling salesman problem. Computers & Operations Research 29:1129–1141

Rinaldi G (1996) RUDY: a generator for random graphs. http://www-user.tu- chemnitz.de/ helmberg/sdp_software.html

Savelsberg M W P, Sol M (1995) The general pickup and delivery problem. Transportation Science 29(1):17–29

Schrijver A (2002) Combinatorial Optimization: Polyhedra and Efficiency. Springer

Siek J, Lee L, Lumsdaine A (2000) Boost graph library. http://www.boost.org/libs/graph/

Sirdey R, Kerivin H L M (2007) Polyhedral combinatorics of a resource-constrained ordering problem part I: on the partial linear ordering polytope. Tech. rep., PE/BSC/INF/017912 V01

Vygen J (1995) NP-completeness of some edge-disjoint paths problems. Discrete Applied Mathematics 61(1):83–90

Appendix

Proof of Theorem 1 : We prove the NP-completeness of the DPCP by reducing the so-called arc-disjoint paths in quasi-topological digraph problem to it. Before stating precisely this new NP-complete problem, we introduce what a quasi-topological digraph is. Let $G = (V_G, A_G)$ be a simple acyclic digraph with vertex set $V_G = \{v_1, v_2, \ldots, v_n\}$. We suppose that the digraph G contains a Hamiltonian path whose arcs are those of the set $A_H = \{(v_i, v_{i+1}) : i = 1, 2, \ldots, n-1\}$. The arcs of $A_J = A_G \setminus A_H$ are called *jump arcs*. We remark that since G is simple and acyclic, any arc of A_J is of the form (v_i, v_j) with $1 \leq i < j \leq n$ and $j - i > 1$. The digraph G then is *quasi-topological* if all the indegrees and outdegrees are bounded by two, and every vertex is incident with at most three arcs. Due to the definition of G, any vertex of V_G is incident with at most one jump arc. Let $S = \{(s_i, t_i) : i = 1, 2, \ldots, k\}$ be a set of pairs of distinct vertices of G so that every vertex of G belongs to at most one pair, $\deg^{\text{in}}(s_i) \leq 1$ and $\deg^{\text{out}}(t_i) \leq 1$ for all $i = 1, \ldots, k$. The *arc-disjoint paths in Quasi-Topological Digraph Problem (QTDP)* consists of checking whether there exist k arc-disjoint paths $L_1, L_2, \ldots, L_k$ so that L_i is

a s_it_i-path of G for $i = 1, 2, \ldots, k$. In the following claim, we prove that the QTDP is NP-complete.

Claim The arc-disjoint paths in quasi-topological digraph problem is NP-complete.

Proof We prove the NP-completeness of the QTDP by a reduction from the so-called arc-disjoint paths in acyclic digraph problem. This latter can be stated as follows. The *arc-disjoint paths in Acyclic Digraph Problem (ADP)* consists, given an acyclic digraph $G' = (V_{G'}, A_{G'})$ and a set S' of k' pairs of distinct vertices (s'_i, t'_i), of checking if there exist k arc-disjoint paths $L'_1, L'_2, \ldots, L'_{k'}$ so that L'_i is a $s'_it'_i$-path of G' for all $i = 1, \ldots, k'$. This problem is known to be NP-complete (Vygen 1995). An example of an instance of the ADP is depicted in Figure 2, where the information relative to S' is enclosed in parentheses.

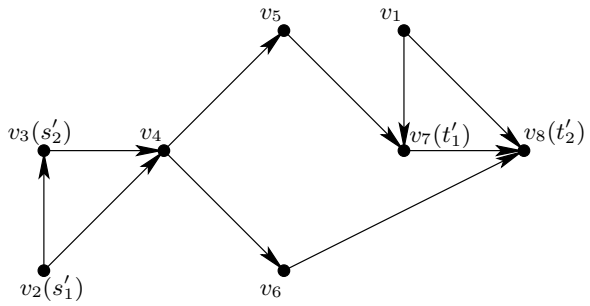

Fig. 2 An instance of the ADP

First, for $i = 1, 2, \ldots, k'$, we create two new vertices s_i and t_i and two new arcs (s_i, s'_i) and (t'_i, t_i). Let $S = \{(s_i, t_i) : i = 1, 2, \ldots, k'\}$. Notice that no vertex belongs to two pairs of S, which might not be the case with S'. Consider the graph $G^* = (V_{G^*}, A_{G^*})$ where $V_{G^*} = V_{G'} \cup \{s_i, t_i : i = 1, 2, \ldots, k'\}$ and $A_{G^*} = A_{G'} \cup \{(s_i, s'_i), (t'_i, t_i) : i = 1, 2, \ldots, k'\}$. Since G^* is acyclic, we can compute a topological order T of G^*. Moreover, T is choosen in such a way that vertex s'_i (t_i, respectively) directly follows s_i (t'_i, respectively) in T for all $i = 1, 2, \ldots, k'$. Let $\tilde{A}$ be the set of arcs (u, v) so that v is immediately after u in T and the arc (u, v) does not belong to A_{G^*}. It is straightforward to see that the digraph $\tilde{G}$ induced by $A_{\tilde{G}} = A_{G^*} \cup \tilde{A}$ contains a Hamiltonian path traversing the vertices of V_{G^*} in the same order as in T. We denote by A'_H the arcs of the Hamiltonian path. The graph obtained from the one of Figure 2 by the foregoing construction is given in Figure 3, where the dashed arcs represent the arc set $\tilde{A}$.

At this point, the digraph $\tilde{G}$ may not be quasi-topological since the degree conditions may not be satisfied or it may contain multiple arcs. Let $\tilde{V}$ be the subset of vertices v of V_{G^*} so that either $r_v = \deg^{\text{in}}_{\tilde{G}}(v) > 2$, $s_v = \deg^{\text{out}}_{\tilde{G}}(v) > 2$,

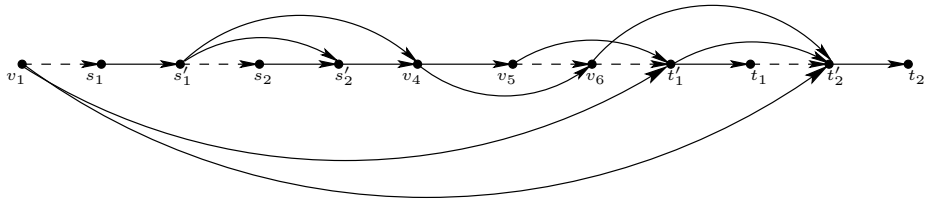

Fig. 3 Digraph $\tilde{G}$

or $d_v = \deg_{\tilde{G}}(v) > 3$. $\tilde{V}$ does not contain any vertex of a couple of S since we have $\deg^{\text{in}}_{\tilde{G}}(v) \leq 1$ and $\deg^{\text{out}}_{\tilde{G}}(v) \leq 1$ for all $v \in \{s_i, t_i : i = 1, 2, \ldots, k'\}$. Let v be a vertex of $\tilde{V}$. The entering (leaving, respectively) arcs of v are ordered in such a way that if there exists an arc of A'_H in $\delta^{\text{in}}(v)$ ($\delta^{\text{in}}(v)$, respectively), then it is the first (last, respectively) arc in the order. We consider $z_v = r_v(s_v + 1) + s_v(r_v + 1)$ new vertices $v^1, v^2, \ldots, v^{z_v}$. We then replace in $\tilde{G}$ the vertex v by the path $((v^1, v^2), (v^2, v^3), \ldots, (v^{z_v-1}, v^{z_v}))$ in the following way: the i^{th} arc of $\delta^{\text{in}}_{\tilde{G}}(v)$ has $v^{i(s_v+1)-s_v}$ as its head for $i = 1, 2, \ldots, r_v$; the i^{th} arc of $\delta^{\text{out}}_{\tilde{G}}(v)$ has $v^{r_v(s_v+1)+i(r_v+1)}$ as its tail for $i = 1, 2, \ldots, s_v$. For $i = 1, 2, \ldots, r_v$, we denote by I^i_v the sequence of the $(s_v + 1)$ consecutive vertices starting from $v^{i(s_v+1)-s_v}$ (corresponding to the head of the i^{th} arc entering v in $\tilde{G}$). Similarly, we define the set O^i_v of vertices by considering the (r_v+1) consecutive vertices finishing by $v^{r_v(s_v+1)+i(r_v+1)}$ for $i = 1, 2, \ldots, s_v$. Furthermore if $s_v \neq 0$ and $r_v \neq 0$, we also add the arcs $(v^{i(s_v+1)+j+1}, v^{r_v(s_v+1)+j(r_v+1)-r_v+i})$ for $i = 0, 1, \ldots, r_v - 1$ and for $j = 1, 2, \ldots, s_v$. If v corresponded to a vertex s_i (t_i, respectively), then the vertex v^1 (v^{z_v}, respectively) becomes the vertex s_i (t_i, respectively). This gadget we use to replace the vertex v_6 in Figure 3 is illustrated on Figure 4.

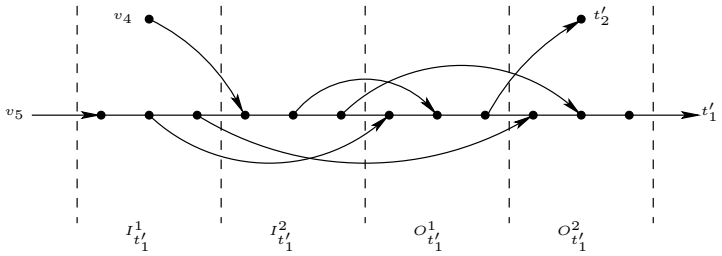

Fig. 4 Replacement gadget of vertex v_6

The digraph we obtain, say $\hat{G}$, contains a Hamiltonian path. We denote by A^*_H the arcs of this path. At this point, $\hat{G}$ may still contain multiple arcs if $\tilde{G}$ contains multiple arcs that are not incident to $\tilde{V}$. However, by definition of $\tilde{W}$, given two vertices u and v of $\hat{G}$, there exist at most two multiple arcs

from u to v and one of them belongs to A^*_H. We then subdivise the arc (u, v) that belongs to A^*_H, *i.e.*, we add a new vertex w and we replace (u, v) in A^*_H by the arcs (u, w) and (w, v). By repeting this operation as long as necessary, we obtain a simple digraph.

The last gadget we use is to replace every arc $a = (u, v)$ of $\tilde{A}$ by a path $((u, s_a), (s_a, t_a), (t_a, v))$, where s_a and t_a are two new vertices. Moreover, the pair (s_a, t_a) is added to the set S. At the end of this replacement procedure, the set S is then composed of $k = k' + |\tilde{A}|$ pairs of vertices. Let G be the resulting digraph. It is clear that all the vertices of G fulfill the degree conditions of a quasi-topological digraph. Therefore, since G is simple, it is quasi-topological. Since no vertex of G belongs to two pairs of S and $\deg^{\text{in}}(s_i) \leq 1$ and $\deg^{\text{out}}(t_i) \leq 1$ for all $i = 1, \ldots, k$, the digraph G and the set S correspond to an instance of the QTDP. As finding a topological order of an acyclic digraph is a polynomial-time solvable problem, the whole construction procedure of G and S from G' and S' can be clearly performed in polynomial time.

We now claim that the ADP has a feasible solution if and only if the QTDP has a feasible one. We first consider a solution to the QTDP, that is, k arc-disjoint paths $L_1, L_2, \ldots, L_k$. Without loss of generality, a path L_i for $i = 1, 2, \ldots, k'$ is associated with a pair of S'. Let $L'_1, L'_2, \ldots, L'_{k'}$ be the restriction of $L_1, L_2, \ldots, L_{k'}$ respectively on the digraph G'. Because of the definition of the pairs (s_a, t_a) with $a \in \tilde{A}$, L'_i corresponds to a path of G' from s'_i to t'_i. As the paths $L_1, L_2, \ldots, L_{k'}$ are arc-disjoint, so are the paths $L'_1, L'_2, \ldots, L'_{k'}$.

To prove the converse, we consider a solution to the ADP given by k' arc-disjoint paths $L'_1, L'_2, \ldots, L'_{k'}$. For $i = 1, 2, \ldots, k'$, every path L'_i can be trivially extended to a path $\tilde{L}_i$ of $\tilde{G}$ by concatenating the arc (s_i, s'_i), the path L'_i and the arc (t'_i, t_i). Clearly, the paths $\tilde{L}_1, \tilde{L}_2, \ldots, \tilde{L}_{k'}$ are arc-disjoint. Moreover, since neither (s_i, s'_i) nor (t'_i, t_i) belongs to $\tilde{A}$, the path $\tilde{L}_i$ does not contain any arc of $\tilde{A}$, for $i = 1, 2, \ldots, k'$. Let v be a vertex of $\tilde{V}$ and q be the number of paths of $\tilde{L}_1, \tilde{L}_2, \ldots, \tilde{L}_{k'}$ traversing v. Without loss of generality, we suppose that these paths are the first q ones. We remark that $1 \leq q \leq \min\{r_v, s_v\}$. Consider any i of $\{1, 2, \ldots, q\}$, and denote by a_i (b_i, respectively) the arc of $\tilde{L}_i$ entering (leaving, respectively) v. As $\tilde{G}$ is acyclic, the arcs a_i and b_i are uniquely defined. Let $I^{i'}_v$ and $O^{i''}_v$ be the subsets of vertices associated with a_i and b_i respectively, as previously defined in the replacement gadget. We remark that in G, the vertex $v^{i'(s_v+1)-s_v}$ is the head of a_i, and the vertex $v^{r_v(s_v+1)+i''(r_v+1)}$ is the tail of b_i. From the definition of the replacement gadget, it is straightforward that there exists a unique path M_i from $v^{i'(s_v+1)-s_v}$ to $v^{r_v(s_v+1)+i''(r_v+1)}$ in the subgraph of G induced by $I^{i'}_v \cup O^{i''}_v$. We then construct the path $\bar{L}_i$ by inserting M_i in between the arcs a_i and b_i in $\tilde{L}_i$. We remark that two distinct paths $\tilde{L}_j$ and $\tilde{L}_{j'}$ with $1 \leq j, j' \leq q$ clearly provide two arc-disjoint paths M_j and $M_{j'}$. Since the paths $\tilde{L}_1, \tilde{L}_2, \ldots, \tilde{L}_q$ are arc-disjoint, so are the paths $\bar{L}_1, \bar{L}_2, \ldots, \bar{L}_q$. By reiterating this local insertion procedure to every vertex of $\tilde{V}$, we obtain k' paths $L_1, L_2, \ldots, L_{k'}$ of G which are obviously arc-disjoint.

To complete the proof, we consider the path composed of the arc (s_a, t_a) for every pair of S associated with an arc a of $\tilde{A}$. In that way, we have obtained $k = k' + |\tilde{A}|$ arc-disjoint paths of G. □

Using a reduction from the QTDP, we now prove that the demand-paths checking problem is NP-complete. Consider an instance of the QTDP specified by a quasi-topological digraph $G = (V_G, A_G)$ and a set $S = \{(s_i, t_i) : i = 1, 2, \ldots, k\}$ of pairs of distinct vertices of G so that every vertex of G belongs to at most one pair, $\deg^{\text{in}}(s_i) \leq 1$ and $\deg^{\text{out}}(t_i) \leq 1$ for $i = 1, 2, \ldots, k$. To obtain the digraph D' of the instance of the DPCP, we first contract every jump arc of G. (Remember that the jump arcs are those which do not belong to the Hamiltonian path $((v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n))$ of G.) We then add a new vertex v_0 together with two new arcs (v_0, v_1) and (v_n, v_0). Since G is quasi-topological and contracting the jump arcs creates closed walks, the digraph D' thus obtained is clearly Eulerian. The Eulerian closed walk C of D' is constructed as follows. We consider the sequence $((v_0, v_1), (v_1, v_2), \ldots, (v_{n-1}, v_n), (v_n, v_0))$ where $((v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n))$ corresponds to the Hamiltonian path of G. For every arc (v_i, v_j) of A_J, we identify in C the vertices v_i and v_j. It is easy to see that C is an Eulerian closed walk of D'. The set P of demands is derived from S by associating a demand p_i with every pair (s_i, t_i) for $i = 1, 2, \ldots, k$, so that o^{p_i} (d^{p_i}, respectively) is the vertex of D' corresponding to s_i (t_i, respectively) and $q^{p_i} = 1$. Moreover, we set the vehicle capacity Q to one. It is obvious that this construction of the instance of the DPCP is polynomial.

We now claim that the QTDP has a feasible solution if and only if the DPCP has a feasible one. Consider a feasible solution to the QTDP defined by k arc-disjoint paths $L_1, L_2, \ldots, L_k$ of G. We construct a solution to the DPCP as follows. For $i = 1, 2, \ldots, k$, the path K_i of D' associated with the demand p_i is obtained from L_i by removing all the jump arcs. Since the paths $L_1, L_2, \ldots, L_k$ are arc-disjoint, no arc of D' is overloaded. Moreover, the arcs of K_i are traversed in the same order as in C, for $i = 1, 2, \ldots, k$, because G is quasi-topological and C is built upon the Hamiltonian path of G. Therefore, $K_1, K_2, \ldots, K_k$ is a feasible solution to the DPCP.

To prove the converse, we start with a feasible solution $K_1, K_2, \ldots, K_k$ of the DPCP. Since $Q = 1$, the paths K_i, $i = 1, 2, \ldots, k$ of D' are arc-disjoint. Let $i \in \{1, 2, \ldots, k\}$. Consider the restriction L'_i of K_i on G. Clearly, all the arcs of the L'_i belong to A_H. To transform L'_i into a $s_i t_i$-path, we add jump arcs of A_J to connect the different subpaths of L'_i. (Jump arcs may be needed to connect s_i to the first vertex of L'_i and to connect the last vertex to t_i.) We remark that any added jump arc corresponds to a reload of p_i with respect to C. Indeed, all the arcs of the Hamiltonian path between the two extremities of any added jump arc corresponds to a closed walk in C. Moreover, the demand p_i is carried before and after this closed walk but not on any arc of this walk, which implies that it is reloaded on the vertex of D' corresponding to the extremities of the jump arc. Since $L'_1, L'_2, \ldots, L'_k$ are arc-disjoint, a jump arc

can be added at most once. The resulting $s_i t_i$-paths for $i = 1, 2, \ldots, k$ then form a feasible solution to the QTDP. □