The Target Visitation Arc Routing Problem

Jessica Rodríguez-Pereira *[1,2] and Gilbert Laporte[3,4]

[1]Universitat Pompeu Fabra, Barcelona, Spain
[2]Barcelona GSE, Spain
[3]HEC Montréal, Montréal, Canada
[4]School of Management, University of Bath, Bath, United Kingdom

June 16, 2021

ABSTRACT

This paper studies the Target Visitation Arc Routing Problem on an undirected graph. This problem combines the well-known undirected rural postman problem and the linear ordering problem. In this problem, there is a set of required edges partitioned into targets, which must be traversed and there are pairwise preferences for the order in which some targets are serviced, which generates a revenue if the preference is satisfied. The aim is to find a tour that traverses all required edges at least once, and offers a compromise between the revenue generated by the order in which targets are serviced, and the routing cost of the tour. A linear integer programming formulation including some families of valid inequalities are proposed. Despite the difficulty of the problem, the model can be used to solve to optimality around 62 % of the test instances.

1 Introduction

The purpose of this paper is to introduce, model and solve the Target Visitation Arc Routing Problem (TVARP) defined as follows. The TVARP is an undirected arc routing problem in which required edges are partitioned into clusters of targets, which define preferences relative to the order in which they are completely serviced. Thus, in addition to the usual routing cost, there is a revenue associated with ordered pairs of targets, which is realized if the required edges of the first target have been serviced before completing the service of the required edges belonging to the second target. The profit of a tour is equal to the revenue associated with the order in which the edges are serviced, minus the routing cost. The TVARP is to determine a tour traversing each required edge at least once and yielding a maximum profit.

*Corresponding author; jessica.rodriguez@upf.edu

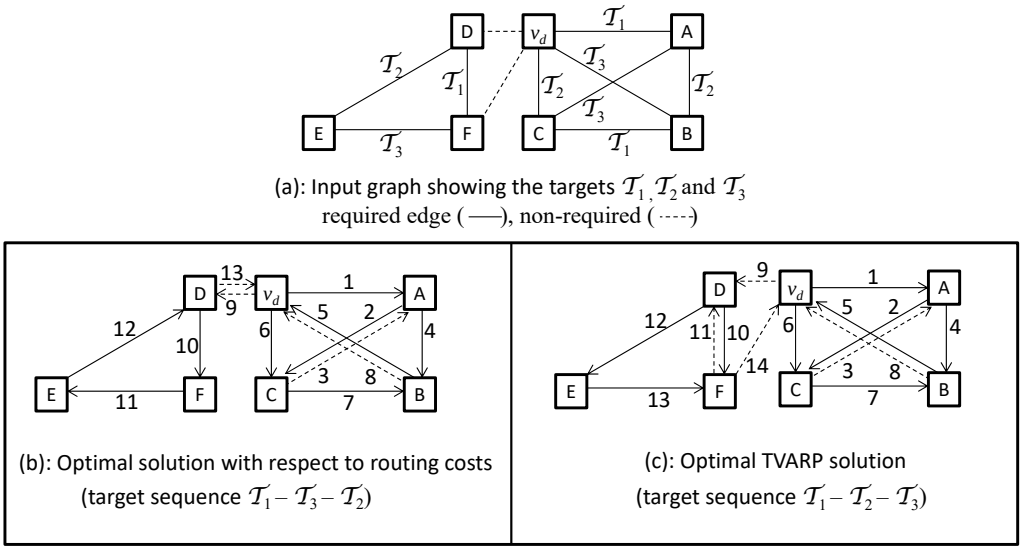

Figure 1: Example of TVARP.

Figure 1 depicts a small instance of the TVARP. Consider the input graph of Figure 1(a) with seven vertices, $V = \{v_d, A, ..., F\}$. The solid lines represent required edges, while the dotted lines represent non-required edges, all of which with unit routing cost. The nine required edges are partitioned into three targets, shown next to the edges. The potential revenue associated with the order in which the service of the targets are finished is 10 if target one is completed before target two, 10 if target one is completed before target three, 5 if target two is completed before target three, and zero for any other ordered pair. Figure 1(b) shows an optimal solution with respect to the routing cost with a routing cost of 13 and a revenue of 20, for a profit of 7. Figure 1(c)depicts an optimal TVARP solution of profit 11, corresponding to a routing cost of 14 and a revenue of 25.

Applications of the TVARP are motivated by civil and military operations. For instance, in patrol control, rescue activities, environmental assessment or town cleaning, there are often priorities for the order in which service is offered. An example arises, for instance, in the context of disaster relief, where there is a collection of damaged roads that must be cleaned, repaired, or reconstructed. The clean-up or reconstruction should be as fast as possible, but some roads or areas may need to be reached earlier than others, depending not only on their own needs, but also on the accessibility they provide to some others. Therefore, proceeding according to the preferences or priorities derived from the relative order in which repairs are carried out can be crucial. In addition, other applications may be inherited from the literature on hierarchical and deadline classes of ARP problems; see, for instance, the case discussed in Li and Eglese (1996) concerning salt spreading.

The TVARP combines two well known and difficult combinatorial optimization problems: the Rural Postman Problem (RPP) (Orloff, 1974; Corberán and Laporte,

2014) and the Linear Ordering Problem (LOP) (Martí and Reinelt, 2011). The (undirected) RRP is an arc routing problem that consists of designing a least cost tour traversing at least once each edge of a set of required edges. The LOP is a node problem that aims to find a ordering of the nodes that induces a spanning acyclic tournament such that the sum of the arcs weights is maximized. As in the RPP, in the TVARP there is a service demand, expressed by a set of required edges of a given input graph, which must be traversed. As in the LOP there are pairwise revenue preferences for the order in which some targets are serviced. The aim of the TVARP is to find a tour that traverses all required edges at least once, which reflects a compromise between the preferences obtained by the order in which targets are serviced and the routing cost of the tour. Similar to the LOP with cluster (Alcaraz et al., 2020) the preferences are associated to clusters of targets. However, in the TVARP the elements of the cluster do not necessarily have to be serviced consecutively. In both the LOP and the TVARP it is necessary to impose constraints in order to ensure that the solution is transitive, i.e. if node i is visited before node j, and node j is visited before k, then node i is visited before node k.

The TVARP is the arc routing counterpart of the Target Visitation Problem (TVP). The TVP is a relatively new problem, introduced in Grundel and Jeffcoat (2004), where service demand is expressed by a set of vertices that must be visited, targets correspond to vertices, and pairwise preferences are given for the order in which the targets are visited. Feasible solutions to the TVP are Hamiltonian tours and the goal is to find a trade-off between the overall preferences for the order in which the targets are visited and the routing cost. The original application that inspired the TVP is the planing of optimal routes for unmanned aerial vehicles in military missions. There are non-military applications as well, like environmental assessment, combat search, or rescue and disaster relief (Hildenbrandt, 2018).

The TVARP and the TVP are related to routing problems with hierarchies on the set of targets, where the order in which the demand must be satisfied is established in advance. One node routing example is the hierarchical traveling salesman problem (Panchamgam et al., 2013). Hierarchies have also been studied in the context of arc routing. The Hierarchical Chinese Postman Problem (HCPP) was introduced by Dror et al. (1987) and studied by other authors, namely Ghiani and Improta (2000); Cabral et al. (2004) and Korteweg and Volgenant (2006). Another related problem is the RPP with deadline classes (RPPDC) (Letchford and Eglese, 1998). The HCPP and the RPPDC consider a single-vehicle arc routing problem in which the required edges are partitioned into a number of classes according to priorities, each class having its own order or deadline. More recent works of these related problems are Colombi et al. (2016) and Colombi et al. (2017). The TVARP and the TVP are more flexible than hierarchical problems since the order in which clusters must be visited is not imposed in advance but is expressed by means of some preference revenue, which affects the objective function value. However, the preference revenue express pairwise priorities and do not necessarily define a total order, which can be very difficult to establish *a priori*. Consequently, it is also necessary to determine the order in which the targets are visited, in order to reach a trade-off between the pairwise

preference revenue and the routing cost, which increases the difficulty of the problem.

To the best of our knowledge, the TVARP is new, and all the relevant literature refers to its node routing counterpart. The TVP was solved in Grundel and Jeffcoat (2004) by means of a greedy randomized adaptive search heuristic solution procedure. Genetic algorithms were proposed in Blázsik et al. (2006) and Arulselvan et al. (2007): a hybrid genetic algorithm in the former, and a random key genetic algorithm in the latter. Exact algorithms for the TVPs, a branch-and-cut algorithm (Hildenbrandt and Reinelt, 2015) and an exact semi-definite optimization algorithm (Hungerländer, 2015).

The scientific contribution of this work is the introduction of a new arc routing problem, the TVARP, for which we study some properties and present a mathematical programming formulation which is computationally tested. We introduce the TVARP assuming that the cost for traversing an edge is not the same when it is serviced and when it is deadheaded. We further assume that the deadheading cost of an edge depends on whether or not it has been previously serviced. Furthermore, we consider three different routing costs associated with each edge: a service cost, as well as pre- and post- service deadheading costs. Addressing this consideration leads to new modeling challenges, since knowing which edges are traversed is no longer sufficient, as their costs depend on *when* they are traversed. Unlike hierarchical problems this order is not known in advance, so the actual costs of edge traversals cannot be precomputed. Additional constraints are needed for associating edge traversals with their correct cost.

The remainder of this paper is organized as follows. Section 2 contains a formal definition of the problem, some illustrative examples as well as some properties. The mathematical programming formulation for the TVARP is presented in Section 3. Computational results are provided and analyzed in Section 4. The paper closes with some conclusions in Section 5.

2 Formal definition of the TVARP

The TVARP is defined on a undirected graph $G = (V, E)$ where V is the vertex set with $|V| = n$, with a distinguished vertex $v_d \in V$ representing the depot, and E is the edge set with $|E| = m$. We denote by $R \subset E$ the set of required edges. The connected components induced by the required edges are indexed in set K and denoted by $C_k = (V_k, E_k)$, $k \in K$. Feasible TVARP solutions are tours starting and ending at the depot, which traverse each required edge at least once. Note that even if G is an undirected graph, a solution will visit the edges in a certain direction, and the tour can therefore be viewed as a sequence of arcs. As usual, we distinguish between edge traversals that do not provide service (*deadheadings*) and those that provide service. Moreover, like in some ARPs (see, for instance, Cabral et al., 2004), we assume that the deadheading cost of an edge may depend on whether or not it has been previously serviced. For each edge $e \in R$ we denote by c_e^+ the cost for dead-

heading e if it has not yet been serviced and by c_e^- if it has already been serviced. The cost of servicing edge $e \in R$ is denoted by $c_e = c_e^+ + \Delta_e$, with $\Delta_e \geq 0$. We asume that $c_e^- \leq c_e^+ \leq c_e$. The non-required edges $e \in E \setminus R$ are always deadheaded. Their deadheading costs are denoted by c_e^- and do not depend on whether or not they have been previously deadheaded. Let $\mathcal{T}_r$, with $\mathcal{T}_r \subset R$, $r \in L$, denote a pairwise disjoint collection of *targets*, each of which consisting of a subset of required edges not necessarily connected. We say that target $r \in L$ *precedes* target $s \in L$ in a given tour if the required edges of $\mathcal{T}_r$ have been serviced before servicing all of the required edges of $\mathcal{T}_s$, i.e., the last required edge serviced in target r is visited before the last required edge serviced in cluster s. We denote by V_r the set of vertices incidents to the edges in $\mathcal{T}_r$. Associated with each pair of targets $r, s \in L$ there is a non-negative real value a_{rs} representing the revenue obtained when $\mathcal{T}_r$ precedes $\mathcal{T}_s$. The *profit* of a tour is the difference between the total revenue collected and the routing cost.

We work in a general setting under the following assumptions:

- The depot is incident to some edge in $\mathcal{T}_1$. This implies that v_d is incident to some required edge. If needed we define a copy of the depot $v_{d'}$, and a required edged $(v_d, v_{d'})$, with $c^-_{v_d v_{d'}} = c^+_{v_d v_{d'}} = c_{v_d v_{d'}} = 0$, which defines $\mathcal{T}_1$.
- The targets induce a partition of the set of required edges, i.e., $\bigcup_{j \in K} \mathcal{T}_r = R$. If needed we define an additional *dummy target*, $\mathcal{T}_{|L|+1} = R \setminus \bigcup_{j \in K} \mathcal{T}_r$, with revenue $a_{r,|L|+1} = a_{|L|+1,r} = 0$ for all $r \in L$.
- The targets $\mathcal{T}_r$ are not necessarily connected, so a target may appear in several connected components.
- The targets are not necessarily vertex-disjoint.

Definition 2.1. *The TVARP is to find a feasible directed tour, serving all required edges, of maximum profit.*

Usually in ARPs, the graph G is preprocessed and simplified following the procedure described in Christofides et al. (1981). This procedure reduces the set V to the vertices incident to edges of R, and modifies the set E to contain R plus additional unrequired edges representing shortest paths in the original graph, connecting every pair of vertices not connected with an edge of R. Note that this procedure cannot be applied to the TVARP. Given that the TVARP distinguishes between pre-, post- and service costs, precomputing the cost of shortest paths that use some required edge is not possible.

We next present a small example to illustrate the difficulty of the TVARP by highlighting its combinatorial nature. Consider again the same input graph as in the previous example, depicted now in Figure 2(a). As the example shows, it is possible to build multiple TVARP solutions, which are directed, all of which use exactly the same arcs. Figure 2(b) depicts the used arcs for a feasible route. Knowing the arcs of

a tour is not enough to determine a TVARP solution. Figure 2(c) shows four different solutions using the same route. The difference between them lies in the labels of the arcs that define the order in which they are traversed, which at the same time determine the sequence of the target visits. The order labels are shown next to the arcs in each solution and the resulting target service sequence below each of the solutions.

Figure 2 also illustrates that for a given set of arcs, there can be multiple sets of labels leading to the same target sequence, i.e., they produce exactly the same TVARP solutions. For instance, solutions S2 and S3, use the same arcs in a different order, (v_d, D, E, F, D, v_d, A, B, v_d, C, A, C, B, v_d) and (v_d, D, E, F, D, v_d, C, B, v_d, A, C, A, B, v_d), respectively, producing the same target sequence ($\mathcal{T}_2$, $\mathcal{T}_3$, $\mathcal{T}_1$).

Finally, it can be observed that since we assume that pre- and post-service routing costs are not necessarily the same, a given route plus a given set of arc-order labels may also not be sufficient to determine a solution and its value. For example, in solution S1 of Figure 2 (c), arc (B, v_d) is serviced in position 10 of the route and deadheaded in position 13 of the route. If we keep exactly the same arc-order labels but "name" as deadheaded the arc in position 10 and as serviced the arc in position 13, then not only the targets sequence changes, as it becomes $(\mathcal{T}_2, \mathcal{T}_1, \mathcal{T}_3)$, but also the routing cost of the solution changes because deadheading arc (B, v_d) in position 13 incurs a routing cost of c^-_{B,v_d}, whereas the routing cost for deadheading it in position 10 is c^+_{B,v_d}.

The TVARP is NP-hard, since it has as particular cases well-known NP-hard problems. For example, it is easy to see that the RPP is a particular case of the TVARP, when all precedence revenues are zero, $a_{rs} = 0\ \forall r, s \in L$. Likewise, it is easy to see that the LOP is also a particular case of the TVARP when all edge costs are zero, $c_e = 0\ \forall e \in E$.

3 Mathematical model and valid inequalities for the TVARP

In this section we present a formulation for the TVARP as an integer linear program (ILP) and we introduce some valid inequalities.

3.1 Mathematical model

We first transform the undirected graph $G = (V, E)$ into a directed graph $N = (V, A)$, where each edge $e = \{u, v\} \in E$ is replaced with two arcs $(u, v), (v, u) \in A$. For each subset $H \subseteq E$, let $A(H) = \{(u, v),\ (v, u) \in A | \{u, v\} \in H\}$. We also use the following usual notation. For any non-empty vertex subset $S \subset V$, $\delta^+(S) = \{(u, v) \in A | u \in S, v \in V \setminus S\}$ denotes the set of arcs from S to $V \setminus S$ and $\delta^-(S) = \{(u, v) \in A | u \in$

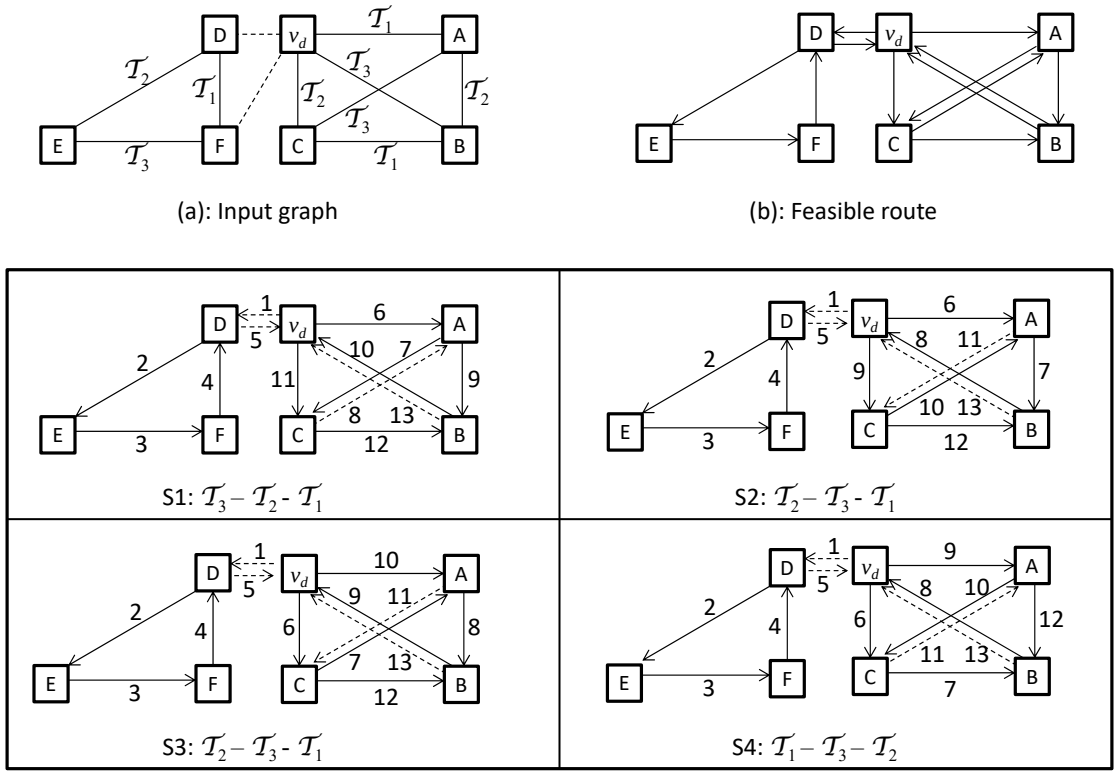

Figure 2: Input graph, route and solutions

$V \setminus S, v \in S\}$ denotes the set of arcs from $V \setminus S$ to S. For a singleton $S = \{v\}$, with $v \in V$, we simply write $\delta^+(v)$ and $\delta^-(v)$ instead of $\delta^+(\{v\})$ and $\delta^-(\{v\})$, respectively. Finally, given an edge set H we define $\delta_H^+(v)$ as $A(H) \cap \delta^+(v)$.

As motivated by the example depicted in Figure 2, in order to establish the preference revenue that will be collected, we associate a *time* component to the arc traversals. This will allow us to identify the time period during which the service of each target is completed. This time period will be referred to as the *completion period* of the target. Therefore, we identify tours with arc sequences in which each arc traversal is associated with one time period and consecutive arcs are traversed in consecutive time periods. We denote by T the index set of the time periods. Since we do not know the exact number of arcs a feasible tour will use (not all feasible tours have the same length), we use an upper bound for the cardinality of $T = \{1, \ldots, \bar{t}\}$, with $\bar{t} = 2|R|D$, where D denotes the maximum number of arcs in a path of N with origin at the depot. This is a crude upper bound on the maximum number of edges in a solution, which would correspond to servicing each required edge using a tour starting and ending at the depot, including that edge, and using the maximum possible number of arcs.

The formulation uses the following sets of binary variables:

- Routing variables:
  - x_{ij}^t and x_{ji}^t: binary variables equal to one if and only if required edge $e = \{i, j\} \in R$, is serviced in period $t \in T$, in the direction from i to j or in the direction from j to i, respectively.
  - y_{ij}^t and y_{ji}^t: binary variables equal to one if and only if edge $e = \{i, j\} \in E$ is deadheaded in period $t \in T$ from i to j or from j to i, respectively.
- Completion variables:
  - o_r^t: binary variable equal to one if and only if t is the completion period of target $\mathcal{T}_r$, $r \in L$.
- Ordering variables:
  - p_{rs}: binary variable equal to one if and only if target $\mathcal{T}_r$ precedes target $\mathcal{T}_s$, $r, s \in L$.
- Pre-service deadheading variables:
  - h_e^t: binary variable equal to one if and only if edge $e \in R$ is deadheaded (in one of its associated directions) in period $t \in T$, but has not yet been serviced.

The formulation is the following:

$$\text{Maximize} \sum_{\substack{r,s \in L \\ r \neq s}} a_{rs} p_{rs} - \sum_{t \in T} \sum_{e=\{i,j\} \in R} c_e (x_{ij}^t + x_{ji}^t) - \sum_{t \in T} \sum_{e=\{i,j\} \in E} c_e^- (y_{ij}^t + y_{ji}^t) - \sum_{t \in T} \sum_{e \in R} (c_e^+ - c_e^-) h_e^t \tag{1}$$

subject to

$$\sum_{t\in T}(x_{ij}^t + x_{ji}^t) = 1 \qquad \{i,j\} \in R \tag{2}$$

$$\sum_{j\in V} x_{vdj}^1 = 1 \tag{3}$$

$$\sum_{\{i,j\}\in R}(x_{ij}^t + x_{ji}^t) + \sum_{\{i,j\}\in E}(y_{ij}^t + y_{ji}^t) \le 1 \qquad t \in T \tag{4}$$

$$\sum_{j|\{i,j\}\in\delta_R^-(i)} x_{ji}^t + \sum_{j|\{i,j\}\in\delta_E^-(i)} y_{ji}^t = \sum_{j|\{i,j\}\in\delta_R^+(i)} x_{ij}^{t+1} + \sum_{j|\{i,j\}\in\delta_E^+(i)} y_{ij}^{t+1} \qquad i \in V\setminus\{d\},\ t = 1,\dots,|T|-1 \tag{5}$$

$$\sum_{t\in T} o_r^t = 1 \qquad r \in L \tag{6}$$

$$\sum_{r\in L} o_r^t \le 1 \qquad t \in T \tag{7}$$

$$\sum_{\{i,j\}\in\mathcal{T}_r}(x_{ij}^t + x_{ji}^t) \le \sum_{t'\ge t} o_r^{t'} \qquad t \in T,\ r \in L \tag{8}$$

$$o_r^t \le \sum_{\{i,j\}\in\mathcal{T}_r}(x_{ij}^t + x_{ji}^t) \qquad t \in T,\ r \in L \tag{9}$$

$$p_{rs} + p_{sr} = 1 \qquad r,s \in L : r \ne s \tag{10}$$

$$p_{rs} + p_{s\ell} + p_{\ell r} \le 2 \qquad r,s,\ell, \in L : r < s,\ s < \ell \tag{11}$$

$$p_{rs} + o_r^t \le 1 + \sum_{p=t+1}^{|T|} o_s^p \qquad r \ne s \in L,\ t = 1,\dots,|T|-1 \tag{12}$$

$$(y_{ij}^t + y_{ji}^t) + \sum_{t'>t}(x_{ij}^{t'} + x_{ji}^{t'}) \le 1 + h_e^t \qquad e = \{i,j\} \in R,\ t \in T \tag{13}$$

$$x_{ij}^t, x_{ji}^t \in \{0,1\} \qquad \{i,j\} \in R,\ t \in T \tag{14}$$

$$y_{ij}^t, y_{ji}^t \in \{0,1\} \qquad \{i,j\} \in E,\ t \in T \tag{15}$$

$$o_r^t \in \{0,1\} \qquad r \in L,\ t \in T \tag{16}$$

$$p_{rs} \in \{0,1\} \qquad r \ne s \in L \tag{17}$$

$$h_e^t \in \{0,1\} \qquad e \in R,\ t \in T. \tag{18}$$

The objective function computes the difference between the revenue generated by satisfying target preferences and the routing cost. The routing cost is computed as the cost of servicing the required edges, plus the deadheading cost. Deadheading variables y_e^t are assigned an objective cost c_e^-, assuming that, in the case of required edges, they have already been serviced. Therefore the auxiliary variables h_e^t are necessary in order to compute the number of times that an edge is deadheaded prior to being serviced. The objective function value can then be computed correctly by assigning to h_e^t an objective coefficient of $c_e^+ - c_e^-$.

Constraints (2)–(5) deal with the feasibility of the routing part. Constraints (2) guarantee that each required edge is serviced exactly once. By (3) the tour starts at the depot and by (4) at most one edge is traversed (serviced or deadheaded) at each time period. The balance constraints (5) guarantee the parity of the vertices and also ensure that consecutive arcs of the tour are assigned to consecutive time periods. The time index in the routing variables prevents the formation of subtours and therefore enforce the connectivity of the routes with the depot. Constraints (6)

assign a completion period to each cluster, and by (7) each period can be the completion time of at most one target. The linkage between the routing variables x and the completion variables o is established by constraints (8) and (9). In particular, (8) ensure that an edge from a target is not serviced in a period unless the completion period of that target occurs in a later period. Constraints (9) impose that the completion period of a target corresponds to a period in which some required edge of that target is serviced. Constraints (10)–(11) are well-known LOP constraints which ensure transitivity. Constraints (10) state that for any pair of targets one must precede the other, whereas the dicycle constraints (11) prevent cycles in target precedences. Constraints (12) relate the variables p and o by imposing that if target r precedes target s, then the completion time of target s has to be after the completion time of target r. Variables h_e^t are activated through constraints (13), if at time period t edge e is deadheaded and it has not yet been serviced. Finally, the binary conditions on the decision variables are imposed through constraints (15)–(17).

3.2 Valid Inequalities

Below we introduce some families of simple valid inequalities that can be used to reinforce the LP relaxation of formulation (1)–(18):

a) Connectivity inequalities for targets and connected components. Even if the connectivity of the solution is already enforced by the time index, we can reinforce the formulation by adding the classical connectivity constraints to the targets and to the connected components induced by the required edges, which ensure that each set has one incoming arc and one outgoing arc:

$$\sum_{t\in T}\left(\sum_{j|\{i,j\}\in\delta_R^+(V_r)} x_{ij}^t + \sum_{j|\{i,j\}\in\delta_E^+(V_r)} y_{ij}^t\right) \geq 1 \quad r \in L \tag{19}$$

$$\sum_{t\in T}\left(\sum_{j|\{i,j\}\in\delta_R^+(V_k)} x_{ij}^t + \sum_{j|\{i,j\}\in\delta_E^+(V_k)} y_{ij}^t\right) \geq 1 \quad k \in K. \tag{20}$$

b) Depot aggregated balance constraints. Balance inequalities (5) are defined in $V \setminus \{d\}$ to ensure the parity of the vertices and that consecutive arcs of the tours are assigned to consecutive time periods. At the depot parity can be reinforced by summing over all periods, we obtain:

$$\sum_{t\in T}\left(\sum_{\{d,i\}\in R} x_{di}^t + \sum_{\{d,i\}\in E} y_{di}^t\right) = \sum_{t\in T}\left(\sum_{\{i,d\}\in R} x_{id}^t + \sum_{\{i,d\}\in E} y_{id}^t\right). \tag{21}$$

c) Consecutive depot time. Constraints (5) guarantee that consecutive arcs of the tour are assigned to consecutive time periods. However, note that this family

of constraints does not apply to the depot, where in general we will have the first and the last arcs. However, in a feasible solution, a tour can visit the depot more than once, in which case it is helpful to ensure no periods jumps at the depot through the following constraints:

$$\sum_{\{d,i\}\in R} x_{di}^t + \sum_{\{d,i\}\in E} y_{di}^t \leq \sum_{\{i,d\}\in R} x_{id}^{t-1} + \sum_{\{i,d\}\in E} y_{id}^{t-1} \quad t = 2,\ldots,|T|. \quad (22)$$

d) Relation H–Y variables. By definition, variable h_e^t can only take value one if the edge $e \in R$ is deadheaded in period $t \in T$, but has not yet been serviced. Variables y_{ij}^t and y_{ji}^t identify whether edge $e = \{i,j\} \in E$ is deadheaded in period $t \in T$, from i to j or from j to i. Therefore, in any feasible solution, variable h_e^t will be equal to one only if there is a variable y_{ij}^t or y_{ji}^t equal to one:

$$h_e^t \leq y_{ij}^t + y_{ji}^t \quad e = \{i,j\} \in R,\, t \in T. \quad (23)$$

e) Reinforcement relation P–O variables. Constraints (12) can be reinforced by considering all periods before t. Given two targets $r, s \in L$ and a period $t \in T$, if target r precedes target s, then the completion time of target s cannot be before the completion time of target r:

$$p_{rs} + \sum_{p\leq t} o_s^p + \sum_{q>t} o_r^q \leq 2 \quad r \neq s \in L,\, t \in T. \quad (24)$$

3.3 Particular case: targets with cardinality of one

In the particular case where all targets $\Im_r$ $r \in L$ have cardinality of one, the model can be written without using the completion variables. The completion period of a target can be directly known through the service period of the unique edge of the target. Hence, the families of constraints (6)–(9), and (16) are not longer needed, and the relation between the routing variables and the precedences variables is given by constraints (32), which follow the same idea as constraints (12), imposing that if target r precedes target s, then target s must be served after target r.

The formulation is the following:

$$\text{Maximize} \sum_{\substack{r,s\in L \\ r\neq s}} a_{rs}p_{rs} - \sum_{t\in T}\sum_{e=\{i,j\}\in R} c_e(x_{ij}^t + x_{ji}^t) - \sum_{t\in T}\sum_{e=\{i,j\}\in E} c_e^-(y_{ij}^t + y_{ji}^t) - \sum_{t\in T}\sum_{e\in R}(c_e^+ - c_e^-)h_e^t \quad (25)$$

subject to

$$\sum_{t\in T}(x_{ij}^t + x_{ji}^t) = 1 \qquad \{i,j\} \in R \tag{26}$$

$$\sum_{j\in V} x_{v_d j}^1 = 1 \tag{27}$$

$$\sum_{\{i,j\}\in R}(x_{ij}^t + x_{ji}^t) + \sum_{\{i,j\}\in E}(y_{ij}^t + y_{ji}^t) \le 1 \qquad t \in T \tag{28}$$

$$\sum_{j|\{i,j\}\in\delta_R^-(i)} x_{ji}^t + \sum_{j|\{i,j\}\in\delta_E^-(i)} y_{ji}^t = \sum_{j|\{i,j\}\in\delta_R^+(i)} x_{ij}^{t+1} + \sum_{j|\{i,j\}\in\delta_E^+(i)} y_{ij}^{t+1} \qquad i \in V\setminus\{d\},\ t = 1,\dots,|T|-1 \tag{29}$$

$$p_{rs} + p_{sr} = 1 \qquad r,s \in L : r \ne s \tag{30}$$

$$p_{rs} + p_{s\ell} + p_{\ell r} \le 2 \qquad r,s,\ell, \in L : r < s,\ s < \ell \tag{31}$$

$$p_{rs} + (x_{ij}^t + x_{ji}^t) \le 1 + \sum_{p=t+1}^{|T|}(x_{kl}^p + x_{lk}^p) \qquad \{i,j\} \in \mathfrak{T}_r,\ \{k,l\} \in \mathfrak{T}_s,\ r \ne s \in L,\ t = 1,\dots,|T|-1 \tag{32}$$

$$(y_{ij}^t + y_{ji}^t) + \sum_{t'>t}(x_{ij}^{t'} + x_{ji}^{t'}) \le 1 + h_e^t \qquad e = \{i,j\} \in R,\ t \in T \tag{33}$$

$$x_{ij}^t, x_{ji}^t \in \{0,1\} \qquad \{i,j\} \in R,\ t \in T \tag{34}$$

$$y_{ij}^t, y_{ji}^t \in \{0,1\} \qquad \{i,j\} \in E,\ t \in T \tag{35}$$

$$p_{rs} \in \{0,1\} \qquad r \ne s \in L \tag{36}$$

$$h_e^t \in \{0,1\} \qquad e \in R,\ t \in T. \tag{37}$$

4 Computational study

We now present the results obtained in the computational experiments we have conducted in order to assess the behavior of our exact algorithms for the TVARP. The algorithm was implemented in C++ and all experiments were run on a 2.80 Giga-Hertz Intel Core i7 machine with 16 Gigabytes of memory. We used the IBM CPLEX 12.7 Concert Technology with default parameters and a maximum computing time of two hours.

4.1 Description of the instances

The set of instances used in the computational experiments are well-known benchmark instances for the RPP adapted to the TVARP. We have worked with 10 instances of each of the following groups: random instances (labeled "P") (Christofides et al., 1981); instances with vertices of degree four (labeled "D"), and grid instances (labeled "G") (Hertz et al., 1999). We have generated two instances from each of the original instances by introducing three or four targets. Furthermore, for each number of targets, we have created two types of instances which differ from each other in the way the targets are defined. In the first type, we randomly divide the required edges among the targets, while in the second type, the targets are formed by one or several connected components C_k, $k \in K$. To define the targets from the second type, we build the minimum spanning tree (MST) associated with the required connected components (each component is a node for building the MST), and we delete

successively the longest edge form the MST until this is divided into the required number of targets. In this way, the targets can be interpreted as common areas or neighborhood. The structure of cost is generated following the procedure applied in Colombi et al. (2016). For the service cost, c_e, the original cost is used. The cost for deadheading an edge is equal to the service cost reduced or increased randomly by 10% to 20% if the edge was already serviced or not yet serviced, respectively. To define the revenue function a_{rs} associated with every pair of clusters $r, s \in L$ we have considered the prize-collecting instance associated with each original RPP instance (Aráoz et al., 2009) and computed for each cluster $r \in L$ the revenue R_r of its required edges. Then, the revenue a_{rs} associated with the pair of clusters $r, s \in L$ takes, with a probability of 0.8, value one if $R_r \geq R_s$, and zero otherwise.

4.2 Results for the TVARP

In order to prioritize revenue and cost, we have multiplied the first term of the objective function by a weight M. In particular, we have used three different values of M: i) unity, $M = 1$; ii) maximum revenue, $M = \max_{r \in L}\{R_r\}$; and iii) total revenue, $M = \sum_{r \in L} R_r$. Given these weights, we move from prioritizing the routing cost (unity) to prioritizing the revenue (total revenue), going by a balanced prioritization (maximum revenue).

Table 1 summarizes the results for the different weight values M. For each type of cluster (randomly or MST generated) and each number of targets (three or four) each group of instances (P, D, and G), columns 3–4, 5–8 and 9–12 provide results for $M = 1$, $M =$ MaxRevenue and $M =$ TotalRevenue, respectively. The last row of the table summarizes the results over all instances. The columns under ♯ Opt show the number of instances in the group that have been optimally solved. The columns Time give the average computing time in seconds over all instances. The columns Gap_{LPR} give the average percentage gap at the root node, showing how good is the upper bound. Hence, given the upper bound provided by the LP relaxation UB_0 and the optimal solution value z^*, then $Gap_{LPR} = 100(UB_0 - z^*)/z^*$. If the optimal solution is unknown, then the computation is performed with respect to the lower bound LB provided by the best know solution value at termination, which overestimates Gap_{LPR}. The columns Gap_{Sol} give the average percentage gap at termination with respect to the best upper bound UB: $Gap_{Sol} = 100(UB - z^*)/UB$. Analogous, if the optimal solution value is unknown, then the best known solution value LB at termination is used. Note that the gaps can be computed only if the denominator is positive, which never happened for $M = 1$.

As can be seen, the optimality of the current solution was proven for 59 to 63 % of the instances depending on the value of M. The results of the computational experiments show that while instances G have high gaps at the root node, showing a

Table 1: Computational Results Summary

		$M = 1$		$M =$ **MaxRevenue**				$M =$ **TotalRevenue**			
		♯ **Opt**	**Time**	♯ **Opt**	Gap_{LPR}	Gap_{Sol}	**Time**	♯ **Opt**	Gap_{LPR}	Gap_{Sol}	**Time**
MST-3	**P**	3/10	5474.54	5/10	15.60	6.91	5232.80	5/10	5.42	3.41	5388.99
	D	5/10	3990.98	6/10	36.29	15.35	3367.81	6/10	16.20	8.38	3679.28
	G	10/10	581.59	10/10	31.33[1]	0.00	386.34	10/10	28.21	0.00	405.16
MST-4	**P**	3/10	5723.86	2/10	15.45	8.55	5845.49	2/10	4.18	3.18	5847.74
	D	5/10	4304.99	5/10	26.01	13.60	4376.03	5/10	9.28	6.26	4658.91
	G	10/10	737.73	10/10	30.44[2]	0.00	1723.27	10/10	11.42	0.00	544.07
RDM-3	**P**	3/10	5594.96	4/10	64.88[3]	18.70	5172.03	4/10	9.96	4.15	5465.40
	D	6/10	4738.20	6/10	134.45	24.71	4579.97	6/10	20.39	9.24	4989.01
	G	10/10	1595.32	10/10	92.92[4]	0.00	585.35	10/10	9.14	0.00	305.92
RDM-4	**P**	2//10	5875.17	3/10	10.86	7.56	5821.06	2/10	3.59	2.93	5806.29
	D	6/10	4315.13	5/10	16.05	4.75	4611.16	5/10	9.06	4.75	4539.06
	G	8/10	1620.16	10/10	36.79	0.00	1210.43	10/10	11.75	0.00	2336.63
ALL		71/120	3712.72	76/120	42.59[5]	8.35	3575.98	75/120	11.56	3.53	3663.87

[1] Computed with 7 out of 10 instances due to negative values of the feasible solutions.
[2] Computed with 8 out of 10 instances due to negative values of the feasible solutions.
[3] Computed with 8 out of 10 instances due to negative values of the feasible solutions.
[4] Computed with 8 out of 10 instances due to negative values of the feasible solutions.
[5] Computed without the previous not used instances.

poor quality of the upper bound, they are also the easiest to resolve in terms of the number of instances in the group optimally solved, the gap at the termination, and the computing time. In contrast, instances P, with smaller gaps at root node, are the most difficult to solve in terms of number of instances solved, and the computing time.

With respect to how the cluster were built, we can observe that, in general, the instances "MST", where the targets are subsets of connected required components, are slightly easier to solve than the others. They yield a smaller gap and a smaller computing time. Regarding the different values of the weight M, as could be expected, the instances are more difficult to solve for the maximum revenue weight when both terms are in the same range. This is reflected by the percentage optimality gaps which are larger than the gaps for total revenue weight.

Table 2: Average number of precedences

	$M = 1$	$M =$ MaxRevenue	$M =$ TotalRevenue
MST-3	1.7	2.9	2.9
MST-4	4.0	5.7	5.8
RDM-3	2.3	2.8	2.8
RDM-4	4.2	5.5	5.6

Table 2 shows for each cluster type and size, the average number of precedences in the optimal or best know solution. The number of possible precedences in an instance is given by $(|L|(|L|-1)/2)$. Since each precedence has a probability of one half, either there is or there is no precedence, the expected value of satisfied precedences is half the number of possible precedences, i.e. $(|L|(|L|-1)/4)$. Hence, the expected values are 1.5 and 3 for three and four targets, respectively. The results clearly illustrate that when routing cost is prioritized ($M = 1$), the number of sat-

isfied precedences is smaller. On the other side, when the weight is prioritizing the revenue (M = TotalRevenue) is when the number of precedences is higher. However, there is no real significant difference in this respect when both terms are in equilibrium (M = MaxRevenue). In fact, on average, we can only see an increase on the instances of size 4.

One could expect the routing cost to increase with M, since as more weight is given to the revenue, it may be interesting to have longer routes in order to satisfy more preferences. To analyze the variation of the routing cost and for each pair of M values, we compute the percentage increase of the route cost. In particular, let $\bar{v}^i$ denote the routing cost for the optimal solution value or the best known solution at termination associated to $i \in M$ for a given instance. Table 3 gives, for each pair of M values $i, j \in M$ $i \neq j$, the averages of the percentage increase of the routing cost $100(\bar{v}^i - \bar{v}^j)/\bar{v}^i$ over all the instances of each set of benchmark instances. As expected, the lowest routing cost is obtained for $M = 1$, where the routing cost is prioritized over the revenue. Furthermore, looking at the comparison "MaxRevenue vs TotalRevenue", it can be observed that the longest routes are obtained for M = TotalRevenue, when the weight on the revenue is the highest. However, for the set of instances MST-3 the average percentage of routing cost increase is slighly higher for the "1 vs MaxRevenue" than for the "1 vs TotalRevenue", which may seem surprising. A further analysis, instance by instance, shows that according to expectations, the routing cost increases with M. However, since for some of the instances the comparison is made between the best known solutions, in some cases the TotalRevenue best solution at termination is better than the MaxRevenue best solution at termination, which explains why "1 vs MaxRevenue" has a higher increase than "1 vs TotalRevenue". In particular, this happens in three instances, where on average the routing cost associated to the best known solutions of the TotalRevenue are 4% smaller than those of the best known solutions of the MaxRevenue. Furthermore, we observe that the instances with more targets are more likely to yield a higher routing cost.

Table 3: Average percentage increase of the routing cost

	1 vs MaxRevenue	1 vs TotalRevenue	MaxRevenue vs TotalRevenue
MST-3	10.4	10.3	0.5
MST-4	16.1	16.2	2.8
RDM-3	1.1	1.7	0.7
RDM-4	9.8	32.8	22.5

5 Conclusions

We have introduced a new arc routing problem, the TVARP, which combines two optimization problems: the RPP and the LOP. We have presented an integer linear formulation for the TVARP, and some valid inequalities to reinforce the LP relaxation. Despite the difficulty of the problem, the formulation is capable of solving to

optimality 59 to 63 % of the instances. Our numerical results illustrate the impact of the cluster pattern and the priorization between routing cost and revenue. Instances with random targets and equal weight for cost and revenue are the most difficult to solve.

Acknowledgements

Thanks are due to the referees for their valuable comments. This research was partially supported by the Spanish Ministry of Economy and Competitiveness through grant MTM2015-63779-R (MINECO/FEDER), and by the Canadian Natural Sciences and Engineering Research Council under the grant 2015-06189. This support is gratefully acknowledged.

References

Alcaraz, J., E. M. García-Nové, M. Landete, and J. F. Monge (2020). The linear ordering problem with clusters: a new partial ranking. *TOP*, 1–26.

Aráoz, J., E. Fernández, and O. Meza (2009). Solving the prize-collecting rural postman problem. *European Journal of Operational Research 196*(3), 886–896.

Arulselvan, A., C. Commander, and P. Pardalos (2007). A hybrid genetic algorithm for the target visitation problem. *Naval Research Logistics*, 1–20.

Blázsik, Z., T. Bartók, B. Imreh, C. Imreh, and Z. Kovacs (2006). Heuristics on a common generalization of TSP and LOP. *Pure Mathematics and Applications 17*(3–4), 229–239.

Cabral, E., M. Gendreau, G. Ghiani, and G. Laporte (2004). Solving the hierarchical Chinese postman problem as a rural postman problem. *European Journal of Operational Research 155*(1), 44–50.

Christofides, N., V. Campos, A. Corberán, and E. Mota (1981). An algorithm for the rural postman problem. Technical report, Imperial College Report IC.O.R.81.5.

Colombi, M., Á. Corberán, R. Mansini, I. Plana, and J. Sanchis (2016). The hierarchical mixed rural postman problem. *Transportation Science 51*(2), 755–770.

Colombi, M., Á. Corberán, R. Mansini, I. Plana, and J. Sanchis (2017). The hierarchical mixed rural postman problem: Polyhedral analysis and a branch-and-cut algorithm. *European Journal of Operational Research 257*(1), 1–12.

Corberán, Á. and G. Laporte (2014). Arc Routing: Problems, Methods, and Applications. *MOS-SIAM Series on Optimization Philadelphia, 20*.

Dror, M., H. Stern, and P. Trudeau (1987). Postman tour on a graph with precedence relation on arcs. *Networks 17*(3), 283–294.

Ghiani, G. and G. Improta (2000). An algorithm for the hierarchical Chinese postman problem. *Operations Research Letters 26*(1), 27–32.

Grundel, D. and D. Jeffcoat (2004). Formulation and solution of the target visitation problem. Technical report, Proceedings of the AIAA 1st Intelligent Systems Technical Conference, Chicago.

Hertz, A., G. Laporte, and P. Nanchen Hugo (1999). Improvement procedures for the undirected rural postman problem. *INFORMS Journal on Computing 11*, 53–62.

Hildenbrandt, A. (2018). Traveling salesman problems with additional ordering constraints. In N. Kliewer, J. F. Ehmke, and R. Borndörfer (Eds.), *Operations Research Proceedings 2017*, Cham, pp. 221–227. Springer International Publishing.

Hildenbrandt, A. and G. Reinelt (2015). Inter programming models for the target visitation problem. *Informatica 39*, 257–260.

Hungerländer, P. (2015). A semidefinite optimization approach to the target visitation problem. *Optimization Letters 9*(8), 1703–1727.

Korteweg, P. and T. Volgenant (2006). On the hierarchical Chinese postman problem with linear ordered classes. *European Journal of Operational Research 169*(1), 41–52.

Letchford, A. N. and R. W. Eglese (1998). The rural postman problem with deadline classes. *European Journal of Operational Research 105*(3), 390–400.

Li, L. Y. O. and R. W. Eglese (1996). An interactive algorithm for vehicle routeing for winter-gritting. *Journal of the Operational Research Society 47*(2), 217–228.

Martí, R. and G. Reinelt (2011). In S. S. Antman, J. E. Marsden, and L. Sirovich (Eds.), *The Linear Ordering Problem: Exact and Heuristic Methods in Combinatorial Optimization*, Volume 175. Springer Science & Business Media.

Orloff, C. S. (1974). A fundamental problem in vehicle routing. *Networks 4*(1), 35–64.

Panchamgam, K., Y. Xiong, B. L. Golden, B. Dussault, and E. A. Wasil (2013). The hierarchical traveling salesman problem. *Optimization Letters 7*, 1517–1524.