The Directed Profitable Rural Postman Problem

Marco Colombi, Renata Mansini

Department of Information Engineering, University of Brescia, Italy

Operations Research Group

Technical Report No. 01, April 2013

Abstract

In this paper we study a generalization of the Directed Rural Postman Problem (DRPP) where not all arcs requiring a service have to be visited provided that a penalty is paid if not served. The problem looks for a tour visiting the selected set of service arcs while minimizing both traversing and penalty costs. The problem is known as Directed Profitable Rural Postman Problem (DPRPP). We propose a matheuristic that uses the optimal solution of a problem relaxation to identify different subsets of service arcs on which to formulate and solve a sequence of DRPPs. A second heuristic approach is then applied as final refinement. The method uses as input the best solution found so forth and exploits the structure of a branch and cut approach where subtours are eliminated through the insertion of cuts heuristically determined and that may exclude feasible solutions. Finally, a branch and cut approach that makes use of subtours elimination cuts introduced in a of "lazy way" is presented. The method is enhanced by a heuristic routine solving suproblems in nodes of the solution tree. All proposed methods have been tested on benchmark instances from the literature and compared to state of the art algorithms. Results show that heuristic methods are extremely effective frequently outperforming existing algorithms, whereas the exact method finds the optimal solution for all but three open instances in less than one hour.

Keywords: Directed Profitable Rural Postman Problem, Arc Routing with profits, Matheuristic.

1 Introduction

Many studies appeared in the literature about the famous Traveling Salesman Problem (TSP) and its variations (see Gutin and Punnen [23]) and about its multi vehicle generalizations (see Toth and Vigo [27] for a survey on Vehicle Routing Problems (VRP)). The general aim of these problems is to find one (or more) optimal tour(s) to serve all or a selected group of customers. In these problems, customer demands are located at vertices of a graph. In Arc Routing Problems (ARPs) demands are located along the edges instead that at vertices. Although the difference between these two classes of problems seems to be minor, the structure of ARPs is different and specialized solution methods need to be devised for these problems (see Dror [16], Eiselt *et al.* [17] and [18] for surveys and Corberán and Prins [12] for an annotated bibliography).

In this paper we analyze a generalization of the Directed Rural Postman Problem (DRPP) (see Eiselt *et al.* [18]) where some required arcs may not be served provided that a penalty is paid for each of them. The problem is defined on a directed graph $G(V, A)$, where $V = \{0, ..., n\}$ is the set of nodes (node 0 represents the depot), and $A = \{(i, j) \mid i \neq j, i, j \in V\}$ is the set of directed arcs. At each arc $(i, j) \in A$ is associated a traveling cost c_{ij}. Let $R \subseteq A$ be the set of arcs that require a service (*service arcs*). We assume that directed paths in G exist between every pair of nodes that are terminal nodes of arcs in R. A penalty p_{ij} is assigned to each arc $(i, j) \in R$ that represents the cost that has to be paid if the corresponding service arc is not selected in the optimal solution. The problem looks for a tour starting and ending at the depot so that the total cost, given by the sum of traveling costs and total penalty paid for non traversing some service arcs, is minimized. This problem was first introduced by Guastaroba *et al.* [22] under the name of Shipper's Lane Selection Problem (SLSP), and recently reformulated by Archetti *et al.* [4] who called it Directed Profitable Rural Postman Problem (DPRPP) since minimizing traveling and penalty costs can be reformulated as maximizing the difference between profit gained serving required arcs and traveling costs.

The problem belongs to the class of arc routing problems with profits. Whereas a large literature exists for node routing problems with profits (see Feillet *et al.* [20] and Vansteenwegen *et al.* [28] for a survey about the TSP with profits and the team orienteering problem, respectively), still very few contributions can be found on their arc routing counterpart. In [15], Deitch and Ladany introduce the one-period Bus Touring Problem (BTP) in which profits are placed on nodes and arcs. In this problem the profits are non-negative attractiveness values and the objective is to maximize the total attractiveness of the tour by selecting a subset of nodes (sites) to be visited and arcs (scenic routes) to be traveled subjected to constraints on touring time, cost and total distance. Feillet *et al.* [19] introduce the Profitable Arc Tour Problem (PATP). The problem objective is to maximize the profit located on arcs minus the travel costs using a fleet of unlimited vehicles with no capacity constraint. No depot is considered and the number of times profit is available on arcs is limited. The use of more vehicles is due to an upper bound on the length of each tour. More recently, Aráoz *et al.* [1] introduce the Privatized Rural Postman Problem (PRPP). The problem can be seen as the undirected version of the DPRPP and looks for a tour starting and ending at the depot while maximizing the difference between the profits gained serving required arcs and the traversing costs. The authors study polyhedral properties of the problem identifying some dominance relations. Later on, Aráoz *et al.* [2] rename the problem Prize-collecting Rural Postman Problem. They identify upper bounds solving iteratively relaxed models with a small number of inequalities and adding violated cuts through exact separation procedures. They also provide lower bounds for the problem generating feasible solutions using the 3T heuristic by Fernández *et al.* [21]. They introduce a problem formulation using two sets of binary variables one indicating when an edge is traversed and served the first time, one indicating the deadheading traversal and show that the number of times an edge can be traversed in an optimal solution is never higher than 2. The latter is a known result also for the Rural Postman Probem (RPP) (see Christofides *et al.* [9]). The RPP objective is to find a tour, starting from the depot, crossing all the requested edges in a set R and minimizing the total cost. RPP was first introduced by Orloff [26] and is known to be NP-hard but for some special cases as when graph induced by the requested edges is connected. Note that, once selected the service arcs to be visited, the DPRPP reduces to a directed RPP.

Starting from the prize-collecting RPP described in Aráoz *et al.* [1], many authors introduce different variants of the problem; see, for instance, the clustered prize-collecting ARP by Aráoz *et al.* [3] and the windy clustered prize-collecting ARP by Corberán *et al.* [13]. More recently, Black *et al.* [7] study the time-dependent prize-collecting Arc Routing Problem (TD-PARP) inspired to the PRPP but with the addition of real constraints. The graph is directed and traversing costs depend on the time they are crossed. Moreover, the problem allows for more requests between the same two nodes.

Motivation

The directed profitable RPP (DPRPP) studied in this paper finds application in the domain of the transportation services procurement where companies need to decide which customers to serve directly and which assign to external operators paying an outsourcing cost. In Guastaroba *et al.* [22] the problem has been first introduced as the problem of a shipper that has to decide which lanes (service arcs) to pick out for a direct service with its own vehicle, and which to assign conveniently at external carriers paying an outsourcing cost. Since lanes outsourcing is made through an auction, the problem objective is the minimization of the sum of traversing costs and outsourcing costs plus the fixed cost of auction set-up. The authors provide a mathematical formulation using binary instead of integer variables, thus allowing a single crossing for each arc. Other works can be found in this application area almost all referring to node routing problems. See for instance Chu [11] who first analyzes the problem of deciding how many private vehicles to employ in truckload mode (and then defining their routing) and how many external carriers to use in less than truckload mode when the customers demand is more than the available vehicle capacity. A slightly different problem formulation is also analyzed by Côté and Potvin [14] who propose a tabu search algorithm to solve it.

Contribution

The DPRPP is an NP-hard problem being a generalization of the directed RPP (DRPP). This motivates the study of heuristic approaches to solve it. Archetti *et al.* [4] introduce an effective tabu search algorithm with the addition of an ex-post ILP-refinement. They also provide a set of benchmark instances for the problem many of which have not been solved to optimality up to now.

This paper provides different contributions. We introduce a tighter problem formulation than that proposed in [4]. We use it to develop a branch-and-cut algorithm where subtours elimination constraints are introduced in a "lazy way", i.e. only when integer solutions containing subtours are determined in the search tree. The method is further enhanced by the introduction of a routine that solves optimally the directed rural postman problem on subset of service arcs selected heuristically. The routine is called iteratively after a predefined number of iterations, and uses information provided by the optimal solutions of the continuous relaxation at nodes of the branch and bound tree to pick out service arcs on which to solve DRPPs. The exact method results to be highly efficient allowing to find the optimal solution, within a time limit of one hour, for 19 out of the 22 instances not solved to optimality yet by Archetti *et al.* [4] on a comparable PC.

The selection of service arcs to be visited can be seen as a first critical step to solve the DPRPP. Once identified a set of service arcs to be visited, a directed Rural Postman problem needs to be solved. We show how the optimal solution of the problem relaxation obtained removing subtours elimination constraints can be used to identify subsets of "promising" service arcs to be visited. In particular, we propose a matheuristic that exploits information provided by this relaxation to identify subsets of service arcs on which to formulate and solve to optimality Directed Rural Postman problems. Each DRPP is solved on a *restricted graph* induced by the selected service arcs. The procedure consists of two main routines, one considering subsets of service arcs of increasing size and the other one of decreasing size. Finally, a refinement is introduced through a heuristic method that receives as input the best solution found so forth and exploits the branch and cut structure of a common MILP solver. The method gets quick convergence through the introduction of heuristic cuts that may exclude feasible solutions of the original problem. More precisely, all the times an integer solution is found in the branch and bound solution tree, the heuristic eliminates smallest subtours as in an exact approach, whereas subtours with a size larger than a predefined number of nodes are excluded in a heuristic way. In the latter case connectivity cuts are inserted forcing the visit to the selected subtours. The resulting constraints are not guaranteed to be valid inequalities for the problem.

The article is organized as follows. In Section 2 the mathematical formulation of the problem is proposed with some simple properties. We also describe the mathematical formulation used to optimally solve the directed RPP when a set of service arcs to be visited is selected (auxiliary problem). Section 3 is devoted to solution algorithms description, whereas in Section 4 we test both our exact algorithm and the heuristic approaches on benchmark instances and compare their performance with state of the art algorithms. In Section 5 some conclusions and future developments are drawn.

2 Mathematical Formulation

We start describing the integer linear programming formulation for the DPRPP. The model uses $O(|V|^2)$ integer variables x to indicate how many times each arc is traversed in the optimal solution and other $O(|V|^2)$ binary variables y to select arcs to serve. Finally $|V|$ binary variables z are used to formulate subtours elimination constraints:

$$x_{ij} \geq 0 \quad \text{number of times arc } (i,j) \text{ is traversed in the optimal solution} \quad (i,j) \in A$$

$$y_{ij} := \begin{cases} 1 & \text{if arc } (i,j) \text{ is served in the optimal solution} \\ 0 & \text{otherwise;} \end{cases} \qquad (i,j) \in R$$

$$z_j := \begin{cases} 1 & \text{if node } j \text{ is visited in the optimal solution} \\ 0 & \text{otherwise.} \end{cases} \qquad j \in V$$

The problem can be formulated as follows:

$$
\begin{array}{llll}
\text{(DPRPP)} & \min \sum_{(i,j)\in A} c_{ij}x_{ij} + \sum_{(i,j)\in R} p_{ij}(1-y_{ij}) & & (1)\\
\text{s. to:} & \sum_{i\in V\setminus\{j\}} x_{ij} = \sum_{i\in V\setminus\{j\}} x_{ji} & j \in V & (2)\\
& x_{ij} \geq y_{ij} & (i,j)\in R & (3)\\
& z_j \leq \sum_{i\in V\setminus\{j\}} x_{ij} \leq (|R|+1)z_j & j\in V & (4)\\
& z_0 \geq z_j & j \in V\setminus\{0\} & (5)\\
& \sum_{i\notin P}\sum_{j\in P} x_{ij} \geq \frac{\sum_{j\in P} z_j}{|P|} & P\subseteq V\setminus\{0\}, P\neq\emptyset & (6)\\
& x_{ij}\geq 0 \text{ integer} & (i,j)\in A & (7)\\
& y_{ij}\in\{0,1\} & (i,j)\in R & (8)\\
& z_j \in \{0,1\} & j\in V & (9)
\end{array}
$$

The objective function (1) minimizes the sum of traveling costs and penalties paid for all service arcs not selected. Constraints (2) are the so called in-degree and out-degree constraints and impose the equivalence between the number of arcs entering and leaving each node $j \in V$. Constraints (3) imply that arc $(i,j) \in R$ can be served ($y_{ij} = 1$), if and only if it has been traversed ($x_{ij} \geq 1$). Constraints (4) control values of z_j variables. If at least one arc is entering node j, then variable z_j is forced to 1 (right inequality), otherwise z_j is forced to zero (left inequality). Constraints (5) impose the selection of variable z_0 when any other node is visited thus requiring, through constraint (4), that arcs have to enter and leave the depot. Constraints (6) prevent the generation of isolated cycles. Finally, constraints (7),(8),(9) define integer and binary conditions.

Note that, as in classical profitable problems, objective function (1) can be reformulated as follows:

$$\max \sum_{(i,j)\in R} p_{ij}y_{ij} - \sum_{(i,j)\in A} c_{ij}x_{ij} - \sum_{(i,j)\in R} p_{ij}.$$

With respect to the formulation proposed in Archetti *et al.* [4] we add a variable z_0 associated to the depot, the set of valid inequalities (5) that forces to pass from the depot when at least a node is visited, and the lower bounds in inequalities (4). These constraints are not necessary, but make stronger the problem continuous relaxation and the relaxation obtained removing connectivity constraints (6). Finally, an alternative formulation for the DPRPP could be obtained generalizing the model proposed by Aráoz *et al.* [1]. In this case two sets of variables are required both defined on the complete set A of arcs. The first set consists of binary variables that indicate the first time arcs (if any) are traversed and served in the optimal solution, whereas the second set consists of integer variables counting the additional times such arcs are traversed.

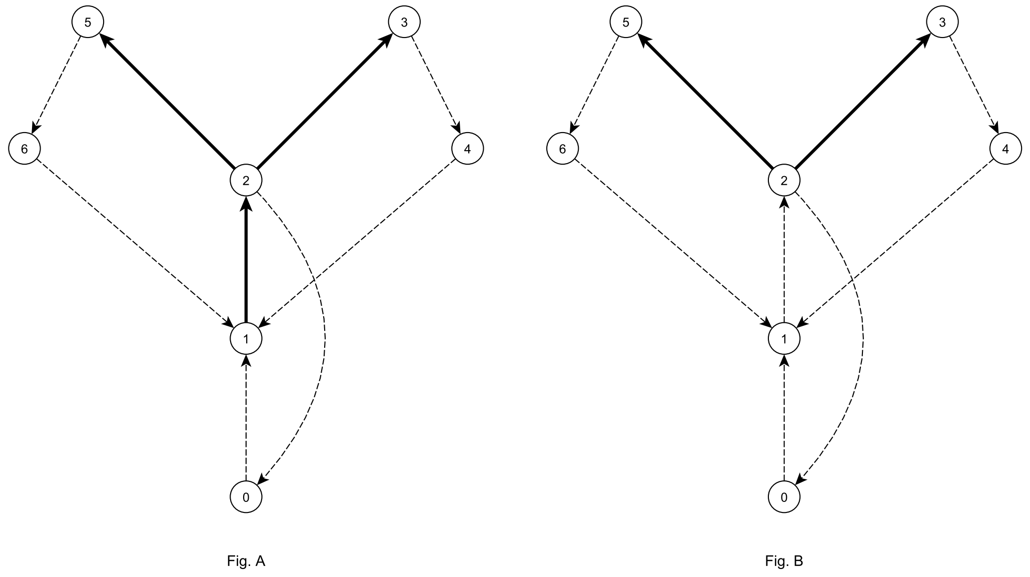

Figure 1: Upper bounds on arc traversals.

Let $R^* \subseteq R$ be the set of service arcs selected in an optimal solution of the DPRPP and x^*_{ij} the optimal value of variable x_{ij}, $(i, j) \in A$. The following proposition is trivially true:

Proposition 1 *In an optimal solution of the DPRPP, $x^*_{ij} \le |R^*|$ if $(i, j) \in R$ and $x^*_{ij} \le |R^*| + 1$ otherwise.*

Instance depicted in Figure 1A shows a DPRPP instance with three service arcs $(1, 2)$, $(2, 3)$, $(2, 5)$ (shown in bold lines). Assuming that penalties and costs are such that all service arcs are selected in an optimal solution, then variable x_{12} will take value $|R^*| = |R| = 3$. Figure 1B shows the opposite case, in which arc $(1, 2)$ does not belong to R. In this case, variable x_{12} will take value $|R^*| + 1$, with $|R^*| = 2$.

Let us consider the DPRPP relaxation obtained excluding subtours elimination constraints (6). From now on, we will refer to such a relaxation as to problem RELAX. We indicate as R_{REL} the set of service arcs selected by the optimal solution of problem RELAX. We have already defined $R^* \subseteq R$ as the set of service arcs selected in an optimal solution of the DPRPP. The following proposition holds:

Proposition 2 *In general, R^* is not included in R_{REL}.*

Proof Trivially, an isolated service arc (i, j) (i.e. a service arc not adjacent to any other service arcs) with $p_{ij} < c_{ij}$ will never be selected by the optimal solution of problem RELAX. On the contrary it will make part of the DPRPP optimal solution if arc (i, j) is necessary to ensure connectivity to other service arcs visited by the optimal integer solution. $\square$

In spite of this negative result, we will show how, the set R_{REL} usually contains *promising service arcs*, that is service arcs that will be likely visited in an optimal solution of the DPRPP. In the following section, we will describe a heuristic algorithm that selects different combinations of service arcs in R_{REL} to and then formulates and solves to optimality the resulting instance of the Directed Rural Postman Problem.

Given a subset $\bar{R} \subseteq R$ of service arcs, let $\bar{G}(\bar{V}, \bar{A})$ be the graph induced by $\bar{R}$, where the set of nodes $\bar{V} \subseteq V$ is the set of endpoints of arcs in $\bar{R}$ plus the depot (if not already included), and the set of arcs is given by the set $\bar{R}$ plus arcs between every pair of nodes in $\bar{V}$. A cost c_{ij} is assigned to each arc in $\bar{A}$ as follows: if the arc $(i,j) \in \bar{R}$ then c_{ij} is its original traversing cost, otherwise c_{ij} is the cost of the shortest path connecting i to j in the original graph G. The directed Rural Postman Problem visiting all required arcs of set $\bar{R}$ in a tour at minimum cost can be formulated as follows:

$$
\begin{array}{rlll}
(AUX) & \min \sum_{(i,j)\in\bar{A}} c_{ij} x_{ij} & & (10)\\
\text{subject to:} & \sum_{i\in\bar{V}\setminus\{j\}} x_{ji} = \sum_{i\in\bar{V}\setminus\{j\}} x_{ij} & j \in \bar{V} & (11)\\
& x_{ij} \geq 1 & (i,j) \in \bar{R} & (12)\\
& \sum_{i\in K}\sum_{j\notin K} x_{ij} \geq 1 & K \subseteq \bar{V}\setminus\{0\}, K \neq \emptyset & (13)\\
& \sum_{j\in\bar{V}\setminus\{0\}} x_{0j} \geq 1 & & (14)\\
& x_{ij} \geq 0 \text{ integer} & (i,j) \in \bar{A} & (15)
\end{array}
$$

Each variable x_{ij} is non negative integer and represents the number of times arc $(i,j) \in \bar{A}$ is traversed. Objective function (10) minimizes the sum of traversing costs. Constraints (11) are the classical in-degree and out-degree constraints. Constraints (12) ensure the traversing of the set of selected service arcs. Constraints (13) are common connectivity constraints which can be used since all nodes in graph $\bar{G}$ insist on at least one arc in $\bar{R}$. Constraint (14) guarantees that at least one arc has to leave the depot. Finally, constraints (15) define the nature of variables x_{ij}. From now on we will refer to this formulation of the directed RPP as to the *auxiliary problem* (AUX). Alternative formulations of the problem can be found in Ball and Magazine [5] and Christofides *et al.* [10].

Auxiliary problem will be largely used both in the exact approach and in the heuristic methods as external routine to find good feasible solutions given a promising subset of service arcs to visit. For its solution we implement a branch and cut approach. First the subproblem without constraints (13) is optimally solved. Note that constraint (14) is not necessary in AUX formulation, but it makes stronger the problem relaxation obtained eliminating constraints (13). The constraints (13) are added when an integer solution is found in the search tree, i.e. they are added in a "lazy way". We will better describe the concept of "lazy way" when introducing our branch and cut method.

3 Solution Algorithms

In this section we describe the two heuristic procedures and the branch and cut approach proposed for solving the DPRPP.

3.1 Exact solution algorithm

Our branch and cut algorithm is different from that proposed in [4] that uses the max-flow algorithm by Boykov and Kolmogorov [8] as separation procedure for inequalities (6) and applies it starting at the root node. Instead of solving continuous relaxation and inserting violated subtours elimination constraints directly at the root node, our algorithm starts by solving to optimality the RELAX problem. As soon as an integer solution is found in the branch and bound search tree, isolated tours are looked for and, if found, they are prevented by adding the inequalities (6) on the subsets P identified by the isolated cycles. A solution becomes feasible when no more isolated tours are found in the integer solution, that is, no more cuts will be discovered on that tree branch. Constraints added to an integer solution are usually weak in a polyhedral sense, since they only remove a single infeasible integral point. For this reason they are used in a "lazy way" by the branch and cut algorithm with the purpose of checking the feasibility of the integral solutions found along the tree. In our case the "lazy constraints" added do not only eliminate current infeasible solution, but also exclude all integral solutions that contain such tours. We believe that the performance of our branch and cut method highly depends on the time saving obtained solving RELAX problem and then introducing subtours elimination constraints in a lazy way instead of solving separation algorithm for subtours elimination from scratch as in [4]. Our method avoids the introduction of the high number of constraints that the second method has to deal with since the beginning.

Moreover, in order to improve our exact method performance, the auxiliary problem AUX is solved on subsets of service arcs determined using information provided by solutions at nodes of the search tree. A heuristic routine takes the optimal solution of the continuous relaxation at a node of the search tree, and rounds the value of variables associated to service arcs (i.e. all y_{ij} variables with fractional value) to the closest integer. Selected service arcs are then used as input to problem AUX. Note that given a set of required arcs the solution of the auxiliary problem AUX will provide the best way to serve them. This may possibly include the selection of additional service arcs located on the shortest path connecting nodes of the reduced graph on which the auxiliary problem is formulated. If the solution found by solving the auxiliary problem is better than the actual incumbent solution, the latter is updated making bounding operations possibly more effective. This heuristic routine is called iteratively along the whole search tree.

3.2 Heuristic solution algorithms

Most of heuristic methods provided in the literature for arc routing problems transform problem instances into node routing instances. We propose two solution algorithms that directly use arc routing formulation of the problem. The first approach is a matheuristic where at each iteration a new subset of service arcs to be visited is selected and then a directed rural postman problem is solved to optimality. Combining heuristics and exact methods appears to be a very promising alternative in solving hard combinatorial problems, we refer interested readers to Maniezzo *et al.* [25]. The second procedure is a final refinement that tries to improve the best solution found by the matheuristic using the structure of a branch and cut search, but getting quick convergence through the introduction of subtours elimination constraints that may exclude feasible solutions.

Relaxation Based Heuristic

The first method solves problem RELAX and uses different combinations of arcs in R_{REL} to construct DRPP instances that are then solved to optimality using the auxiliary problem AUX. We call this algorithm Relaxation Based Heuristic (RBH) and report its pesudocode in Algorithm 1.

At Step 1 of Algorithm 1 the values r^* and w^* are initialized to the case where no service arcs are selected: the objective function value $w*$ is set equal to the sum of all penalties, whereas tour r^* is void. Then the optimal solution (x^*_{REL}, y^*_{REL}) of problem RELAX is determined and the set of service arcs R_{REL} visited by such a solution identified (Step 2). If the optimal solution of problem RELAX does not select any arcs, then initialization values at Step 1 are optimal for the DPRPP and the algorithm stops providing them. This is also the case if the optimal solution of problem RELAX contains a unique tour closed to the depot (Steps 3-5). Otherwise, the three subroutines ISA, IDA and IIA are called and the best tour found r^* with total cost w^* is returned at the end.

Pseudo code of the Initial Solution Algorithm (ISA) is described in Algorithm 2. At Step 1 this subroutine finds a first feasible solution solving a DRPP using as required arcs the complete set R_{REL} of service arcs selected by the optimal solution of problem RELAX.

The optimal solution of problem RELAX can have isolated tours. If an isolated tour contains only one service arc, we call such an arc an *isolated service arc*. To be selected in the optimal solution of problem RELAX an isolated service arc must have a penalty larger than the traversing cost of the corresponding tour. The selection of isolated service arcs in an optimal solution of the DPRPP depends on how convenient they remain after imposing their connection to the depot tour. Whereas it may happen that an isolated service arc is selected in the optimal solution of the DPRPP, we notice that this is usually not the case.

Algorithm 1 Relaxation Based Heuristic (RBH)

```
Require: Graph G=(V,A), set $R$ of service arcs. Maximum computational time $T_{max}$.
Ensure: a feasible tour $r^*$ with total cost $w^*$.
 1: Set $r^* := \emptyset$ and total cost $w^* := \sum_{(i,j)\in R} p_{ij}$.
 2: Solve problem RELAX. Let $(x^*_{REL}, y^*_{REL})$ be its optimal solution and let $R_{REL} \subseteq R$ be
    the subset of service arcs selected in $(x^*_{REL}, y^*_{REL})$.
 3: if $y^*_{REL} = \underline{0}$ then return $(r^*, w^*)$.
 4: if $(x^*_{REL}, y^*_{REL})$ contains a unique cycle $\bar{r}$ closed on the depot with total cost $\bar{w}$ then
 5:     Set $r^* := \bar{r}$ and total cost $w^* := \bar{w}$.
 6:     return $(r^*, w^*)$
 7: else
 8:     $(r', w') :=$ ISA($(x^*_{REL}, y^*_{REL})$, $R_{REL}$).
 9:     if $w' < w^*$ then set $w^* := w'$, $r^* := r'$.
10:     $(r', w') :=$ IDA($(x^*_{REL}, y^*_{REL})$, $R_{REL}$).
11:     if $w' < w^*$ then set $w^* := w'$, $r^* := r'$.
12:     $(r', w') :=$ IIA($(x^*_{REL}, y^*_{REL})$).
13:     if $w' < w^*$ then set $w^* := w'$, $r^* := r'$.
14:     return $(r^*, w^*)$.
15: end if
```

Algorithm 2 Initial Solution Algorithm - ISA((x^*_{REL}, y^*_{REL}), R_{REL})

```
1: Solve problem AUX using set $R_{REL}$ as required arcs. Let $r$ be the tour found and $w$ its
   total cost.
2: Let $\tilde{R}$ be the set of isolated service arcs selected in $(x^*_{REL}, y^*_{REL})$.
3: if $\tilde{R} \neq \emptyset$ then
4:     Solve problem AUX using set $R_{REL} := R_{REL} \setminus \tilde{R}$. Let $\tilde{r}$ be the tour found and $\tilde{w}$ its
       total cost.
5: end if
6: if $\tilde{w} < w$ then set $w := \tilde{w}$, $r := \tilde{r}$.
7: return $(r, w)$.
```

So, we decide (Steps 2–5) to eliminate all isolated service arcs from set R_{REL} and then solve problem AUX on the remaining set. Algorithm 2 terminates providing the best between the two solutions found.

Routines IDA and IIA represent the two main phases of the Relaxation Based Heuristic. Their pseudo codes are illustrated in Algorithms 3 and 4.

Isolated subtours are selected in the optimal solution of RELAX problem when the sum

Algorithm 3 Improvement by Decreasing Algorithm - IDA((x^*_{REL}, y^*_{REL}))

```
1: Set $r = \emptyset$ and $w = \sum_{(i,j)\in R} p_{ij}$.
2: Let $\tilde{n}$ be the # of isolated tours in $(x^*_{REL}, y^*_{REL})$ with more than 2 service arcs.
3: for $i = 1, ..., \tilde{n}$ do
4:     Let $R_i$ be the set of service arcs belonging to tour $i$. Set $R'_{REL} := R_{REL} \setminus R_i$.
5:     Solve problem AUX using set $R'_{REL}$ as required arcs. Let $r_i$ be its solution and $w_i$ its
       total cost.
6:     if $w_i < w$ then set $w := w_i$, $r := r_i$.
7: end for
8: Let $r_0$ be the tour passing through the depot in $(x^*_{REL}, y^*_{REL})$ and $w_0$ its total cost.
9: if $w_0 < w$ then set $w := w_0$, $r := r_0$.
10: return $(r, w)$.
```

of traversing costs is lower than the sum of penalties. Nevertheless, their actual convenience for the optimal solution depends on the cost to connect them to the depot. To evaluate the convenience not to serve an isolated tour found by the RELAX problem optimal solution, Algorithm IDA removes, in turn, the service arcs of each isolated tour from the set R_{REL}. At each iteration of the **for cycle** (Steps 3–7 of Algorithm 3) a new set of required arcs R_i is determined and then eliminated from R_{REL}. The resulting set of service arcs R'_{REL} is provided as required arcs to construct a DRPP instance (problem AUX) solved to optimality. If no convenient paths exist to connect isolated tours to the depot, it may happen that the tour passing through the depot in the optimal solution of problem RELAX is already optimal for the original problem. Algorithm IDA takes this into account in Step 8.

Subroutine Improvement by Increasing (IIA) aims at evaluating the convenience to connect each isolated tour with the tour passing through the depot. This should represents a measure of the convenience to insert it into the optimal solution. At this aim during each iteration of the first **for cycle** (Steps 4–7) the service arcs selected by an isolated tour are added to those (if any) in the tour passing through the depot and used as input to problem AUX. For each set $R_i \cup R_0$ of required arcs defined at Step 5, we keep the objective function value (total cost) of the solution found by the problem AUX and in Step 8 we sort isolated tours in non decreasing order of such values. Then, following such an order, the iterations of the second **for cycle** starting at Step 10, evaluate solution obtained by increasingly including the set of service arcs of each isolated tour, one after the other, to the set of service arcs of the tour passing through the depot. Thus, at each iteration a new instance of problem AUX is solved receiving as input an increasing set of required arcs represented by the set of service arcs considered in the previous iteration plus the arcs of the isolated tour under consideration.

Algorithm 4 Improvement by Increasing Algorithm - IIA((x^*_{REL}, y^*_{REL}))

```
1: Set $r = \emptyset$ and $w = \sum_{(i,j)\in R} p_{ij}$.
2: Let $\tilde{n}$ be the # of isolated tours in $(x^*_{REL}, y^*_{REL})$ with more than 2 service arcs.
3: Let $R_0$ be the set of service arcs selected by the tour passing through the depot in $(x^*_{REL}, y^*_{REL})$.
4: for $i = 1, ..., \tilde{n}$ do
5:     Let $R_i$ be the set of service arcs selected by tour $i$. Solve problem AUX using set $R_i \cup R_0$. Let $r_i$ be the solution tour and $w_i$ its total cost.
6:     if $w_i < w$ then set $w := w_i$, $r := r_i$.
7: end for
8: Sort sets $\{R_i, i = 1, ..., \tilde{n}\}$ in non decreasing order of solution costs $\{w_i, i = 1...\tilde{n}\}$.
9: Set $\bar{R} := R_1 \cup R_0$.
10: for $i = 2$ to $\tilde{n} - 1$ do
11:     Solve problem AUX using set $\bar{R} := \bar{R} \cup R_i$. Let $\bar{r}$ be its solution tour and $\bar{w}$ its total cost.
12:     if $\bar{w} < w$ then set $w := \bar{w}$, $r := \bar{r}$.
13: end for
14: return $(r, w)$.
```

Post-processing of RBH

As final post-processing the whole Algorithm 1 is repeated changing the solution found by the RELAX problem at Step 2 through the insertion of a constraint forcing the selection of a larger number of service arcs by a defined percentage γ, as follows:

$$\sum_{(i,j)\in R} x_{ij} \geq |R_{REL}| * (1 + \gamma).$$

If $\gamma|R_{REL}| > |R| - |R_{REL}|$, then the right hand side is set to $(1 - \gamma)(|R| - |R_{REL}|)$. If $|R_{REL}| = |R|$ post processing is not applied.

Heuristic Cuts Search (HCS)

The best solution found by procedure RBH is used as input to an algorithm that exploits the structure of a branch and cut algorithm but inserts cuts for subtours elimination in a heuristic manner. Resulting cuts may not be valid inequalities thus excluding integer feasible solutions. In particular, at each node of the branch and bound tree when an integer solution is available the procedure identifies isolated subtours. If the cardinality of the set P of nodes making part of the subtour is larger than $\lceil \alpha \frac{|A|}{\lfloor \frac{|A|}{|V|} \rfloor} \rceil$ the following cut is introduced:

$$\textstyle\sum_{i\in P}\sum_{j\notin P} x_{ij} \geq 1 \quad \forall\ P \subseteq V \setminus \{0\}, P \neq \emptyset \tag{16}$$

In all the other cases, exact violated cuts are added.

Introduction of an inequality (16) prevents the creation of a particular isolated tour forcing its visit (right hand side of each constraint (16) is a constant). This may cause the exclusion of some problem feasible integer solutions. The rationale for their introduction is that larger isolated tours are more likely to belong to an optimal solution and thus they are forced directly without identification of the valid violated cut. The algorithm stops when a time limit is met. As our branch and cut algorithm also this heuristic makes an intensive use of the heuristic routine solving problem AUX on predefined sets of service arcs at nodes of the search tree. We call this procedure Heuristic Cuts Search (HCS).

4 Experimental analysis

In this section we present the results obtained testing our algorithms on the set of benchmark instances provided by Archetti et al. [4] that can be downloaded at the web site `http://www.unibs.it/sites/default/files/ricerca/allegati/13004DPRPP.zip`. We reorganize these benchmark instances in the following sets:

- Set 1: instances DPRPP_ϵ_δ_valXA originally come from those by Benavent *et al.* [6] modified by introducing penalties. Parameters (ϵ, δ) define the interval $[\epsilon c_{ij}, \delta c_{ij}]$ in which penalty p_{ij} for each arc (i, j) is randomly generated and are set equal to $(1.0, 2.0)$, $(1.5, 2.5)$ and $(2.0, 3.0)$, respectively. The total number of instances is equal to 30.

- Set 2: instances DPRPP_ϵ_δ_egl-eY-A and DPRPP_ϵ_δ_egl-sY-A, originally taken from Li and Eglese [24], are obtained setting parameters (ϵ, δ) equal to $(1.0, 2.0)$, $(1.5, 2.5)$ and $(2.0, 3.0)$, respectively. This gives rise to 24 instances altogether.

- Set 3: 26 instances DPRPP_P.

- Set 4: 36 instances DPRPP_D.

- Set 5: 36 instances DPRPP_G.

- Set 6: 20 instances DPRPP_R.

Sets 3–6 come from Aráoz *et al.* [2], where the original profits of these problems have been used as penalties for the DPRPP. At the website `http://www.ing.unibs.it/~orgroup/`

instances.html we provide new best known values and new optimal solutions we get on these benchmark instances.

All tests have been performed using a Laptop with 1.73 Ghz processor in Windows Seven operating system. Algorithms have been coded in C++ and models implemented in ILOG Concert Technology 2.0 and solved using CPLEX 10.1.

4.1 Exact results

In this section we show the results obtained by our exact procedure. We set a computational time limit of 3600 seconds after which the branch and cut search is stopped and the best feasible solution found as well as its percentage gap with respect to the best lower bound are provided as output.

In Tables 1-6 for each set of instances we report three variants of our exact procedure. The first one, named "Basic", uses the same problem formulation as that proposed in Archetti *et al.* [4]. The second one, named "New", uses our formulation of the DPRPP that includes new inequalities. Finally, the last one, named "New + Aux", uses formulation New along with the heuristic routine that solves the problem AUX iteratively at nodes of the search tree and uses as starting initial solution the payment of all penalties. In the last variant, after some preliminary tests we have decided to set to 10 seconds the maximum time available to solve the auxiliary problem (AUX) at a tree node. Tests have shown that, on average, CPLEX can optimally solve this problem in less than 10 seconds. Moreover, we set to 2% the percentage gap between incumbent integer solution and best bound, under which the auxiliary problem is not solved anymore. Preliminary tests have shown that, on average, it is difficult that a solution of this problem can improve the best incumbent under this gap value. Finally, to limit the number of times the auxiliary problem is solved, we set a CPLEX condition that calls the heuristic only if the current iteration is a multiple of 100.

All Tables 1-6 have the same structure. The first column provides instance name, whereas columns "$|V|$" and "$|R|$" indicate the size of solved instance, namely the number of nodes and of service arcs, respectively. Column "Opt." shows the optimal solution value of an instance if it is found by at least one variant; the entry is highlighted in bold style when a new (not previously known) optimal solution value is found. If at least one variant finds a solution value that improves the best known value the entry in column "Opt" is in italic font. Columns "Time" and "# cuts" report the time in seconds required by each variant to get the optimal solution and the number of cuts added to prevent isolated tours. If a variant does not find the optimal solution within the computational time limit of 3600 seconds in column "Time" we report (into brackets) the best integer solution found and its percentage gap from the best lower bound. These values into brackets are also reported when the algorithm goes out of memory. In such a case the computing time is put before the bracket, whereas a symbol "*" is put in the name of the instance when all tested variants terminates out of memory.

Table 1 shows the results on Set 1 of instances. DPRPP_ϵ_δ_valXA are very easy instances: their solution always requires less than 1 second. Note that in variants New and New+Aux the number of added cuts is lower than in the formulation Basic. Since these instances are extremely easy the introduction of a heuristic routine solving DRPPs seem to be meaningless. In Table 2 are shown the results for the instances in Set 2. DPRPP_ϵ_δ_egl-eY-A and DPRPP_ϵ_δ_egl-sY-A instances are among the most difficult ones. Our algorithm has been able to find the optimal solution for 6 previously open instances (bold entries). In this set two instances are still not solved to optimality (namely instances DPRPP_1.0_2.0_egl-s2-A and DPRPP_1.5_2.5_egl-s1-A) but, in one case, formulation New+AUX found a new best known integer solution value. Comparing the Basic variant with the New+AUX variant, one can note that both the average computational time and the average number of cuts are lower for the second one. Thus, the introduction of new constraints and of a heuristic routine solving the auxiliary problem (AUX) seem to help Branch and Cut algorithm. Table 3 shows the results obtained on instances of the Set 3. If we exclude instances DPRPP_ALBAIDAANoRPP and DPRPP_ALBAIDABNoRPP that are the most difficult problems in the set, the other ones are still quite easy instances solved, on average, in less than 1 second and always requiring a limited number of cuts. Table 4 reports solutions found for Set 4 (instances DPRPP_D). This set shows instances with a large deviation in terms of number of nodes and service arcs. The complexity of such instances increases with their size from the first to the last one. Our algorithm finds the optimal solution using all the three variants for all the instances and has been able to close all the four open instances. Variant New has reduced the computational time by about 7 times with respect to variant Basic and halved the number of cuts that have to be inserted to get optimal solution. Comparing the columns "# cuts" for variants New and New + Aux can be noted that in some instances the solution of the auxiliary problem seem to be helpfull. In Table 5 are shown results on Set 5 (instances DPRPP_G). As for DPRPP_D instances, size and complexity increase from the first to last one. Our algorithm performs well always finding the optimal solution but for one instance and closing 7 out of 8 previously open instances. Variants New and New + Aux both behave quite well almost halving the average computational time and the average number of cuts with respect to variant Basic. Finally, Table 6 shows the results on Set 6 (instances DPRPP_R) where variant New provides the best performance immediately followed by variant New + Aux. Our algorithm has been able to find the optimal solution for all the instances, closing two previously optimally unsolved instances. In conclusion, our exact algorithm, in its various variants, has been able to solve all instances but 3. The experimental analysis shows that the introduction of new constraints on variables z_i create a stronger formulation that significantly help Branch and Cut resolution. The use of a heuristic routine solving directed rural postman problems on subsets of arcs (problem AUX) seem to help algorithm performance. It is evident how the use of this heuristic does not seem to heavly affect the average computational time. Besides, in those instances not solved to optimality it has allowed the exact algorithm to find the best known solution values.

4.2 Heuristic Results

In this section we analyze the performance of our heuristic approaches. After some preliminary tests, we have decided to set parameter γ for heuristic RBH to 0.30. The maximum computing time assigned to heuristic HCS has been set to 50 seconds. To limit the number of times the auxiliary problem is solved, we set a CPLEX condition that calls the corresponding heuristic only if the current iteration is a multiple of 10. A maximum computing time of 10 seconds is assigned to the solution of each AUX problem and the percentage gap under which this problem is not solved anymore is set to 2% as for our exact algorithm.

In the following tables we compare the performance of the tabu search algorithm and the ILP-refinement (indicated as ILP-ref. in the tables) proposed by Archetti *et al.* [4] with our procedures RBH and HCS, the latter tested with $\alpha = 0$ and $\alpha = 0.05$.

Tables 7–12 have all the same structure. The first column indicates the name of the instance (reduced to the identification number). Column "Best Known" shows the best known solution value for each instance, this corresponds to the optimal solution value in all but three instances identified by symbol "(*)". Column "Gap (%)" provides for each heuristic the percentage gap of its solution value (UB) from the best known value (BK) as follows $\frac{UB-BK}{UB}$ (computed as in Archetti *et al.* [4] in order to guarantee a comparison). Column "Time" indicates the computational time in seconds required by each algorithm. This means that to correctly evaluate the time of the ILP-refinement that receives as input the solution found by the Tabu search, one has to sum the computational time of the two procedures. Thus, differently from Archetti *et al.* [4], the "Time" column of ILP-refinement is the sum of tabu search and ILP-refinement computing times. Similarly, the computational time for our heuristic HCS that receives as input the best solution found by procedure RBH is computed as the sum of the times required by the two procedures. Column "Routine" shows which routine of the method RBH has found the best final solution. If the name of the routine has "(γ)" at the end, it means that the best solution has been found by this routine during the post-processing phase of RBH procedure. For algorithm HCS column "# cuts" provides the number of heuristic cuts used and column "# AUX" shows the number of time the auxiliary problem AUX has been solved and the number of times its solution has improved the best incumbent integer solution (value into brackets).

Table 7 presents results for Set 1. Procedure HCS finds the optimal solution for all instances with both $\alpha = 0.05$ and $\alpha = 0$. In particular, almost all the optimal solutions have been found by routine ISA. Algorithm RBH has always required a computational time lower than 1 second finding an average error almost null, whereas tabu search and ILP-refinement show a performance slightly worse with an average computational time of about 13 and 16 seconds respectively. Table 8 shows heuristic results on instances of Set 2. In this set tabu search performs very well providing an average gap equal to 0.28, whereas ILP-refinement has no effect at all. Our algorithm RBH performs very well but in few instances. Average error provided by RBH is strongly reduced by HCS. It is worth noticing that HCS($\alpha = 0.05$) behaves better than HCS($\alpha = 0$) where all subtours (also the smallest) are forced to be visited. This seems to confirm our idea that smallest subtours selected by problem RELAX are more unlikely to belong to the optimal solution. To conclude, on this set of instances, the performance of HCS and that of ILP-refinement are comparable. On instances of the Set 3 (see Table 9) our algorithm HCS with $\alpha = 0.05$ outperforms state of the art algorithms. Its average computational time and the average gap are equal to 4.55 seconds and 0.08%, respectively, whereas the corresponding values increase to 19.84 seconds and to 0.38% for ILP-refinement by Archetti *et al.* [4]. In Table 10 heuristic results on instances of the Set 4 are presented. Also in this set our algorithm outperforms state of art algorithms. HCS with $\alpha = 0.05$ requires, on average, 32 seconds and gets an average gap equal to 0.29%, whereas ILP-refinement shows an average gap equal to 0.71% and requires about 40 seconds. In Set 5 (see Table 11) the average time of our HCS with $\alpha = 0.05$ is higher by about 8 seconds than that of ILP-refinement, whereas its percentage gap is 0.12% instead of 0.18%. It is worth noticing that, for these instances, HCS is able, on average, to strongly improve results obtained by RBH in less than 3 seconds. Finally, better results can be found on Set 6 in Table 12, where both RBH and HCS algorithms get an average gap better than tabu search and ILP-refinement, respectively and requiring computational times significantly lower. The average computing time of our HCS is lower than 1 second with an average gap equal to 0.25% against the 40 seconds required by the ILP-refinement getting an average gap of 0.46%.

In Table 13 we report a summary comparing all results for ILP-refinement and HCS. The first column indicates the set of instances, then for each algorithm we provide the average (Avg.) and the maximum (Max) gap and the average and the maximum time computed out of all instances in each set. On all solved instances our algorithms provides a better average performance both in terms of time and percentage gap.

Table 14 illustrates the number of times each routine in RBH obtains the best solution. For each routine (i.e. for each row) we provide the number of times it gets the best value (final solution) for algorithm RBH (first column) and for the refinement HCS (second column), respectively. For instance, routine ISA gets the best solution value provided by RBH in 69 instances out of the 172 available, whereas for HCS this happens 56 times, that is in 13 out of 69 times HCS was able to improve the best result obtained by ISA. These 13 instances are included in the 85 where HCS got the best final result improving the result obtained by RBH.

Finally, in Table 15 we provide the number of times each algorithm finds the best solution value in each set of instances, and show into brackets the number of times such values coincide with the best know solution values. If the best known value is reached by more than one algorithm we count it for all of them. For instance, the first row of DPRPP$_\epsilon$ $_\delta_$val results shows that tabu search and ILP-refinement get 21 best heuristic solutions that coincide with the best known values while RBH and HCS find 28 and 30 best solutions corrisponding to best known values, respectively. To conclude, HCS performance is very good and RBHis an easy and fast matheuristic able to find promising solutions in some sets of instances even better than those obtained by more complicated meta-heuristic algorithms as the tabu search used as benchmark.

	Instance Details			Basic		New		New + Aux	
Instances	$\|V\|$	$\|R\|$	Opt.	Time	# cuts	Time	# cuts	Time	# cuts
DPRPP_1.0_2.0_val1A	24	39	195	0.02	2	0.03	1	0.03	1
DPRPP_1.0_2.0_val2A	24	34	265	0.05	18	0.03	1	0.05	1
DPRPP_1.0_2.0_val3A	24	35	88	0.03	1	0.02	0	0.05	0
DPRPP_1.0_2.0_val4A	41	69	426	0.03	3	0.03	1	0.05	1
DPRPP_1.0_2.0_val5A	34	65	472	0.03	5	0.08	4	0.03	3
DPRPP_1.0_2.0_val6A	31	50	253	0.02	3	0.03	3	0.02	3
DPRPP_1.0_2.0_val7A	40	66	340	0.02	5	0.05	5	0.03	5
DPRPP_1.0_2.0_val8A	30	63	423	0.03	0	0.02	0	0.05	0
DPRPP_1.0_2.0_val9A	50	92	342	0.02	0	0.03	0	0.02	0
DPRPP_1.0_2.0_val10A	50	97	446	0.05	4	0.05	0	0.03	0
DPRPP_1.5_2.5_val1A	24	39	216	0.03	2	0.02	2	0.02	2
DPRPP_1.5_2.5_val2A	24	34	291	0.00	1	0.02	1	0.02	1
DPRPP_1.5_2.5_val3A	24	35	101	0.02	0	0.02	0	0.02	0
DPRPP_1.5_2.5_val4A	41	69	463	0.02	1	0.02	0	0.02	0
DPRPP_1.5_2.5_val5A	34	65	517	0.02	0	0.03	0	0.03	0
DPRPP_1.5_2.5_val6A	31	50	275	0.03	0	0.03	0	0.02	0
DPRPP_1.5_2.5_val7A	40	66	374	0.02	0	0.03	0	0.03	0
DPRPP_1.5_2.5_val8A	30	63	461	0.00	0	0.02	0	0.00	0
DPRPP_1.5_2.5_val9A	50	92	368	0.03	0	0.05	0	0.03	0
DPRPP_1.5_2.5_val10A	50	97	485	0.03	0	0.03	0	0.03	0
DPRPP_2.0_3.0_val1A	24	39	221	0.02	0	0.02	0	0.02	0
DPRPP_2.0_3.0_val2A	24	34	298	0.02	0	0.00	0	0.00	0
DPRPP_2.0_3.0_val3A	24	35	105	0.02	0	0.06	0	0.03	0
DPRPP_2.0_3.0_val4A	41	69	476	0.03	0	0.03	0	0.03	0
DPRPP_2.0_3.0_val5A	34	65	528	0.03	0	0.03	0	0.02	0
DPRPP_2.0_3.0_val6A	31	50	281	0.02	0	0.05	0	0.03	0
DPRPP_2.0_3.0_val7A	40	66	383	0.03	0	0.19	0	0.03	0
DPRPP_2.0_3.0_val8A	30	63	471	0.02	0	0.03	0	0.02	0
DPRPP_2.0_3.0_val9A	50	92	373	0.00	0	0.02	0	0.02	0
DPRPP_2.0_3.0_val10A	50	97	492	0.02	0	0.03	0	0.14	0
Average				0.02	1.50	0.03	0.60	0.03	0.57

Table 1: Variants comparison: exact results on the instances of Set 1.

Instances	Instance Details		Opt.	Basic		New		New + Aux	
	$\|V\|$	$\|R\|$		Time	# cuts	Time	# cuts	Time	# cuts
DPRPP_1.0_2.0_egl-e1-A	77	51	**2183**	28.91	862	0.56	65	0.72	88
DPRPP_1.0_2.0_egl-e2-A	77	72	2616	488.55	3403	10.30	670	8.38	408
DPRPP_1.0_2.0_egl-e3-A	77	87	3026	12.26	637	0.33	27	0.27	26
DPRPP_1.0_2.0_egl-e4-A	77	98	3413	87.80	1438	3.07	196	2.54	212
DPRPP_1.0_2.0_egl-s1-A	140	75	2125	65.24	1633	25.85	738	21.06	741
DPRPP_1.0_2.0_egl-s2-A *	140	147	*4755*	1732.06 (4804, 3.50%)	5563	2607.86 (4835, 3.75%)	5558	2152.94 (4755, 2.17%)	4880
DPRPP_1.0_2.0_egl-s3-A	140	159	**4853**	135.38	1352	51.39	904	52.71	491
DPRPP_1.0_2.0_egl-s4-A	140	190	5772	6.10	197	2.14	106	1.17	65
DPRPP_1.5_2.5_egl-e1-A	77	51	2494	0.50	59	0.05	5	0.06	5
DPRPP_1.5_2.5_egl-e2-A	77	72	3016	1.45	166	0.08	7	0.08	7
DPRPP_1.5_2.5_egl-e3-A	77	87	3518	0.50	56	0.09	5	0.06	5
DPRPP_1.5_2.5_egl-e4-A	77	98	3817	0.61	61	0.34	25	0.27	25
DPRPP_1.5_2.5_egl-s1-A *	140	75	2596	648.56 (2598, 1.02%)	4591	804.03 (2596, 0.50%)	4690	616.45 (2596, 0.90%)	4547
DPRPP_1.5_2.5_egl-s2-A	140	147	**5434**	43.79	861	61.11	843	93.30	426
DPRPP_1.5_2.5_egl-s3-A	140	159	**5644**	58.44	211	8.25	132	54.96	89
DPRPP_1.5_2.5_egl-s4-A	140	190	6492	0.11	5	0.17	5	4.40	5
DPRPP_2.0_3.0_egl-e1-A	77	51	2622	0.03	1	0.06	1	0.06	1
DPRPP_2.0_3.0_egl-e2-A	77	72	3109	0.03	1	0.05	1	0.06	1
DPRPP_2.0_3.0_egl-e3-A	77	87	3659	0.03	0	0.05	0	0.03	0
DPRPP_2.0_3.0_egl-e4-A	77	98	3931	0.03	0	0.05	0	0.05	0
DPRPP_2.0_3.0_egl-s1-A	140	75	**2688**	13.03	325	16.93	430	12.42	321
DPRPP_2.0_3.0_egl-s2-A	140	147	**5662**	1.05	34	1.53	34	1.23	35
DPRPP_2.0_3.0_egl-s3-A	140	159	5838	0.30	8	0.36	9	0.30	9
DPRPP_2.0_3.0_egl-s4-A	140	190	6721	0.06	0	0.08	0	0.14	0
Average				138.53	894.33	149.78	602.13	125.99	516.13

Table 2: Variants comparison: exact results on the instances of Set 2.

	Instance Details			Basic		New		New + Aux	
Instances	$\|V\|$	$\|R\|$	Opt.	Time	# cuts	Time	# cuts	Time	# cuts
DPRPP_ALBAIDAANoRPP	102	159	14332	4.20	196	18.05	459	54.48	830
DPRPP_ALBAIDABNoRPP	90	144	12599	0.62	47	1.62	89	1.11	67
Average				2.41	121.50	9.84	274.00	27.79	448.50
DPRPP_P01NoRPP	11	11	60	0.03	5	0.02	5	0.03	5
DPRPP_P02NoRPP	14	30	306	0.02	5	0.02	6	0.02	6
DPRPP_P03NoRPP	28	48	160	0.06	11	0.09	12	0.06	11
DPRPP_P04NoRPP	17	30	123	0.02	4	0.03	4	0.06	4
DPRPP_P05NoRPP	20	31	222	0.02	5	0.02	5	0.02	5
DPRPP_P06NoRPP	24	38	155	0.05	10	0.05	7	0.03	7
DPRPP_P07NoRPP	23	44	221	0.05	13	0.06	13	0.08	13
DPRPP_P08NoRPP	17	36	178	0.00	1	0.05	0	0.00	0
DPRPP_P09NoRPP	14	23	105	0.02	0	0.03	0	0.03	0
DPRPP_P10NoRPP	12	18	103	0.02	3	0.00	3	0.02	3
DPRPP_P11NoRPP	9	10	31	0.00	4	0.02	3	0.02	3
DPRPP_P12NoRPP	7	11	24	0.00	0	0.03	0	0.02	0
DPRPP_P13NoRPP	7	9	41	0.02	1	0.00	1	0.02	1
DPRPP_P14NoRPP	28	72	465	0.02	4	0.03	4	0.02	4
DPRPP_P15NoRPP	26	36	512	0.02	12	0.03	12	0.03	12
DPRPP_P16NoRPP	31	89	434	0.02	6	0.05	6	0.05	6
DPRPP_P17NoRPP	19	40	234	0.02	5	0.02	4	0.02	4
DPRPP_P18NoRPP	23	33	200	0.11	37	0.08	24	0.05	19
DPRPP_P19NoRPP	33	48	344	0.03	7	0.03	7	0.03	6
DPRPP_P20NoRPP	50	95	745	0.05	10	0.06	10	0.03	10
DPRPP_P21NoRPP	49	103	620	0.27	21	0.28	28	1.14	28
DPRPP_P22NoRPP	50	178	1692	0.05	0	0.25	0	0.05	0
DPRPP_P23NoRPP	50	147	913	0.20	7	0.17	8	0.14	8
DPRPP_P24NoRPP	41	119	945	0.08	11	0.09	11	0.09	11
Average				0.05	7.58	0.06	7.21	0.08	6.92

Table 3: Variants comparison: exact results on the instances of Set 3.

	Instance Details			Basic		New		New + Aux	
Instances	$\|V\|$	$\|R\|$	Opt.	Time	# cuts	Time	# cuts	Time	# cuts
DPRPP_D0NoRPP	16	30	1191	0.05	0	0.05	0	0.05	0
DPRPP_D1NoRPP	16	31	1348	0.05	7	0.03	1	0.03	1
DPRPP_D2NoRPP	16	31	1312	0.03	9	0.05	9	0.05	9
DPRPP_D3NoRPP	16	30	1398	0.05	6	0.06	4	0.03	4
DPRPP_D4NoRPP	16	31	1595	0.05	12	0.05	4	0.03	5
DPRPP_D5NoRPP	16	31	1329	0.02	6	0.03	6	0.02	6
DPRPP_D6NoRPP	16	31	1517	0.08	13	0.11	16	0.09	12
DPRPP_D7NoRPP	16	31	1592	0.06	25	0.03	10	0.03	10
DPRPP_D8NoRPP	16	31	1650	0.03	7	0.05	7	0.03	7
DPRPP_D9NoRPP	36	69	1889	0.16	34	0.09	14	0.11	16
DPRPP_D10NoRPP	36	69	1765	0.03	9	0.03	8	0.03	8
DPRPP_D11NoRPP	36	69	2375	0.16	27	0.06	5	0.03	5
DPRPP_D12NoRPP	36	70	2172	0.05	10	0.06	13	0.05	13
DPRPP_D13NoRPP	36	69	2391	0.95	86	0.30	33	0.27	31
DPRPP_D14NoRPP	36	71	2536	0.02	2	0.05	2	0.05	2
DPRPP_D15NoRPP	36	71	2506	0.11	10	0.41	10	0.06	10
DPRPP_D16NoRPP	36	72	2458	0.03	4	0.05	1	0.03	1
DPRPP_D17NoRPP	36	71	2754	0.06	8	0.20	7	0.06	7
DPRPP_D18NoRPP	64	126	2520	4.60	186	4.95	141	3.95	104
DPRPP_D19NoRPP	64	122	2522	0.67	68	0.64	44	0.44	32
DPRPP_D20NoRPP	64	120	**2618**	40.54	391	5.09	127	5.26	133
DPRPP_D21NoRPP	64	125	2898	0.87	61	1.56	91	1.11	80
DPRPP_D22NoRPP	64	125	3045	0.87	77	1.53	81	16.02	91
DPRPP_D23NoRPP	64	125	2892	0.44	33	0.61	30	0.45	31
DPRPP_D24NoRPP	64	126	3341	0.08	13	0.12	13	0.09	13
DPRPP_D25NoRPP	64	125	3317	0.11	17	0.34	25	0.34	25
DPRPP_D26NoRPP	64	123	3772	0.08	8	0.09	8	0.08	8
DPRPP_D27NoRPP	100	190	**3388**	133.88	293	6.18	119	8.50	151
DPRPP_D28NoRPP	100	192	**3896**	74.19	519	14.62	345	21.20	424
DPRPP_D29NoRPP	100	188	**3265**	516.28	1853	58.78	758	147.67	930
DPRPP_D30NoRPP	100	196	4037	13.96	189	18.46	284	21.48	241
DPRPP_D31NoRPP	100	195	4175	3.12	96	0.62	37	0.48	22
DPRPP_D32NoRPP	100	193	3857	4.12	77	2.03	71	1.95	55
DPRPP_D33NoRPP	100	194	4578	0.16	6	0.16	6	0.17	6
DPRPP_D34NoRPP	100	195	4753	0.45	23	0.11	4	0.12	4
DPRPP_D35NoRPP	100	193	4437	0.14	8	0.16	9	0.19	9
Average				22.13	116.47	3.27	65.08	6.40	69.61

Table 4: Variants comparison: exact results on the instances of Set 4.

Instances	Instance Details			Basic		New		New + Aux	
	$\lvert V\rvert$	$\lvert R\rvert$	Opt.	Time	# cuts	Time	# cuts	Time	# cuts
DPRPP_G0NoRPP	16	3	6	0.03	1	0.02	1	0.02	0
DPRPP_G1NoRPP	16	5	11	0.20	2	0.03	2	0.02	0
DPRPP_G2NoRPP	16	4	9	0.03	3	0.03	1	0.03	0
DPRPP_G3NoRPP	16	8	15	0.02	1	0.02	1	0.02	1
DPRPP_G4NoRPP	16	7	16	0.03	10	0.00	3	0.00	3
DPRPP_G5NoRPP	16	7	16	0.03	10	0.00	3	0.00	2
DPRPP_G6NoRPP	16	13	23	0.02	6	0.02	7	0.02	7
DPRPP_G7NoRPP	16	8	17	0.02	7	0.20	6	0.03	3
DPRPP_G8NoRPP	16	9	19	0.03	12	0.02	2	0.02	2
DPRPP_G9NoRPP	36	11	23	0.05	8	0.03	6	0.03	4
DPRPP_G10NoRPP	36	13	26	0.06	11	0.05	4	0.03	4
DPRPP_G11NoRPP	36	15	29	0.17	32	0.22	65	0.28	76
DPRPP_G12NoRPP	36	26	46	0.16	37	0.03	23	0.05	23
DPRPP_G13NoRPP	36	23	44	1.25	159	1.31	222	1.48	205
DPRPP_G14NoRPP	36	25	45	1.25	172	0.94	158	0.94	127
DPRPP_G15NoRPP	36	35	51	0.03	7	0.08	11	0.08	11
DPRPP_G16NoRPP	36	30	51	1.29	200	0.81	136	0.73	133
DPRPP_G17NoRPP	36	34	54	0.03	8	0.05	7	0.05	7
DPRPP_G18NoRPP	64	24	**51**	1.03	98	0.20	26	0.20	29
DPRPP_G19NoRPP	64	27	53	0.58	40	0.87	77	0.64	48
DPRPP_G20NoRPP	64	27	**54**	3.45	184	0.31	35	0.51	46
DPRPP_G21NoRPP	64	46	75	0.33	31	0.05	10	0.06	10
DPRPP_G22NoRPP	64	47	76	6.07	456	0.12	11	0.25	9
DPRPP_G23NoRPP	64	50	82	0.09	23	0.08	18	0.08	17
DPRPP_G24NoRPP	64	68	110	0.16	36	0.27	33	0.20	30
DPRPP_G25NoRPP	64	61	94	0.69	122	0.76	89	0.30	48
DPRPP_G26NoRPP	64	66	101	0.75	67	0.06	14	0.06	14
DPRPP_G27NoRPP	100	41	**82**	734.73	1696	211.47	1425	236.47	1440
DPRPP_G28NoRPP	100	49	**94**	(94, 2.57%)	6031	133.63	2178	71.82	923
DPRPP_G29NoRPP	100	44	89	2828.43 (91, 5.48%)	8657	3539.13 (90, 4.07%)	8462	(89, 3.16%)	7371
DPRPP_G30NoRPP	100	73	**122**	17.91	888	6.36	335	23.24	434
DPRPP_G31NoRPP	100	77	**131**	3.28	208	6.44	283	7.58	172
DPRPP_G32NoRPP	100	82	**139**	1.03	78	2.48	156	12.11	113
DPRPP_G33NoRPP	100	113	168	1.28	88	2.87	107	4.98	107
DPRPP_G34NoRPP	100	107	162	0.19	19	0.17	18	0.17	18
DPRPP_G35NoRPP	100	109	164	0.41	25	0.44	31	0.73	38
Average				200.14	539.81	108.60	387.94	110.09	318.75

Table 5: Variants comparison: exact results on the instances of Set 5.

	Instance Details			Basic		New		New + Aus	
Instances	$\|V\|$	$\|R\|$	Opt.	Time	# cuts	Time	# cuts	Time	# cuts
DPRPP_R0NoRPP	20	37	45900	0.08	19	0.02	4	0.02	4
DPRPP_R1NoRPP	20	47	69694	0.05	19	0.03	11	0.03	8
DPRPP_R2NoRPP	20	47	66712	0.09	30	0.05	15	0.08	14
DPRPP_R3NoRPP	20	75	140259	0.12	12	0.08	10	0.09	10
DPRPP_R4NoRPP	20	60	86507	0.05	10	0.03	0	0.02	0
DPRPP_R5NoRPP	30	70	72779	0.14	19	0.03	9	0.05	9
DPRPP_R6NoRPP	30	110	134780	0.19	23	0.39	42	0.27	27
DPRPP_R7NoRPP	30	70	83653	0.37	47	0.20	29	0.27	35
DPRPP_R8NoRPP	30	111	143229	0.81	42	0.42	32	0.48	31
DPRPP_R9NoRPP	30	111	138809	0.22	26	0.03	1	0.02	1
DPRPP_R10NoRPP	40	130	119540	1.59	80	0.17	13	0.20	13
DPRPP_R11NoRPP	40	103	93686	0.23	15	0.25	20	0.44	35
DPRPP_R12NoRPP	40	82	**79132**	8.02	127	1.44	110	3.14	121
DPRPP_R13NoRPP	40	203	266344	0.64	22	0.39	20	0.48	20
DPRPP_R14NoRPP	40	203	258759	0.25	11	0.12	7	0.17	7
DPRPP_R15NoRPP	50	203	212498	26.49	176	0.48	29	1.03	37
DPRPP_R16NoRPP	50	162	140184	2.40	73	1.51	78	2.18	79
DPRPP_R17NoRPP	50	130	**109525**	12.48	367	6.10	252	13.03	246
DPRPP_R18NoRPP	50	203	206141	2.87	55	2.11	72	2.50	58
DPRPP_R19NoRPP	50	203	203643	0.06	2	0.09	2	0.09	2
Average				2.86	58.75	0.70	37.80	1.23	37.85

Table 6: Variants comparison: exact results on the instances of Set 6.

		Tabu		ILP-ref.		RBH			HCS (α=0)				HCS (α=0.05)			
Instances	Best Known	Gap (%)	Time	Gap (%)	Time	Gap (%)	Time	Routine	Gap (%)	Time	# cuts	# Aux. Pr.	Gap (%)	Time	# cuts	# Aux. Pr.
1.0_2.0_val1A	195.00	0.00	4.63	0.00	5.55	0.00	0.09	ISA	0.00	0.14	0	0 (0)	0.00	0.12	0	0 (0)
1.0_2.0_val2A	265.00	0.00	4.48	0.00	17.67	0.00	0.08	IIA	0.00	0.11	1	0 (0)	0.00	0.08	1	0 (0)
1.0_2.0_val3A	88.00	0.00	4.64	0.00	5.16	0.00	0.08	ISA	0.00	0.09	0	0 (0)	0.00	0.09	0	0 (0)
1.0_2.0_val4A	426.00	0.47	12.89	0.47	14.03	0.00	0.35	ISA	0.00	0.45	0	0 (0)	0.00	0.41	0	0 (0)
1.0_2.0_val5A	472.00	0.42	11.45	0.42	12.09	0.21	0.24	ISA (γ)	0.00	0.34	5	0 (0)	0.00	0.28	5	0 (0)
1.0_2.0_val6A	253.00	0.00	7.67	0.00	8.20	0.39	0.17	IDA	0.00	0.23	2	1 (1)	0.00	0.23	1	1 (1)
1.0_2.0_val7A	340.00	0.29	11.67	0.29	14.20	0.00	0.46	ISA (γ)	0.00	0.56	0	0 (0)	0.00	0.50	0	0 (0)
1.0_2.0_val8A	423.00	0.00	13.83	0.00	14.30	0.00	0.15	ISA	0.00	0.19	0	0 (0)	0.00	0.16	0	0 (0)
1.0_2.0_val9A	342.00	0.29	19.17	0.29	19.86	0.00	0.55	ISA	0.00	0.76	0	0 (0)	0.00	0.66	0	0 (0)
1.0_2.0_val10A	446.00	0.22	22.97	0.22	23.60	0.00	0.61	ISA	0.00	0.94	0	1 (1)	0.00	0.83	0	1 (1)
1.5_2.5_val1A	216.00	0.00	7.67	0.00	9.22	0.00	0.12	ISA	0.00	0.12	0	0 (0)	0.00	0.12	0	0 (0)
1.5_2.5_val2A	291.00	0.00	4.11	0.00	4.47	0.00	0.09	ISA	0.00	0.14	0	0 (0)	0.00	0.11	0	0 (0)
1.5_2.5_val3A	101.00	0.00	3.78	0.00	4.17	0.00	0.08	ISA	0.00	0.11	0	0 (0)	0.00	0.09	0	0 (0)
1.5_2.5_val4A	463.00	0.22	14.39	0.22	15.84	0.00	0.35	ISA	0.00	0.45	0	0 (0)	0.00	0.39	0	0 (0)
1.5_2.5_val5A	517.00	0.19	11.33	0.19	11.77	0.00	0.23	ISA	0.00	0.28	0	0 (0)	0.00	0.27	0	0 (0)
1.5_2.5_val6A	275.00	0.00	7.06	0.00	7.45	0.00	0.15	ISA	0.00	0.22	0	0 (0)	0.00	0.19	0	0 (0)
1.5_2.5_val7A	374.00	0.00	18.34	0.00	55.86	0.00	0.40	ISA	0.00	0.41	0	0 (0)	0.00	0.45	0	0 (0)
1.5_2.5_val8A	461.00	0.00	16.30	0.00	17.19	0.00	0.24	ISA	0.00	0.26	0	0 (0)	0.00	0.27	0	0 (0)
1.5_2.5_val9A	368.00	0.00	18.84	0.00	19.73	0.00	0.83	ISA	0.00	0.86	0	0 (0)	0.00	1.00	0	0 (0)
1.5_2.5_val10A	485.00	0.21	36.64	0.21	38.31	0.00	0.91	ISA	0.00	0.93	0	0 (0)	0.00	1.11	0	0 (0)
2.0_3.0_val1A	221.00	0.00	7.31	0.00	8.00	0.00	0.13	ISA	0.00	0.14	0	0 (0)	0.00	0.14	0	0 (0)
2.0_3.0_val2A	298.00	0.00	6.13	0.00	6.72	0.00	0.12	ISA	0.00	0.12	0	0 (0)	0.00	0.16	0	0 (0)
2.0_3.0_val3A	105.00	0.00	6.39	0.00	7.03	0.00	0.13	ISA	0.00	0.14	0	0 (0)	0.00	0.16	0	0 (0)
2.0_3.0_val4A	476.00	0.00	17.05	0.00	20.27	0.00	0.47	ISA	0.00	0.48	0	0 (0)	0.00	0.56	0	0 (0)
2.0_3.0_val5A	528.00	0.00	18.05	0.00	18.61	0.00	0.29	ISA	0.00	0.30	0	0 (0)	0.00	0.37	0	0 (0)
2.0_3.0_val6A	281.00	0.00	10.38	0.00	11.11	0.00	0.24	ISA	0.00	0.25	0	0 (0)	0.00	0.28	0	0 (0)
2.0_3.0_val7A	383.00	0.00	16.61	0.00	17.33	0.00	0.46	ISA	0.00	0.47	0	0 (0)	0.00	0.55	0	0 (0)
2.0_3.0_val8A	471.00	0.00	15.52	0.00	18.10	0.00	0.24	ISA	0.00	0.25	0	0 (0)	0.00	0.30	0	0 (0)
2.0_3.0_val9A	373.00	0.00	28.19	0.00	28.82	0.00	0.93	ISA	0.00	0.97	0	0 (0)	0.00	1.11	0	0 (0)
2.0_3.0_val10A	492.00	0.20	31.05	0.20	35.27	0.00	0.90	ISA	0.00	1.05	0	1 (1)	0.00	1.20	0	1 (1)
Average		0.08	13.62	0.08	16.33	0.02	0.34		0.00	0.39	0.27		0.00	0.41	0.23	

Table 7: Heuristic results on instances of Set 1.

		Tabu		ILP-ref.		RBH			HCS (α=0)				HCS (α=0.05)			
Instances	Best Known	Gap (%)	Time	Gap (%)	Time	Gap (%)	Time	Routine	Gap (%)	Time	# cuts	# Aux. Pr.	Gap (%)	Time	# cuts	# Aux. Pr.
1.0_2.0_egl-e1	2183.00	0.00	5.75	0.00	6.27	0.00	0.43	ISA	0.00	0.56	37	0 (0)	0.00	0.89	42	0 (0)
1.0_2.0_egl-e2	2616.00	0.00	10.64	0.00	11.05	0.53	0.90	ISA	0.53	0.95	20	0 (0)	0.00	12.07	1177	0 (0)
1.0_2.0_egl-e3	3026.00	0.07	16.64	0.07	20.92	0.00	1.46	ISA	0.00	1.90	9	0 (0)	0.00	1.59	10	0 (0)
1.0_2.0_egl-e4	3413.00	0.20	18.70	0.20	64.81	0.96	8.51	ISA	0.23	11.15	107	2 (2)	0.00	10.33	188	2 (2)
1.0_2.0_egl-s1	2125.00	0.00	9.64	0.00	10.03	0.42	1.51	IDA	0.00	1.64	20	0 (0)	0.00	49.05	1690	0 (0)
1.0_2.0_egl-s2	4755.00 *	0.34	61.88	0.34	324.36	1.74	174.72	IDA	1.80	231.04	131	5 (0)	1.65	224.72	136	11 (0)
1.0_2.0_egl-s3	4853.00	1.58	71.27	1.58	404.43	1.36	134.73	IIA	1.36	157.83	329	0 (0)	1.36	184.73	103	15 (0)
1.0_2.0_egl-s4	5772.00	1.20	109.94	1.20	110.64	0.53	376.75	IIA (γ)	0.53	384.35	53	0 (0)	0.00	378.46	22	0 (0)
1.5_2.5_egl-e1	2494.00	0.00	6.72	0.00	7.30	0.24	1.45	IDA (γ)	0.16	1.73	5	0 (0)	0.00	1.53	1	0 (0)
1.5_2.5_egl-e2	3016.00	0.00	12.64	0.00	13.14	0.13	1.62	ISA	0.10	2.54	8	0 (0)	0.00	1.81	0	0 (0)
1.5_2.5_egl-e3	3518.00	0.00	17.06	0.00	18.51	0.17	2.26	ISA	0.00	3.87	5	0 (0)	0.00	2.50	0	0 (0)
1.5_2.5_egl-e4	3817.00	0.00	19.92	0.00	21.06	0.00	3.65	ISA	0.00	5.40	18	0 (0)	0.00	4.04	17	0 (0)
1.5_2.5_egl-s1	2596.00 *	0.00	13.39	0.00	14.45	2.48	65.64	IIA (γ)	2.48	66.84	19	0 (0)	1.59	115.64	1204	60 (1)
1.5_2.5_egl-s2	5434.00	2.74	43.47	2.74	44.20	1.52	177.56	IIA (γ)	1.40	256.75	104	5 (0)	1.40	227.56	18	5 (0)
1.5_2.5_egl-s3	5644.00	0.55	57.81	0.55	61.26	0.69	168.37	IDA (γ)	0.09	210.33	83	1 (0)	0.69	218.37	51	5 (0)
1.5_2.5_egl-s4	6492.00	0.03	105.69	0.03	106.42	0.03	50.03	ISA	0.00	62.64	5	0 (0)	0.00	53.32	1	0 (0)
2.0_3.0_egl-e1	2622.00	0.00	9.94	0.00	10.82	0.00	0.71	ISA	0.00	0.84	1	0 (0)	0.00	0.86	0	0 (0)
2.0_3.0_egl-e2	3109.00	0.00	17.58	0.00	18.78	0.00	1.24	ISA	0.00	1.69	1	0 (0)	0.00	1.54	0	0 (0)
2.0_3.0_egl-e3	3659.00	0.00	22.52	0.00	24.46	0.00	2.81	ISA	0.00	3.82	0	0 (0)	0.00	3.43	0	0 (0)
2.0_3.0_egl-e4	3931.00	0.00	28.97	0.00	29.89	0.00	2.85	ISA	0.00	4.63	0	0 (0)	0.00	3.39	0	0 (0)
2.0_3.0_egl-s1	2688.00	0.00	12.14	0.00	12.67	0.00	42.58	IIA	0.00	43.25	7	0 (0)	0.00	49.58	249	0 (0)
2.0_3.0_egl-s2	5662.00	0.00	40.95	0.00	41.37	0.19	114.25	IDA	0.00	176.89	30	0 (0)	0.00	125.47	19	0 (0)
2.0_3.0_egl-s3	5838.00	0.00	127.02	0.00	129.90	1.20	82.33	IDA	0.00	103.85	8	0 (0)	0.00	96.50	5	1 (1)
2.0_3.0_egl-s4	6721.00	0.00	176.20	0.00	178.58	0.00	28.74	ISA	0.00	48.03	0	0 (0)	0.00	33.57	0	0 (0)
Average		0.28	42.35	0.28	70.22	0.51	60.21		0.36	74.27	41.67		0.28	75.04	205.54	

Table 8: Heuristic results on instances of Set 2.

		Tabu		ILP-ref.		RBH			HCS (α=0)				HCS (α=0.05)			
Instances	Best Known	Gap (%)	Time	Gap (%)	Time	Gap (%)	Time	Routine	Gap (%)	Time	# cuts	# Aux. Pr.	Gap (%)	Time	# cuts	# Aux. Pr.
ALBAIDAA	14332.00	2.35	58.45	1.82	59.15	0.16	34.07	IDA	0.16	35.35	12	0 (0)	0.00	54.60	1142	0 (0)
ALBAIDAB	12599.00	0.34	44.19	0.25	44.83	2.04	31.76	IDA	0.90	38.08	53	0 (0)	0.00	33.03	80	0 (0)
Average		1.35	51.32	1.03	51.99	1.10	32.91		0.53	36.72	32.50		0.00	43.81	611.00	
P01	60.00	0.00	2.69	0.00	3.08	0.00	0.04	IDA	0.00	0.06	3	0 (0)	0.00	0.17	3	0 (0)
P02	306.00	0.00	6.44	0.00	6.78	3.16	0.08	ISA (γ)	0.00	0.08	3	0 (0)	0.00	0.08	3	0 (0)
P03	160.00	0.00	6.27	0.00	6.77	3.03	0.42	IDA	0.00	0.51	9	0 (0)	0.00	0.45	11	0 (0)
P04	123.00	0.00	3.88	0.00	4.35	2.38	0.11	ISA (γ)	0.00	0.13	4	0 (0)	0.00	0.11	4	0 (0)
P05	222.00	0.00	3.86	0.00	4.52	0.00	0.11	IDA	0.00	0.12	9	0 (0)	0.00	0.10	1	0 (0)
P06	155.00	0.00	4.05	0.00	4.60	0.64	0.14	ISA	0.00	0.17	7	0 (0)	0.00	0.14	7	0 (0)
P07	221.00	0.00	4.63	0.00	5.07	1.34	0.09	IIA	0.00	0.11	11	0 (0)	0.00	0.09	10	0 (0)
P08	178.00	0.00	4.75	0.00	5.20	2.73	0.07	IIA	0.00	0.09	0	0 (0)	0.00	0.08	0	0 (0)
P09	105.00	0.00	2.67	0.00	3.14	0.00	0.04	ISA	0.00	0.05	0	0 (0)	0.00	0.05	0	0 (0)
P10	103.00	0.00	2.45	0.00	2.83	0.00	0.04	ISA	0.00	0.05	3	0 (0)	0.00	0.05	3	0 (0)
P11	31.00	0.00	2.31	0.00	2.67	8.82	0.04	ISA	0.00	0.05	3	0 (0)	0.00	0.04	3	0 (0)
P12	24.00	0.00	2.22	0.00	2.69	0.00	0.02	ISA	0.00	0.03	0	0 (0)	0.00	0.03	0	0 (0)
P13	41.00	0.00	2.27	0.00	2.69	0.00	0.02	ISA	0.00	0.03	1	0 (0)	0.00	0.03	1	0 (0)
P14	465.00	1.27	10.97	1.27	11.47	3.93	1.20	IDA (γ)	0.00	1.42	4	0 (0)	0.00	1.22	4	0 (0)
P15	512.00	0.19	3.42	0.19	4.03	0.00	0.09	IDA	0.00	0.12	14	0 (0)	0.00	0.11	2	0 (0)
P16	434.00	1.59	13.77	1.36	14.22	1.59	0.61	ISA (γ)	0.00	0.76	6	0 (0)	0.00	1.45	6	0 (0)
P17	234.00	0.85	4.45	0.85	5.06	0.00	0.07	ISA (γ)	0.00	0.08	3	0 (0)	0.00	0.11	2	0 (0)
P18	200.00	0.00	4.80	0.00	5.16	3.38	0.11	IDA (γ)	1.96	0.12	5	0 (0)	1.96	0.16	17	0 (0)
P19	344.00	0.00	4.73	0.00	5.67	0.00	0.34	ISA	0.00	0.44	7	0 (0)	0.00	0.37	3	0 (0)
P20	745.00	0.13	17.67	0.13	27.48	3.62	13.82	IDA	0.13	15.96	9	0 (0)	0.00	14.09	6	1 (1)
P21	620.00	0.80	36.28	0.80	36.91	1.27	4.40	IDA	0.00	5.21	18	0 (0)	0.00	4.59	17	0 (0)
P22	1692.00	0.65	112.34	0.65	117.58	0.00	0.69	ISA	0.00	0.97	0	0 (0)	0.00	0.84	0	0 (0)
P23	913.00	0.65	88.41	0.65	88.86	0.98	3.04	ISA (γ)	0.00	3.70	7	0 (0)	0.00	3.18	7	0 (0)
P24	945.00	1.97	30.78	1.97	41.08	2.48	3.10	ISA (γ)	0.00	3.81	11	0 (0)	0.00	3.18	11	0 (0)
Average		0.42	18.41	0.38	19.84	1.60	3.63		0.12	4.13	7.77		0.08	4.55	51.65	

Table 9: Heuristic results on instances of Set 3.

		Tabu		ILP-ref.		RBH			HCS (α=0)				HCS (α=0.05)			
Instances	Best Known	Gap (%)	Time	Gap (%)	Time	Gap (%)	Time	Routine	Gap (%)	Time	# cuts	# Aux. Pr.	Gap (%)	Time	# cuts	# Aux. Pr.
D0	1191.00	0.00	2.95	0.00	3.42	0.00	0.04	ISA	0.00	0.06	0	0 (0)	0.00	0.06	0	0 (0)
D1	1348.00	0.00	3.05	0.00	3.41	0.00	0.04	ISA	0.00	0.05	1	0 (0)	0.00	0.05	0	0 (0)
D2	1312.00	0.00	3.44	0.00	3.88	0.00	0.09	IDA	0.00	0.09	6	0 (0)	0.00	0.09	4	0 (0)
D3	1398.00	0.00	4.25	0.00	4.63	1.89	0.04	IDA	1.89	0.05	7	0 (0)	1.89	0.05	7	0 (0)
D4	1595.00	0.00	3.50	0.00	3.89	1.66	0.06	IDA	0.00	0.08	4	0 (0)	0.00	0.08	3	0 (0)
D5	1329.00	0.00	3.25	0.00	3.73	0.00	0.07	IDA	0.00	0.08	7	0 (0)	0.00	0.07	7	0 (0)
D6	1517.00	0.00	3.08	0.00	3.44	0.00	0.17	IDA	0.00	0.19	8	0 (0)	0.00	0.17	8	0 (0)
D7	1592.00	0.00	3.24	0.00	4.02	2.51	0.08	IDA (γ)	0.00	0.11	11	0 (0)	0.00	0.09	8	0 (0)
D8	1650.00	0.00	3.17	0.00	3.58	0.00	0.06	IDA	0.00	0.08	7	0 (0)	0.00	0.06	5	0 (0)
D9	1889.00	0.26	7.03	0.26	7.45	0.11	0.25	IDA	0.11	0.31	4	0 (0)	0.11	0.28	7	0 (0)
D10	1765.00	0.00	6.63	0.00	7.02	0.73	0.15	IDA	0.00	0.20	5	0 (0)	0.00	0.17	4	0 (0)
D11	2375.00	0.00	16.41	0.00	16.88	2.06	0.20	IIA (γ)	0.63	0.27	6	0 (0)	0.63	0.22	4	0 (0)
D12	2172.00	0.00	7.72	0.00	8.06	1.63	0.62	IIA	1.63	0.73	12	0 (0)	1.63	0.62	9	0 (0)
D13	2391.00	0.00	9.75	0.00	10.13	0.95	0.18	IDA	0.00	0.34	25	0 (0)	0.00	0.28	25	0 (0)
D14	2536.00	0.00	11.91	0.00	12.38	0.00	0.26	ISA	0.00	0.34	5	1 (1)	0.00	0.30	1	1 (1)
D15	2506.00	2.26	10.25	0.24	10.89	0.08	3.49	IDA	0.08	4.04	4	0 (0)	0.08	3.51	3	0 (0)
D16	2458.00	0.04	10.23	0.04	10.75	0.24	0.24	IIA	0.00	0.31	1	0 (0)	0.00	0.27	1	0 (0)
D17	2754.00	0.33	11.70	0.33	12.08	0.83	0.70	IDA	0.33	0.86	7	0 (0)	0.33	0.76	7	0 (0)
D18	2520.00	1.64	23.81	1.64	27.12	0.94	2.48	IIA (γ)	0.40	3.23	24	1 (1)	0.04	3.43	59	3 (1)
D19	2522.00	1.91	23.34	0.86	24.95	0.00	4.33	IIA (γ)	0.00	5.15	16	0 (0)	0.00	4.54	22	0 (0)
D20	2618.00	1.36	25.94	1.36	27.33	2.39	4.99	IIA (γ)	1.36	7.99	246	1 (0)	1.13	7.60	306	1 (1)
D21	2898.00	0.65	31.88	0.65	32.55	0.24	7.17	IDA	0.14	8.97	65	0 (0)	0.14	7.88	64	0 (0)
D22	3045.00	1.23	31.61	1.23	32.25	3.39	39.50	ISA	0.13	44.04	60	4 (0)	0.00	41.37	88	5 (0)
D23	2892.00	0.99	47.17	0.99	47.64	1.47	40.00	IDA	0.72	46.74	26	0 (0)	0.10	40.26	18	0 (0)
D24	3341.00	1.99	34.36	0.83	37.80	4.08	13.60	IDA	0.00	17.13	10	0 (0)	0.00	13.85	7	0 (0)
D25	3317.00	1.69	34.56	0.18	35.08	1.25	5.16	IDA	0.30	8.36	20	0 (0)	0.24	5.52	10	0 (0)
D26	3772.00	0.19	66.99	0.19	67.52	1.82	2.57	IDA	0.00	4.31	5	0 (0)	0.00	2.84	4	0 (0)
D27	3388.00	3.03	119.33	2.89	121.46	2.70	93.07	IIA	2.70	99.84	38	0 (0)	1.85	96.25	39	1 (0)
D28	3896.00	1.17	93.48	0.76	94.12	4.04	123.38	IIA (γ)	1.54	177.36	1177	3 (2)	1.54	173.38	432	19 (4)
D29	3264.00	2.07	80.28	2.07	80.97	2.36	219.02	IIA	1.15	254.86	648	6 (0)	0.31	269.02	1034	30 (2)
D30	4037.00	1.78	108.91	1.78	109.50	1.85	69.14	IDA	0.32	93.97	267	5 (3)	0.25	76.97	260	1 (1)
D31	4175.00	3.67	106.17	2.48	109.16	2.52	85.70	ISA (γ)	0.00	117.86	12	0 (0)	0.00	86.53	23	0 (0)
D32	3857.00	2.87	118.78	1.91	119.34	3.45	229.80	IIA	0.00	244.22	31	0 (0)	0.00	231.44	43	0 (0)
D33	4578.00	3.30	97.94	2.47	102.61	2.26	34.93	IDA	0.13	37.19	6	0 (0)	0.00	36.15	3	0 (0)
D34	4753.00	0.46	121.86	0.38	123.14	0.61	10.15	ISA	0.29	11.00	10	0 (0)	0.00	10.98	1	0 (0)
D35	4437.00	2.83	110.75	1.94	111.41	1.51	25.44	ISA	0.56	37.99	9	0 (0)	0.31	26.41	1	0 (0)
Average		0.99	38.85	0.71	39.82	1.38	28.25		0.40	34.12	77.50		0.29	31.71	69.92	

Table 10: Heuristic results on instances of Set 4.

		Tabu		ILP-ref.		RBH			HCS (α=0)				HCS (α=0.05)			
Instances	Best Known	Gap (%)	Time	Gap (%)	Time	Gap (%)	Time	Routine	Gap (%)	Time	# cuts	# Aux. Pr.	Gap (%)	Time	# cuts	# Aux. Pr.
G0	6.00	0.00	2.38	0.00	2.39	0.00	0.02	ISA	0.00	0.05	0	0 (0)	0.00	0.03	0	0 (0)
G1	11.00	0.00	2.41	0.00	2.43	0.00	0.02	ISA	0.00	0.08	10	1 (1)	0.00	0.05	10	1 (1)
G2	9.00	0.00	2.66	0.00	3.05	0.00	0.04	ISA	0.00	0.05	0	0 (0)	0.00	0.04	0	0 (0)
G3	15.00	0.00	2.66	0.00	3.13	0.00	0.04	ISA	0.00	0.06	1	0 (0)	0.00	0.04	1	0 (0)
G4	16.00	0.00	2.19	0.00	2.22	0.00	0.04	ISA	0.00	0.05	3	0 (0)	0.00	0.04	3	0 (0)
G5	16.00	0.00	2.67	0.00	3.12	0.00	0.04	IDA	0.00	0.05	2	0 (0)	0.00	0.04	2	0 (0)
G6	23.00	0.00	2.67	0.00	3.09	0.00	0.04	ISA	0.00	0.08	5	1 (1)	0.00	0.04	5	1 (1)
G7	17.00	0.00	2.50	0.00	2.89	0.00	0.02	ISA	0.00	0.06	3	0 (0)	0.00	0.04	3	0 (0)
G8	19.00	0.00	2.81	0.00	3.17	0.00	0.06	IDA	0.00	0.06	2	0 (0)	0.00	0.06	2	0 (0)
G9	23.00	0.00	2.83	0.00	3.19	0.00	0.06	ISA	0.00	0.09	3	0 (0)	0.00	0.06	1	0 (0)
G10	26.00	0.00	2.67	0.00	3.03	0.00	0.07	ISA	0.00	0.19	14	1 (1)	0.00	0.09	0	1 (1)
G11	29.00	0.00	2.81	0.00	3.20	0.00	0.09	IDA	0.00	0.17	11	0 (0)	0.00	0.12	10	0 (0)
G12	46.00	0.00	3.55	0.00	4.02	13.21	12.54	IDA	2.13	14.16	22	0 (0)	2.13	12.59	12	0 (0)
G13	44.00	0.00	3.42	0.00	3.84	4.35	0.32	IDA (γ)	2.22	1.17	134	14 (1)	2.22	0.51	56	1 (1)
G14	45.00	0.00	3.56	0.00	3.95	6.25	0.31	ISA	0.00	0.64	44	2 (1)	0.00	0.56	67	4 (1)
G15	51.00	0.00	4.64	0.00	5.00	1.92	0.25	IDA	0.00	0.41	8	0 (0)	0.00	0.30	8	0 (0)
G16	51.00	0.00	3.86	0.00	4.31	7.27	0.78	IDA (γ)	0.00	1.44	109	3 (1)	0.00	0.94	48	1 (1)
G17	54.00	0.00	4.47	0.00	4.92	3.57	0.79	IDA (γ)	0.00	1.15	7	1 (1)	0.00	0.84	5	1 (1)
G18	51.00	0.00	2.99	0.00	3.43	0.00	1.42	IDA	0.00	1.95	17	0 (0)	0.00	1.51	9	1 (0)
G19	53.00	0.00	3.70	0.00	4.26	5.36	1.00	IDA	1.85	1.65	26	1 (1)	0.00	1.78	54	5 (2)
G20	54.00	0.00	2.98	0.00	3.36	3.57	0.54	IDA	1.82	0.84	27	0 (0)	0.00	0.73	14	2 (1)
G21	75.00	0.00	7.50	0.00	7.92	5.06	6.28	IDA (γ)	0.00	8.52	10	0 (0)	0.00	6.35	6	0 (0)
G22	76.00	0.00	6.34	0.00	6.73	0.00	2.35	ISA	0.00	3.37	9	0 (0)	0.00	2.57	5	0 (0)
G23	82.00	0.00	6.84	0.00	7.26	1.20	4.18	IDA (γ)	0.00	5.82	7	0 (0)	0.00	4.27	6	0 (0)
G24	110.00	0.00	20.78	0.00	21.20	2.65	27.44	IDA (γ)	0.90	31.12	26	0 (0)	0.00	27.81	14	0 (0)
G25	94.00	0.00	8.70	0.00	9.25	3.09	10.76	IDA (γ)	0.00	14.96	57	0 (0)	0.00	11.19	20	1 (1)
G26	101.00	0.98	10.94	0.98	11.35	2.88	7.63	ISA	0.98	10.67	16	0 (0)	0.00	7.86	7	0 (0)
G27	82.00	0.00	4.16	0.00	6.88	2.38	4.47	IDA	2.38	22.18	347	29 (0)	0.00	35.99	420	130 (0)
G28	94.00	0.00	5.53	0.00	5.89	5.05	23.24	IIA (γ)	5.05	46.24	647	21 (0)	0.00	40.95	844	25 (2)
G29	89.00 *	0.00	4.38	0.00	6.40	0.00	15.08	IDA	0.00	41.78	312	30 (0)	0.00	65.08	528	99 (0)
G30	122.00	0.00	15.63	0.00	16.35	5.43	53.74	IDA (γ)	0.00	57.30	135	0 (0)	0.00	57.70	136	4 (3)
G31	131.00	0.00	15.50	0.00	15.97	6.43	40.46	IDA (γ)	0.76	64.12	122	3 (2)	0.00	45.32	75	4 (2)
G32	139.00	0.71	14.70	0.71	15.22	7.95	46.24	IDA (γ)	0.71	67.17	74	3 (1)	0.00	47.86	51	1 (1)
G33	168.00	2.89	38.89	2.89	39.42	1.75	68.47	IDA (γ)	0.00	82.38	69	1 (0)	0.00	71.59	37	1 (1)
G34	162.00	1.22	38.19	1.22	38.69	4.14	96.70	IDA (γ)	0.00	97.98	23	0 (0)	0.00	97.48	10	0 (0)
G35	164.00	0.61	53.33	0.61	53.78	2.38	95.28	IDA (γ)	0.00	122.57	24	1 (1)	0.00	101.04	20	1 (1)
Average		0.18	8.77	0.18	9.29	2.66	14.47		0.52	19.46	64.61		0.12	17.87	69.14	

Table 11: Heuristic results on instances of Set 5.

		Tabu		ILP-ref.		RBH			HCS (α=0)				HCS (α=0.05)			
Instances	Best Known	Gap (%)	Time	Gap (%)	Time	Gap (%)	Time	Routine	Gap (%)	Time	# cuts	# Aux. Pr.	Gap (%)	Time	# cuts	# Aux. Pr.
R0	45900.00	0.00	3.63	0.00	4.04	0.00	0.04	ISA	0.00	0.08	2	0 (0)	0.00	0.06	1	0 (0)
R1	69694.00	0.00	4.83	0.00	5.24	0.63	0.06	IIA	0.63	0.11	6	0 (0)	0.63	0.08	5	0 (0)
R2	66712.00	0.00	8.05	0.00	8.63	3.43	0.08	IDA (γ)	1.30	0.19	19	1 (0)	1.30	0.12	19	1 (0)
R3	140259.00	0.00	9.20	0.00	9.70	0.00	0.09	IDA	0.00	0.17	5	0 (0)	0.00	0.11	5	0 (0)
R4	86507.00	0.70	6.20	0.70	6.72	0.00	0.04	ISA	0.00	0.11	0	0 (0)	0.00	0.06	0	0 (0)
R5	72779.00	0.00	7.48	0.00	8.06	4.84	0.15	IDA	0.00	0.28	14	0 (0)	0.00	0.19	13	0 (0)
R6	134780.00	0.84	18.58	0.84	19.63	0.00	0.08	ISA	0.00	0.22	10	0 (0)	0.00	0.16	9	0 (0)
R7	83653.00	0.00	7.83	0.00	8.21	0.00	0.17	IDA	0.00	0.28	11	0 (0)	0.00	0.23	10	0 (0)
R8	143229.00	0.18	18.80	0.18	19.24	0.00	0.26	IDA (γ)	0.00	0.45	24	0 (0)	0.00	0.34	24	0 (0)
R9	138809.00	1.90	18.36	1.10	19.72	0.47	0.10	ISA (γ)	0.00	0.17	1	0 (0)	0.00	0.14	1	0 (0)
R10	119540.00	4.57	30.16	0.54	33.97	1.62	0.40	ISA	0.00	0.62	7	0 (0)	0.00	0.47	7	0 (0)
R11	93686.00	0.75	14.19	0.48	15.27	0.48	0.22	IDA	0.48	0.31	6	0 (0)	0.48	0.27	11	0 (0)
R12	79132.00	1.52	8.70	0.64	9.11	1.52	0.17	IDA	1.52	0.81	58	9 (1)	1.52	0.78	81	11 (2)
R13	266344.00	3.55	101.44	0.44	101.99	0.56	0.23	ISA (γ)	0.00	0.66	19	0 (0)	0.00	0.48	18	0 (0)
R14	258759.00	1.96	97.19	1.36	112.19	2.12	2.12	IIA	0.00	3.00	7	0 (0)	0.00	2.20	4	0 (0)
R15	212498.00	1.30	79.53	1.01	79.97	1.06	0.42	ISA (γ)	0.58	1.39	38	0 (0)	0.00	0.94	26	0 (0)
R16	140184.00	0.71	35.50	0.53	41.30	0.72	0.64	IIA	0.72	0.94	21	0 (0)	0.72	0.76	16	0 (0)
R17	109525.00	0.13	71.75	0.00	78.66	0.39	0.25	IDA	0.39	2.09	111	11 (0)	0.39	1.62	117	10 (0)
R18	206141.00	1.54	120.17	0.82	124.22	2.63	0.54	ISA (γ)	0.51	2.65	117	4 (1)	0.00	1.36	70	1 (0)
R19	203643.00	1.53	75.63	0.62	76.99	0.48	0.31	IDA	0.00	0.50	2	0 (0)	0.00	0.39	2	0 (0)
Average		1.06	36.86	0.46	39.14	1.05	0.32		0.31	0.75	23.90		0.25	0.54	21.95	

Table 12: Heuristic results on instances of Set 6.

	Tabu Search - ILP-ref.				RBH - HCS (α=0.05)			
	Gap		Time		Gap		Time	
Instances	Avg.	Max	Avg.	Max	Avg.	Max	Avg.	Max
DPRPP_ϵ_δ_val	0.08	0.47	16.33	55.86	0.00	0.00	0.41	1.20
DPRPP_ϵ_δ_egl	0.29	2.74	70.22	404.43	0.28	1.65	75.04	378.46
DPRPP_P	0.38	1.97	19.84	117.58	0.08	1.96	4.55	54.60
DPRPP_D	0.71	2.89	39.82	123.14	0.29	1.89	31.71	269.02
DPRPP_G	0.18	2.89	9.29	53.78	0.12	2.22	17.87	101.04
DPRPP_R	0.46	1.36	39.14	124.22	0.25	1.52	0.54	2.20
All instances	0.35	2.89	30.48	404.43	0.17	2.22	21.67	378.46

Table 13: Summary Table

	RBH	HCS (α=0.05)
ISA	69	56
IDA	48	14
IIA	13	12
ISA (γ)	13	2
IDA (γ)	20	2
IIA (γ)	9	1
HCS	-	85
Total	172	172

Table 14: Performance analysis of RBH routines and HCS

	#	Tabu Search	ILP Ref.	RBH	HCS (α=0.05)
Set 1 - DPRPP_ϵ_δ_val	30	21 (21)	21 (21)	28 (28)	30 (30)
Set 2 - DPRPP_ϵ_δ_egl	24	18 (16)	18 (16)	10 (9)	21 (19)
Set 3 - DPRPP_P	26	15 (15)	15 (15)	10 (10)	25 (25)
Set 4 - DPRPP_D	36	15 (15)	17 (17)	10 (10)	31 (31)
Set 5 - DPRPP_G	36	31 (31)	31 (31)	15 (15)	34 (34)
Set 6 - DPRPP_R	20	6 (6)	10 (7)	7 (6)	15 (14)
All instances	172	106 (104)	112 (107)	80 (78)	156 (153)

Table 15: Comparison with benchmark

5 Conclusions

In this paper we analyze a generalization of the Directed Rural Postman problem where not all arcs requiring a service has to be visited provided that a penalty is paid for those not served. This generalization is called Profitable Directed Rural Postman problem.

We propose a branch and cut algorithm and two heuristic methods. The exact algorithm introduces cuts in a lazy way and this has allowed to close all but three open benchmark problems available in the literature. Heuristics behave very well outperforming state of the art algorithms in many sets of instances. The core of the first procedure is the solution of

Directed Rural Postman Problems constructed on subsets of service arcs previously selected by the optimal solution of a problem relaxation. The second heuristic procedure is a quite general method based on a branch and cut structure exploiting heuristic cuts that may eliminate feasible solutions. We believe that the main ideas behind these two heuristic algorithms deserve to be further developed also for other arc routing problems.

References

[1] Aráoz, J., Fernández, E., Zoltan, C., (2006). "Privatized Rural Postman Problem". Computers and Operations Research 33(12), 3432-3449.

[2] Aráoz, J., Fernández, E., Meza, O., (2009). "Solving the Prize-collecting Rural Postman Problem". European Journal of Operational Research 196(3), 886-896.

[3] Aráoz, J., Fernández, E., Franquesa, C., (2009). "The Clustered Prize-Collecting Arc Routing Problem". Transportation Science 43(3), 287-300.

[4] Archetti C., Guastaroba G., Speranza M.G., "An ILP-refined tabu search for the Directed Profitable Rural Postman Problem", Discrete Applied Mathematics, Available online 30 June 2012.

[5] Ball, M.O., and Magazine, M.J., (1988). "Sequencing of insertions in printed circuit board assembly". Oper. Res. 36(2), 192-201.

[6] Benavent, E., Campos, V., Corberán, A., Mota, E., (1992). "The capacitated Chinese postman problem: Lower bounds". Networks, 22(7), 669-690.

[7] Black, D., Eglese, R., Wøhlk, S., (2013). "The time-dependent prize-collecting arc routing problem". Computers & Operations Research 40, 526-535.

[8] Boykov, Y. and Kolmogorov, V., (2004). An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision". IEEE Transaction on Pattern Analysis and Machine Intelligence 26(9), 1124-1137.

[9] Christofides N., Campos V., Coberán A. and Mota E., (1981). "An algorithm for the rural postman problem", Imperial college report ICOR 81.5.

[10] Christofides N., Campos V., Coberán A. and Mota E., (1986). "An algorithm for the rural postman problem on a directed graph", Mathematical Programming Study 26, 155-166.

[11] Chu, C.-W., (2005). "A heuristic algorithm for the truckload and less-than-truckload problem." European Journal of Operational Research 165(3), 657-667.

[12] Corberán, A. and Prins C., (2010). "Recent results on arc routing problems: An annotated bibliography". Networks 56(1), 50-69.

[13] Corberán, A., Fernández, E., Franquesa, C., Sanchis, J. M., (2011). "The Windy Clustered Prize-Collecting Arc-Routing Problem". Transportation Science 45(3), 317-334.

[14] Côté J. F. and Potvin J. Y., (2009). "A tabu search heuristic for the vehicle routing problem with private fleet and common carrier". European Journal of Operational Research 198(2), 464-469.

[15] Deitch, R., and Ladany, S. P., (2000). "The one-period bus touring problem: Solved by an effective heuristic for the orienteering tour problem and improvement algorithm". European Journal of Operational Research 127, 69-77.

[16] Dror M., (2000). "Arc routing: Theory, solutions and applications". Dordrecht: Kluwer Academic Publishers.

[17] Eiselt, H. A., Gendreau, M., Laporte, G., (1995). "Arc Routing Problems, Part I: The Chinese Postman Problem". Operations Research 43(2), 231-242.

[18] Eiselt, H. A., Gendreau, M., Laporte, G., (1995). "Arc Routing Problems, Part II: The Rural Postman Problem". Operations Research 43(3), 399-414.

[19] Feillet, D., Dejax, P., Gendreau, M., (2005). "The profitable arc tour problem: Solution with a branch and price algorithm". Transportation Science 39, 539-552.

[20] Feillet, D., Dejax P., Gendreau M., (1981). "Traveling salesman problems with profits". Transportation Science 39(2), 188-205.

[21] Fernández, E., Meza, O., Garfinkel, R., Ortega, M., (2003). "On the undirected rural postman problem: Tight bounds based on a new formulation". Operations Research 51, 281291.

[22] Guastaroba, G., Mansini, R., Speranza, M. G., (2009). "Modeling the Pre-Auction Stage: The Truckload Case", Innovations in Distribution Logistics. J. A. E. E. Nunen, M. G. Speranza and L. Bertazzi, Springer Berlin Heidelberg 619, 219-233.

[23] Gutin, G. and Punnen, A., (2002). "Traveling salesman problem and its variations". Dordrecht: Kluwer Academic Publishers.

[24] Li, L. and Eglese, R., (1996). "An iterative algorithm for vehicle routing for winter gritting". Journal of Operational Research Society 47(2), 217-228.

[25] Maniezzo, V., Stützle, T. and Voss, S., (2009). Matheuristics: Hybridizing Metaheuristics and Mathematical Programming. Annals of Information Systems, vol. 10, Springer, Heidelberg,

[26] Orloff, C. S., (1974). "Fundamental Problem in Vehicle Routing". Networks 4, 35-64.

[27] Toth, P. and Vigo, D.,(2002) "The vehicle routing problem", Philadelphia: SIAM, Society for Industrial and Applied Mathematics.

[28] Vansteenwegen, P., Souffriau, W., Van Oudheusden, D., (2011). "The orienteering problem: A survey". European Journal of Operational Research. Volume 209. Issue 1.