Available online at www.sciencedirect.com

SciVerse ScienceDirect

Electronic Notes in Theoretical Computer Science

Electronic Notes in Theoretical Computer Science 281 (2011) 85–100

www.elsevier.com/locate/entcs

A Tabu Search Heuristic for the Prize-collecting Rural Postman Problem

Guillermo Palma[1]

Departamento de Computación y Tecnología de la Información
Universidad Simón Bolívar
Caracas, Venezuela

Abstract

I present a heuristic based in tabu search, in order to generate feasible solutions for solving the Prize-collecting Rural Postman Problem. This problem was recently defined and is a generalization of other arc routing problems. The numerical results from a series of computational experiments with various types of instances show the good behavior of the proposed algorithm in comparison with previous works.

Keywords: Arc routing, metaheuristic, tabu search, combinatorial optimization.

1 Introduction

The Prize-Collecting Arc Routing Problems (PARP), were introduced by Aráoz, Fernández and Zoltán [6] and they are a generalization of the Arc Routing Problems (ARP). The Arc Routing Problems aim to find the greatest benefit of traverse from some edges or arcs of a graph, subject to certain restrictions. The main Arc Routing Problems are the Chinese Postman Problem (CPP), and the Rural Postman Problem (RPP). The book edited by Dror [14] is the main reference about the ARPs. The PARP is defined about a graph where there is a vertex d called the deposit. Each edge has a profit function and a cost function. The profit function is taken into account only when an edge is traversed for first time. The objective the PARP is find a maximum profit cycle passing through d. Aráoz et al. [6] show that the PARPs are NP-hard and that are generalizations of many Arc Routing Problems and of the Traveling Salesman Problem. The basic PARP is the Prize-Collecting Rural Postman Problem. Other PARPs are defined by adding constraints.

The PRPP is to find the maximum profit cycle passing through the deposit in an undirected graph. At the PRPP the demand service is on the edges of graph.

[1] Email: gpalma@ldc.usb.ve

doi:10.1016/j.entcs.2011.11.027

When perform a service to a edge not only incurs a cost but a profit is received. The cost is to traverse the edge. The profit is due to the service provided to the edge. In a cycle is taken into account the cost of traversing all edges of the same. Is possible that a edge is traversed more than once. Only when an edge is traversed for the first time in a route, a reward is earned. The objective the PRPP is to find a route that begin and finish in the deposit. The objective the PRPP is to find a route that begin and ending in the deposit, such that maximizes the total profit to serve the edges, less the the cost of traverse them.

The figure 1 show an example of a graph that is a instance the PRPP. The figure 2 shown the cycle that is optimal solution of problem of figure 1.

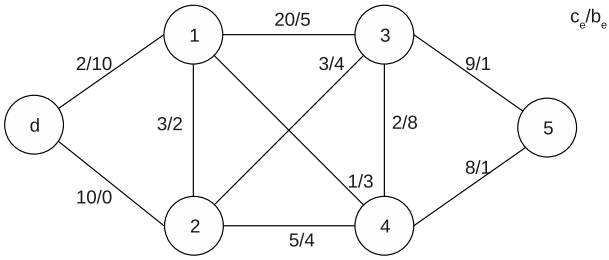

Figure 1. A PRPP instance. The vertex d is the deposit and c_e/b_e are the values cost/profit of a edge e

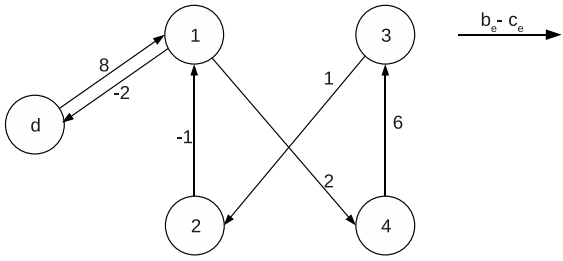

Figure 2. Optimal solution of the figure 1. The cycle is d-1-4-3-2-1-d with profit equal to 14

Aráoz, Fernández and Meza [5] proposed the first algorithmic solution for the PRPP. Their algorithm has two phases and in each phase used a diferent solver. The first phase give a approximate solution of the PRPP through a linear inequalities system along with a heuristic. The heuristic is an adaptation of the heuristic 3T for the RPP [16]. The second phase uses the first solution to obtain the optimal solution of the problem.

Besides the PRPP, there are four problemas that belong to the family of the PARPs. The first is the Weighted PARP (WPARP) [9], where there is a weight associated with the service of each edge and there is a limit in the total weight that can have a route when it is served. A solution can have more than one route. The second is the Clustered Prize-collecting Arc Routing Problem (CPARP) [3]. The objetive the CPARP is to find a cycle that passes for d, such that satisfies the demand of a subset of the cluster of edges. Franquesa [17] present a complete study of the CPARP and of the algorithms that solved it. Aráoz, Fernández and Franquesa [3] recently, proposed an exact algorithm based in linear programming

and a heuristic for solved the CPARP. The third problem of the PARPs family is the Windy Clustered Prize-collecting Arc Routing Problem (WCPARP). This problem have all features of the CARP. The only difference is that the cost de traverse a arc depend the direction in that traversed it. Franquesa [17] studied the features of the problem and he proposes an exact algorithm to solve it. Aráoz et al. [4] present an exact algorithm based in branch-and-cut to solve the WCPARP and a simple heuristic to obtain feasible solutions. The fourth and final problem that belong to the PARPs family is the Profitable Capacitated Rural Postman (PCRPP). The PCRPP was introduced in 2009 for Stefan Irnich [24] For the PCRPP apply the same definition of the PRPP. Additionally when traversing a edge for first time is consumed a time q_e, and the others times is consumed a time R_e. The objective the PCRPP is to find a cycle of maximum profit, such that the cycle has a time less or equal that Q.

In addition to the studies about the PRPP and the related problems I have knowledge of six works of arc routing problems with profits. Deitch and Ladany [13] were interested in determining a attractive transport route for a turistic region. Feillet, Dejax and Gendreau [15] introduces a new problem named the Profitable Arc Tour Problem (PATP). The objetive of the PATP is to find a set of cycles in the graph that maximize the profit less the cost of the route, with restrictions on the number of times that profit is in the edges and the maximum length of the cycles. Malandraki and Daskin [25] introduces the Maximum benefit Chinese postman problem (MBCPP) and the Maximum benefit traveling salesman problem (MBTSP). The MBCPP is analogous to the PRPP, the one difference is that the MBCPP is definite with a direct graph and that each arc has a profit function b_{en}. This function returns a profit when an arc e is traversed for nth time. The MBTSP definition is similar the main difference is that the benefit is obtained by passing through a node. Finally Hertz et al. introduce the Undirected capacitated arc routing problems with profits [8] [7] (UCARPP). The UCARPP is defined on an undirected graph in which each edge has a reward associated with them, a demand and travel time. A fleet of vehicles serving the edges. The goal is to find a set of routes of maximum benefit that satisfies all constraints.

There are potential practical applications for the PRPP. Applications of arc routing problems trying to minimize costs. It was decided that the service demand is exactly in some places. Because the goal is not to decide which edges will be served, is to determine the least cost to traverse them. With the rural postman problem can be modeled more complex practical problems. A company that wants to maximize profits, can decide that the demand of a edge will not be served, unless this provided a benefit for the company. An example where this principle is applied, is in the collection of recycling bins by private companies. Another example is the mail service managed by private companies. Companies may choose the districts to which they will serve. Because of the success that has been obtained by applying the Tabu Search to several ARPs [23,12,11,2,19,1,10,8,7] this metaheuristic is chosen as the basis of algorithmic solution for the PRPP.

This paper is organized as follows. The section 2 describes formally the PRPP

and some of its properties. The section 3 presents the algorithms developed to solve the PRPP. The experimental results and discussion of them is done in section 4. Section 5 presents the conclusions of the work.

2 The Prize-collecting Rural Postman Problem

In this section I define formally PRPP and present some notations and definitions to be used in this work.

Definition 2.1 Let $G(V, E)$ be an undirected graph with a distinguished d, called deposit, and with two functions on the edge set E in $\mathbb{R}^+$, the profit function b and the cost function c. The objective of the problem is to find a cycle $\mathcal{C}^*$ that maximizes the value of

$$\sum_{e \in E(\mathcal{C})} (b_e - t_e c_e) \tag{1}$$

where $\mathcal{C}$ is an cycle in G passing through d, which is not necessarily simple, edges can be repeated, t_e is the number of times that edge e is traversed in $\mathcal{C}$ and $E(\mathcal{C})$ is the edges set of cycle $\mathcal{C}$.

The following notations to use and other definitions.

$V(H)$**:** Vertex set of an subgraph H. If $H = G$ then $V(G) = V$.

$E(H)$**:** Edge vertex of an subgraph H. If $H = G$ then $E(G) = E$.

$\gamma(S)$**:** Let S be a set such that $\emptyset \subset S \subseteq V$ we denote by $\gamma(S) = \{e \in E \mid e = \{u, v\}, u, v \in S\}$ the edges set with both vertices in S.

$\delta(S)$**:** Let S be a set such that $\emptyset \subset S \subseteq V$ we denote by $\delta(S) = \{e \in E \mid e = \{u, v\}, u \in S, v \in V \setminus S\} = \delta(V \setminus S)$ the edges in the cut between S and $V \setminus S$.

For a singleton set, I do not use the brackets. For example the following expressions are valid $\delta(v) \equiv \delta(\{v\})$ and $\gamma(v) \equiv \gamma(\{v\})$.

The following functions are defined on the edges e of a graph G.

- $\varphi_e = b_e - c_e$. φ_e is the profit you get when a edge is traversed for first time.
- $\psi_e = b_e - 2c_e$. ψ_e is the profit you get when a edge is traversed two times.

The edges set is divided into three sets

$$P = \{e \in E \mid \varphi_e < 0\} \qquad R = \{e \in E \mid \psi_e \geq 0\} \qquad Q = \{e \in E \setminus R \mid \varphi_e \geq 0\}$$

Definition 2.2 [Even Graph] A Even Graph is a graph in the which all vertices have even degree.

We denote by (G, d, b, c) a PRPP instance, where G is the graph associate to the problem, d is the deposit vertex, b is the profit function and c is the cost function.

G_R**:** $G_R \equiv (V(R) \cup d, R)$ is is the subgraph obtained with the edges set R and the deposit d. Where $V(R)$ is vertex set incidents with the edges of type R.

C_k**:** Are the connected components of graph G_R, where $k \in \{0, \ldots, n\}$. We assume that $d \in C_0$.

$V(C_k) = V_k$**:** Vertex set of connected component C_k.

$\gamma_R(V_k)$**:** With $\gamma(V_k)$ we obtain a set of edges corresponding to the original graph. While with $\gamma_R(V_k)$ we obtain a set of edges corresponding to the graph G_R. Therefore $\gamma_R(V_k) = \gamma(V_k) \cap R$.

Here are some properties that have the PRPP optimal solutions. These were derived and proved by Aráoz, Fernández and Zoltán [6]. Let $\mathscr{C}^*$ be a cycle that is optimal solution of the PRPP.

Dominance 1 *Neither edge $e \in E$ is present more than twice in $\mathscr{C}^*$.*

Dominance 2 *Let $e \in \mathscr{C}^*$ be, if for any connected component C_k of the graph G_R, we have that $V(e) \cap V_k \neq \emptyset$ then all edges of $\gamma_R(V_k)$ is in $\mathscr{C}^*$.*

Remark 2.3 The Dominance 2 implies that for each connected component C_k, we have that the edges in $\gamma_R(V_k)$, or all them are in the optimal solution $\mathscr{C}^*$, or none of them are in $\mathscr{C}^*$. The Dominance 2 also implies that if exist a edge that is not in the set R, is in cycle $\mathscr{C}^*$ and that is incident with any vertex V_k, then all the edges are in $\mathscr{C}^*$.

Preprocessing 1 *Let C_k be the connected components of G_R and let $e \in \gamma(V_k) \setminus R$ for some k. Then we have that the edge e is traversed at most once in the optimal solution $\mathscr{C}^*$*

Remark 2.4 Dominance 2 and Preprocessing 1 imply that if a edge $e \in \gamma(V_k) \setminus R$ is in $\mathscr{C}^*$, then all the edges $\gamma_R(V_k)$ are in $\mathscr{C}^*$

3 Algorithm to solving the PRPP

I developed an algorithm based on Tabu Search to solve the PRPP. This algorithm has two phases. In the first phase generates two solutions feasible for PRPP by two constructive algorithms. These algorithms are called *Union of Connected Components* and *Successive Elimination of Connected Components*. The second phase is the application of a algorithm of Tabu Search that tries to improve the initials solutions.

The algorithm 1 *Union of Connected Components* is based in two algorithms. These are the algorithm presented by Pearn and Wang for the *Maximum Benefit Chinese Postman Problem* [26] and the approximate algorithm for the RPP proposed by Frederickson [18]. The algorithm constructs a feasible solution for the PRPP, that has all the edges of graph except the edges of the input list. The idea of the Tabu Search is to create an initial feasible solution containing the edges that give profit when them are traversed for first time. The idea of the Tabu Search is to create an initial feasible solution containing the edges that give profit when them are traversed for first time, these are the edges that belong to set $R \cup Q$. Aráoz et al. [6] prove that the edges $e \in R \cup Q$ are not necessarily part of the optimal solution of the PRPP. However it is considered a best strategy lose profit for including these edges in the solution, instead to lose profit for not including them. One reason for

using this technique is because with the Tabu Search algorithm is possible to determine if the elimination of edges of a cycle, that is a feasible solution, is obtained a cycle of greater benefit. One reason for using this technique is because with the Tabu Search algorithm is possible to determine if with the elimination of edges of a cycle, that is a feasible solution, is obtained a cycle of greater benefit.

In the Tabu Search the algorithm 1 *Union of Connected Components* provides a feasible solution for the PRPP that contains all the edges that provide benefits, this is the edges of type R and Q. The algorthm 3 *Successive Elimination of Connected Components* has a strategy to rule out the edges of R and Q types that would not provide benefit to a solution of PRPP. Let $G_t = (V_t, E_t)$ be of *step 2* of the algorithm 1 *Union of Connected Components*. Then in *step* 3 of the algorithm 1 is obtained *minimum spanning tree* of the graph G_t with the set of edges E_{st}. With this set of edges can be created the graph $G_{mst} = (V_t, E_{st})$ which corresponds to the tree. Each vertex $v_i \in V_t$ corresponds to a connected component C_i. Each edge of E_{st} correspond to minimum cost paths between the connected components. C_0 is the connected component containing the deposit. Traverse the graph G_{mst} with a search algorithm from vertex $v_0 \in V_t$ that is in the connected component C_0. When visiting each vertex v_i inV_t can be obtained the cost c_{v_i} of reaching the vertex v_i and the connected component C_i. Can also be calculate the maximum benefit that can be obtained from a connected component C_i, this is $\varphi(\gamma(C_i))$. If the maximum benefit of the connected component is less than cost the path in which this component is reached, that is $c_{v_i} > \varphi(\gamma(C_i))$ then this component should not be part of the solution. This is because the inclusion of this set of edges could provide losses to a solution of the PRPP. This is the basic idea of the algorthm 3 *Successive Elimination of Connected Components*. Connected components are sorted in ascending order based on the value of the total profit minus the cost of reaching the component, such that is not mandatory for the solution to have the edges of the connected component C_i. From all solutions obtained, we choose the of greatest profit.

Tabu Search was introduced by Fred Glover [20] and is one of the metaheuristics more successful in solving combinatorial optimization problems. Good descriptions of the concepts and applications of the Tabu Search is presented by Glover and Laguna [21] and by Melián and Glover [22]. The Tabu Search is an metaheuristic algorithm based on adaptive memory and on the adoption of intelligent general principles for the resolution problems. Tabu Search is used to guide a slocal earch in the attempt to find a global optimum using memory structures that help to avoid falling into local optima and to visit previously reached states.

The basic idea of the algorithm 5 *Búsqueda Tabú para el PRPP* is to start with a feasible solution for the PRPP to do a local search that has adaptive memory and has the ability to reset at different points of the search space, if necessary. Before explaining how to work the algorithm will describe some of their structures and basic procedures.

First we will explain how it generates a set of neighboring solutions from a feasible solution of PRPP. It will use an exchange scheme of edges by paths inspired by the one presented by Coberán et al. for the Mixed RPP [12]. In

Algorithm 1: Union of Connected Components

Input: A graph $G = (V, E)$ and an edge set L_e
Output: A cycle $\mathscr{C}$, feasible solution the PRPP

STEP 1: Create a graph $G_s = (V_s, E_s)$, where $E_s = E \backslash L_e$. If the deposit $d \notin V_s$ then is added to the graph G_s and will be considered the connected components C_0. Do $G' \leftarrow G_s$. If G' is eulerian (connected and even) go to step **6**. If G' is only connected, do $E'_{st} \leftarrow \emptyset$ and go to step **4**

Step 2: Let $C_0, C_1, \ldots, C_t$ the two connected components of G'. Construct a complete graph $G_t = (V_t, E_t)$, where V_t has as elements each of connected components C_i. Each edge $e_t \in E_t$ has a cost given by the function $c_{e_t} : E_t \rightarrow \mathbb{R}$ This function is defined as

$$c_{e_t}(e_t) = \min\{dcc(v_i, v_j) \mid v_i \in C_i \wedge v_j \in C_j \wedge i \neq j\}$$

where *dcc* is a function that return the value of minimum cost path between the vertices v_i and v_j in the graph G. We called to the minimum cost path as $MC_{v_i v_j}$. Let MCP be a set with the $MC_{v_i v_j}$. We use the following function to obtain the minimum cost paths $MCP \leftarrow$ `find-minimum-cost-path`(G, G')

Step 3: We calculate the *minimum spanning tree* of the graph G_t, to obtain the edges E_{st} corresponding to the edge set of the tree. Let E'_{st} be the edge set $e \in E$ that are obtained by determining the paths of the graph G corresponding to each of the edges of E_{st}. E'_{st} is added to the graph G' to obtain $G' = (V_s, E_S \cup E'_{st})$

Step 4: We obtain the set of vertices of odd degree of the graph G' and is called V_o. We build a complete graph $G_m = (V_o, E_o)$, in which each edge $e_o = (i, j) \mid e_o \in E_o$ will have as cost c_{e_o} the value of the minimum cost path, called MC_{ij}. We have that MC_{ij} is the minimum cost path between the vertices $i, j \in V_o$ in the graph G Let MCP be a set of MC_{ij}, we have the minimum cost paths are obtained with the function
$MCP \leftarrow$ `find-minimum-cost-path`(G, G')

Step 5: Obtain the minimum cost perfect matching in the graph G_m. Let E_m be the set of edges of G_m corresponding that to optimal matching. Let E'_m be the set of edges $e \in E$ that obtained by determining the paths of the graph G that corresponds to each edge of E_m. E'_m is adding to the graph G' to obtain $G' = (V_s, E_s \cup E_{st} \cup E_m)$. We have G' is a eulerian graph (even and connected) therefore it has a feasible solution of the PRPP

Step 6: The eulerian cycle $\mathscr{C}$ is determined with the algorithm *Eulerian-Cycle* presented in Dror book [14]. We have that $\mathscr{C}$ is a cycle that is a feasible solution for the PRPP in G

Step 7: return $\mathscr{C}$

Function 2: find-minimum-cost-path

Input: The graphs $G = (V, E)$, and $G_s = (V_s, E_s)$
Output: Set of minimum cost paths MCP

begin
  $MCP \leftarrow \emptyset$ `// Set of minimum cost path`
  $EdgeSet \leftarrow \emptyset$ `// Set of edges`
  Let $MC_{v_iv_j}$ be the minimum cost path between the vertices $v_i, v_j \in V_s$ in the graph G. Each edge $e \in E$, has a cost given by function $c_{st} : E \rightarrow \mathbb{R}$ that will depend on the paths $MC_{v_iv_j}$ previously found.
  foreach $v_i, v_j \in V_s$ **do**
    Determine $MC_{v_iv_j}$ with $c_{st}(e) = \begin{cases} c_e, & \text{if } e \in E_s \cup EdgeSet \\ -\varphi_e, & \text{otherwise} \end{cases}$
    $MCP \leftarrow MCP \cup \{MC_{v_iv_j}\}$
    $EdgeSet \leftarrow EdgeSet \cup$ the edges of the $MC_{v_iv_j}$
  return MCP

the function 4 obtenerVecinos, we present the algorithm to generate the set of neighboring solutions. Let be $\mathscr{C}$ a cycle that is a feasible solution of the PRPP represented by a sequence of edges which correspond to the order on that the edges of graph are traversed. The cycle $\mathscr{C}$ has the following form $\mathscr{C} = ((d,a)(a,b),\ldots,(i,j),\ldots,(l,m),\ldots,(p,q),\ldots,(s,t),(t,d))$, where d is the deposit. Given any edge $e = (l,m)$, such that belong to the solution $\mathscr{C}$, starting from this edge the solution $\mathscr{C}$ is traversed from the left and right of the edge e. When we traversed to the left of the solution $\mathscr{C}$, we want to find one edge h of the type that provides benefit when traversing for the first time, this is $h \in R \cup Q$. If we not found a edge $h \in R \cup Q$ we take the edge that is connected to the reservoir, in this case in the solution $\mathscr{C}$ we have that $h = (d,a)$. Suppose if we find one edge $h \in R \cup Q$ in $\mathscr{C}$, this edge is denoted as $h = (i,j) \in \mathscr{C}$. We do the same procedure starting from edge, we traversed the solution from the right until to find one edge or by default, the rightmost edge that has the deposit, this is $k = (t,d) \in \mathscr{C}$. Once we have identified these three edges $h = (i,j)$, $e = (l,m)$ and $k = (p,q)$ in the solution $\mathscr{C}$, we find the minimum cost path between the vertices j and p in the graph G that we call MCP_{jp}. Then we replace the path of cycle by the minimal cost path between j and p, this is MCP_{jp}. This will result in a new feasible solution for the PRPP $\mathscr{V}$, which has the form $\mathscr{V} = ((d,a)(a,b),\ldots,(i,j),CCM_{jp},(p,q),\ldots,(s,t),(t,d))$. This procedure is applied to all edges of the solution $\mathscr{C}$, thereby generates a set of feasible solutions of PRPP. The set of neighboring solutions will be composed of tuples $(\mathscr{V}, e)$, where $\mathscr{V}$ is a feasible solution and $e = (l,m) \in \mathscr{C}$ is the edge that was selected at the beginning of the algorithm and that allowed the generation of the solution $\mathscr{V}$. The function 4 obtenerVecinos shows the algorithm to get the set of neighboring solutions as a set of tuples. Let $\mathscr{C}$ be the feasible solution of PRPP with which you get the set neighboring solutions. Let $G = (V, E)$ be the graph that represents the instance PRPP to resolve. In obtaining the minimum cost paths

Algorithm 3: Successive Elimination of Connected Components

Input: A graph $G = (V, E)$
Output: A cycle $\mathscr{C}$ feasible solution of the PRPP

begin
  tabu-edges ← All the edges of the graph G of type P
  Apply steps 1 and 2 of the algorithm 1 *Union of Connected Components* with input, the graph G and tabu-edges. Is obtained a graph $G_t = (V_t, E_t)$.
  ncc ← $|V_t|$ `// Number of connected components`
  best-solution ← *Union of Connected Components* with input G and tabu-edges.
  if ncc $= 1$ **then return** best-solution.
  Calculate the *minimum spanning tree* of the graph G_t resulting in a set of edges E_{st}. Create the graph $G_{mst} = (V_t, E_{st})$.
  Traverse the graph G_{mst} with the depth first search algorithm from the vertex the vertex that corresponds to the component C_0 (contains the deposit d) and calculate the cost c_{C_i} of reaching each connected component $C_i \in V_t$.
  foreach *connected component* $C_i \in V_t$ **do**
    Calculate the maximum profit can be obtained $\varphi(\gamma(C_i))$.
    $profit_{C_i} \leftarrow \varphi(\gamma(C_i)) - c_{C_i}$
    profit-connec-component ← profit-connec-component $\cup\{(C_i, profit_{C_i})\}$
  It is ordered ascending the set of profits of the connected components profit-connec-component.
  foreach *tuple* $(C_i, profit_{C_i}) \in$ profit-connec-component **do**
    edges-connec-component ← Obtain the edges in the graph G that form the connected component C_i.
    tabu-edges ← tabu-edges ∪ edges-connec-component.
    solution ← *Union of Connected Components* with input G and tabu-edges.
    if profit(solution) > profit(best-solution) **then** best-solution ← solution.
  return best-solution

MCP_{jp} each edge has a cost given by the function c (2).

$$c(e) = \begin{cases} \infty, & \text{if } e \in edgesPreprocessing1 \cup TabuList \\ c_e, & \text{if } e \in \mathscr{C} \\ 0, & \text{of } e \in R \cup Q \cup edgesDominance2 \\ -\varphi_e, & \text{otherwise (the edge is type } P). \end{cases} \tag{2}$$

Once generated the set of neighboring solutions, we want get a solution which will be the best current solution, in other words the best solution of iteration that is running. The function 4 OBTENERSOLUCION choose a solution, probabilistically,

Algorithm 4: Functions used by the tabu search for the PRPP

func `obtainNeighbors`(G : *A graph*, CurrentSol: *Sol. PRPP*, BestSolution: *Sol. PRPP*, TabuList: *Set of edges*) $\longrightarrow$ Neighbors: *Set of Sol. PRPP*

begin

foreach *edge* $e \in$ CurrentSol **do**

Let $e = (l, m)$ be a edge and the cycle CurrentSol is represented as a sequence of edges which are traversed the following form:
CurrentSol
$= ((d, a)(a, b), \ldots, (i, j), \ldots, (l, m), \ldots, (p, q), \ldots, (s, t), (t, d))$;

From the edge $e = (l, m)$ the solution CurrentSol is traversed to the left to find a edge $(i, j) \in R \cup Q$. If no such edge then we take the more extreme edge of the solution, This is one edge (d, a) that has the deposit;

From the edge $e = (l, m)$ the solution CurrentSol is traversed to the right to find a edge $(p, q) \in R \cup Q$. If no such edge then we take the more extreme edge of the solution, This is one edge (t, d) that has the deposit;

Find the minimum cost path MCP_{jp} between the vertices j and p;

Obtain a new feasible solution to the PRPP by replacing the path $[j, l](l, m)[m, p]$ of CurrentSol by the minimum cost path MCP_{jp} in CurrentSol. We obtain a solution with the following form:
NewSolution $\leftarrow ((d, a)(a, b), \ldots, (i, j), MCP_{jp}, (p, q), \ldots, (s, t), (t, d))$;

if $e \in$ TabuList **then**

`// Aspiration Criterion`

if $beneficio$(NewSolution) $>$ `profit`(BestSolution) **then**

Neighbors = Neighbors $\cup$ {(NewSolution, e)};

else

Neighbors = Neighbors $\cup$ {(NewSolution, e)};

return Neighbors;

func `obtainSolution`(Neighbors: *Set of soluc.*, TabuList: *Set of edges*, BestSolution: *Sol. PRPP*) $\longrightarrow \mathscr{C}$: *Sol. PRPP*

begin

(ChosenSol, e) $\leftarrow$ find within Neighbors the tuple with the feasible solution with the greatest profit;
ChosenSol $\leftarrow$ Obtain of the tuple (ChosenSol, e) the solution of the PRPP;
if `profit`(ChosenSol) $\leq$ `profit`(BestSolution) **then**

(ChosenSol, e) $\leftarrow$ Choose a tuple of probabilistic manner, using the technique of *roulette wheel* in which is considered the profit of the solutions of the PRPP;
ChosenSol $\leftarrow$ Obtain of the tuple (ChosenSol, e) the solution of the PRPP;

$e \leftarrow$ Obtain of the tuple (ChosenSol, e) the edge e associated to the PRPP solution, e is the edge with which the solution ChosenSol en `obtainNeighbors` is built ;
TabuList $\leftarrow$ TabuList $\cup$ {e};
return ChosenSol;

of the set of neighboring solutions.

In the algorithm 5 shows the Tabu Search for the PRPP. At first, the algorithm generates two feasible solutions with the algorithms 1 and 3. The solution generated by the algorithm 3 *Successive Elimination of Connected Components* is the initial solution which will iterate in the Tabu Search. The solution of the algorithm 1 *Union of Connected Components* can be used later in the Tabu search in case we need to diversify the search. In line 6 begins the iteration of the algorithm. The maximum number of iterations, $MaxIter$, which were allowed was 120. Once obtained the set of neighboring solutions, they pass through *improvement algorithms of solutions*. These are three algorithms. The first eliminates the cycles of negative benefit of a solution. The second eliminates duplicate edges of the solutions. The third take each one of the pairs of edges of a solution and examines whether there is a lowest cost path between the vertices extremes on both edges. It is considered that two solutions are equal if they have the same profit. There is a multiset of solutions called $BeneficioMultiConjunto$, which has the frequency of occurrence of a solution during the search. In lines 12-19 we see that if a solution is presented three times the search is trying to change the search space, making an intensification of the search since the best solution found so far or performing a diversification since the solution of algorithm 1 *Union of Connected Components*. We remember that the solution of algorithm 1 has all the edges that give benefit when traversing them for the first time. In lines 20 and 21 edges are obtained that are associated with the properties of PRPP, which are known as *Dominance 2* and *Preprocessing 1*. The idea with the edges of EDGESDOMINANCE2 is to try to promote their inclusion in the feasible solution, because if one edge R belongs to the optimal solution is likely that some of adjacent edges also belong to the optimal solution. Due to this we see that the cost of these edges in the function 2 is 0. The purpose of having the edges of EDGESPREPROCESSING1 is try to avoid having to include these edges in the feasible solution, as shown in the function 2. In lines 26-33 is observed what makes the algorithm in case you have a number iterations without finding a better global solution. The first time there will be a diversification of the search in a completely different state space, which is obtained by continuing with a tabu search feasible solution generated randomly. In case of falling back into this situation will make a intensification of the search by choosing as current solution, the solution algorithm 3 *Successive Elimination of Connected Components*.

4 Experimental Results and Discussion

The algorithms were implemented in `C` and are publicly available on the `http://www.ldc.usb.ve/~gpalma`. All experimental tests were performed on a computer SUN ULTRA 10 modelo 440, with a processor UltraSPARC-IIi de 440 MHz with 1.0 GB DRAM.

So far the only instances of PRPP available are those formulated by Aráoz, Fernández and Meza [5]. They are a set of 118 instances that were generated from a set of instances, widely used, of the Rural Postman Problem (RPP). We will use

Algorithm 5: Tabu Search for the PRPP

Input: A graph G
Output: A cycle $\mathscr{C}$ feasible solution of the PRPP

```
1  begin
2      iterWithoutImprovement ← 10,   MultisetProfit ← ∅
3      SECCSol ← Apply the algorithm 3 with input G
4      CurrentSol ← SECCSol,   BestSolution ← CurrentSol
5      UCCSol ← Apply the algorithm 1 with input G and the edges of type P
       in G
6      while iter < MaxIter do
7          Neighbors ← getNeighbors(G, CurrentSol, BestSolution, TabuList)
8          Apply the improvements algorithms to all solutions of Neighbors
9          CurrentSol ← getSolution(Neighbors, TabuList, BestSolution)
10         PCS ← profit(CurrentSol)
11         MultisetProfit ← MultisetProfit ∪ {PCS}
12         if number of occurrences of PCS in MultisetProfit = 3 then
13             if useBestSolution then
14                 useBestSolution ← FALSE
15                 CurrentSol ← BestSolution
16             else
17                 useBestSolution ← TRUE
18                 CurrentSol ← UCCSol
19             Delete all occurrences of PCS in MultisetProfit
20         edgesDominance2 ← Get all the edges of type Q and P that are
           adjacent to the edges of type R of CurrentSol and not belonging to it
21         edgesPreprocessing1 ← Get all the edges γ(V_k) ∪ δ(V_k) that are part of
           CurrentSol (Preprocessing 1)
22         if profit(CurrentSol) > profit(BestSolution) then
23             BestSolution ← CurrentSol
24             iter ← iter − 10,   iterWithoutImprovement ←
               iterWithoutImprovement − 9
25         Remove one unit the number of iterations that should be the edges
           e ∈ TabuList in TabuList
26         iter ← iter+1,   iterWithoutImprovement ← iterWithoutImprovement+1
27         if iter = iterWithoutImprovement then
28             if UseRandomSol = TRUE then
29                 CurrentSol ← Get a randomly generated feasible solution
30                 UseRandomSol ← TRUE
31             else
32                 CurrentSol ← SECCSol
33                 UseRandomSol ← FALSE
34             MultisetProfit ← ∅,   TabuList ← ∅
35             iterWithoutImprovement ← iterWithoutImprovement + 9
36     return BestSolution
```

these same instances to test the algorithms implemented. The 118 instances were grouped into 15 classes of problems, depending on their type and its number of vertices. The table 1 shows the characteristics of the problems to solve.

The results obtained with the Tabu Search Algorithm for the PRPP were compared with the results of the Heuristic of cutting plane algorithm proposed by Aráoz, Fernández and Meza [5], described in the introduction. All the results of both algorithms were obtained using the same computer SUN ULTRA 10.

Table 1
Summary of the characteristics of the problems to be solved. $|R|$ and $|P|$ are the number of edges type R and type P for each type of problem

Problem	#instances	$\|V\|$	$\|E\|$	$\|R\|$	$\|P\|$
AA	1	102	5151	160	10
AB	1	90	4005	144	11
P	24	7-50	21-1225	13-184	2-8
D16	9	16	120	31-32	2-5
D36	9	36	630	72	4-11
D64	9	64	2016	128	5-15
D100	9	100	4950	200	9-22
G16	9	16	120	24	3-5
G36	9	36	630	60	5-9
G64	9	64	2016	112	4-14
G100	9	100	4950	180	4-20
R20	5	20	190	37-75	3-4
R30	5	30	435	70-111	4-6
R40	5	40	780	82-203	5-9
R50	5	50	1225	130-203	7-12

Due to the probabilistic elements of Tabu Search, it is possible to obtain solutions different when solving an instance. To test the robustness and reliability of the algorithm Tabu Search was performed 30 runs for each of the 118 instances and in the table 4 are reported the results of the mean values obtained and the best solutions found. When comparing the results obtained with the the heuristic can see that the percentage of total deviation of the heuristic (10.98) is greater than the percentage of total deviation from the mean value (10.22) and greater than the best value of the Tabu Search (2.85). This indicates that, in general, the Tabu Search get better solutions than the heuristic of [5]. If we observe the best solutions found by the Tabu Search algorithm we have to find the best solution in 13 of the 15 types

of problems. The heuristic of [5] , just find a better solution than the Tabu Search in the problems $G64$ and $G100$. The Tabu Search get the optimal solution in 12 of the 15 types of problems, which is very satisfactory. The computation time of the Tabu Search was higher in small instances and much lower in larger problems. This is because in the Tabu search generates two initial solutions, which is expensive in time. However we consider that the computational effort of the Tabu Search is good if we see time total spent by both heuristics.

Table 2
Results of Tabu Search and the Heuristic of [5] in the instances of the PRPP. Where V_o: optimal value, %dHeur: percentage deviation of the heuristic, %dMeanTS: percentage deviation of the mean value of the Tabu Search, %dBestTS: percentage deviation of the best value found by the Tabu Search, tHeur: time of the heuristic, tTS: average time of the tabu search

Problema	V_o	%dHeur	%dMeanTS	%dBestTS	tHeur (seg)	tTS (seg)
AA	6266	0.30	0.08	0.00	471.65	124.60
AB	4372	0.00	0.03	0.00	102.59	109.40
P	2567	1.25	0.08	0.00	97.13	240.31
D16	2076	0.00	0.01	0.00	3.71	7.09
D36	5162	2.23	0.01	0.00	210.12	66.47
D64	8843	0.44	0.07	0.00	509.85	310.94
D100	11646	2.00	0.36	0.05	6714.30	1385.33
G16	20	0.00	0.00	0.00	2.74	2.77
G36	116	0.00	0.69	0.00	249.65	20.61
G64	280	0.36	3.92	0.71	1094.70	75.80
G100	478	1.05	3.93	2.09	9876.65	182.19
R20	47402	0.84	0.00	0.00	1.55	2.71
R30	54551	0.00	0.54	0.00	10.03	14.32
R40	89208	2.23	0.37	0.00	10.18	34.58
R50	97935	0.28	0.13	0.00	81.83	63.66
Totales	-	10.98	10.22	2.85	19436.68	2640.78

5 Conclusions

Tabu Search Algorithm for PRPP get high quality solutions beating to the Heuristic of [5] in most of the problems studied. The strategies used in Tabu Search algorithm can be applied to other arc routing problems with profits and costs. One of the main contributions of this work is the mechanism of generation of neighboring

solutions in Tabu Search for the PRPP. This is a key process for the Tabu Search performance and is so general that can be applied to other arc routing problems with profits and costs. The computational effort of the tabu search algorithm is moderate. Among the future work is to apply the scheme of Tabu Search algorithm to other arc routing problems with profits, such as the *Clustered Prize-collecting Arc Routing Problem* (CPARP) and the *Undirected capacitated arc routing problems with profits* (UCARPP).

References

[1] Ahr, D. and G. Reinelt, *A tabu search algorithm for the min–max k-Chinese postman problem*, Computers & Operations Research **33** (2006), pp. 3403–3422.

[2] Amberg, A., W. Domschke and S. Voß, *Multiple center capacitated arc routing problems: A tabu search algorithm using capacitated trees*, European Journal of Operational Research **124** (2000), pp. 360–376.

[3] Aráoz, J., E. Fernández and C. Franquesa, *The clustered prize-collecting arc routing problem*, Transportation Science **47** (2009), pp. 287–300.

[4] Aráoz, J., E. Fernández and C. Franquesa, *The windy clustered prize-collecting arc routing problem.*, in: *Conference on System Modelling and Optimization*, IFIP TC7, 2009.

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

[6] Aráoz, J., E. Fernández and C. Zoltan, *Privatized rural postman problems*, Computers & Operations Research **33** (2006), pp. 3432–3449.

[7] Archetti, C., D. Feillet, A. Hertz and M. Speranza, *The Capacitated Arc Routing Problem with Profits C. Archetti, D. Feillet*, Technical report, GERAD (2009).

[8] Archetti, C., D. Feillet, A. Hertz and M. Speranza, *The undirected capacitated arc routing problem with profits*, Computers and Operations Research (2009).

[9] Aráos, J., E. Fernández, C. Franquesa and O. Meza, *Prize-collecting arc routing problems and extensions*, in: *Odysseus Conference*, Valencia, España, 2006.

[10] Brandão, J. and R. Eglese, *A deterministic tabu search algorithm for the capacitated arc routing problem*, Computers & Operations Research **35** (2008), pp. 1112–1126.

[11] Corberán, A., R. Martí, E. Martínez and D. Soler, *The Rural Postman Problem on mixed graphs with turn penalties*, Computers & Operations Research **29** (2002), pp. 887–903.

[12] Corberán, A., R. Martí and A. Romero, *Heuristics for the mixed rural postman problem*, Computers & Operations Research **27** (2000), pp. 183–203.

[13] Deitch, R. and S. Ladany, *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** (2000), pp. 69–77.

[14] Dror, M., "ARC Routing: Theory, Solutions and Applications," Kluwer Academic Publishers, 2000.

[15] Feillet, D., P. Dejax and M. Gendreau, *The Profitable Arc Tour Problem: Solution with a Branch-and-Price Algorithm*, Transportation Science **39** (2005), p. 539.

[16] Fernández, E., O. Meza, R. Garfinkel and M. Ortega, *On the Undirected Rural Postman Problem: Tight Bounds Based on a New Formulation*, Operations Research **51** (2003), pp. 281–291.

[17] Franquesa, C., "The Clustered Prize-collecting Arc Routing Problem," Ph.D. thesis, Universidad Politécnica de Cataluña (2008).

[18] Frederickson, G., *Approximation algorithms for some postman problems*, Journal of the ACM (JACM) **26** (1979), pp. 538–554.

[19] Ghiani, G., F. Guerriero, G. Laporte and R. Musmanno, *Tabu Search Heuristics for the Arc Routing Problem with Intermediate Facilities under Capacity and Length Restrictions*, Journal of Mathematical Modelling and Algorithms **3** (2004), pp. 209–223.

[20] Glover, F., *Future paths for integer programming and links to artificial intelligence*, Computers & Operations Research **13** (1986), pp. 533–549.

[21] Glover, F. and M. Laguna, "Tabu Search," Springer, 1997.

[22] Glover, F. and B. Melián-Batista, *Búsqueda tabú*, Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial **19** (2003), pp. 29–48.

[23] Hertz, A., G. Laporte and M. Mittaz, *A Tabu Search Heuristic for the Capacitated Arc Routing Problem*, Operations Research **48** (2000), pp. 129–135.

[24] Irnich, S., *New models and methods for arc routing*, in: C. Barnhart, U. Clausen, U. Lauther and R. H. Möhring, editors, *Models and Algorithms for Optimization in Logistics*, number 09261 in Dagstuhl Seminar Proceedings (2009).
URL http://drops.dagstuhl.de/opus/volltexte/2009/2163

[25] Malandraki, C. and M. Daskin, *The maximum benefit Chinese postman problem and the maximum benefit traveling salesman problem*, European Journal of Operational Research **65** (1993), pp. 218–234.

[26] Pearn, W. and K. Wang, *On the maximum benefit Chinese postman problem*, Omega **31** (2003), pp. 269–273.