SPLITTING PROCEDURE AND A RELAXED ITERATED LOCAL SEARCH FOR THE GENERALIZED VEHICLE ROUTING PROBLEM

Christian Prins, Andréa Cynthia Santos, H. Murat Afsar
ICD-LOS, UMR CNRS STMR, Université de Technologie de Troyes
12, rue Marie Curie, BP 2060, 10010, Troyes, France
{christian.prins, andrea.duhamel, murat.afsar}@utt.fr

ABSTRACT

The Generalized Vehicle Routing Problem (GVRP) deals with a set of clients grouped into clusters. The problem consists in minimizing the total cost for a set of routes, such that each cluster is visited exactly once and its total demand is delivered to one of its nodes. The GVRP extends several classical vehicle routing problems and has interesting applications like humanitarian logistics. In this work, the split procedure is further detailed for the GVRP and an Iterated Local Search based metaheuristic are proposed. A new benchmark is suggested in this work and results are reported for four test sets.


Main areas: Vehicle routing, Metaheuristic.
1 Introduction

Let $G = (V, E)$ be an undirected weighted graph with a set $V$ of vertices and a set $E$ of edges. $V$ is the union of $m$ clusters $C_1 \cup C_2 \cup \ldots \cup C_m = V$ and $C_i \cap C_j = \emptyset \ \forall i, j \leq m \ ; \ i \neq j$. A demand $d_i$ is associated with each demand node $i$. The Generalized Vehicle Routing Problem (GVRP) consists in defining a set of routes with minimum total distance such that each cluster is visited once, and its total demand is delivered to one of its nodes. The total demand does not exceed the vehicle capacity $Q$ and the fleet of vehicles is homogeneous and unlimited.

The GVRP is NP-hard since it extends two well known NP-hard problems, the Generalized Traveling Salesman Problem (GTSP) where capacity constraints are relaxed and the Capacitated Vehicle Routing Problem (CVRP) which clusters contain only one customer. It models real-world applications as for transport logistics: distribution of supplies (food, water, medicines, etc) in natural and human disasters, distribution of goods by sea for potential harbors and postbox collection with several vehicles, Kara and Bektas (2003); Afsar et al. (2012).

Ghiani and Improta (2000) introduce the GVRP and propose a transformation to the Capacitated Arc Routing Problem. The authors solve an instance with 50 nodes, 24 clusters and one artificial cluster containing the depot. Baldacci et al. (2010) reduce the GVRP into a number of classical problems including several VRP extensions and the Traveling Salesman with profits. Integer linear formulations are proposed by Kara and Bektas (2003) for the GVRP as well as for the Generalized Traveling Salesman Problem (GTSP). Recently, Bektas et al. (2011) design a branch-and-cut procedure and a large neighbourhood search for the GVRP with limited fleet. Exact and heuristic methods for the GVRP are propose by Afsar et al. (2012). Furthermore, different versions of the GVRP have been treated by Moccia et al. (2011) and Pop et al. (2010). Moccia et al. (2011) develop a tabu search for the GVRP with time windows and limited fleet and test it on instances with and without time windows. The work has been motivated by an application for locating stops in school bus routes. Pop et al. (2010) propose a genetic algorithm in which the total demand of a cluster can be supplied by more than one vehicle.

In this work, the splitting procedure is further detailed for the GVRP. Moreover, the local search and the Iterated Local Search (ILS) based metaheuristic proposed by Afsar et al. (2012) have been improved. The classic 2-opt move have been added to the local search and the ILS has been modified by allowing to change the incumbent solution whenever it fills a threshold streaming. The paper is organized as follows: the splitting procedure is described in Section 2, followed by the metaheuristic strategy in Section 3. Computational experiments and concluding remarks are respectively given in Section 4 and Section 5.

2 Splitting procedure for the GVRP

A route-first approach is used and a non-trivial and polynomial splitting procedure for the GVRP, called Split has been developed. This procedure is used in a simple local search explained in section 2.1 and in the ILS metaheuristic presented in Section 3.

Conceptually, the principle is rather simple. Let $T = (T_1, T_2, \ldots, T_m)$ be an ordering (giant tour) of the $m$ clusters. We build a weighted auxiliary graph $H = (X, A, Z)$ with nodes 0 to $n$. Any subsequence of clusters $(T_i, T_{i+1}, \ldots, T_j)$ which can be visited by a feasible GVRP route (total demand compatible with vehicle capacity) is modelled by one arc $(i - 1, j)$ in $A$. The weight $z_{i-1,j}$ of this arc is computed using the method of Bontoux et al. (2010), i.e., we compute a shortest path in another auxiliary graph $F$ with $j - i + 3$ node layers: one layer reduced to the depot, $j - i + 1$ layers corresponding to the customers of clusters $T_i, T_{i+1}, \ldots, T_j$, and one final layer with the depot.

Figure 1 gives a small example for one giant tour $T$ with four clusters and vehicle capacity $Q = 10$. The left part shows the simplified network, with the arcs compatible with the given ordering and possible connections with the depot. The Bontoux et al. (2010) method is illustrated for subsequence $(T_2, T_3, T_4)$. The auxiliary graph $F$ is here the subgraph induced by the customers
of \( T_2, T_3 \) and \( T_4 \). The best route to visit these three clusters (bold segments) is \((0, 2, 3, 8, 0)\), with cost 17. The lower part illustrates the auxiliary graph \( H \), in which each possible route is represented by one arc weighted by the route cost. For instance, the best route computed for \((T_2, T_3, T_4)\) is represented by arc \((1, 4)\). The shortest path in \( H \) is composed of the two thick arcs, with cost 35. The right part of the figure shows the resulting GVRP solution with two trips.

The non-trivial task is to implement \textit{Split} with a low polynomial complexity. Algorithm 1 gives an efficient implementation in which no auxiliary graph is generated explicitly. It is based on two nested loops (lines 4 and 7) that browse all feasible subsequences of clusters \((T_i, T_{i+1}, \ldots, T_j)\), i.e., each arc \((i-1, j)\) of the implicit auxiliary graph \( H \).

For each subsequence of clusters, the total demand is computed in \textit{load} while the minimum cost of the route is calculated by a procedure \textit{Compute\_cost}. Label \( V_j \) represents the cost of a shortest path from node 0 to node \( j \) in the implicit auxiliary graph \( H \) (value on top of each node in Figure 1). Whenever a better path is found to reach node \( j \), \( V_j \) is updated, \( P_j \) stores the predecessor of \( j \) on this path, and a \textit{Store\_route} procedure records the last route of the path (route associated with the current subsequence). \textit{Compute\_cost}, \textit{Store\_route} and the extraction of the GVRP solution are explained in the sequel.

\textit{Compute\_cost} is described by Algorithm 2. The trick for a low complexity is to deduce the best route for the input subsequence of clusters \((T_i, T_{i+1}, \ldots, T_j)\) from intermediate results computed in the previous call for \((T_i, T_{i+1}, \ldots, T_{j-1})\). To do so, a label \( W_u \) is calculated for each customer contained in the clusters of the subsequence: if \( u \) belongs to cluster \( T_k \), \( W_u \) is the cost of a shortest path that leaves the depots, visits one customer in clusters \( T_i \) to \( T_{k-1} \) and ends at customer \( u \in T_k \). This path corresponds to a route, but without the return to the depot. The predecessor of \( u \) on this route is stored in \( B_u \). The vectors \( W \) and \( B \) are stored as global variables to be preserved between two calls.

For each node \( u \in T_j \), the best route to \( u \) is arc \((0, u)\) if \( j = i \) (line 3). Otherwise (lines 7-13), the procedure adds arc \((v, u)\) to the routes ending at each node \( v \in T_{j-1} \) and updates label \( W_u \) when improved. Lines 14-17 add arc \((u, 0)\) to return to the depot and memorize the best route as a pair \((\text{cost}, y)\), where \( y \) being the last customer. These results are necessary for tracing the route back.
Algorithm 1: General structure of Split for one giant tour $T$

```
1 $V(0) \leftarrow 0$;
2 $P(0) \leftarrow 0$;
3 for $i \leftarrow 1$ to $m$ do $V_i \leftarrow \infty$;
4 for $i \leftarrow 1$ to $m$ do
5     $j \leftarrow i$;
6     load $\leftarrow 0$;
7     repeat
8         load $\leftarrow$ load + $q(T(j))$;
9         if load $\leq Q$ then
10            Compute_cost ($T$, $i$, $j$, cost, $y$);
11            if $V(i-1) + cost < V(j)$ then
12                $V(j) \leftarrow V(i-1) + cost$;
13                $P(j) \leftarrow i-1$;
14                Store_route ($y$, $S$, $j$);
15            end
16         end
17     $j \leftarrow j + 1$;
18     until (load $> Q$) or ($j > m$);
19 end
20 return ($V_m$, $P$, $B$)
```

Algorithm 2: Algorithm for Compute_cost

```
1 cost $\leftarrow \infty$;
2 for each $u \in T(j)$ do
3     if $j = i$ then
4         $W_u \leftarrow c_{0u}$;
5         $B_u \leftarrow 0$;
6     else
7         $W_u \leftarrow \infty$;
8         for each $v \in T_{j-1}$ do
9             if $W_v + c_{vu} < W_u$ then
10                $W_u \leftarrow W_v + c_{vu}$;
11                $B_u \leftarrow v$;
12             end
13         end
14     if $W_u + c_{u0} < cost$ then
15         cost $\leftarrow W_u + c_{u0}$;
16         $y \leftarrow u$;
17     end
18 end
```
\textit{Store\_route} (Algorithm 3) records the best route for the subsequence of clusters \((T_i, T_{i+1}, \ldots, T_j)\). Using the last customer \(u\) and the predecessors stored in \(B\) (computed by \textit{Compute\_cost}), the route is traced backwards and its customers are stored in row \(j\) of a \(m \times m\) matrix \(S\) of customers.

\begin{verbatim}
1 \( u \leftarrow j - i + 2; \)
2 \( k \leftarrow y; \)
3 \textbf{repeat}
4 \( u \leftarrow u - 1; \)
5 \( S_{ju} \leftarrow k; \)
6 \( k \leftarrow B_k; \)
7 \textbf{until} \( k = 0; \)
\end{verbatim}

\textbf{Algorithm 3: Algorithm for \textit{Store\_route}}

Finally, using Algorithm 4, the GVRP solution can be extracted from the shortest path computed in the auxiliary graph \(H\). The algorithm begins at the last node (cluster) \(m\) and backtracks using the predecessors \(P_j\). The solution is encoded as a list containing the customers of the successive trips, separated by copies of the depot node. For each node \(j\) of \(H\), the trip corresponding to the arc ending at \(j\) on the shortest path has been stored in \(S\) by \textit{Store\_route}.

\begin{verbatim}
1 \( L \leftarrow \emptyset; \)
2 \( j \leftarrow m; \)
3 \textbf{repeat}
4 \( i \leftarrow P_j + 1; \)
5 \( \text{insert at the beginning of } L \text{ the route } (0, S_{j1}, S_{j2}, \ldots, S_{j,j-i+1}, 0); \)
6 \( j \leftarrow P_j; \)
7 \textbf{until} \( j = 0; \)
\end{verbatim}

\textbf{Algorithm 4: Algorithm to extract the GVRP solution}

To analyze the complexity of \textit{Split}, let \(d^+_i\) be the number of clusters in the longest feasible subsequence starting with cluster \(T_i\), i.e., the number of outgoing arcs or out-degree of node \(i\) in the auxiliary graph \(H\). The number of arcs in \(H\) is then \(|A| = \sum_{i=1}^m d^+_i\). There is one call per arc to \textit{Compute\_cost}, in \(O(|T_i|)\) if \(i = j\) and in \(O(|T_j||T_{j-1}|)\) if \(j > i\). Therefore, \textit{Split} runs in:

\[ O \left( \sum_{i=1}^m \left( |T_i| + \sum_{j=i+1}^{i+d^+_i-1} |T_j||T_{j-1}| \right) \right) \]  \hfill (1)

We get a simpler expression for balanced clusters with \(O(n/m)\) customers:

\[ \sum_{i=1}^m \left( |T_i| + \sum_{j=i+1}^{i+d^+_i-1} |T_j||T_{j-1}| \right) = \sum_{i=1}^m \left( \frac{n}{m} + \sum_{j=i+1}^{i+d^+_i-1} \frac{n^2}{m^2} \right) = O(n^2|A|/m^2) \]  \hfill (2)

If \(b\) is the average number of clusters per feasible route (average out-degree of the nodes in \(H\)), we have \(|A| = mb\) arcs in \(H\) and the complexity reduces to \(O(n^2b/m)\). Note that \(b\) can be approximated by \(\lceil Q/\bar{q} \rceil\), where \(\bar{q}\) is the average demand per cluster. If \(m = n\), each cluster contains one customer only and we find the \(O(nb)\) complexity.

Concluding, the split for the GVRP allows to define the optimal clusters order for a given giant tour \(T\), and also selects the clients to be visit in each clusters and builds optimal routes.
2.1 A simple local search based on Split

Preliminary tests have shown that Split is fast enough to solve by complete enumeration instances with up to 11 clusters and 50 customers, in less than 30 seconds on a 2 GHz PC: all permutations of the \(m\) clusters are enumerated and evaluated by Split, and the best GVRP solution is returned at the end.

This gave us the idea of a local search on the space of giant tours. The initial giant tour is computed by a nearest neighbor heuristic: starting from the depot, each iteration adds to the emerging route the closest customer contained in a cluster not yet visited. The resulting list of customers is replaced by the list of corresponding clusters to give the first giant tour \(T\).

Each iteration of the local search examines all ordered pairs of distinct clusters \((T_i, T_j)\) and evaluates the following moves. String relocations remove a string of one or two clusters starting with \(T_i\), to reinsert it after \(T_j\). These moves consider also relocations after the depot (before the first customer) and reinsertions of \((T_i, T_{i+1})\) as \((T_{i+1}, T_i)\). Swap moves exchange \(T_i\) and \(T_j\) if \(i < j\). 2-Opt moves invert the string \((T_i, T_{i+1}, \ldots, T_j)\) if \(i < j\). The \(O(n^2)\) resulting giant tours are evaluated by Split in \(O(n^2b/m)\). The first improving move, if any, is executed. The local search stops when no improvement can be found.

3 An Iterated Local Search based metaheuristic

Iterated local search (ILS) has been successfully applied to a number of optimization problems Lourenço et al. (2002), including vehicle routing problems Prins (2009); Nguyen et al. (2012). Starting from one initial local optimum \(S\), each iteration takes a copy of \(S\), applies a perturbation procedure and improves the perturbed copy using a local search procedure. \(S\) is updated if the resulting solution is better. We propose a Relaxed Iterated Local Search (R-ILS) for the GVRP. The general idea of the R-ILS is to allow changing the incumbent solution \(S\) at each iteration after the local search procedure, even if it is worse than the current solution. R-ILS has a dynamic threshold parameter to control the acceptance criteria, Tarantilis et al. (2004).

The R-ILS for the GVRP alternates between the giant tours and the space of GVRP solutions. The internal ILS begins with one pair \((S, T)\), where \(S\) is a GVRP solution computed via a heuristic and \(T\) the giant tour obtained by concatenating its routes. Then, each iteration performs four steps: I) a random perturbation is applied to a copy \(T'\) of \(T\); II) \(T'\) is converted into a GVRP solution \(S'\) via Split; III) \(S'\) is improved using a local search procedure; IV) if the cost of \(S'\) satisfies the acceptance criteria, it replaces \(S\) and its routes are concatenated to get a giant tour \(T\), giving the pair \((S, T)\) for the next iteration. The ILS is embedded in a main loop which restarts it from several initial solutions. The accepting procedure checks if a solution \(S'\) degrades the incumbent solution in up to a percentage defined by the threshold parameter. This parameter is reduced during the search process. It is set to 15% at the begging of the R-ILS iterations, and it is reduced in 5% at each 1/4 of the total number of iterations. The more R-ILS progresses, the harder the accepting procedure is. This non-standard ILS generates a trajectory in which the cost may temporarily increase, which allows to overcome trap local optima. Additional, after a 20 runs without improving the best solution value found, the R-ILS is restarted. All components are detailed below.

3.1 Initial giant tour

The initial giant tour of each ILS is built by a method inspired by the geometric sweep heuristic for the CVRP Laporte et al. (2000). The use of spatial (geographical) information is a tactical decision, particularly in the context of disaster logistics management. The heuristic initially computes the centroid for each cluster as follows: if \((x_i, y_i)\) denote the geographical coordinates of each node \(i \in V\setminus\{0\}\), the centroid coordinates \((\bar{x}_j, \bar{y}_j)\) of each cluster \(j = 1 \ldots m\) are given by Equations 3 and 4. Then, the polar angle of each centroid is computed, taking the depot as origin,
and the clusters are sorted in increasing order of these angles. Finally, the giant tour starts at the depot, visits the ordered clusters and returns to the depot.

\[ \bar{x}_j = \frac{\sum_{i \in C_j} x_i}{|C_j|} \quad \forall j = 1 \ldots m \]  

(3)

\[ \bar{y}_j = \frac{\sum_{i \in C_j} y_i}{|C_j|} \quad \forall j = 1 \ldots m \]  

(4)

### 3.2 Perturbation, restart and local search

The perturbation is applied to giant tours, not to GVRP solutions. It consists in exchanging two randomly selected clusters. Moreover, the restart procedure shifts the first position of the initial polar order.

Swap, relocation, and 2-opt moves are used in the local search procedure which is applied to each new GVRP solution. A swap is an exchange of two clusters, while a relocation removes one cluster to reinsert it in a different position. 2-opt moves invert the sequence of clusters \((E_1, E_{i+1}, \ldots, E_j)\) when applied to intra-routes. Routes \(j \leq 3\) rely on basic swap moves. Thus, intra-route 2-opt moves are performed for routes with more than three clusters. The clusters order is not inverted if the move occurs using different routes. The three moves are applied to one or two routes. Vehicle capacity must be checked for moves involving only inter-routes. 2-opt, relocation and swap moves are evaluated in this order. The cost variation of each move is quickly computed, assuming that the active nodes of involved clusters are preserved. As soon as an improving move is detected, it is accepted following a first-improvement strategy. A further improvement can be obtained by recomputing for each affected route the best active nodes in visited clusters, using the dynamic programming method developed by Bontoux et al. (2010) for the GTSP. This process is repeated until no improvement is possible.

Concerning complexity, the number of swap moves is \(O(m^2)\) since they are applied to each pair of effective clusters (the cluster containing the depot is not considered). There are also \(O(m^2)\) relocations, because the effective clusters can be inserted in \(O(m)\) different positions. The are also \(O(m^2)\) 2-opt moves. Each move can be evaluated in \(O(1)\). For a route containing \(k\) effective clusters \(C_1, C_2, \ldots, C_k\), the complexity of the dynamic programming procedure applied after each accepted move is linear in the number of arcs \(|C_1| + \sum_{i=1}^{k-1} |C_i| \cdot |C_{i+1}| + |C_k|\).

### 4 Computational experiments

The computational experiments were carried out on an Intel Core i3 Duo with 2.27 GHz clock and 4 GB of RAM. The R-ILS metaheuristic has been developed in ANSI C and tested over four test sets. The first test set (APS) has been introduced in Afsar et al. (2012) for which instances range from 12 clusters and 24 customers to 100 clusters and 504 customers. Moreover, it includes short trips (ST) (2 or 3 clusters per trip on average) and longer trips (LT) (5 to 7 clusters per trip). The second test set (P) contains instances used by Pop et al. (2011). As they handled the split GVRP, they selected small vehicle capacities to visit the same cluster using several routes. We kept the networks, demands and clusters but set vehicle capacity to the maximum cluster demand, multiplied by \(\alpha \in \{1.5, 2.0\}\). The third test set corresponds to only one instance from Ghiani and Imporota (2000). Finally, the fourth test set (S) reuses the networks and demands of three VRPTW instances with 101 nodes (c101, r101 and rc101) proposed by Solomon (1987). Time windows are dropped while clusters and vehicle capacities are determined as for set (P).

The best or the optimal values provided in Afsar et al. (2012) for the four test sets are used as reference to measure the improvements obtained with the R-ILS version. The number of R-ILS iterations is set to 1000 using the giant tour in polar order, and another set of 1000 R-ILS
iterations are performed by inverting the polar order. In this case, the threshold is reinitialized to 15%. Table 1 depicts results for Pop, Araque, Solomon and APS instances. The instance names are followed by the reference results. Lower bounds \((LB)\) and upper bounds \((UB)\) are those produced respectively by the column generation and the R-ILS found in Af
car et al. (2012). Optimal values are identified by symbol “*” in column \(UB\). For some instances, mostly with long tours, the linear program at the root of the B&P tree cannot be solved in one hour. Then, results for the simple local search procedure (S-LS) introduced in subsection 2.1 are given followed by the results for the R-ILS. \((UB)\), \((t(s))\) and \((Gap\%\) stands respectively fot the best solution value, the running time in seconds and the relative gap in percentage to the lower bound produced by the column generation strategy. Values in bold have been improved with the modification in the R-ILS, i.e, using the dynamic threshold and an improved local search.

<table>
<thead>
<tr>
<th>Instances</th>
<th>(m)</th>
<th>(Q)</th>
<th>(LB)</th>
<th>(UB)</th>
<th>(*LB)</th>
<th>(*UB)</th>
<th>(Gap%)</th>
<th>(UB)</th>
<th>(t(s))</th>
<th>(Gap%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>12-24-ST</td>
<td>12</td>
<td>200</td>
<td>504.37</td>
<td>*504.37</td>
<td>*504.37</td>
<td>0.00</td>
<td>*504.37</td>
<td>0.09</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>12-24-LT</td>
<td>12</td>
<td>200</td>
<td>354.64</td>
<td>*354.64</td>
<td>*354.64</td>
<td>0.00</td>
<td>*354.64</td>
<td>0.15</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>12-72-ST</td>
<td>12</td>
<td>200</td>
<td>506.08</td>
<td>*506.08</td>
<td>515.29</td>
<td>1.82</td>
<td>*506.08</td>
<td>0.11</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>12-72-LT</td>
<td>12</td>
<td>200</td>
<td>290.29</td>
<td>*290.29</td>
<td>294.00</td>
<td>1.28</td>
<td>*290.29</td>
<td>0.23</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>25-54-ST</td>
<td>25</td>
<td>200</td>
<td>1,006.29</td>
<td>*1,006.29</td>
<td>1,040.61</td>
<td>3.41</td>
<td>*1,006.29</td>
<td>0.48</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>25-54-LT</td>
<td>25</td>
<td>200</td>
<td>585.88</td>
<td>589.27</td>
<td>611.29</td>
<td>4.34</td>
<td>589.27</td>
<td>0.48</td>
<td>0.58</td>
<td></td>
</tr>
<tr>
<td>25-162-ST</td>
<td>25</td>
<td>200</td>
<td>858.75</td>
<td>*858.75</td>
<td>*858.75</td>
<td>0.00</td>
<td>*858.75</td>
<td>0.66</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>25-162-LT</td>
<td>25</td>
<td>200</td>
<td>560.88</td>
<td>581.28</td>
<td>581.28</td>
<td>3.64</td>
<td>581.28</td>
<td>0.85</td>
<td>3.64</td>
<td></td>
</tr>
<tr>
<td>50-121-ST</td>
<td>50</td>
<td>500</td>
<td>1,566.95</td>
<td>1,595.50</td>
<td>1,619.01</td>
<td>3.32</td>
<td>*1,575.98</td>
<td>2.25</td>
<td>0.58</td>
<td></td>
</tr>
<tr>
<td>50-121-LT</td>
<td>50</td>
<td>500</td>
<td>-</td>
<td>1,147.26</td>
<td>1,173.74</td>
<td>1.144</td>
<td>1,144.39</td>
<td>2.51</td>
<td></td>
<td></td>
</tr>
<tr>
<td>50-283-ST</td>
<td>50</td>
<td>500</td>
<td>1,655.23</td>
<td>1,668.27</td>
<td>1,665.58</td>
<td>0.63</td>
<td>*1,664.23</td>
<td>3.56</td>
<td></td>
<td></td>
</tr>
<tr>
<td>50-283-LT</td>
<td>50</td>
<td>500</td>
<td>-</td>
<td>1,120.69</td>
<td>1,166.12</td>
<td>1.130</td>
<td>1,130.35</td>
<td>2.45</td>
<td></td>
<td></td>
</tr>
<tr>
<td>75-150-ST</td>
<td>75</td>
<td>500</td>
<td>2,788.43</td>
<td>2,821.24</td>
<td>2,926.17</td>
<td>4.94</td>
<td>*2,803.68</td>
<td>0.48</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>75-150-LT</td>
<td>75</td>
<td>500</td>
<td>-</td>
<td>1,461.64</td>
<td>1,506.99</td>
<td>1,439</td>
<td>1,397.65</td>
<td>7.58</td>
<td></td>
<td></td>
</tr>
<tr>
<td>75-400-ST</td>
<td>75</td>
<td>500</td>
<td>2,643.42</td>
<td>2,666.54</td>
<td>2,771.34</td>
<td>4.84</td>
<td>2,660.20</td>
<td>8.78</td>
<td>0.63</td>
<td></td>
</tr>
<tr>
<td>75-400-LT</td>
<td>75</td>
<td>500</td>
<td>-</td>
<td>1,398.16</td>
<td>1,435.85</td>
<td>1,439.75</td>
<td>9.34</td>
<td>0.58</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100-216-ST</td>
<td>100</td>
<td>500</td>
<td>4,130.81</td>
<td>4,213.20</td>
<td>4,283.03</td>
<td>3.68</td>
<td>4,169.71</td>
<td>22.24</td>
<td>0.94</td>
<td></td>
</tr>
<tr>
<td>100-216-LT</td>
<td>100</td>
<td>500</td>
<td>-</td>
<td>1,958.01</td>
<td>2,024.64</td>
<td>1,941.64</td>
<td>16.67</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100-504-ST</td>
<td>100</td>
<td>500</td>
<td>3,661.55</td>
<td>3,711.60</td>
<td>3,759.81</td>
<td>2.68</td>
<td>3,720.39</td>
<td>17.20</td>
<td>1.61</td>
<td></td>
</tr>
<tr>
<td>100-504-LT</td>
<td>100</td>
<td>500</td>
<td>-</td>
<td>2,182.75</td>
<td>2,241.24</td>
<td>2,158.58</td>
<td>14.47</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.5-eil51</td>
<td>11</td>
<td>186</td>
<td>258.24</td>
<td>*258.24</td>
<td>-</td>
<td>-</td>
<td>*258.24</td>
<td>0.11</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>1.5-eilA76</td>
<td>16</td>
<td>267</td>
<td>376.82</td>
<td>*376.82</td>
<td>-</td>
<td>-</td>
<td>*376.82</td>
<td>0.23</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>1.5-eilA101</td>
<td>21</td>
<td>320</td>
<td>414.86</td>
<td>*414.86</td>
<td>-</td>
<td>-</td>
<td>*414.86</td>
<td>0.40</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>2-eil51</td>
<td>11</td>
<td>248</td>
<td>239.53</td>
<td>*239.53</td>
<td>-</td>
<td>-</td>
<td>*239.53</td>
<td>0.17</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>2-eilA76</td>
<td>16</td>
<td>356</td>
<td>329.31</td>
<td>*329.31</td>
<td>-</td>
<td>-</td>
<td>*329.31</td>
<td>0.28</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>2-eilA101</td>
<td>21</td>
<td>426</td>
<td>375.82</td>
<td>386.71</td>
<td>-</td>
<td>-</td>
<td>387.17</td>
<td>0.49</td>
<td>3.02</td>
<td></td>
</tr>
<tr>
<td>1.5-rc101</td>
<td>21</td>
<td>320</td>
<td>624.86</td>
<td>626.56</td>
<td>-</td>
<td>-</td>
<td>624.86</td>
<td>0.45</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>2-rc101</td>
<td>21</td>
<td>426</td>
<td>596.68</td>
<td>596.82</td>
<td>-</td>
<td>-</td>
<td>598.47</td>
<td>0.50</td>
<td>0.30</td>
<td></td>
</tr>
</tbody>
</table>

Tabela 1: Numerical results for APS, Pop, Araque and Solomon instances

In spite of the relative Gaps, S-LS remains a simple way to investigate the space of giant tours and to get some preliminary results. For these test sets, R-ILS respectively finds optimal solutions for 17 instances out of 33. For the others, when the column generation lower bound is available, solutions are obtained in less than 25 seconds and the gaps are within 4%. It is important to mention that for some instances like 25-162-LT the CG approach is not optimal, so the gaps...
computed using the CG lower bound overestimate the real gap. Moreover, R-ILS improves 12 solutions from the reference results highlighted in bold. APS instances are those more impacted by the new R-ILS version. This happen because results for instances from the other test sets was already of very good quality.

5 Concluding remarks and perspectives

In this paper, the split procedure is adapted and detailed for the GVRP and it has been integrated in an ILS-based metaheuristic. The R-ILS algorithm has been improved by using a threshold streaming and a 2-opt move into the local search. The proposed algorithm can be useful for a problem which has still a scarce literature, in spite of promising applications in disaster relief operations.

This work also brings important contributions in terms of heuristics. A tour splitting procedure has been designed for the GVRP and included in the metaheuristics. It differs of the split for classic VRP since two levels of decisions are needed: deciding the optimal clusters order and routes by selecting the clients to be visited in each clusters. It allows searching the space of giant tours instead of the wider set of GVRP solutions. The initial giant tours are built using a constructive heuristic based on geographical information.

The R-ILS allows to update the incumbent solution whenever it fills a threshold streaming. The R-ILS produces high-quality results in a few seconds. The performance of the R-ILS probably resides in the fact that accepted solutions contain optimal subsequences, since these are local optima relatively close to the incumbent solution (thanks to the small perturbation applied). This strategy avoids being trapped in a few attraction basins.

We are working to couple the metaheuristic approach with the exact method proposed in Afsar et al. (2012). Moreover, we intend as future work to address other objective functions to better model emergency and equity in disaster interventions, and also integrate stochastic issues to tackle uncertainty.

Referências


