A Multi-Start Simulated Annealing Algorithm for the Vehicle Routing Problem with Time Windows

Humberto César Brandão de Oliveira¹, Germano Crispim Vasconcelos² and Guilherme Bastos Alvarenga³

¹,²Center for Informatics, Federal University of Pernambuco - Brazil
³Department of Computer Science, Federal University of Lavras - Brazil
¹hcbo@cin.ufpe.br, ²gcv@cin.ufpe.br, ³guilherme@dcc.ufla.br

Abstract

Vehicle Routing Problems have been analyzed to reduce transportation costs of people and goods. More particularly, the Vehicle Routing Problem with Time Windows (VRPTW) imposes the period of time of customer availability as a constraint, a very common characteristic in real world picking up and delivery problems. Using minimization of the total distance as the main objective to be fulfilled, this work implements an efficient hybrid system which associates a non-monotonic Simulated Annealing technique to a Hill Climbing Strategy with Random Restart (Multi-Start). The algorithm performance is evaluated by comparing the results achieved with the best published works found in the literature of the 56 Solomon instances. The results outperformed or paired the individual best previous results in 36 out of the 56 instances.

1. Introduction

Many studies found in the literature of Vehicle Routing Problems (VRP) have contributed to practical advances in this field [22][24]. Since in real world many constraints and particularities have to be considered, some parameters have been studied as close as possible to the real case situations. Two of these constraints are the capacity of the vehicle and the time window in which the customers must be reached. This class of problems is known as the Vehicle Routing Problem with Time Windows (VRPTW) and, at the moment, it is the most popular class of the vehicle routing problem studied [19].

The costs related to people and goods transportation are very significant and are growing rapidly [1]. Studies suggest that from 10% to 15% of the final value of traded goods correspond to its transportation cost [7]. Part of these costs can be reduced with the treatment of diverse vehicle routing problems, where the VRPTW is a particular important case.

2. The Vehicle Routing Problem

The Vehicle Routing Problem with Time Windows (VRPTW) is a combinatorial optimization problem. It is a particular case of the Vehicle Routing Problem (VRP) [5]. In the VRP, the vehicle fleet must to visit and deliver a service to a set of customers. Every vehicle starts and finishes at a unique depot. For each pair of customers there is an associated cost. This cost denotes how expensive it is for a vehicle to move from a costumer to another, with the constraint that each customer must be visited exactly once. Additionally, each customer demands a specific number of goods (weight of the load). For each vehicle in a fleet, there is an upper limit of load capacity supported. In the basic case all the vehicles are of the same type and have the same capacity. Then, the objective of the VRP is to find a set of customers attended for each vehicle in order to minimize transportation costs [9].

In the VRPTW, each costumer has an associated time window that determines an interval within which a vehicle has to begin and finish the service to a specific customer. In the VRP as well as in the VRPTW several types of optimization objectives have been investigated in the literature. The total distance traveled is one of the most typical cost measures to be minimized and is the main target of the work here. De Backer et al. investigated iterative improvement techniques within a Constraint Programming (CP) framework [3]. The improvement techniques are coupled to Tabu Search (TS) and Guided Local Search (GLS) to avoid the search of being trapped into local minima. Riise and Stølevik [21] studied GLS and Fast Local Search (FLS) combined with simple move operators that relocate single tasks. Kilby et al. [20] introduced a deterministic GLS that use local search
operators (2-opt, relocate, exchange and 2-opt*) with a so-called best-acceptance strategy [23]. Alvarenga [1] studied the use of a Set Partitioning (SP) formulation after generating several solutions through a genetic algorithm (GA).

3. The Proposed Hybrid System (HS)

A strategy based on simulated annealing and hill climbing was first considered inspired on the capability of simulated annealing to both evolve solutions to a given problem and escape from local minima and on the capacity of hill climbing to refine initially defined solutions. To complete the method, a technique called ‘random restart’ [14] of the system is applied in order to cope with the idea of producing solutions to varied configurations of the VRPTW, returning the best solution from the executed restarts. Such strategy performs multiple system restarts with the association of simulated annealing and hill climbing and finds better results by diminishing the variance between the different executions of the system, and consequently enhancing the robustness of the method. The characteristics and definitions of this combined approach are described in details in the next sections. The parameter set used in this hybrid system was defined by manual tuning. As a future research, the parameter control will be adjusted in runtime, using techniques described in [6].

3.1. Simulated annealing (SA)

Simulated Annealing is a probabilistic metaheuristic algorithm [8] (being a local search method), that accepts search movements that temporarily produces degradations in a current solution found to a problem as a way to escape from local minima.

This meta-heuristic is based on a natural method which uses an analogy to the thermodynamics simulating the cooling of a set of heated atoms, in a operation known as Annealing [8].

The simulated annealing begins its search from a random initial solution. The iteration loop that characterizes the main procedure randomly generates in each iteration only one neighbor $s'$ of the current solution $s$. The variation $\Delta$ for the value of the objective function $f(x)$ is tested for each neighbor generation. To test this variation, $\Delta = f(s') - f(s)$, is computed.

If the value of $\Delta$ is less than $0$ (zero), then the new solution $s'$ is automatically accepted to replace $s$. Otherwise, accepting the new solution $s'$ will depend on the probability established by the Metropolis Criteria, which is given by: $e^{-\Delta/T}$, where $T$ is a temperature parameter, a key variable for the method.

The Metropolis Criteria accepts with higher probability solutions which have lower values of $\Delta$. Higher values of $\Delta$ will have lower chances if compared to lower values of $\Delta$. The higher the temperature, the higher is the probability of accepting the solution $s'$ as the new solution, explaining the algorithm analogy to the solid cooling.

3.1.1 Starting point for the simulated annealing. For defining an initial SA solution, this work used the algorithm known as Push-Forward Insertion Heuristic (PFIH) [15]. The PFIH has an efficient constructive strategy for calculating the cost of a new customer in a route [9]. This cost is computed according to its geographic position, the end of its time window and the angle between it and the central depot. Consider the Figure 1 as a current solution before the insertion of a customer C5.

![Figure 1. Current incomplete solution before the insertion of a new customer in a route](image1)

Besides the computation of the costs, the insertion of the customer is examined to guarantee that it does not violate any of the restrictions involved in the VRPTW. If any of the existing (from the smallest to the biggest cost) route solutions (Figure 2) does not violate any

![Figure 2. Feasible solutions found by the PFIH algorithm for the introduction of a new customer](image2)

Besides the computation of the costs, the insertion of the customer is examined to guarantee that it does not violate any of the restrictions involved in the VRPTW. If any of the existing (from the smallest to the biggest cost) route solutions (Figure 2) does not violate any
constraint of capacity, load of the vehicle or attendance
time to the customers, then this route becomes the
definite solution for inclusion of the new costumer (Figure 3).
Otherwise, the current routes are discarded and a new route is created for representing the new costumer.

Figure 3. Solution selected when inserting the customer C5

The order in which the customers are inserted into
the VRPTW solution directly defines the quality of the
final solution. Keeping that in mind, Solomon
developed in his work [15] a heuristic to determine the
order in which the customers should be considered into
the solution, according to the cost Equation:

\[ C_i = -\alpha d_{oi} + \beta t_i + \gamma \left( \frac{p_i}{360} \right) d_{oi} \] (1)

Where: \( \alpha = 0.7; \beta = 0.1; \gamma = 0.2; d_{oi} = \) the
distance between the central depot and customer \( i; t_i = \) the
upper limit of the time window for arrival of customer
\( i; p_i = \) polar coordinate angle of the customer \( i, \) with
respect to the central depot. The constant values for \( \alpha, \beta \) and \( \gamma \) were defined empirically in [15].

From the first chosen customer, the remaining
customers are tested one by one with respect to each
possible route solution for construction, according to
the Equation 1. The position and the customer that
resulted in the lowest increase in the total traveled
distance, without violating the time window capacity,
are chosen. After there are no more customers to insert
in the route under construction, this particular route is
closed and the same process is restarted again with a
new empty route, being the first customer the one with
the lowest cost according to Equation 1, among those
customers yet to be routed.

A variation in the original cost formula was
introduced, where, instead of being treated as constants
the \( \alpha, \beta \) and \( \gamma \) elements are turned into PFIH
parameters. The new values change at each execution
being captured by a normal distribution \( N(\mu, \sigma), \) with
an average in the points \( (\alpha \mu = 0.7; \beta \mu = 0.1; \gamma \mu =
0.2), \) suggested as optimal by Solomon, and with a
deviation by the unit \( (\sigma = 1). \) With this variation of the
heuristic orders, the customers being inserted maintain
a good arrangement because the averages are centered
on the optimal values obtained empirically and the
small variations cause perturbations that create
different initial solutions for the SA method. In resume,
this parametric variation on the cost function
(Equation 1) offers the possibility of the SA search to
start itself in different parts of the solution space.

3.1.2. Neighborhood operators applied to the
simulated annealing. Given a feasible point \( f \in F \) in a
particular problem, it is useful in many situations to
define a set \( N(f) \) of points that are ‘close’ in the same
sense to the point \( f \) [13], where \( F \) represents the set of
any solution that satisfies the problem. For example, if
\( F = R^n, \) then the set of points within a fixed Euclidean
distance provides a natural neighborhood solution for \( F \)
[13].

As this work focus its solutions on ordered lists
without repetitions (routes with its respective customers),
it is guaranteed, according to [6], that for
this type of representation four basic permutation
operators can describe a generic way to capture the
neighborhood of a solution \( f. \) These operators perform
permutations between the elements in order to capture
the neighborhood of a solution \( f \) and were originally
implemented as mutation operators in evolutionary
algorithms [6]. These have been adapted here to be
employed in the SA for the neighborhood definition.

The first neighborhood operator is called Swap
Mutation [6], \( swap(f), \) and can be described as:

\[ swap(f) = \{g : g \in F, \) obtained from \( f \) swapping 2
\]
customers \( (c_1, c_2) \) of any routes \( (r_1, r_2) \).\n
The second neighborhood operator is the Insert
Mutation, formally defined as:

\[ insert(f) = \{g : g \in F \) and \( g \) may be obtained by
removing one customer from any route of \( f \) and
inserting it again in any position of any route of \( f \}. \]

The third operator is the Scramble Mutation, which
is another random operator defined as:

\[ scramble(f) = \{g : g \in F \) and \( g \) may be obtained by
choosing any continuous sequence \( q \) of customers in a
route \( r \) chosen randomly from \( f \) and mixing the
customers of \( q \) in order to create a sequence \( q' \) which
will replace \( q \) in \( r \}. \]

The fourth operator, totally random, is based on the
customers’ inversion and may be explained as:

\[ inversion(f) = \{g : g \in F, \) obtained by choosing a
sequence \( s \) of customers in a route \( r, \) randomly chosen
from \( f \) and after inverting them systematically for
generating a new sequence \( s' \) that will replace \( s \) in \( r \}. \]

The fifth operator [12] is also a mutation operator
for evolutionary algorithms. Firstly, \( m \) customers are
withdrawn from each route of solution \( f. \) The number
of withdrawn customers varies for each route $r$ and is chosen by selecting a value from a uniform distribution that varies from 0 to the number of customers present in $r$. After selecting the customers withdrawn from $f$ (creating an incomplete solution $h$), all the selected customers are inserted back into $h$ through the PFIIH method, until a complete solution is found.

The $f'$ solution generated through the $f$ solution after the application of any neighborhood operator is accepted only if $f'$ satisfies every constraint of the VRPTW. In any case of constraint violation, the solution $f$ is kept into the SA for the next iteration of the system. This possibility may occur with the operators swap, insert, scramble and inversion.

For each iteration of the SA, an operator is chosen by withdrawing any positive number from a uniform distribution, which varies from 1 to $k$, being $k$ the number of operators of the system (in this work $k$ is equal to 5). The operators are stored in a vector of $k$ positions and the one which index is raffled is applied.

3.1.3. Temperature Control. The temperature $T$ used initially on the system was set to 100 and, on every 100 iterations, it was reduced to 95% of its current value: $T_{\text{new}} = 0.95 \times T_{\text{old}}$. In addition to the temperature decrease, the algorithm can also produce a temperature increase, characterizing the non-monotonic aspect of the $T$ variable. This increase in temperature (by the unit) occurs when there is no improvement in the best solution in the last 1000 iterations of the system. In the total, 30000 iterations were used for each execution of the simulated annealing.

3.2. Hill Climbing (HC) Strategy

After the conclusion of the SA, the hill climbing (HC) strategy takes places to compose the hybrid solution (HS). The motivation to introduce the HC strategy comes from the observation that the solution of the SA may be found in the system when its temperature is considerably high, and, in this case, the neighborhood close (which may contain a better solution) to the best solution of the SA will probably never be explored. This happens because, when the temperature is high, the Metropolis Criteria will tend to carry out a drastic locomotion in the solution space to look for the solution and worse solutions may be accepted by the method with high probability. Taking that into account, it was considered the application of the HC strategy to find a local minimum equal or better than the returned solution of the SA method. When the best solution of the SA is found under low temperatures, however, this local search (HC) turns itself to be unnecessary since it is basically executed by the SA method itself, considering that the Metropolis Criteria will only accept worse solutions with very low probabilities.

The HC process is then executed three times at the end of the SA method, each execution corresponding to 1000 iterations, to consider that different executions of a hill climbing may drive to different regions of the search space (solutions). This is the case because the implemented hill climbing is non deterministic and the execution with the best result is returned by the method. The neighborhood operators are the same as those described in Section 3.1.2.

3.3 Random Restart

With respect to random initialization, different works have followed different paths. In the work of [12] it was identified that short executions of the system, repeated many times, produced more robust results for the VRPTW. Since simulated annealing is also a stochastic algorithm, the quality of the final solutions over a number of runs shows a certain variance, and the same strategy of random was adopted to build the proposed HS (SA with HC). 30 system restarts were applied and the solution that presented the shorter total distance among all the system restarts was considered the final HS solution to the VRPTW.

4. Test Bed for Performance Evaluation

There are many publications using heuristics and meta-heuristics in the resolution of the VRPTW. For evaluating the quality and robustness of the different algorithms, these are frequently applied over the Solomon instances [15] (http://neo.lcc.uma.es/RADI-AEB/WebVRP/data/instances/solomon/solomon_100.zip). Likewise, the tests of this work were executed over the 56 Solomon instances with 100 customers each.

The Solomon instances are divided into six classes: R1, R2, C1, C2, RC1 and RC2. The R1 and R2 instances present customers with random Euclidean coordinates. Instances C1 and C2 present customers grouped in clusters. Instances RC1 and RC2 present a mix of the two characteristics (spared and clustered). In the R1, C1 and RC1 instances few customers have to be attended by each vehicle, introducing the need for more vehicles to attend all the demand. The types R2, C2 e RC2 present few vehicles in the solution, to attend a great number of customers in each route.
5. Experimental Results

5.1. Best known results

The best result found in the literature for each instance was compared with the best results of this work. They all consider the total traveled distance minimization as the main target.

For ease of visualization, Tables 1, 2, 3, 4, 5 and 6 point out if the hybrid system proposed in this work obtains equal or better results when compared to the best results found in the literature with respect to the total traveled distance minimization for the VRPTW. The instances marked with ** denote the situations where the best previous individual result for that instance (picked up from many different authors) were overcame by the proposed method, whereas those marked with * represent the cases where the results of the method paired the previous results.

<table>
<thead>
<tr>
<th>Instance</th>
<th>Vehicles</th>
<th>Distance</th>
<th>Work</th>
<th>Vehicles</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>** R101</td>
<td>20</td>
<td>1642.88</td>
<td>[1]</td>
<td>20</td>
<td>1642.88</td>
</tr>
<tr>
<td>R102</td>
<td>18</td>
<td>1472.62</td>
<td>[1]</td>
<td>18</td>
<td>1475.35</td>
</tr>
<tr>
<td>R103</td>
<td>14</td>
<td>1213.62</td>
<td>[17]</td>
<td>15</td>
<td>1222.68</td>
</tr>
<tr>
<td>R104</td>
<td>11</td>
<td>986.10</td>
<td>[1]</td>
<td>11</td>
<td>990.78</td>
</tr>
<tr>
<td>R105</td>
<td>15</td>
<td>1360.78</td>
<td>[1]</td>
<td>15</td>
<td>1363.74</td>
</tr>
<tr>
<td>R106</td>
<td>13</td>
<td>1241.52</td>
<td>[1]</td>
<td>13</td>
<td>1244.58</td>
</tr>
<tr>
<td>R107</td>
<td>11</td>
<td>1076.13</td>
<td>[1]</td>
<td>11</td>
<td>1081.88</td>
</tr>
<tr>
<td>R108</td>
<td>10</td>
<td>948.57</td>
<td>[1]</td>
<td>10</td>
<td>952.37</td>
</tr>
<tr>
<td>R109</td>
<td>13</td>
<td>1151.84</td>
<td>[1]</td>
<td>12</td>
<td>1153.89</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Instance</th>
<th>Vehicles</th>
<th>Distance</th>
<th>Work</th>
<th>Vehicles</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>** R201</td>
<td>5</td>
<td>1148.48</td>
<td>[1]</td>
<td>8</td>
<td>1147.80</td>
</tr>
<tr>
<td>** R202</td>
<td>9</td>
<td>1042.35</td>
<td>[12]</td>
<td>8</td>
<td>1039.32</td>
</tr>
<tr>
<td>** R203</td>
<td>5</td>
<td>876.94</td>
<td>[12]</td>
<td>6</td>
<td>874.87</td>
</tr>
<tr>
<td>** R204</td>
<td>4</td>
<td>736.66</td>
<td>[12]</td>
<td>5</td>
<td>735.80</td>
</tr>
<tr>
<td>** R205</td>
<td>5</td>
<td>960.07</td>
<td>[12]</td>
<td>5</td>
<td>954.16</td>
</tr>
<tr>
<td>** R206</td>
<td>4</td>
<td>887.90</td>
<td>[12]</td>
<td>5</td>
<td>884.25</td>
</tr>
<tr>
<td>** R207</td>
<td>4</td>
<td>811.93</td>
<td>[12]</td>
<td>4</td>
<td>797.99</td>
</tr>
<tr>
<td>** R208</td>
<td>3</td>
<td>707.01</td>
<td>[12]</td>
<td>4</td>
<td>705.62</td>
</tr>
<tr>
<td>** R209</td>
<td>5</td>
<td>860.11</td>
<td>[12]</td>
<td>5</td>
<td>860.11</td>
</tr>
<tr>
<td>** R210</td>
<td>6</td>
<td>912.48</td>
<td>[12]</td>
<td>5</td>
<td>910.98</td>
</tr>
<tr>
<td>** R211</td>
<td>4</td>
<td>761.75</td>
<td>[12]</td>
<td>4</td>
<td>755.82</td>
</tr>
</tbody>
</table>

5.2. Comparisons between different works

Table 7 compares in more details the best works ([20],[21],[3] and [1]) found to the algorithm proposed here. The columns represent the algorithm, whereas the lines show the average number of vehicles (NV) and the total traveled distance (TD) for each class. The last two lines represent the cumulative number of vehicles (CNV) and total distance (CTD) for each class of the Solomon’s benchmark, respectively over all 56 test problems.
Technical Information about work, hardware, runs, and average time to instance, are respectively as follows: [20] DEC Alpha, 3 runs, 48.3 minutes. [21] Pentium 200 MHz, 1 run, 29 minutes. [3] Hardware, number of runs and time not reported. [1] Pentium 4 2.4GHz, 3 runs, 60 minutes. [This work] Centrino 1.7 GHz, 15 runs, 11 minutes.

6. Final Remarks

This work presented a hybrid system that combines simulated annealing with non-monotonic temperature control, random start and a hill climbing strategy for the optimization of the total traveled distance of the VRPTW. The hybrid system offers a relevant contribution in the research of the best techniques to solve the VRPTW. Its main gain, in relation to the previous works, is a substantial improvement in solving the R2 problem classes (R2, C2 and RC2) of the Solomon’s benchmark. The solutions found by the HS for each instance contains few routes and many customers for the attendance. In the type 2 classes, this work obtained 17 new best results when compared to the ones found in the literature on minimization of the total traveled distance, and equaled other 10 best results (obtaining success in all but one instance (RC208)) out of the 28 tested. With respect to the classes C1, R1 and RC1, where the solutions for each instance contain many routes and few customers for attendance, this work only paired the best results of the C1 class and the instance R101, being inferior in 19 out of the 29 tested instances. A general comparison with other works that focus on the total traveled distance of the VRPTW ([20],[21],[3] and [1]) shows the superiority of this work in the R2 and RC2 classes, and a general advantage when considered the sum of the distances achieved for all the Solomon’s instances. These results encourage a further continuation of this research to refine the algorithm towards a performance improvement for the general classes of the VRPTW.

7. References