A HYBRID METAHEURISTIC FOR THE VEHICLE ROUTING PROBLEM WITH SIMULTANEOUS PICK-UP AND DELIVERY

Anand Subramanian (UFPB)
anandsubraman@hotmail.com
Lucídio dos Anjos Formiga Cabral (UFPB)
lucidio@de.ufpb.br
Gustavo Rezende Carvalho (UFPB)
grcjp@hotmail.com

This paper deals with the Vehicle Routing Problem with Simultaneous Pick-up and Delivery (VRPSPD). Due to the combinatorial nature of the problem, heuristics methods are commonly used to generate good quality solutions in an acceptable computational time. Therefore, a greedy randomized adaptive search procedure that uses a variable neighborhood descent method to perform the local search is proposed. The algorithm was tested in benchmark problems found in the literature related to the VRPSPD. The results appear to be competitive compared to the best known solutions.

Keywords: Vehicle Routing Problem with Simultaneous Pick-up and Delivery, GRASP, VND.
1. Introduction

The Vehicle Routing Problem with Pick-up and Delivery (VRPPD), i.e., where people or objects should be collected and distributed, constitutes an important category of the well-known Vehicle Routing Problem (VRP). In the late 80’s, Min (1989) proposed a new variation for the VRPPD where, in certain situations, the pick-up and delivery services must be carried out simultaneously for each customer, characterizing one of the most important variants of this class: the Vehicle Routing Problem with Simultaneous Pick-up and Delivery (VRPSPD).

Some applications of the VRPSPD can be observed in the beverage industry, where filled bottles are delivered while the empty ones are collected; in grocery stores, where pallets or containers are collected for re-use in merchandise transportation, etc. It is important to mention that some clients can demand that the pick-up and delivery services should not be carried out separately, since, in certain cases, this can result in additional costs and operational efforts for these customers.

Thus, one should consider not only the Distribution Logistics, but also the management of the reverse flow. It is in this context, that the concept of Reverse Logistics arises, which can be defined as the process of planning, implementing and controlling the return of raw materials, inventories under process, finished products and information related to the point of consumption until the point of origin. Therefore, the Distribution Logistic and Reverse Logistic should act together with an aim to guarantee the synchronization between the pick-up and delivery operations, as well as their impact on the company’s supply chain, resulting in the customer’s satisfaction and minimization of the operational efforts.

However, this is not a simple task, since the VRPSPD is NP-hard (DETHLOFF, 2001), and the determination of the optimum solution, by means of an exact method, in an acceptable time, is almost impossible. Due to combinatorial nature of the problem, heuristics techniques have been often applied in order to obtain good quality solutions in an acceptable time. Hence, this paper proposes a greedy randomized adaptive search procedure (GRASP) which uses a variable neighborhood descent (VND) in the local search phase.

2. Literature Review

The VRPSPD was first proposed by Min (1989), where the author shows a real-life application through a case study carried over in public library’s distribution system. Since then, little work has been done related to this variant. Only a decade later, Salhi and Nagy (1999) suggested some insertion heuristics, also capable of solving the problem with multi-depots. Recently the same authors have developed another procedure (NAGY & SALHI, 2005) which involves solutions with certain degree of feasibility.

Dethloff (2001) treats VRPSPD under various aspects of reverse logistics and proposes a constructive procedure (insertion heuristic) based on the cheapest feasible insertion criteria, radial surcharge, and the residual capacity, where the last one is an adaptation of the load-base approach (CASCO et al., 1988). Vural (2003) makes use of two methods based on Genetics Algorithms and, as per our best knowledge, this was the first work where a metaheuristic was applied to solve the VRPSPD. Gokçê (2004) developed a four phase heuristics based on the Ant Colony metaheuristic.

Crispim and Brandão (2005) present a hybrid procedure where Tabu Search (TS) and VND
are combined. Montané and Galvão (2006) proposed a TS algorithm involving multiple neighborhood structures: reallocation, interchange (swap), crossover and 2-opt. The same metaheuristic was implemented by Gribkovskaia et al. (2006), for the case where only one vehicle is considered.

Chen and Wu (2006) developed an insertion heuristic to generate initial solutions and a local search procedure based on the record-to-record travel approximation and tabu lists. Bianchessi and Righini (2007) suggest some constructive algorithms and local search heuristics as well as a TS procedure that uses a variable neighborhood structure, in which the node-exchange-based and arc-exchange-based movements were combined.

Dell’Amico et al. (2005) make use of an exact approach based on the branch-and-price technique. The problem is treated in two different ways, both under the dynamic programming scope. The same technique is applied by Angelelli and Mansini (2003), where the authors consider the VRPSPD with time-windows constraints.

3. Proposed Algorithm

The algorithm developed has two main steps. The first one corresponds to the greedy randomized construction and the second one is related to improvement phase, where the variable descent method is used. The pseudocode is shown in Figure 1, where $s^*$ represents the best solution and $\gamma$ is a parameter described in detail in subsection 3.1.

```
Procedure GRASP/VND(Max_iterations, seed, $\alpha$, $\gamma$)
1. LoadData( );
2. for $k = 1,..., Max\_iterations$ do
3.    $s \leftarrow$ Greedy_Randomized_Construction(seed, $\alpha$, $\gamma$);
4.    $s \leftarrow$ VND(N($s$), $f(s)$, $r$, $s$);
5.    Update_Solution($s$, $s^*$);
6. end;
7. return $s^*$;
End GRASP/VND
```

Figure 1 – Procedure GRASP/VND

3.1 Constructive Procedure

The method used to construct an initial feasible solution has a structure analogous to the first phase of the GRASP metaheuristic (FEO & RESENDE, 1995). To start with, the number of routes ($v$) to be used to obtain the initial solution is determined. In this case, the least number of routes ($v_{min}$) necessary is considered. Next, all routes are filled with a client $e$ randomly selected from the candidate list (CL). Later, the clients belonging to the CL are evaluated according to the insertion criteria expressed by the equation (1).

$$ g(e^v) = (C_{ak} + C_{kj} - C_{ij}) - \gamma(C_{0k} + C_{k0}) $$

The first part of the expression (1) is related to the cheapest feasible insertion criteria, which consists of a greedy approach that takes into account the least additional cost regarding the insertion of the client $k$ between the clients $i$ and $j$ of the route $v$. Naturally, only the feasible insertions are admitted. The second part corresponds to a surcharge used to avoid late insertions of customers remotely located (CASCO et al., 1988). The distance from the depot and back is weighted by a factor $\gamma \in [0,1]$. 
The client $e$ is then added to a restricted candidate list (RCL) in case its cost $g(e)$ is lesser or equal then the value given by the expression $g^{\min} + \alpha(g^{\max} - g^{\min})$, with $\alpha \in [0,1]$. Following this, one client is randomly selected and added to the solution $s$. The constructive procedure ends when all clients have been added to the solution $s$.

\begin{center}
\textbf{Procedure} Greedy_Randomized_Construction(seed, $\alpha$, $\gamma$)

1. $s \leftarrow \emptyset$; 
2. Initialize Candidate List (CL); 
3. Consider $s = \{s^1, s^2, \ldots, s^{\min}\}$ the solution set composed by $v$ empty routes; 
4. $v \leftarrow 1$; 
5. \textbf{while} $v \leq v^{\min}$ \textbf{do} 
6. \hspace{1em} $s^v \leftarrow e \in \text{CL}$ randomly selected; 
7. \hspace{1em} Update CL; 
8. \hspace{1em} $v \leftarrow v + 1$; 
9. \textbf{end while}; 
10. \textbf{while} CL $\neq \emptyset$ \textbf{do} 
11. \hspace{1em} Evaluate the value of each cost $g(e)$ for $e \in$ CL; 
12. \hspace{1em} $g^{\min} \leftarrow \min\{g(e) \mid e \in \text{LC}\}$; 
13. \hspace{1em} $g^{\max} \leftarrow \max\{g(e) \mid e \in \text{LC}\}$; 
14. \hspace{1em} RCL $\leftarrow \{e \in \text{LC} \mid g(e) \leq g^{\min} + \alpha(g^{\max} - g^{\min})\}$; 
15. \hspace{1em} Randomly select an element $n \in$ RCL; 
16. \hspace{1em} $s \leftarrow s \cup \{n\}$; 
17. \hspace{1em} Update CL; 
18. \textbf{end while}; 
19. \textbf{return} $s$; 
\textbf{End} Greedy_Randomized_Construction
\end{center}

Figure 2 – Procedure Greedy_Randomized_Construction

\subsection*{3.2 Local Search}

The local search phase, responsible to improve the initial solution is performed by a heuristic based on the VND algorithm (Figure 3). Mladenović and Hansen (1997) proposed the variable neighborhood descent method which systematically modifies the neighborhood structures that belong to a set $N = \{N^{(1)}, N^{(2)}, N^{(3)}, \ldots, N^{(r)}\}$, in a deterministic way.

\begin{center}
\textbf{Procedure} VND($N(.)$, $f(.)$, $r$, $s$)

1. Consider $r$ the number of different neighborhood structures; 
2. $k \leftarrow 1$; 
3. \textbf{while} $k \leq r$ \textbf{do} 
4. \hspace{1em} Find the best neighbor $s'$ of $s \in N^k(s)$; 
5. \hspace{1em} \textbf{if} $f(s') < f(s)$ 
6. \hspace{1em} \textbf{then} 
7. \hspace{2em} $s \leftarrow s'$; 
8. \hspace{2em} $k \leftarrow 1$; 
9. \hspace{1em} \textbf{else} 
10. \hspace{2em} $k \leftarrow k + 1$; 
11. \textbf{end if}; 
\end{center}
Four neighborhood structures, similar to the ones adopted by Montané and Galvão (2006), were considered. Three of them realize movements between routes (reallocation, swap and crossover), while one executes the movement inside its own route.

**Re allocation** \((N^{(1)})\) – In this case, one node is removed from one of the routes and inserted in another one, as shown in Figure 4. It can be observed that the node 7 was removed from one of the routes and added to the other.

**C lossover** \((N^{(2)})\) – This movement corresponds to the permutation of route’s fragments. The Figure 5 shows a situation where the path 2-3-7 is exchanged by the path 8-1.

**S wap** \((N^{(3)})\) – The movement performed in this neighborhood is associated to a node exchange. The Figure 6 exhibits an example where de nodes 4 and 7 are interchanged.
2-opt ($N^{(4)}$) – In this situation, two nodes, belonging to the same route are permuted. Figure 7 illustrates an example, in which the nodes 2 and 3 have their positions exchanged.

### 4. Computational Results

The algorithm was implemented in C++ programming language, using the Borland C++ Builder 6.0 compiler and executed in a Notebook Intel Pentium Centrino 1.86Ghz with 1024 of RAM memory and operation system Windows XP – Home Edition. The procedure was tested in benchmark problems found in the literature related to the VRPSPD. A comparison was made with the best known results (MONTANÉ & GALVÃO, 2006).

The number of iterations ($Max_{\text{iterations}}$) performed for all cenaries was 150. Ten executions were realized for each one of the different parameterizations of $\alpha$ and $\gamma$. In all problems, the least number of necessary vehicles were used. In order to show the influence of the variation of these parameters, a graph was mounted (Figure 8) in which $\gamma$ was considered to be 0.8 and 1.0 for a constant value of $\alpha = 0.4$. 
From Figure 9, it can be verified that the most of the solutions obtained for $\gamma = 1.0$ are slightly better in quality than the ones found using $\gamma = 0.8$, as they show lesser gap in comparison with the results of Montané and Galvão (2006).

Table 1 shows the best results found by GRASP/VND in the instances generated by Dethloff (2001). The number of clients in all problems is 50.

![Graph](image-url)
<table>
<thead>
<tr>
<th>Problem</th>
<th>Nº of clients / vehicles</th>
<th>Montané and Galvão</th>
<th>GRASP/VND Best</th>
<th>Gap in %</th>
<th>GRASP/VND Avg. Cost</th>
<th>Std. Dev.</th>
<th>α</th>
<th>γ</th>
<th>Avg. Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMT1X</td>
<td>50 / 3</td>
<td>472</td>
<td>473.84</td>
<td>0.39</td>
<td>488.26</td>
<td>9.41</td>
<td>0.2</td>
<td>0.7</td>
<td>1.96</td>
</tr>
<tr>
<td>CMT1Y</td>
<td>50 / 3</td>
<td>470</td>
<td>469.90</td>
<td>-0.02</td>
<td>487.10</td>
<td>10.59</td>
<td>0.4</td>
<td>0.8</td>
<td>3.47</td>
</tr>
<tr>
<td>CMT2X</td>
<td>75 / 7</td>
<td>695</td>
<td>711.41</td>
<td>2.36</td>
<td>731.84</td>
<td>10.55</td>
<td>0.3</td>
<td>0.9</td>
<td>6.09</td>
</tr>
<tr>
<td>CMT2Y</td>
<td>75 / 7</td>
<td>700</td>
<td>699.16</td>
<td>-0.12</td>
<td>720.54</td>
<td>9.98</td>
<td>0.2</td>
<td>1.0</td>
<td>5.87</td>
</tr>
<tr>
<td>CMT3X</td>
<td>100 / 5</td>
<td>721</td>
<td>739.35</td>
<td>2.55</td>
<td>769.20</td>
<td>12.00</td>
<td>0.2</td>
<td>1.0</td>
<td>17.30</td>
</tr>
<tr>
<td>CMT3Y</td>
<td>100 / 5</td>
<td>719</td>
<td>745.07</td>
<td>3.63</td>
<td>775.91</td>
<td>13.27</td>
<td>0.5</td>
<td>1.0</td>
<td>16.87</td>
</tr>
<tr>
<td>CMT12X</td>
<td>100 / 6</td>
<td>675</td>
<td>686.81</td>
<td>1.75</td>
<td>703.20</td>
<td>14.29</td>
<td>0.2</td>
<td>1.0</td>
<td>15.98</td>
</tr>
<tr>
<td>CMT12Y</td>
<td>100 / 5</td>
<td>689*</td>
<td>687.93</td>
<td>-0.16</td>
<td>721.66</td>
<td>35.54</td>
<td>0.1</td>
<td>0.9</td>
<td>16.51</td>
</tr>
<tr>
<td>CMT11X</td>
<td>120 / 4</td>
<td>900</td>
<td>902.61</td>
<td>0.29</td>
<td>984.28</td>
<td>54.30</td>
<td>0.3</td>
<td>0.9</td>
<td>35.56</td>
</tr>
<tr>
<td>CMT11Y</td>
<td>120 / 4</td>
<td>910*</td>
<td>922.01</td>
<td>1.32</td>
<td>996.11</td>
<td>43.83</td>
<td>0.3</td>
<td>0.8</td>
<td>30.95</td>
</tr>
<tr>
<td>CMT4X</td>
<td>150 / 7</td>
<td>880</td>
<td>919.10</td>
<td>4.44</td>
<td>980.04</td>
<td>26.81</td>
<td>0.1</td>
<td>1.0</td>
<td>59.20</td>
</tr>
<tr>
<td>CMT4Y</td>
<td>150 / 7</td>
<td>878</td>
<td>920.10</td>
<td>4.79</td>
<td>970.10</td>
<td>22.46</td>
<td>0.4</td>
<td>0.9</td>
<td>61.84</td>
</tr>
<tr>
<td>CMT5X</td>
<td>199 / 11</td>
<td>1098</td>
<td>1123.90</td>
<td>2.36</td>
<td>1183.59</td>
<td>35.20</td>
<td>0.2</td>
<td>0.9</td>
<td>123.88</td>
</tr>
<tr>
<td>CMT5Y</td>
<td>199 / 10</td>
<td>1083</td>
<td>1140.65</td>
<td>5.32</td>
<td>1221.02</td>
<td>51.87</td>
<td>0.5</td>
<td>0.8</td>
<td>158.42</td>
</tr>
</tbody>
</table>

(*) The authors considered one extra vehicle.

Table 1 – Results obtained by GRASP/VND on Dethloff’s instances

The best results found by GRASP/VND in the instances generated by Salhi and Nagy (1999), are shown in Table 2 where only the ones that do not include the maximum distance constraint were considered.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Nº of clients / vehicles</th>
<th>Montané and Galvão</th>
<th>GRASP/VND Best</th>
<th>Gap in %</th>
<th>GRASP/VND Avg. Cost</th>
<th>Std. Dev.</th>
<th>α</th>
<th>γ</th>
<th>Avg. Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIN22</td>
<td>22 / 2</td>
<td>88*</td>
<td>88</td>
<td>0.0</td>
<td>90.1</td>
<td>1.66</td>
<td>0.6</td>
<td>0.1</td>
<td>0.34</td>
</tr>
</tbody>
</table>

(*) Optimum Solution.

Table 2 – Results obtained by GRASP/VND on Salhi and Nagy’s instances

Finally, Table 3 illustrates the results related to the real-life problem proposed by Min (1989).

<table>
<thead>
<tr>
<th>Problem</th>
<th>Nº of clients / vehicles</th>
<th>Montané and Galvão</th>
<th>GRASP/VND Best</th>
<th>Gap in %</th>
<th>GRASP/VND Avg. Cost</th>
<th>Std. Dev.</th>
<th>α</th>
<th>γ</th>
<th>Avg. Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIN22</td>
<td>22 / 2</td>
<td>88*</td>
<td>88</td>
<td>0.0</td>
<td>90.1</td>
<td>1.66</td>
<td>0.6</td>
<td>0.1</td>
<td>0.34</td>
</tr>
</tbody>
</table>

From Table 1, it can be verified that the GRASP/VND procedure has yielded significantly better results than those found in the literature with an average improvement of 0.28%. It
should be emphasized that, in the calculation of the average gap, cases in which the number of vehicles involved was different were not considered. It can also be observed that among the 37 Dethloff’s problems, where a straightforward comparison can be made, 18 cases show improved results while in 9 other cases the results equaled those in the literature.

However, the proposed heuristic did not have the same robustness in the problems illustrated in Table 2, although the average gap was worse than the literature by only 2.52%. Again, this value was computed without considering the problems where the number of vehicles was different. It is also important to point out that, in both cases, the average gap was calculated taking into account the least cost found in each problem.

For Min’s problem, where the dimension is relatively small, the GRASP/VND procedure had achieved the optimal solution.

5. Conclusion

This paper presented a hybrid application of the GRASP and VND metaheuristics to the VRPSPD. The results found using this approach have proven to be competitive, particularly in the problems involving 50 clients (DETHLOFF, 2001), where an average gap of -0.28% was obtained, signifying an average gain in the quality of the set of solutions generated.

Although this research is still in its initial phase, it is sufficiently encouraging given that improvements have been achieved in the cases tested. Certainly, further improvements can be expected on using various neighborhood structures randomly, i.e., the effort in each neighborhood will follow a probability distribution. There is also a possibility of adopting new neighborhood structures, which can improve the quality of the solutions achieved.

References


GÖKÇE, E. I. A revised ant colony system approach to vehicle routing problems, Masters Dissertation, Graduate School of Engineering and Natural Sciences, Sabanci University, Turkey, 2004.


GRIBKOVSKAIA, I., HALSKAU, Ø., LAPORTE, G. & VLCEK, M. General solutions to the single vehicle


VURAL, A. V. *A GA based meta-heuristic for capacitated vehicle routing problem with simultaneous pick-up and deliveries*, Masters Dissertation, Graduate School of Engineering and Natural Sciences, Sabanci University, Turkey, 2003.