Solving Vehicle Routing Problem with Multiple Trips using Iterative Local Search with Variable Neighborhood Search

Belver, M.¹, Gomez, A.¹, Lopez, J.¹, De la Fuente, D.¹ and Ponte, B.²

¹Administracion de empresas. Campus de Viesques, University of Oviedo, 33204 Gijon, Spain.

Abstract - The aim of this paper is to describe the use of Iterate Local Search (ILS) for solving the Vehicle Routing Problem with Multiple Trips. The proposed algorithm uses Variable Neighborhood Search as local search for the best solution improvement. This local search is split in two consecutive VNS procedures in order to improve the solution in each step. For this purpose, the first VNS makes motions among customers located in different routes and the second one makes movements between customers in the same one. Different perturbations have been developed for the purpose of avoiding minimum locals. The proposed algorithm has been tested against the set of well-known problems and a brief comparison between our results and results from the literature is made in order to evaluate the quality of our algorithm.

Keywords: ILS, VRPMT, VRP, Logistic

1 Introduction

The classical Vehicle Routing Problem (VRP) has been deeply studied along the years due to the importance of reducing the costs of transporting manufactures between different points. Nowadays, is widely known that an important percentage of the final cost of a product is related to the movement, so trying to reduce this cost takes a big importance in a globalization world.

The goal of the classical VRP is, let a number of clients, create the minimum routes as possible and with the less cost, starting in the depot and finishing there. Usually to achieve this purpose is necessary to respect the limit capacity of each vehicle (CVRP) and this is one of the constraints in these kind of problems. One complete solution for the VRP is a result that visits all customers of the problem.

There are different versions for the classical VRP that incorporate their own particularities. For instance, in the VRPTW (VRP with Time Windows) the aim is attend to the customers inside a time strip. This problem has been one of the most studied in the literature along the time. The Vehicle Route Problem with Multiple Depots (MDVRP) is other common issue which incorporates multiple depots in the solution. Other variation of the VRP is the HFVRP (Heterogeneous Fleet VRP) where each vehicle has different characteristics like capacity or gas emissions. All these variations have the fact that the vehicle perform only one trip as a common constraint. This may not be true in cases where the float is limited since is not possible satisfy all demand with the available vehicles or because it produces a reduction in the trucks number used reducing therefore the operation costs.

In this paper we will focus on the VRPMT, which is a generalization of the CVRP where a vehicle can be used more than once. Typically, in the VRPMT the number of vehicles is limited by a number so in many cases some vehicles will have to cover more than one route. Since the VRP problem is a NP-Hard problem [1], all of its generalizations are NP-Hard. For its nature, problems NP-Hard involve a big computational complexity so it is important to develop heuristics that allow us to reach good solutions in a reasonable running time.

The next section provides a brief review of literature, followed by a presentation of the proposed algorithm, its features and particularities and how we adapt an ILS heuristic to the VRPMT in order to reach good solutions. Finally, we make different experiments, providing the correct parameter tuning, with the set of benchmark and we compare the results obtained against the published results from the literature.

2 State of the Art

The classical VRP and its different generalizations have been deeply studied in the literature. However, the possibility of considerer a limited number of vehicles has been less studied and there are only a few publications about it. The first work to focus in this feature was made by Salhi [2] who proposed a study case where the routes, restricted to double–trips, were assigned to vehicles using a matching algorithm within a refinement process. Fleischmann proposed a different view [3] about the approach trying to generate a solution in a one-phase algorithm using a greedy-type heuristic and assigning routes to vehicles through the Best Fit Decreasing (BFD) heuristic. Taillard et al [4] proposed a two phase approach. A set of VRP solution are built in the first phase.
using Tabu Search heuristic for making several routes and in the second phase the tours are assigned to vehicles through the mentioned BFD. Brandão and Mercer [5] introduced a novel Tabu Search for the multi-trip vehicle routing and scheduling problem (MTVRSP) considering different constraints as time windows and vehicle fleet mix. Later on, they modified this algorithm [6] in order to compare their results with Taillard et al. In this modification they propose an algorithm which is split into two phases. In the first, the algorithm gives a solution combining nearest neighbor and insertion concepts. In the last one, a two-phases tabu search is applied allowing unfeasible solutions regarding the time but not the load. For this purpose, they use a penalty function that punishes unfeasible solutions. Petch and Salhi [7] elaborated a multi-phase heuristic. In the first phase a set of CVRP solutions are constructed using a saving approach and later improving each solution with two-optimum and three-optimum procedures. Later on, each CVRP solution is transformed to a VRPMT solution through solving multiples bin-packing problems.

Olivera and Viera proposed an adaptive memory algorithm based on the Adaptive Memory Procedure [8] of Rochat and Taillard [9] but with the difference this algorithm allows work with non-feasible solution. This approach is based in the fact that worthy results may be constructed by the combination of different components of other good solutions. In each step the algorithm applies a local search procedure based on swap and insertion movements in order to improve the solution. Salhi and Petch [10] designed a hybrid Genetic Algorithm which uses a non-binary chromosome representation and genetic operators for adapting the VRPMT to a population heuristic.

The most recently works about VRPMT has been made by Cataruzza et al [11] and by Cheick et al [12]. Cataruzza suggested a population heuristic based on a Memetic Algorithm with two variations. We have used these results for establishing a comparison between Cataruzza’s results and the ones returned by our algorithm. Cheick et al proposed a variable neighborhood search for solving the VRPMT. In this work they used different neighborhood in the improvement phase but the quality of solutions they achieved are not good enough.

3 Formatting instructions

As we said before, VRPMT is a generalization of classical VRP with certain particularities. The VRPMT can be defined as follows. Let an undirected graph $G = (V, E)$ where $V = \{v_0, v_1, v_2, \ldots, v_n\}$ is the set of vertices representing the cities and $E \subseteq V \times V$ is the set of arcs. If $(i, j) \in E$ then it is possible to travel from city $i$ to city $j$ with a certain cost let by $c_{ij}$. A homogenous fleet of $m$ vehicles $K = \{k_1, k_2, \ldots, k_m\}$ is defined with a determined capacity $Q$ and is available in $v_0$ which represents the depot. Each route $r$ will be denoted as a sequence of nodes $r = (v_0, v_1, \ldots, v_{nr+1})$ where $v_0 = v_{nr+1} = 0$ and $(v_i, v_{i+1}) \in E \forall i \in [0, nr]$.

For each route $r_{nr}$ is the number of customers it visits during a tour and $d_{r} = \sum_{i=1}^{nr} q_{v_{i}}$ is the demand covered by the route, $c_{r} = \sum_{i=0}^{nr} c_{v_{i}+v_{i+1}}$ is the cost of the route and $t_{r} = \sum_{i=0}^{nr} t_{v_{i}+v_{i+1}}$ is the duration.

A solution is a set of routes $s = (R_1, \ldots, R_m)$ where $R_k$ represents the routes assigned to the vehicle $k$. With this definition we can define the cost of a solution as follows:

$$F_{R}(s) = \sum_{r \in R(s)} c_{r}$$

This mathematical model was proposed by [8]. A VRPMT solution is subject to a number of constraints:

- Each route starts and ends at the depot $v_0$.
- Each customer has to be visited exactly once by just one vehicle.
- The total demand of the customers in a route cannot exceed the capacity $Q$ of the vehicle.
- The total duration of a tour does not exceed the travelling time defined by $L$. In this part service times has to be taken into account.

The objective of the problem is minimizing function (1) taking into account the set of constraints defined above. According to the nature and the complexity of the problem in any moment we allow unfeasible solutions in order to explore a wider search space and to find a feasible solution. Unfeasible solutions will have to be penalized by means of the modification of function (1)

4 Iterated Local Search

The Iterated Local Search was proposed by Lourenço et al [13] and the main idea behind the algorithm is make a local search applying a perturbation in each iteration to allow that the algorithm can escape from local minima.

4.1 Initial solution

For building an initial solution the Nearest Neighbor algorithm is used on step 1. First a feasible CVRP solution is generated, considering that a non-optimal CVRP solution is built through this algorithm. After that, the BFD algorithm is used in step 2 to create a VRPMT solution from the CVRP solution. This heuristic assigns the routes to the $m$ vehicles and starts sorting all routes by length. The longest route is assigned to the vehicle with the less load. This process is done until all routes have been assigned. Finally, a VRPMT non-feasible solution is obtained. This VRPMT solution will be feasible according to the capacity of each route but, nevertheless, it will have overtime problems in some of the vehicles.
4.2 Local Search

We have implemented two different VNDs heuristics in the improvement phase (step 5 of algorithm 1). Both heuristics exploiting the neighborhood of the current solution doing movements between customers. The first VND (called GVND1) makes movements only between customers from different routes. The last VND (called GVND2) tries to optimize each route applying movements between customers from the same route.

The following neighborhoods are implemented in the group GVND1:

- **Shift (1,0)**. Selects a customer from one route and inserts it in another route.
- **Swap (1,1)**. Selects two customers from different routes and interchanges positions between them.
- **Insert new route**. Selects a customer from one route and create with it a new route in a different vehicle.
- **Swap (2,2)**. Selects two groups of two customers and interchanges their positions in a similar way that in a `Swap(1,1)`
- **Two_Optimum**. Removes two route connections and creates a new organization based on the best connection possible.

The group GVND2 also uses the `Swap(1,1)` and the Two_Optimum operators. Moreover, it implements a new neighborhood.

- **Shift(2,0)**. Select two customers position and relocate them in another position of the same route.

Only improvement movements are accepted. Each time current solution is improved the whole neighborhood is reset and the local search starts again. On step 4 and 7, the algorithm selects the neighborhood order randomly according to Coelho et al [14]. All the proposed algorithms are selected once.

For the purpose of reducing the number of movements in the local search a move is only examined if vj is one of the P closest customers to v, being P a parameter. It allows reduce the search space during the local search and the algorithm becomes faster.

4.3 Perturbations

Perturbations play an important role in the ILS algorithm because they make possible that the algorithm leaves the current local optimum and starts the search in another zone. The perturbation strength is crucial for this reason. Small changes could make the algorithm not able to escape from the local minimum and it may generate cycles during the search, wasting computational time. Otherwise large changes could lead the algorithm to lose the good properties of the local optimum. Three different perturbations are selected randomly in each iteration during the search

- **Insertion**. A random customer and his ∂-1 neighbor’s customers are selected and removed from the solution. Later, these customers are inserted again according to the less expensive position avoiding customers previous location.
- **Swap**. For each vehicle a random customer c1 is selected. Then another customer c2 is selected from the neighborhood of c1. The interchange is made between both customers. This perturbation is applied up to ∂ times.
- **Chains**. The perturbation selects two different routes and makes a change between them. The size of each chain is randomly selected in an interval of [2, Rsize] where R is the current route associate with each chain.

In perturbations a) and b) the value of parameter ∂ is critical as it determines the perturbation strength.

4.4 Intensification and Diversification

These are two important concepts in the control of the intensification and diversification during the search. Once the algorithm starts the current solution is explored in order to exploit the good characteristics of this solution. As the search advance it is common that after a number of iterations without improvements the current solution loses the good characteristics so we need to change the current solution. To achieve this after a certain number of iterations without improvements the best solution is perturbed and the algorithm continuous from there. The balance between intensification and diversification is made by the Ω parameter. A higher value allows a big diversification due to a wider search space is explored. On the other hand, smaller values focus on the intensification of the current space and the algorithm could not explore promising areas [15].

5 Experiments

The algorithm was tested over the benchmark formed by 104 problems. 92 of them were proposed by Taillard et al. [4] and the 12 remaining were proposed in [16]. The results have been compared with the results obtained for Cataruzza et al. [11] since these haven been the best one obtained recently. Cataruzza et al. proposed a hybrid genetic algorithm for solving the VRPMT problem. In addition to optimize the packing of routes they introduced a slight modification to the algorithm with the goal of introduce a combined local search to get better results. This local search was called CLS
(Combined Local Search) and this modification allowed them get best results over the whole benchmark but with greater times. We use both algorithms for testing the quality of our solutions.

### 5.1 VND and perturbation analysis.

To establish an impact about the use of different neighborhood in the local search and the number of perturbations used by our algorithm, we have run our algorithm against benchmark for comparative purpose. Each problem was run 5 times with different configurations. First, we have analyzed the VND neighborhood impact and the amount of movements done for each operator in our local search. Table 1 shows the percentage of changes done for each operator during the search.

Table 1. Neighborhood impact in local search.

<table>
<thead>
<tr>
<th>Neighbourhood heuristic</th>
<th>% Improvements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shift(1,0) – M1_0</td>
<td>32.46%</td>
</tr>
<tr>
<td>Insert New Route – MNR</td>
<td>0.72%</td>
</tr>
<tr>
<td>TwoOPT – MTO</td>
<td>24.3%</td>
</tr>
<tr>
<td>Swap(1,1) – M1_1</td>
<td>27.58%</td>
</tr>
<tr>
<td>Shift(2,0) – M2_0</td>
<td>9.16%</td>
</tr>
<tr>
<td>Swap(2,2) – M2_2</td>
<td>5.66%</td>
</tr>
</tbody>
</table>

Operators MNR and M2_2 have a small impact over the local search considering that they achieve a small percentage of changes in comparison with the rest of operators. On the other hand, the operators M1_0, MTO and M1_1 make lots of changes over the solution.

For the analysis of the perturbation impact over our algorithm we have ran our algorithm against the whole benchmark testing each perturbation separately taking the number of feasible solutions obtained and the total execution time. Time is expressed as the time average taken in solving the 104 instances.

Table 2. A comparison between the different perturbations. Time is expressed in seconds.

<table>
<thead>
<tr>
<th>Perturbation</th>
<th>#fs</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insertion</td>
<td>375/460</td>
<td>1145</td>
</tr>
<tr>
<td>Swap</td>
<td>382/460</td>
<td>1754</td>
</tr>
<tr>
<td>Chains</td>
<td>382/460</td>
<td>2485</td>
</tr>
<tr>
<td>All</td>
<td>393/460</td>
<td>1459</td>
</tr>
</tbody>
</table>

Using single perturbations, the execution time in some cases is slightly greater and the amount of feasible solutions obtained is lower. The mix of all perturbations returns the best results.

### 5.2 Results

This section presents the results obtained after run our algorithm against the benchmark previously mentioned. The main idea is compare the results obtained by our algorithm against the results obtained by Cataruzza et al. two proposals, the basic MA and the MA with Combined Local Search (CLS). We compare the amount of feasible solutions found by Cataruzza et al. with the number of feasible solutions that our algorithm is able to find during the search. Although our proposal achieves fewer solutions, it takes less time for solving the whole benchmark in some cases.

Table 3 shows for each problem the number of instances executed and the number of feasible solutions found by the approach proposed by Cataruzza and the number of possible results found by our algorithm. Time column is expressed in seconds and represents the average time for solving each problem.

We have grouped the instances with the purpose of displaying the evolution time when the size of the problem increases in order to make a comparison between the Cataruzza’s times and the times returned by our algorithm. Problems CMT1 and CMT12 have the same number of customers so they are represented together taking the average time shows in Table3.

Table 3. A brief comparison between the number of feasible solutions found by Cataruzza’s algorithms and the amount of feasible solutions found by our proposal.

<table>
<thead>
<tr>
<th>Instance</th>
<th>MA+CLS (Cataruzza)</th>
<th>ILSVND</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>#fs</td>
<td>%fs</td>
</tr>
<tr>
<td>CMT1</td>
<td>8</td>
<td>30</td>
</tr>
<tr>
<td>CMT2</td>
<td>14</td>
<td>63</td>
</tr>
<tr>
<td>CMT3</td>
<td>12</td>
<td>60</td>
</tr>
<tr>
<td>CMT4</td>
<td>16</td>
<td>74</td>
</tr>
<tr>
<td>CMT5</td>
<td>20</td>
<td>100</td>
</tr>
<tr>
<td>CMT11</td>
<td>10</td>
<td>48</td>
</tr>
<tr>
<td>CMT12</td>
<td>12</td>
<td>52</td>
</tr>
</tbody>
</table>

Using single perturbations, the execution time in some cases is slightly greater and the amount of feasible solutions obtained is lower. The mix of all perturbations returns the best results.

### 6 Conclusions

An Iterated Local Search based on Variable Neighborhood Search has been proposed to solving the VRPMT. In this local search the VNS is decomposed in two consecutive VNDs in order to improve the solution. Besides the concepts of intensification and diversification are introduced and it has been explained how the algorithm controls both terms with the goal of exploring a wider search space. Three different kinds of perturbations are proposed for allowing the algorithm to escape from local minimum.
Furthermore, we have analyzed the impact of the different neighborhood as well as the different perturbations used during the search. The use of less neighborhood structures lead us to get best execution times in exchange for reduce the number of feasible solutions found. Moreover, we have showed that the use of a mix of perturbations produce best results that the use of individual changes.

The proposed algorithm was tested against the benchmark and the solutions obtained were compared with the most significant previous studies done in the literature. Although we obtained fewer solutions in comparison with results from Cataruzza our algorithm gets acceptable solutions in reasonable execution time.

7 References


