Random Neighborhood Search for the Vehicle Routing Problem with Time Windows

Francesco Carrabs* Raffaele Cerulli† Monica Gentili*

*Computing Science Department
University of Salerno
P.te Don Melillo, 84084, Fisciano (Sa), Italy
{fcarrabs, mgentili}@unisa.it

†Computing and Mathematics Department
University of Salerno
P.te Don Melillo, 84084, Fisciano (Sa), Italy
raffaele@unisa.it

1 Introduction

The Vehicle Routing Problem with Time Windows (VRPTW) models the situation in which a fleet of vehicles must visit a collection of customers requiring a service. Each request consists in the specification of a quantity of goods to be either picked-up or delivered to a specified location during one or more time windows. Vehicles must be routed in order to service all requests, and, each route must satisfy both time windows and vehicle capacity constraints. The primary objective function is to minimize the total number of vehicles used, and, secondary to minimize the total travelled distance. The problem is known to be NP-complete, and, several efforts have been devoted to develop efficient local search heuristics to solve either the Pick-up and Delivery version or the single Delivery or Pick-up version.

Most of the recently published VRPTW heuristics use two-phase approaches. In the first phase, a construction heuristic is used to generate a feasible initial solution. During the second phase, an improvement heuristic is applied to the initial solution. These route-improvement methods modify the current solution iteratively by performing local searches for better neighboring solutions.

One of the most critical features of local search is the definition of (i) the neighborhood structure and (ii) the exploration strategy. For this problem many different neighborhoods have been considered and experimented. Many successful approaches existing in the literature use a multi neighborhood structure. In particular, in [1] a Reactive Variable Neighborhood Search (RVNS) has been recently proposed and it obtains very good results. It is an extension of the Variable Neighborhood Search (VNS) proposed in [6]. More in details, the classical

VNS explores increasingly different neighborhoods of the current solution jumping from one neighborhood to an other one when an improvement in the objective function is achieved. If exploring all the neighborhoods no improvement from current solution is attained the procedure stops. In the RVNS, once all the neighborhood are explored, a set of parameters limiting the search space is increased in order to perform a more thorough search.

In particular, such approaches that consider different neighborhood structures have two main elements to take care of: (i) the dimension of the search space (defined both by the number of the neighborhoods and by their dimension) and (ii) the policy of exploration (i.e., mainly it can be an exhaustive search or a random one). Obviously, there is a trade off between these choices. Indeed, when the dimension of the search space is not too large an exhaustive exploration can be performed; however, for larger instances we need a bigger search space that does not allow a complete search. The RVNS applies an exhaustive search to several neighborhood structures whose dimension is increased iteratively and compatibly with a reasonable running time.
We propose to use a multi-neighborhood structure, where, however, at each iteration a neighborhood is randomly selected (Random Neighborhood Search) to give the same selection chance to any point of the multi-neighborhood structure. Moreover, we propose to use a shaking phase based on RNS to enforce the route-improvement scheme, differently from other approaches that use a single neighborhood structure like ejection chains proposed firstly by Glover [5] and used, later, by Bräysy [1].

The sequel of the paper is organized as follows. Next section briefly describes our overall technique. Section 3 contains a description of some of the operators used to generate the multi-neighborhood. Section 4 reports a summary of our results obtained on Solomon's instances.

2 The Algorithm

Let $G = (V \cup \{v_0\}, E)$ be a connected digraph consisting of $|V \cup \{v_0\}| = n + 1$ vertices, where V is a set of customers, v_0 is a special vertex representing a central depot and E is a set of arcs to which a nonnegative weight, denoting the travel time, is associated. Each customer $i \in V$ requires a service that consists in the specification of a quantity q_i of goods to be picked-up or delivered to a specified vertex during one or more time windows. Vehicles of equal capacity C must be routed to service all the customers. A feasible vehicle route $R = \{v_0, i_1, i_2, \dots, i_{l-1}, i_l, v_0\}$ of length l is an ordered sequence of customers to be serviced such that the total capacity of the vehicle is not exceeded and all the time windows constraints are respected. A set $S = \{R_1, R_2, \dots, R_h\}$ of h feasible routes is feasible for the VRPTW if all the customers are serviced and each customer is visited by a single route. We look for a feasible set S of minimum size.

We construct an initial feasible solution S in the following way. We start with a set of n routes, that is a route r_i starts from the depot v_0, visits client v_i and comes back to the depot. In order to decrease the number of routes, we apply the following two intra-route exchange operators to each of the n route to obtain the initial feasible solution S:

Let S be an initial feasible solution; iter=1;
While iter < MaxIterations do
    let $N_1(S), N_2(S), \dots, N_h(S)$ be h different neighborhoods of S;
    Select i at random from 1 to h;
    Let $S' \in N_i(S)$ such that $d(S') \leq d(S)$;
    Let S'' the returned solution of Route-elimination phase applied on S';
    $S = S''$;
    If $\vert S''\vert < \vert S'\vert$ then iter =1;
    else iter=iter+1;
End while

Table 1: *Our heuristic algorithm.*

- Move a customer k from a route R to a route R';
- Move a customer k_1 from route R_1 into position of costumer k_2 in R_2 and move k_2 into one other route different from R_1 and R_2.

Starting from this initial feasible solution S, we define its multi-neighborhood, $N_1(S), \dots, N_h(S)$, by using different operators (some of these operators are described in the next section). We explore the solution space by a Random Neighborhood Search (RNS), that is, we select at random one of the h neighborhoods of S, say $N_i(S)$. Then, we select a new feasible solution $S' \in N_i(S)$ such that the total travelled distance, $d(S')$, is less than or equal to $d(S)$. The route elimination procedure is then applied to decrease the total number of routes in S' to obtain S''. We return to the initial step with the new solution S'' such that $|S''| \leq |S'|$ and repeat the procedure a predefined number of iterations. To deeply explore the search space we reset the number of iterations whenever $|S''| < |S'|$. More in details, in the route elimination procedure, a route $R_i \in S'$ is selected and we try to move one by one the customers served by R_i to any other route by using different exchange operations. Any time it is not possible to eliminate a customer from R_i, we shake $S' \backslash \{R_i\}$ by using RNS and obtain a different solution S_i that belongs to a random selected neighborhood of $S' \backslash \{R_i\}$. Then, we try to move the blocked customer of R_i to the routes of S_i; if this is not the case, then, the shaking phase is carried out again, until a predefined maximum number of iterations is reached. Table 1 gives the pseudo-code of our approach.

3 Operators

We used 8 different operators to define the multi-neighborhood of a feasible solution S. In particular we used 4 intra-route operators and 4 inter-route operators. Given a feasible solution S, an inter-route operator defines new solutions by changing the ordered sequence of customers of each route of S; while an inter-route operator exchanges customers among routes. For each of this operators we consider also a reverse version: instead of analyzing the route by considering the ordered sequence $(v_0, i_1, i_2, \dots, i_{l-1}, i_l, v_0)$, the reverse sequence $(v_0, v_l, v_{l-1}, \dots, v_2, v_1, v_0)$ is analyzed.

The intra-route operators we used are:

1. Before-k-Reposition: remove, from a route R, a customer k_1 in some position after

customer k, and, if the resulting route is feasible, put k_1 immediately before customer k.

2. After-k-Reposition: remove, from a route R, a customer k_1 in some position after customer k and, if the resulting route is feasible, put k_1 immediately after customer k.

3. 2-Exchange: swaps two customers k and k_1 that belong to the same route if a distance saving is obtained.

4. Or-opt: move from a route R a sequence of at most three adjacent customers and relocate them in a different position. In our implementation the sequence's length is 1.

The inter-route operators we used are:

1. Relocate: move a customer k from a route R in a route R' if this relocation is distance saving.

2. 2-Exchange: swaps customer k from route R with customer k_1 in route R'.

3. k-to-1 Exchange: swaps customer k from route R with two consecutive customers in route R' if this relocation is distance saving.

4. 1-to-1+1 Exchange: (our new operator for inter-route exchanges) replaces a customer k in a route R with two customers k_1 and k_2, that belong to two different route R' and R'', and tries either to insert k in the k_1 position either to insert k in the k_2 position.

4 Computational Results

We tested our RNS approach on Solomon's instances. These instances consist of six sets (R1, C1, RC1, R2, C2, RC2), each of which contains between eight and twelve 100-node problems over a service area defined on a $100x100$ grid for a total of 56 different instances. For R1 and R2, the customer locations are distributed uniformly over the service area. Sets C1 and C2 have clustered customers, and sets RC1 and RC2 have a combination of clustered and randomly located customers. In addition, R1, C1, and RC1 have tight time windows and a vehicle capacity of 200 units; R2, C2, and RC2 have a long scheduling horizon and vehicle capacity of 1000, 700, and 1000 units, respectively. The time window and the vehicle capacity constraints in problem sets R1, C1, and RC1 allow only a small number of customers to be served by each vehicle. The opposite is true for R2, C2, and RC2.
Our procedures was implemented in C language, the operating system was Suse linux 9.0. The computational experiments were carried out on Xeon 2.4 Ghz bi-processor with 1024 Mb of RAM.
Table 2 above gives our results obtained on all Solomon's instances. We reports the final solutions produced by the proposed RNS method compared with the results of the best metaheuristics proposed recently by other authors. The notation CNV indicates the Cumulative Number of Vehicles over all 56 test problems. The first column in the table contains the class of instances and all the other columns report the best known results in the literature. For each class and each author a couple of values is given: the bold number represents the average number of vehicles computed on all the instances of the class while the other is the

corresponding average total travelled distance. Note that none of the proposed methods in the literature dominates the others in all the classes and we are competitive with the best results of the best methods.

Table 2: *Comparison of our methodology with existing approaches on Solomon's instances*

PROB	TBGGP	CR	LS	GTA	HG	RGP	CLM	RNVS	RNS
R1	**12.17**	**12.17**	**12.17**	**12.00**	**11.92**	**12.08**	**12.08**	**11.92**	**12.17**
	1209.35	1204.19	1249.57	1217.73	1228.06	1210.21	1210.14	1222.12	1253.85
R2	**2.82**	**2.73**	**2.82**	**2.73**	**2.73**	**3.00**	**2.73**	**2.73**	**2.67**
	980.27	986.32	1016.58	967.75	969.95	941.08	969.57	975.12	865.11
C1	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**
	828.38	828.38	830.06	828.38	828.38	828.38	828.38	828.38	839.77
C2	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**
	589.86	591.42	591.03	589.86	589.86	589.86	589.86	589.86	591.24
RC1	**11.50**	**11.88**	**11.88**	**11.63**	**11.63**	**11.63**	**11.50**	**11.50**	**11.62**
	1389.22	1397.44	1412.87	1382.42	1392.57	1382.78	1389.78	1389.58	1440.17
RC2	**3.38**	**3.25**	**3.25**	**3.25**	**3.25**	**3.38**	**3.25**	**3.25**	**3.25**
	1117.44	1229.54	1204.87	1129.19	1144.43	1105.22	1134.52	1128.38	1243.38
CNV	**410**	**411**	**412**	**407**	**406**	**412**	**407**	**405**	**411**

TBGGP: Taillard et al [10],CR: Chiang and Russell [2], LS: Liu and Shen [8], GTA: Gambardella et al. [4], HG: Homberger and Gehring [7], RGP: Rousseau et al. [9], CLM: Cordeau et al. [3], Bräysy [1].

5 Conclusions and Further Research

In this paper we propose a multi-neighborhood approach to solve VRPTW where a random selection of a neighborhood is carried out in order to give the same selection chance to any neighboring solution. We proposed to use the RNS both in the building phase, where we search for a feasible solution improving the total distance, and in the route-reduction phase. We test our method on Solomon's instances obtaining competitive results compared with the best known results existing in the literature.
We are going on with our experiments in order to understand the importance of the role of the RNS method in the overall strategy. That is, we would like to understand the relevance of looking for new solutions near local optima or if it is more effective to choose new solutions in a completely random fashion. Therefore, we are performing new extensive experiments by using different exploration strategies in the building phase (e.g., VNS and its variants) and different shaking phases in the route elimination (e.g., ejection chains or a combination of methods).

References

[1] O. Bräysy, "A Reactive Variable Neighborhood Search Algorithm for the Vehicle Routing Problem with Time Windows", to appear in *INFORMS Journal on Computing* 15:4 (2003).

[2] W. C. Chiang, R. A. Russell, " A reactive tabu search metaheuristic for the vehicle routing problem with time windows", *INFORMS Journal on Computing* 9, 417−430 (1997).

[3] J.-F. Cordeau, G. Laporte, A. Mercier, "A unified tabu search heuristic for vehicle routing problems with time windows", *Journal of the Operational Research Society* 52, 928−936 (1997).

[4] L. M. Gambardella, E. Taillard, G. Agazzi, "MACS-VRPTW: a multiple ant colony system for vehicle routing problems with time windows", in *New Ideas in Optimization*, D. Corne, M. Dorigo, F. Glover, (eds), 63−76, McGraw-Hill, London, U.K.

[5] F. Glover, "New ejection chain and alternating path methods for traveling salesman problems", in *Computer Science and Operations Research: New Developments in Their Interfaces*, O. Balci, R. Sharda, S. Zenios (eds), 449−509, Pergamon Press, Oxford, UK. Potvin, J.-Y., J.-M. Rousseau. 1993.

[6] P. Hansen, N. Mladenovic, "Variable Neighborhood Search" in *Handbook of Metaheuristics*, GERAD and cole des Hautes tudes Commerciales Montral, Canada, 2001.

[7] J. Homberger, H. Gehring, "Two evolutionary meta-heuristics for the vehicle routing problem with time windows", *INFOR* 37, 297−318 (1999).

[8] F.-H. Liu, S.-Y. Shen, "A route-neighborhood-based metaheuristic for vehicle routing problem with time windows", *European Journal of Operational Research* 118, 485−504 (1999).

[9] L.-M. Rousseau, M. Gendreau, G. Pesant, "Using constraint based operators to solve the vehicle routing problem with time windows", *Journal of Heuristics* 8, 43−58 (2002).

[10] E. Taillard, P. Badeau, M. Gendreau, F. Guertin, J.-Y. Potvin, "A tabu search heuristic for the vehicle routing problem with soft time windows", *Transportation Science* 31, 170−186 (1997).