A Multi-Objective Density Restricted Genetic Algorithm for the Vehicle Routing Problem with Time Windows

Abel Garcia-Najera
School of Computer Science
University of Birmingham
Birmingham, B15 2TT
UK
A.G.Najera@cs.bham.ac.uk

John A. Bullinaria
School of Computer Science
University of Birmingham
Birmingham, B15 2TT
UK
J.A.Bullinaria@cs.bham.ac.uk

Abstract

The Vehicle Routing Problem with Time Windows (VRPTW) is a complex combinatorial optimisation problem which can be seen as a combination of two well-known sub-problems: the Travelling Salesman Problem and the Bin Packing Problem. Relevant objectives include minimising the number of vehicles and the total travelling distance for delivering demand to customers, while complying with capacity and time constraints. This paper proposes a novel density restricted genetic algorithm for solving VRPTW as a bi-objective problem, incorporating a diversity ratio which gives information about, and enables restriction of, the density of solutions. We have applied our algorithm to a publicly available set of benchmark instances, resulting in solutions that are competitive compared with others previously published.

1 Introduction

Combinatorial optimisation problems can be found in many real-life applications. Moreover, many of these problems have not only one, but several objectives to be optimised, which are frequently in conflict. So, instead of looking for a single permutation to give the optimal solution, we search for arrangements to provide sets of solutions that allow *trade-offs* between objectives.

There are many theoretical combinatorial problems that can be directly applied to real-life, one of them being the well known Vehicle Routing Problem (VRP). This problem can be adopted in transportation logistics like post, parcel and distribution services.

The VRP's main objective is to obtain the lowest-cost set of routes to deliver demand to customers, but we can also think about reducing the cardinality of the set of routes. This means that we can consider VRP as a multi-objective problem. Moreover, VRP has several variants of increased difficulty, in particular, the one with time windows (VRPTW), that forms the main problem to be studied in this paper, has time as well as capacity constraints.

Optimal solutions for small instances of VRPTW can be obtained using exact methods, but the computation time required increases considerably for larger sizes [5]. This is why many published works have made use of heuristic methods, such as local search, ant colony systems, and genetic algorithms.

Among those publications working with genetic algorithms is that of Potvin and Bengio [10], who developed the GENEetic ROUting System (GENEROUS), which is based on the natural evolution paradigm. They used *sequence-based crossover* and *route-based crossover* in their algorithm, and also developed three mutation operators: *one-level exchange*, *two-level exchange*, and one based on local search. Zhu [13] developed an adaptive genetic algorithm, which automatically adapted the crossover probability and the mutation rate to the changing population dynamics. The adaptive control maintained population diversity at user-defined levels, and therefore prevented premature convergence. There have been some hybrid approaches proposed as well, like those of Berger et al. [1], Le Bouthillier and Crainic [8] and Homberger and Gehring [7].

There are a couple of recent studies which are of special relevance here, because they considered VRPTW as a multi-objective problem and used a genetic algorithm for solving it. The first is due to Tan et al. [12], who used the dominance rank scheme to assign fitness to individuals. They designed a crossover operator for the specific problem called *route-exchange crossover* and used a multi-mode mutation which considered swapping, splitting and merging of routes. Also, they used three local search heuristics which were applied every 50 generations. The second is the study of Ombuki et al. [9], who presented a multi-objective genetic algorithm to minimise the number of vehicles and the total cost using two approaches: weighted sum and dominance depth. They also proposed the specific-problem genetic operators *best cost route crossover* and *constrained route reversal mutation*, which is an adaptation of the widely used inversion method. The recent surveys by Bräysy and Gendreau [2, 3] provides a complete list of studies of VRPTW and a comparison of the results obtained.

As we have noted already, only two of the cited papers have considered VRPTW as a multi-objective problem [12, 9], and only one has taken into account the diversity of solutions in the population [13]. The work presented in this paper involves solving VRPTW as a multi-objective problem by means of using a density restricted genetic algorithm (drGA), which incorporates the concept of *diversity preservation* to obtain good solutions to the problem. Moreover, even though it has been suggested that it is unnecessary to introduce new parameters into genetic algorithms, or evolutionary approaches in general [4], the inclusion of diversity preservation led us to introduce a *diversity ratio* parameter, which restricts the density of solutions. We have tested our algorithm on publicly available benchmark instances, and compared our results with those from other publications, and our algorithm appears very promising.

The remainder of this paper is organized as follows. First, in Section 2, we introduce VRPTW in more detail. In Section 3 we provide a brief description of what multi-objective optimisation problems are, and explain the concepts of fitness assignment and diversity preservation. Our proposed drGA for solving VRPTW as a multi-objective problem is described in Section 4. In Section 5 we present the experimental set-up and results from our preliminary work, as well as the comparison with some others already published. Finally, we give our conclusions in Section 6 and present some ideas for future work.

2 VRPTW

The Vehicle Routing Problem with Time Windows (VRPTW) is a complex combinatorial optimisation problem which can be regarded as a combination of two well-known sub-problems: the Travelling Salesman Problem and the Bin Packing Problem. So, it is at least as difficult as each of them. VRPTW can be formally defined as follows. Given:

- a set $\mathcal{V} = \{v_1, \ldots, v_n\}$ of nodes, called customers, with known demands $d_i > 0$, $\forall\ i \in \{1, \ldots, n\}$,
- a special node v_0, called the *depot*, with $d_0 = 0$,
- a symmetric cost ω_{ij} assigned to the distance between any pair of customers, or between the depot and any customer, $\forall\ i, j \in \{0, \ldots, n\}, i < j$,
- a time window or interval $[b_i, e_i]$ associated with each customer $v_i \in \mathcal{V}$ during which the customer has to be supplied,
- a service or unload time s_i associated with each customer $v_i \in \mathcal{V}$, and
- a fleet of identical vehicles with capacity $Q \geq \max\{d_i : i \in \{1, \ldots, n\}\}$,

we have to design a minimum-cost set of m routes, so that each route begins and ends at the depot and each customer is serviced by exactly one vehicle.

Since every customer has a service time window, a solution becomes infeasible if customer v_i is supplied after e_i. Moreover, if a vehicle arrives at customer v_i before b_i, a waiting time has to be added to the travel time. A solution also becomes infeasible if the total load on any vehicle is greater than Q.

Let $\mathcal{R} = \{r_1, \ldots, r_m\}$ be the set of m routes. Also, set $x_{ijk} = 1$ if the arc (i, j) between any pair of customers, or any customer and the depot, is considered in route r_k, otherwise $x_{ijk} = 0$. Then we can define

$$C = \sum_{k=1}^{m} \sum_{i=0}^{n} \sum_{j=0}^{n} \omega_{ij} x_{ijk} \tag{1}$$

as the total cost associated with the set $\mathcal{R}$.

3 Multi-objective Optimisation

Any multi-objective optimisation can, without loss of generality, be defined as a minimisation problem of the form:

$$\text{minimise } \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_k(\mathbf{x})) \tag{2}$$

subject to constraint functions:

$$g_i(\mathbf{x}) \leq 0, \qquad \forall\ i = 1, 2, ..., m \tag{3}$$
$$h_j(\mathbf{x}) = 0, \qquad \forall\ j = 1, 2, ..., p \tag{4}$$

where $\mathbf{x} = (x_1, x_2, ..., x_n) \in \mathcal{X}$ is the vector of decision variables, $\mathcal{X}$ is the parameter space, and $f_i : \mathbb{R}^n \to \mathbb{R}, i = 1, ..., k$, are the objective functions. Functions $g_i, h_j : \mathbb{R}^n \to \mathbb{R}$ in (3) and (4) restrict $\mathbf{x}$ to consider only feasible solutions.

A decision vector $\mathbf{x} \in \mathcal{X}$ is said to *dominate* a decision vector $\mathbf{y} \in \mathcal{X}$ (written as $\mathbf{x} \prec \mathbf{y}$) if and only if $f_i(\mathbf{x}) \leq f_i(\mathbf{y})\ \forall\ i = 1, 2, ..., k$ and $\exists\ j \in \{1, 2, ..., k\} : f_j(\mathbf{x}) < f_j(\mathbf{y})$. Similarly, we say that a decision vector $\mathbf{x} \in \mathcal{X}$ is *non-dominated* if there is no decision vector $\mathbf{y} \in \mathcal{X}$ such that $\mathbf{y} \prec \mathbf{x}$.

A decision vector $\mathbf{x} \in \mathcal{X}$ is said to be *Pareto optimal* if it is non-dominated with respect to $\mathcal{X}$. The *Pareto optimal set* is defined as $\mathcal{P}_s = \{\mathbf{x} \in \mathcal{X} \mid \mathbf{x} \text{ is Pareto optimal}\}$. Finally, the *Pareto front* is defined as $\mathcal{P}_f = \{\mathbf{f}(\mathbf{x}) \in \mathbb{R}^n \mid \mathbf{x} \in \mathcal{P}_s\}$.

3.1 Fitness Assignment and Diversity Preservation

The task of approximating the Pareto optimal set involves: minimising the distance of the generated solutions to the Pareto optimal set, and maximising the diversity of the achieved Pareto set approximation. When applying a genetic algorithm for solving this kind of problem, the first goal is mainly related to the task of assigning a scalar fitness value to individuals. The second goal concerns how to handle the selection, because it is desirable to avoid identical solutions in the resulting set [14].

In the single-objective case, fitness is assigned directly to an individual according to its objective function evaluation. On the other hand, in the multi-objective case, this assignment cannot be done straightforwardly, due to there being not only one objective function, but at least two of them. In this scenario we may use one of the following methods [14]:

Dominance count This method considers the number of solutions dominated by a certain individual.

Dominance depth The population is divided into several fronts and the depth specifies which front an individual belongs to.

Dominance rank This method takes into account the number of individuals by which a solution is dominated.

Diversity in the Pareto approximation is important because it is desirable that solutions within it are different. Density information gives us a good metric to handle this diversity. To measure the density of solutions, three relevant categories of density estimation have been used in the past [14]:

Kernel The distance between one solution and all others is calculated and a *Kernel* function is applied. Its density estimate will be the sum of all these evaluations.

Nearest neighbour This method takes into account the distance between a given point and its k-th nearest neighbour to estimate the density in its neighbourhood.

Histogram Techniques in this category define k-dimensional grids as neighbourhoods in the k-dimensional space. The number of individuals in the same grid area as a given solution is then its density estimate.

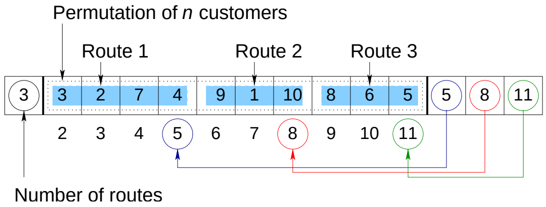

Figure 1: Example of the genetic representation for a solution to VRPTW.

4 Multi-objective drGA

We present in this section our proposed drGA for solving VRPTW as a multi-objective problem. We detail the genetic representation, and the stages of processing involved. We also describe our main contribution, which is the incorporation of a diversity preservation method.

4.1 Genetic Representation and Initial Population

The idea for the genetic representation was inspired by [6], and is depicted in Figure 1, which shows an example with $n = 10$. It consists of a chromosome with three parts. The first gene in the chromosome indicates the number of routes in the solution (3 in this example). The following n genes are a permutation of the n customers $(1, \ldots, 10$ in this example). The rest of the genes specify the gene of the last customer in every route. In this example, the first route includes customers $3, 2, 7, 4$, with customer 4 the last customer of the route, as specified in gene 12. The other two routes are $9, 1, 10$ and $8, 6, 5$. The order in which these customers are serviced is exactly as they appear in the chromosome.

The initial population is built with random feasible solutions. Each of these solutions contains a set of randomly generated routes. Such routes are constructed in the following way. First, a customer is randomly selected and placed as the first location to visit on that route. A second customer is randomly chosen and, if capacity and time constraints are met, it is placed after the previous one. If any of the constraints are not met, a new route is created and this customer will be the first location to visit in the new route. This process is repeated until all customers are assigned to a route.

4.2 Crossover

The crossover operator takes a traditional form suggested by [6], and works as follows. The first step selects from the first parent a random number of adjacent routes and copies them into the offspring. The next step copies all those routes from the second parent which do not include any customers already copied from the first parent. If there remain unassigned customers, these are allocated, in the order they appear in the second parent, to any of the existing routes. If a solution would become infeasible after inserting such a customer, a new route is created.

4.3 Mutation

For the mutation stage, six operators were implemented:

Insertion One sub-route is randomly chosen and its customers are inserted into another route.

Swap Two sub-routes are randomly selected and their customers are swapped.

Inversion One sub-route is randomly chosen and its customers are inverted.

Displace One sub-route is randomly selected and its customers are displaced one position.

Split One route is randomly chosen and split.

Merge Two routes are randomly chosen. The second route is appended after the last customer in the first route.

4.4 Fitness, Diversity and Selection

Fitness is assigned to individuals using the dominance depth criterion reviewed in Section 3. We have considered the minimisation of the number of routes and the travel distance.

The VRPTW uses a relatively low number of vehicles. This aspect makes the definition of niche areas problematic, since most good solutions reside in a very small portion of the vehicle number dimension [9]. This is why, as an innovative feature to help in the selection process, we consider the diversity of solutions, using the nearest neighbour method to estimate density. Moreover, we consider the number of equal solutions to estimate and restrict density.

In other words, we introduce a new parameter, which we call the *diversity ratio* (δ), to restrict the density of equal solutions from growing indiscriminately. This parameter is defined as

$$\delta = \frac{\#\text{dif. solns.}}{\textit{pop\_size}} \tag{5}$$

where the numerator refers to the number of different solutions in the population. We use this parameter to force the algorithm to preserve, at least, a fixed number of different solutions in the population. That is

$$\text{min. }\#\text{dif. solns.} = \textit{pop\_size} \cdot \delta \tag{6}$$

For example, if $\textit{pop\_size} = 100$ and $\delta = 0.1$, we are forcing the algorithm to preserve at least 10 different solutions, but if $\textit{pop\_size} = 200$, the algorithm will preserve at least 20. So, the maximum number of equal solutions in the population will be given by

$$\begin{aligned}\text{max. }\#\text{equal solns.} &= \frac{\textit{pop\_size}}{\text{min. }\#\text{dif. solns.}} \\ &= \frac{\textit{pop\_size}}{\textit{pop\_size} \cdot \delta} = \frac{1}{\delta}\end{aligned} \tag{7}$$

If after the crossover and mutation stages the density of equal solutions has grown to more than the maximum allowed, further individuals are dropped until the density is rectified.

The selection process is carried out twice every iteration: to select parents in the crossover stage and to select the individuals for the next generation. For the former, we select individuals using the tournament selection method. Fitness is the first criterion to use and density the second. For the latter, we select individuals belonging to the first fronts to be carried into to the next generation.

5 Experiments and Results

To test our algorithm, we used the publicly available benchmark set due to Solomon [11], which includes 56 instances of size $n = 100$. These instances are categorised as Random, Clustered, and mixed: R, C and RC. Solomon [11] provides a complete description of the test data, and the data-sets themselves are publicly available from his web site[1].

We ran our algorithm on each instance for three different values of δ: 0.01, 0.05 and 0.1. Figures 2 and 3 display the average convergence obtained for two typical problem instances (C204 and RC208 respectively).

We can observe from this figure that drGA with $\delta = 0.1$ could find better solutions, on average, for those instances. The same behaviour

1. `http://w.cba.neu.edu/~msolomon/home.htm`

Ref.	C1	C2	R1	R2	RC1	RC2	Accum.
[10]	10.00	3.00	12.60	3.00	12.10	3.40	422.20
	838.00	589.90	1296.80	1117.70	1446.20	1360.60	62571.90
[13]	10.00	3.00	12.80	3.00	13.00	3.70	434.20
	828.90	589.90	1242.70	1016.40	1412.00	1201.20	59177.70
[1]	10.00	3.00	11.92	2.73	11.50	3.25	405.00
	828.48	589.93	1221.10	975.43	1389.89	1159.37	57952.00
[8]	10.00	3.00	12.08	2.73	11.50	3.25	406.99
	828.38	589.86	1210.14	969.57	1389.78	1134.52	57555.65
[7]	10.00	3.00	11.91	2.73	11.50	3.25	405.00
	828.38	589.38	1212.73	955.03	1386.44	1108.52	57192.00
[12]	10.00	3.00	12.92	3.55	12.38	4.25	441.00
	828.74	590.69	1187.35	951.74	1355.37	1068.26	56290.48
[9]	10.00	3.00	13.17	4.55	13.00	5.63	471.00
	828.48	590.60	1204.48	893.03	1384.95	1025.31	55740.33
drGA	10.00	3.00	13.17	3.64	13.00	4.38	451.00
	857.49	591.03	1248.55	954.47	1413.95	1099.09	58031.73
ΔL	3.51	0.28	5.15	6.88	4.32	7.20	4.11
ΔH	0.00	0.00	3.86	17.10	2.28	23.79	7.82

Table 1: Averaged best results grouped by instance set.

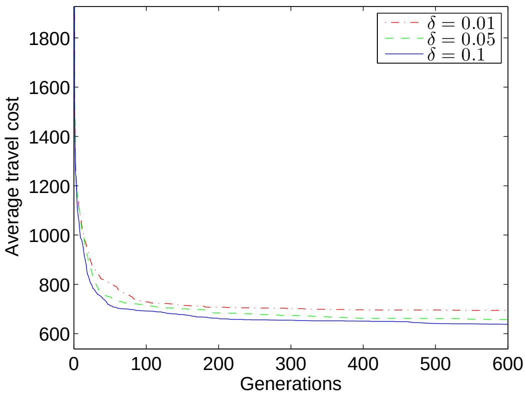

Figure 2: Average convergence instance C204.

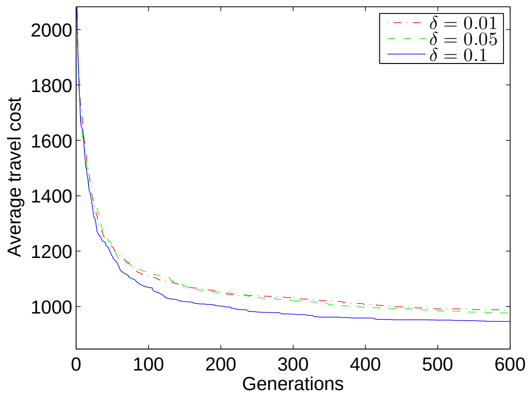

Figure 3: Average convergence instance RC208.

was found in the majority of the instances, so we naturally chose to work with this parameter value. The other parameters were set to: *pop_size* = 200, generations = 600, tournament size = 5, crossover rate = 0.8, mutation rate = 0.2 (insertion = 0.3, swap = 0.3, inversion = 0.2, displacement, = 0.15 and split = 0.05).

We ran our algorithm 10 times for every problem instance and recorded the best solutions in each run, but because of space limitations, we cannot present them all here (though they are publicly available from our research web site[2]). We can note that drGA has found two new best solutions and another 23 similar to the best-known. In Table 1 we summarize the results for the instances grouped in sets. They compare our averaged best results with those from recent publications. We present for each author and instance set the average number of routes (upper) and the average travel distance (lower). The last column presents the total accumulated sum, indicating the total number of vehicles and the total travel distance for all 56 instances. In the last two rows we compute the percent difference between our results and the lowest (ΔL), and the highest (ΔH) travel distance for each instance set.

Analysing the results in Table 1, we can see that for instance sets C1 and C2, our algorithm obtained, on average, the highest costs, but the gap between these and the lowest results is narrow. On the other hand, for the other sets, although the results from our algorithm are not the overall best, they do show considerable improvement over several of the other algorithms. Moreover, in the case of the accumulated travel distance, the difference between our results and the highest is 7.86%, despite our algorithm using a larger number of routes. An-

2. `http://www.cs.bham.ac.uk/~agn/research/2008/ukci/results.pdf`

other interesting observation is that our results for sets R and RC and accumulated confirm the multi-objective nature of VRPTW, in that we obtained lower travel distances, compared with other authors, using more routes.

6 Conclusions

We have proposed in this paper our novel density restricted Genetic Algorithm (drGA) for solving VRPTW as a multi-objective problem. Its main characteristic involves guaranteeing a minimum number of different solutions in the population at all times, depending on the value of a new diversity ratio parameter δ which controls the density of equal solutions. We have applied our algorithm to a set of benchmark problems with three different values for this parameter and demonstrated its importance for obtaining good results. We have also compared our results with those from recent publications also using the genetic paradigm. Although our results are not the overall best, they are better than some, and, on average, competitive. Our drGA also managed to find solutions such that the accumulated travel distance is better than others, despite the number of routes being larger, indicating the trade-offs underlying the multi-objective aspects of VRPTW.

Given this promising start, we are now pursuing our approach further with improved variations on the diversity ratio theme, and also exploring more systematically what improvements might be possible by varying the various other associated design decisions made in this study. We are also comparing our results with those obtained using another kinds of heuristics.

Acknowledgements

The first author acknowledges support from CONACYT through a scholarship to pursue his graduate studies.

References

[1] J. Berger, M. Barkaoui, and O. Bräysi. A route-directed hybrid genetic approach for the vehicle routing problem with time windows. *INFOR*, 41:179–194, 2003.

[2] O. Bräysy and M. Gendreau. Vehicle routing problem with time windows, part I: Route construction and local search algorithms. *Trans. Sci.*, 39(1):104–118, 2005.

[3] O. Bräysy and M. Gendreau. Vehicle routing problem with time windows, part II: Metaheuristics. *Trans. Sci.*, 39(1):119–139, 2005.

[4] K. Deb. Evolutionary multi-objective optimization without additional parameters. In Fernando G. Lobo, Cláudio F. Lima, and Zbigniew Michalewicz, editors, *Parameter setting in evolutionary algorithms*, pages 241–257. Springer Verlag, Berlin, Germany, 2007.

[5] M. Desrochers, J. Desrosiers, and M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Opns. Res.*, 40(2):342–354, 1992.

[6] E. Falkenauer. A hybrid grouping genetic algorithm for bin packing. *J. of Heuristics*, 2(1):5–30, 1996.

[7] J. Homberger and H. Gehring. A two-phase hybrid metaheuristic for the vehicle routing problem with time windows. *Eur. J. of Opnl. Res.*, 162:220–238, 2005.

[8] A. Le Bouthillier and T. G. Crainic. A cooperative parallel metaheuristic for vehicle routing with time windows. *Comps. & Opns. Res.*, 32:1685–1708, 2005.

[9] B. Ombuki, B. J. Ross, and F. Hanshar. Multi-objective genetic algorithms for vehicle routing problem with time windows. *Appd. Intell.*, 24:17–30, 2006.

[10] J. Y. Potvin and S. Bengio. The vehicle routing problem with time windows — part II: Genetic search. *INFORMS J. on Computing*, 8:165–172, 1996.

[11] M. M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Opns. Res.*, 35(2):254–265, 1987.

[12] K. C. Tan, Y. H. Chew, and L. H. Lee. A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows. *Comb. Opt. and Apps.*, 34:115–151, 2006.

[13] K. Q. Zhu. A diversity-controlling adaptive genetic algorithm for the vehicle routing problem with time windows. In *Proc. 2003 IEEE Intnl. Conf. on Tools with Art. Intell.*, pages 176–183, Los Alamitos, CA, USA, 2003. IEEE Computer Society.

[14] E. Zitzler, M. Laumanns, and S. Bleuler. A tutorial on evolutionary multiobjective optimization. In Xavier Gandibleux, Marc Sevaux, Kenneth Sörensen, and Vincent T'kindt, editors, *Metaheuristics for Multiobjective Optimisation*, pages 3–37. Springer. Lecture Notes in Economics and Mathematical Systems Vol. 535, 2004.