Bi-objective Optimization for the Vehicle Routing Problem with Time Windows: Using Route Similarity to Enhance Performance

Abel Garcia-Najera and John A. Bullinaria

School of Computer Science, University of Birmingham,
Birmingham B15 2TT, United Kingdom
{A.G.Najera,J.A.Bullinaria}@cs.bham.ac.uk

Abstract. The Vehicle Routing Problem with Time Windows is a complex combinatorial optimization problem which can be seen as a fusion of two well known sub-problems: the Travelling Salesman Problem and the Bin Packing Problem. Its main objective is to find the lowest-cost set of routes to deliver demand, using identical vehicles with limited capacity, to customers with fixed service time windows. In this paper, we consider the minimization of the number of routes and the total cost simultaneously. Although previous evolutionary studies have considered this problem, none of them has focused on the similarity of solutions in the population. We propose a method to measure route similarity and incorporate it into an evolutionary algorithm to solve the bi-objective VRPTW. We have applied this algorithm to a publicly available set of benchmark instances, resulting in solutions that are competitive or better than others previously published.

Key words: Vehicle routing problem, multi-objective optimization, evolutionary algorithm, similarity of solutions.

1 Introduction

There are many theoretical combinatorial problems that can be directly applied to real-life, one of them being the Vehicle Routing Problem (VRP) [19], which is relevant to transportation logistics such as 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. In addition, we can contemplate other objectives like the makespan, workload balance, etc. [8]. This means that it is often useful to consider the VRP as a multi-objective problem. Moreover, VRP has several important variants of increased difficulty, in particular, the one with time windows (VRPTW) which has time as well as capacity constraints, and is the main problem to be studied in this paper.

Optimal solutions for small instances of VRPTW can be obtained using exact methods, but the computation time required increases considerably for larger instances [5]. This is why many published studies have made use of heuristic

methods. The recent surveys by Bräysy and Gendreau [2, 3] provides a complete list of studies of VRPTW and a comparison of the results obtained.

Over the years, there have been several publications employing evolutionary algorithms to solve VRPTW as a single-objective optimization problem [13, 18, 20, 1]. Recently, Le Bouthillier and Crainic [9] presented a parallel cooperative multi-search method for VRPTW, based on the solution warehouse strategy, in which several search threads cooperate by asynchronously exchanging information on the best solutions identified. Each of these search methods implements a different meta-heuristic, an evolutionary algorithm or a tabu search procedure. Homberger and Gehring [7] proposed a two-phase hybrid meta-heuristic to solve VRPTW, where the first phase was aimed at the minimization of the number of routes by means of a (μ,λ)-evolution strategy, whereas in the second phase the total distance is minimized using a tabu search algorithm.

In the past few years, a couple of studies have been published that are of special relevance to us because they considered VRPTW as a bi-objective optimization problem, minimizing the number of vehicles and the total travel distance, and used a genetic algorithm for solving it. The first is due to Tan et al. [17], 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. They also used three local search heuristics which were applied every 50 generations. The second is the study of Ombuki et al. [12]. They also proposed the problem-specific genetic operators *best cost route crossover* and *constrained route reversal mutation*, which is an adaptation of the widely used inversion method. However, unlike the work we present later in this paper, neither of these two studies considered using a method to measure similarity of solutions and hence preserve diversity.

It is also worth noting that many existing well known and successful multi-objective evolutionary approaches, such as SPEA2 [21] and NSGA-II [4], are not suitable here because they require the definition of niche spaces, which would be problematic since most good solutions of the VRPTW reside in a very small portion of the vehicle number dimension [12].

In our preliminary work [6], it became clear that the lack of population diversity was leading our algorithm to become stuck in suboptimal solutions, and so we proposed a method to restrict the number of clones in the population. This algorithm eventually forced the population to have no clones at all, but the solutions were still not good enough. Consequently, in this paper, we look for a mechanism to improve further both the quality and diversity of solutions.

The approach we shall follow will focus on the similarity of solutions, based in the genotype space. Several methods for calculating the similarity between two solutions using a permutation representation exist in the literature [14, 10, 15], but, as we are not using a permutation encoding, we cannot apply any of them. So, the need to find a suitable similarity measure arose.

The work presented in this paper is concerned with the solution of VRPTW as a bi-objective problem using an evolutionary algorithm (BiEA), which incor-

porates a similarity measure applied in the genotype space, based on Jaccard's similarity coefficient, to select parents for the recombination process, leading to the finding of good solutions to the problem. We have tested this algorithm on publicly available benchmark instances, and when our results are compared with those from recent publications, our algorithm appears very competitive.

The remainder of this paper is organized as follows. First, in Section 2, we introduce VRPTW in more detail. In Section 3 we present our proposed similarity measure for solutions to VRPTW. Our proposed BiEA for solving VRPTW as a bi-objective problem is described in Section 4. In Section 5 we present the results achieved by our algorithm, as well as the comparison with some others already published. Finally, we give our conclusions in Section 6.

2 The Vehicle Routing Problem with Time Windows

The Vehicle Routing Problem (VRP) is a complex combinatorial optimization problem, which can be seen as a fusion of two well-known problems: the Travelling Salesman Problem (TSP) and the Bin Packing Problem (BPP). So, it is at least as difficult as each of them. VRP has several variants of increased difficulty, in particular, the one with time windows (VRPTW) which has both capacity and time constraints.

Before defining VRPTW, we need to specify the information involved in an instance of this problem. First of all, we have a set $\mathcal{V} = \{v_1, \ldots, v_n\}$ of vertices, called customers. We know that customer v_i, $\forall\, i \in \{1, \ldots, n\}$, is geographically located at position (x_i, y_i), has a demand of goods $d_i > 0$, has a time window $[b_i, e_i]$ during which it has to be supplied, and requires a service time s_i to unload goods. There exists a special vertex v_0, called the *depot*, located at (x_0, y_0), with $d_0 = 0$, and time window $[0, e_0 \geq \max\{e_i : i \in \{1, \ldots, n\}\}]$, from which customers are serviced utilising a fleet of identical vehicles with capacity $Q \geq \max\{d_i : i \in \{1, \ldots, n\}\}$.

The travel between vertices v_i and v_j has an associated symmetric cost $c_{ij} = c_{ji}, \forall\, i, j \in \{0, \ldots, n\}$, which is usually considered to be the Euclidean distance. In addition to distance, time also plays an important role, as it is not possible to supply a customer before or after its time window. A vehicle could arrive early at the customer location, but then it has to wait until the beginning of the time window. Arriving late is not allowed. It is common to take the time t_{ij} to travel between vertexes v_i and v_j to simply be $t_{ij} = c_{ij}$.

The problem consists of designing a minimum-cost set of routes, so that each route begins and ends at the depot, and each customer is serviced by exactly one vehicle. Thus, each vehicle is assigned a set of customers that it has to supply, but the sum of their demands can not exceed the vehicle capacity Q.

Let us denote as $r_k = \langle u_1^k, \ldots, u_{n_k}^k \rangle$ the k-th designed route that supplies n_k customers, with u_i^k the i-th vertex to visit in the route. Note that in this notation we are omitting the depot, but we have to consider it before the first customer and after the last customer. Then, the customers demand D_k associated with

route r_k is given by

$$D_k = \sum_{i=1}^{n_k} d_{u_i^k} \leq Q \ . \tag{1}$$

Likewise, we can define the cost C_k associated with route r_k as

$$C_k = c_{0u_1^k} + \sum_{i=1}^{n_k-1} c_{u_i^k u_{i+1}^k} + c_{u_{n_k}^k 0} \ . \tag{2}$$

Once we have defined the problem, we can identify at least two objective functions that could be minimized. If $\mathcal{R} = \{r_1, \ldots, r_m\}$ is the set of designed routes, we can consider minimizing the number of routes

$$f_1(\mathcal{R}) = |\mathcal{R}| \tag{3}$$

and the total cost

$$f_2(\mathcal{R}) = \sum_{k=1}^{|\mathcal{R}|} C_k \ . \tag{4}$$

It is these two objectives that we concentrate on in this paper

3 Measuring similarity of solutions to the VRPTW

Different solution representations require different distance measures. For example, for binary representations, the *Hamming distance* is the most common measure, for representations using a vector of real numbers, a variation of the *Minkowski-r-distance* can be employed [15], and we can find in the literature many methods for solutions represented as a permutation, like the *exact match distance* and *deviation distance* [14], the *R-type distance* [10] and the *edit distance* [15]. Since we are not using any of these representations, and considering distance in the phenotype space is likely to be misleading, we need to identify a suitable new similarity measure.

Taking the above into consideration, we have designed a new similarity measure, based on *Jaccard's similarity coefficient*. This is applied in the genotype space, and consequently provides a more reliable diversity measure than a phenotype-oriented method. Moreover, probably most interestingly and importantly, this new similarity measure is independent of the solution encoding.

The Jaccard's similarity coefficient is a statistic used for comparing the similarity of two sets. It is defined as the cardinality of the intersection of the sets divided by the cardinality of the union of them, i.e.

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \ . \tag{5}$$

It is easy to see that if sets A and B contain the same elements, $A = B = A \cap B = A \cup B$, so Jaccard's similarity coefficient $J(A, B) = 1$. On the other hand, if sets A and B do not share any element at all, $|A \cap B| = 0$, so $J(A, B) = 0$.

Now we can define the similarity between two solutions to VRPTW, according to the Jaccard's similarity coefficient, simply as the ratio of the number of shared arcs to the number of total arcs used in both solutions.

Let $y_{ijk} = 1$ if arc (i, j) from vertex i to vertex j is used by any vehicle in solution r_k, and $y_{ijk} = 0$ otherwise. Then the similarity ς_{pq} between solutions p and q is

$$\varsigma_{pq} = \frac{\sum_{i=0}^{n} \sum_{j=0}^{n} y_{ijp} \cdot y_{ijq}}{\sum_{i=0}^{n} \sum_{j=0}^{n} \text{ sign } (y_{ijp} + y_{ijq})} \ , \tag{6}$$

where $y_{ijp} \cdot y_{ijq} = 1$ iff arc (i, j) is used by both solutions, and sign $(y_{ijp} + y_{ijq}) = 1$ if any of the solutions use it. If solutions p and q are the same, the sum in the numerator will equal the sum in the denominator, and therefore $\varsigma_{pq} = 1$. On the other hand, if they are two completely different solutions with no arc in common, the numerator will equal 0, and then $\varsigma_{pq} = 0$.

In the same manner, if we want to compute the similarity $\mathcal{S}_p$ of individual p with the rest of the population $\mathcal{P}$ of size $N - 1$, we have to calculate the average similarity of p with every other individual $q \in \mathcal{P}$, that is

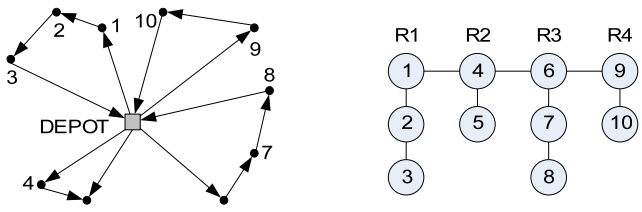
$$\mathcal{S}_p = \frac{1}{N-1} \sum_{q \in \mathcal{P}} \varsigma_{pq} \ . \tag{7}$$

4 Evolutionary approach

We present in this section our proposed EA for solving VRPTW as a bi-objective problem. We detail the encoding of the solutions, and the stages of processing involved. We also describe our main contribution, which is the incorporation of the similarity measure method presented above.

4.1 Solution encoding

We are using a tree representation, in which each node has at most two children. The left child represents the following customer to visit in a route. The right child points to the next route in the solution. A solution to an example instance and its representation are shown in Figure 1. The allocation of customers to



(a) Solution (b) Encoding

Fig. 1. Solution to an example instance of the VRPTW and its encoding.

routes, and the sequence they will be serviced within each route, proceeds as follows: customers 1, 2 and 3 to the first route, customers 4 and 5 to the second, 6, 7 and 8 to the third, and 9 and 10 to the fourth.

4.2 Fitness assignment

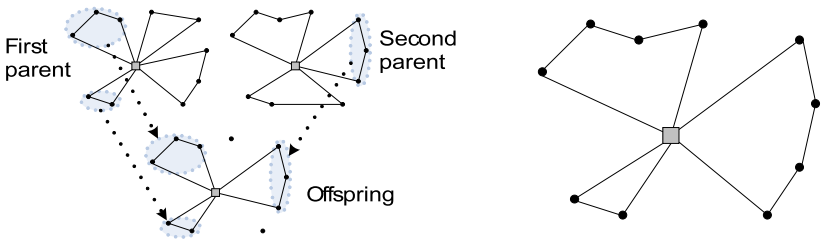
When solving a single-objective problem using an evolutionary algorithm, fitness is assigned to an individual according to its objective function evaluation. In the multi-objective case, this assignment cannot be done straightforwardly, due to there being not only one objective function, but at least two of them. We have used in this work the non-dominance sort criteria [4] to assign fitness to solutions, where the population is divided into several non-dominated fronts and the depth specifies the fitness of the individuals belonging to them. In this case, the lower the front, the fitter the solution.

4.3 Evolutionary process

Our algorithm starts with a set of feasible random solutions, each containing a set of randomly generated routes. These routes are constructed using the following process: First, a customer is selected and placed as the first location to visit on that route. Then, a second customer is 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 have been assigned to a route.

Then, the objective functions are evaluated for every solution in the population and these are assigned a fitness value. Finally, the similarity of each solution with respect to the rest of the collection is computed.

The evolution proceeds with the recombination of two parents that are selected using a standard tournament method, but under different criteria: fitness is used to select the first parent and similarity the second. The recombination of two example parents is shown in Figure 2(a). Here, the algorithm aims at preserving routes from both parents. First, a random number of routes are chosen



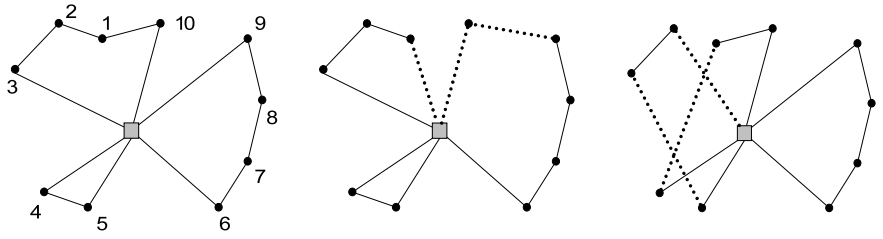
(a) Copying routes from parents (b) Offspring

Fig. 2. The recombination process.

from the first parent and copied into the offspring. Next, all those routes from the second parent which are not in conflict with the customers already copied from the first, are replicated into the offspring. In this case, both routes on the left from the first parent were selected to be copied into the offspring, and we can only copy from the second parent the route on the right, as the other two contain customers already present in the offspring. If there remain unassigned customers, these are allocated, in the order they appear in the second parent, to the route where the lowest travel distance is achieved, like in the example given in Figure 2(b). If a solution would become infeasible after inserting such a customer, a new route is created. This means that there is no need for a repair process to correct invalid individuals.

Once an offspring has been generated, it is submitted to the mutation process. In our algorithm, we have five possible mutation operators, which can be categorised as inter- and intra-route. In the former, the algorithm will perform changes between two routes, thus modifying the assignment of customers to routes, and in the latter, the changes will be done within a route, hence affecting the travel sequence.

In the first category we can identify two viable processes which are: (*i*) removing a sequence of customers from a route and *inserting* it into another, and (*ii*) *swapping* two sequences of customers from different routes. In the case of intra-route, we use three operations: (*i*) the *inversion* of the sequence of a subroute, (*ii*) the *shift* of one customer, and (*iii*) *splitting* a route. Examples of these operations are shown graphically in Figure 3. The dotted lines in each figure rep-



(a) Original offspring (b) Insertion (c) Swap

(d) Inversion (e) Shift (f) Split

Fig. 3. The mutation operators used in BiEA.

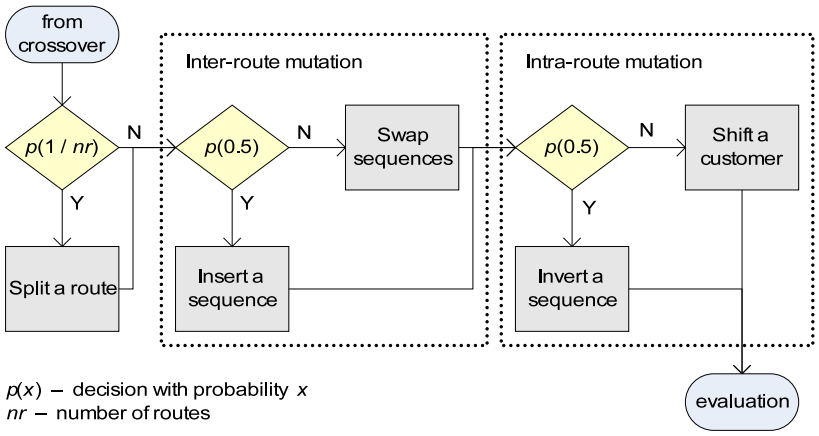


Fig. 4. The full mutation process.

resent the changes in the sequences. In Figure 3(b), customer 10 was removed from the route on the left and has been inserted in the route on the right. Figure 3(c) shows the swap of customer 4 with customers 2 and 3. In Figure 3(d) we have the inversion of the sequence of customers 7, 8 and 9. Figure 3(e) shows how customer 9 has been shifted between customers 6 and 7. Finally, in Figure 3(f), the route on the right has been split between customers 7 and 8.

Not all of the mutation operators are applied each time an offspring is mutated. First, the split operator is performed with a probability equal to the inverse of the number of routes in the solution. Then the solution is submitted to one of the inter-route operators. The decision of which to apply is random. Finally, one of the intra-route operators is applied to the solution. The complete mutation process is shown in Figure 4.

After this process, the algorithm evaluates the objective functions for each solution in the offspring population and combines both parent and offspring populations to assign fitness. Those solutions having the highest fitness are taken to the next generation. If one front is in conflict with the population size, similarity is computed for those solutions in that front, and the less common are considered for the next iteration. Similarity is computed again and the whole process is repeated for $maxGen$ generations.

5 Experimental set-up and results

To test our new algorithm, we used the publicly available benchmark set due to Solomon [16], which includes 56 instances of size $n = 100$. These instances are categorised as Clustered (C1, C2), Random (R1, R2), and Mixed (RC1, RC2). Solomon [16] provides a complete description of the test data, and the data-sets themselves are publicly available from his web site[1].

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

These data-sets have been previously studied in detail, and a recent analysis by Tan et al. [17] suggests that categories C1 and C2 have positively correlating objectives, which means that the travel cost of a solution increases with the number of vehicles. However, many of the instances in categories R1, R2, RC1 and RC2 have conflicting objectives.

Following the discussion above, the analysis of our results has three objectives: (*i*) to compare results from single- and bi-objective algorithms, (*ii*) to examine the difference between results from our algorithm with and without considering the similarity measure, and (*iii*) to study the performance of our algorithm when compared with others previously published.

We ran our algorithm 30 times for every instance and recorded the solutions in the Pareto approximation each time. The parameters of our algorithm were set to convenient round numbers that worked well as follows:

population size = 100	crossover rate = 0.9
number of generations = 500	mutation rate = 0.1
tournament size = 10	

5.1 Single- and bi-objective optimization

In this section we compare the results from our algorithm with those from a single-objective genetic algorithm (GA), namely the version of our BiEA which only minimizes one of the two objective functions. For simplicity, the one that minimizes the number of routes will be called GAR, and the one that minimizes the total cost will be called GAC. Note that we are performing this comparison to first determine whether VRPTW really does behave as a multi-objective problem, and then, if it does, to determine whether we can achieve better results by considering it as a multi-objective problem.

In Table 1 is presented the average best results grouped by category set. We have averaged the best results found over all iterations, and averaged these over the set category. We show for each algorithm and instance set the average number of routes (upper) and the average total cost (lower). The last column presents the total accumulated sum, indicating the total number of routes and the total cost for all 56 instances. In this table we are also showing the results from the version of our algorithm that does not include the similarity measure (BiEA_{NS}), but we will not analyse these until the following section.

We can see that the number of routes from GAR is lower than that from GAC in four of the six categories, and only in two compared with BiEA. Our BiEA algorithm achieved fewer routes in all cases compared with GAC. Comparing the total cost, GAR obtained the highest costs, while our algorithm surpassed GAC in four of the six groups. The last column in the table shows the overall accumulated results for both objectives. In this matter, GAR has the lowest accumulated number of routes, and BiEA the lowest accumulated total cost, reducing by 47% and 3% that achieved by GAR and GAC, respectively.

Given the narrow difference in the total cost from GAC and BiEA for sets C2, R2 and RC2, we decided to analyze in more detail the performance of these

Table 1. Comparison of the average best results, averaged by set category, from the single- and bi-objective algorithms.

Alg.	C1	C2	R1	R2	RC1	RC2	Accum.
GAR	10.43	3.07	13.08	3.14	12.91	3.60	441.95
	1685.22	898.33	1550.45	1448.03	1742.10	1769.78	84982.41
GAC	10.05	3.01	13.52	4.00	13.51	4.81	467.33
	908.21	601.42	1273.83	954.24	1464.37	1111.69	59376.27
BiEA	10.00	3.00	12.96	3.79	12.65	4.53	448.70
	842.29	597.52	1222.13	968.89	1378.40	1136.16	57800.63
BiEA_{NS}	10.03	3.00	13.26	3.63	13.26	4.28	453.52
	918.86	604.23	1279.74	977.27	1471.58	1143.58	60131.52

algorithms. In Figures 5 to 7 we present box-and-whisker plots for all instances in these sets, with the total cost normalized to the median of the results from BiEA. For each instance there are two boxes, with the one on the left corresponding to the results from GAC, and the one on the right to BiEA. The boxes have lines at the lower quartile, median, and upper quartile values. Notches display the variability of the median. The width of the notches are computed so that box plots which notches do not overlap have different medians at the 5% significance level [11].

In the case of the instances in category C2, depicted in Figure 5, there is not a clear difference for instances C201, C202, C205 and C207 where the median and its variability appears to be the same for both algorithms. For instances C206 and C208, despite notches that are overlapping, the variability of the median of the results from GAC is larger than that of the results from BiEA. In the case of instances C203 and C204, boxes from BiEA are lower and shorter that those from GAC.

For instances in category R2, shown in Figure 6, we can observe that notches from five out of 11 instances are overlapping (R204, R306, R207, R209 and R210), where the variability of the median is pretty much similar. The median of the results from GAC is lower in five of the six remaining instances and only in one (R212) the median from BiEA is below that from GAC.

Finally, for instances in category RC2, in Figure 7, we can see overlapping notches for half of the instances (RC203, RC204, RC206 and RC207), lower boxes from GAC, consequently lower medians and their variability, for three of the remaining instances, and only in one instance (RC208) results from BiEA surpassed that from GAC.

5.2 Effect of the similarity measure

The same series of experiments was carried out using our algorithm but without the similarity measure (BiEA_{NS}). In this case, the selection of parents for the recombination process took only the fitness into account. The purpose of this

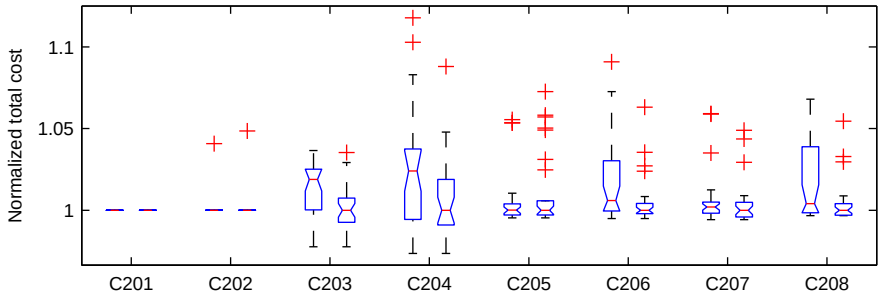


Fig. 5. Box-and-whisker plot for instances in category C2.

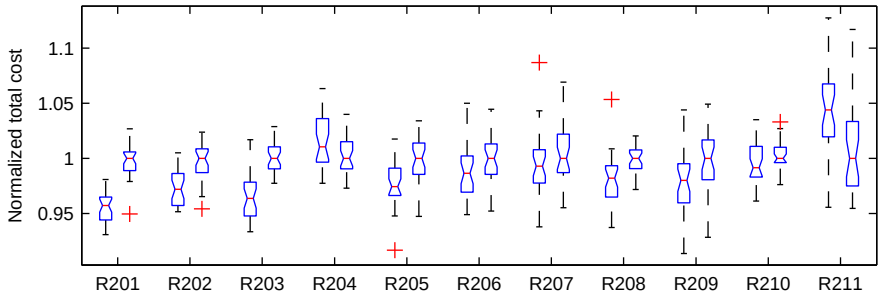


Fig. 6. Box-and-whisker plot for instances in category R2.

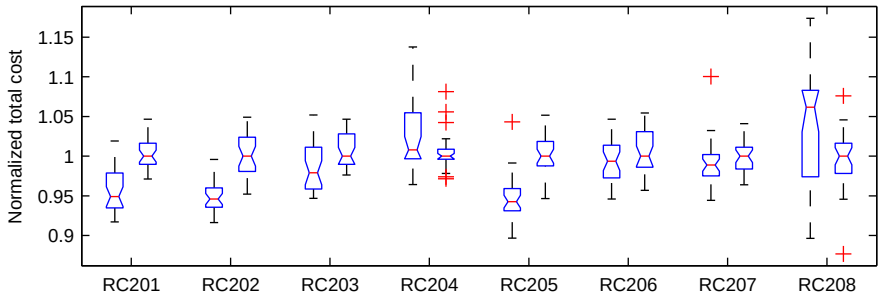
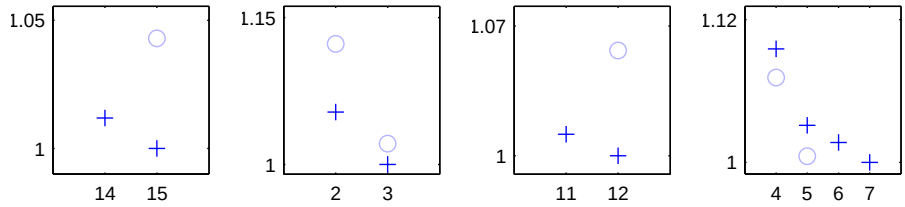


Fig. 7. Box-and-whisker plot for instances in category RC2.

analysis was two-fold: to determine the performance of BiEA with and without the similarity measure, and to determine whether the similarity measure is accomplishing the goal of diversifying the population.

In Table 1, we can see that BiEA_{NS}, compared with BiEA, could find, on average, solutions with a lower number of routes only for instances in category R2 and RC2, and higher in all others. These solutions have a higher total cost for all categories. The difference in these results is the effect of including the similarity measure, as we are selecting one of the parents to be not so similar



(a) R105 (b) R208 (c) RC103 (d) RC201

Fig. 8. Solutions in the Pareto approximation for four instances. Solutions from BiEA are represented with '+', and solutions from BiEA_{NS} with '∘'.

to the rest of the population, and hence looking for solutions in other areas of the search space. This selection could result, after recombination, in an offspring with good quality and different from current individuals.

In Figure 8 we consider the solutions in the Pareto approximations, from one of the repetitions. Because of the space limitations, only four instances are shown, two in category R and two in category RC. In each case, the horizontal axis shows the number of routes, and the vertical axis shows the total cost. Values for the total cost are normalized to the minimum value across both sets. It can be seen that solutions from BiEA are dispersed over at least two values in the number of routes, in contrast with the solutions found by BiEA_{NS}, which are in two at most. We can also observe that, in three of the four instances, the approximation set from BiEA completely cover that from BiEA_{NS}. Although we are displaying the results for only four instances, nearly all of the 39 instances in categories R and RC exhibit similar results. This characteristic is due to the fact that BiEA searches over a wide range of values, no matter if the number of vehicles is higher, as long as the similarity of solutions remains low.

5.3 Comparison with recently published results

Unfortunately, other recent publications dealing with the same problem have not presented Pareto approximations, even when their authors considered their approach to be multi-objective. Consequently, we cannot compare our results with theirs from a proper multi-objective point of view. Instead, we have averaged the best result in all iterations over the instances in each category set, as this appears to be the most common way in the literature to present and compare results. Our results are shown in Table 2, which has the same format as Table 1. This table also includes the results from four recent studies that minimize both objectives, the number of routes and the total cost, either one after another [7, 9] or simultaneously [17, 12]. Additionally, in the last two rows, we show the percentage difference between our results and the lowest total cost (ΔL) and the highest total cost (ΔH) for each instance set.

Analysing the results in Table 2, we can see that, for instance set C1, our algorithm obtained, on average, the highest cost, but the gap between this and

Table 2. Comparison of the best results, averaged by set category, with others previously published.

Author	C1	C2	R1	R2	RC1	RC2	Accum.
[9]	10.00	3.00	12.08	2.73	11.50	3.25	407.00
	828.38	589.86	1209.19	960.95	1386.38	1133.30	57412.37
[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
[17]	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
[12]	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
BiEA	10.00	3.00	12.50	3.18	12.38	4.00	430.00
	830.64	589.86	1191.22	926.97	1349.81	1080.11	56125.35
ΔL	0.27	0.08	0.33	3.80	0.00	5.35	0.69
ΔH	0.00	0.14	1.81	3.67	2.71	4.92	2.29

the lowest result is very narrow, only 0.27%. On the other hand, for set RC1, our algorithm managed to find the lowest costs, and the difference between ours and the highest is 2.71%. For the other categories, although the results from our algorithm are not the overall best, they do show considerable improvement over some of the other algorithms. In fact, they occupy the second or third place among the five. Moreover, in the case of the accumulated total cost, the difference between our results and the lowest is 0.69%, second best, and the difference with the highest is 2.29%, despite our algorithm using a larger number of routes. Another interesting observation is that our results for sets R, RC, and Accumulated confirm that VRPTW really is a multi-objective problem, in the sense that we obtained lower costs, compared with other the authors, despite using more routes.

Finally, although we do not have room to show the details of our results in this paper, we can note that our algorithm has found 11 new best solutions and another 36 similar to the best published.

6 Conclusions

We have proposed in this paper an evolutionary algorithm for solving VRPTW as a bi-objective problem, simultaneously minimizing the number of routes and the total cost. Importantly, this EA includes a similarity measure, which is used to select one of the two parents for the recombination process. As the other parent is selected according to the quality of the solution, the resulting offspring inherits the quality from this parent, while searching for solutions in a non-common area of the search space. As a consequence, solutions in the resulting

population are diverse, covering more than one value in the number of routes dimension.

We have compared the results from our algorithm with those from a single-objective genetic algorithm, with those from our algorithm without the similarity measure, and with algorithms from four recent publications by other authors.

In the first case, we showed that, because the single-objective GA focuses on only one of the two objectives, they do not take into account the likely improvement that could be achieved if the other objective would also be minimized, like the bi-objective evolutionary algorithm does. For this reason, it is highly possible that the single-objective algorithm gets stuck in a suboptimal solution.

In the second case, we have demonstrated the high importance that the similarity measure has, in the sense that the exploration of the search space is wider. Moreover, the solutions from our algorithm are more dispersed, for example, over two values in the number of routes, in contrast with those obtained without considering similarity, which are concentrated in only one.

In the last case, we have shown that, when compared with other algorithms from recent publications, although our results are not the overall best, they are better than some, and, on average, competitive. Our algorithm also managed to find solutions such that the accumulated travel distance is better than others, despite the number of routes being larger, indicating the multi-objective nature of VRPTW.

Given the promising performance of our algorithm, we are now looking at further ways to exploit the similarity measure, further similarity measures, and more rigorous comparisons of our results with other evolutionary multi-criterion optimization methods using multi-objective performance metrics such as coverage and convergence. We are also exploring the extension of our approach to the minimization of at least one more objective, which could be the makespan or the waiting time. Finally, we are also planning to apply our BiEA to other variants of the VRP, such as one with pick-ups and deliveries.

Acknowledgements

The first author acknowledges support from the Mexican National Council of Science and Technology (CONACYT) through scholarship 179372 to pursue his graduate studies. Both authors are thankful for the valuable comments from the anonymous reviewers.

References

1. Berger, J., Barkaoui, M., Bräysi, O.: A Route-directed Hybrid Genetic Approach for the Vehicle Routing Problem with Time Windows. INFOR. 41, 179–194 (2003).
2. Bräysy, O., Gendreau, M.: Vehicle Routing Problem with Time Windows, Part I: Route Construction and Local Search Algorithms. Transport. Sci. 39(1), 104–118 (2005).

3. Bräysy, O., Gendreau, M.: Vehicle Routing Problem with Time Windows, Part II: Metaheuristics. Transport. Sci. 39(1), 119–139 (2005).
4. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE T. Evolut. Comput. 6(2), 182–197 (2002).
5. Desrochers, M., Desrosiers, J., Solomon, M.: A New Optimization Algorithm for the Vehicle Routing Problem with Time Windows. Oper. Res. 40(2), 342–354 (1992).
6. Garcia-Najera, A., Bullinaria, J.A.: A Multi-Objective Density Restricted Genetic Algorithm for the Vehicle Routing Problem with Time Windows. In: 2008 UK Workshop on Computational Intelligence. Leicester, UK (2008).
7. Homberger, J., Gehring, H.: A Two-phase Hybrid Metaheuristic for the Vehicle Routing Problem with Time Windows. Eur. J. Oper. Res. 162, 220–238 (2005).
8. Jozefowicz, N., Semet, F., Talbi, E.: Multi-objective Vehicle Routing Problems. Eur. J. Oper. Res. 189, 293–309 (2008).
9. Le Bouthillier, A., Crainic, T.G.: A Cooperative Parallel Metaheuristic for Vehicle Routing with Time Windows. Comput. Oper. Res. 32, 1685–1708 (2005).
10. Martí, R., Laguna, M., Campos, V.: Scatter Search vs. Genetic Algorithms: An Experimental Evaluation with Permutation Problems. In: Rego, C., Alidaee, B. (eds.) Metaheuristic Optimization Via Adaptive Memory and Evolution: Tabu Search and Scatter Search, pp. 263–282. Kluwer Academic (2005).
11. McGill, R., Tukey, J.W., Larsen, W.A.: Variations of Box Plots. Am. Stat. 32(1), 12–16 (1978).
12. Ombuki, B., Ross, B.J., Hanshar, F.: Multi-Objective Genetic Algorithms for Vehicle Routing Problem with Time Windows. Appl. Intell. 24(1), 17–30 (2006).
13. Potvin, J.Y., Bengio, S.: The Vehicle Routing Problem with Time Windows — Part II: Genetic Search. INFORMS J. Comp. 8, 165–172 (1996).
14. Ronald, S.: More distance functions for order-based encodings. In: 1998 IEEE International Conference on Evolutionary Computation, pp. 558-563, IEEE Press, Piscataway, NJ, USA (1998).
15. Sörensen, K.: Distance Measures Based on the Edit Distance for Permutation-type Representations. J. Heuristics 13(1), 35–47 (2007).
16. Solomon, M.M.: Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. Oper. Res. 35(2), 254–265 (1987).
17. Tan, K.C., Chew, Y.H., Lee, L.H.: A Hybrid Multiobjective Evolutionary Algorithm for Solving Vehicle Routing Problem with Time Windows. Comput. Optim. Appl. 34(1), 115–151 (2006).
18. Tavares, J., Machado, P., Pereira, F.B., Costa, E.: On the Influence of GVR in Vehicle Routing. In: 2003 ACM Symposium on Applied Computing, pp. 753–758. ACM Press, New York, NY, USA, (2003).
19. Toth, P., Vigo, D.: The Vehicle Routing Problem. SIAM, Philadelphia, PA, USA (2001).
20. Zhu, K.Q.: A Diversity-Controlling Adaptive Genetic Algorithm for the Vehicle Routing Problem with Time Windows. In: 15th IEEE International Conference on Tools with Artificial Intelligence, pp. 176–183. IEEE Computer Society, Washington, DC, USA (2003).
21. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization. In: Giannakoglou, K., Tsahalis, D., Periaux, J., Papailiou, K., Fogarty, T. (eds.) Evolutionary Methods for Design, Optimisation and Control, pp. 19–26. CIMNE (2002).