An improved multi-objective evolutionary algorithm for the vehicle routing problem with time windows

Abel Garcia-Najera[*,a], John A. Bullinaria[a]

[a]*School of Computer Science, University of Birmingham*
Birmingham, B15 2TT, United Kingdom

Abstract

The Vehicle Routing Problem with Time Windows is a complex combinatorial problem with many real-world applications in transportation and distribution logistics. Its main objective is to find the lowest distance set of routes to deliver goods, using a fleet of identical vehicles with restricted capacity, to customers with service time windows. However, there are other objectives, and having a range of solutions representing the trade-offs between objectives is crucial for many applications. Although previous research has used evolutionary methods for solving this problem, it has rarely concentrated on the optimization of more than one objective, and hardly ever explicitly considered the diversity of solutions. This paper proposes and analyzes a novel multi-objective evolutionary algorithm, which incorporates methods for measuring the similarity of solutions, to solve the multi-objective problem. The algorithm is applied to a standard benchmark problem set, showing that when the similarity measure is used appropriately, the diversity and quality of solutions is higher than when it is not used, and the algorithm achieves highly competitive results compared with previously published studies and those from a popular evolutionary multi-objective optimizer.

Key words: Vehicle routing problem, combinatorial optimization, multi-objective optimization, evolutionary algorithms.

1. Introduction

The Vehicle Routing Problem (VRP) is one of the most important and widely studied combinatorial optimization problems, with many real-world applications in distribution and transportation logistics [1]. Its objective is to obtain the lowest-cost set of routes to deliver demand to customers. But what does *lowest-cost* mean? Since the problem was proposed by Dantzig and Ramser [2] as a generalization of the Traveling Salesman Problem, cost has mostly been associated with the travel distance, but there are several other types of cost involved, such as the number of vehicles and delivery time [3]. The VRP also has several variants that involve different constraints. The variant *with Time Windows* (VRPTW) has vehicles with limited capacity and specific delivery time windows, and is particularly relevant to practical applications. With such constraints, the minimization of one objective rarely corresponds to the minimization of all of them, and the optimization process needs to provide a range of solutions that represent the trade-offs between the objectives, rather than a single solution.

Optimal solutions for small instances of the VRPTW can be obtained using exact methods, but the computation time required increases considerably for larger instances [4], so heuristic methods are usually employed. Cordeau et al. [5] review a number of different approaches, and the recent surveys by Bräysy and Gendreau [6, 7] provide a complete list of studies utilizing a number of heuristics and a comparison of the results obtained. Although numerous metaheuristic methods have been proposed, this paper concentrates on those using evolutionary computation methods, since they have a natural approach for dealing with multi-objective problems, and have been successful in many practical situations [8]. Evolutionary Algorithms (EA) are based on Darwin's theory of evolution by natural selection: A *population* (set) of *individuals* (solutions) is maintained, and the EA *selects*, *recombines*, and *mutates* the *fittest* (best solutions) to replace the least fit, in the hope of producing solutions of increased *fitness* (quality). The evolutionary operations are repeated until the quality of solutions stops increasing, or some fixed number of *generations* (cycles) has been reached. A number of evolutionary and hybrid algorithms tackling the VRPTW are analyzed by Bräysy et al. [9]. Section 3 reviews the key approaches from that survey, as well as some more recent studies.

The remainder of this paper is organized as follows: Section 2 provides definitions of the key issues relevant to

[*]Corresponding author
Email addresses: `A.G.Najera@cs.bham.ac.uk` (Abel Garcia-Najera), `J.A.Bullinaria@cs.bham.ac.uk` (John A. Bullinaria)

this study, namely the formal specification of the VRPTW, and a brief introduction to the technicalities of multi-objective optimization and the associated performance metrics. Then Section 3 reviews the previous studies in these areas. In Section 4 is described the multi-objective evolutionary algorithm proposed here for solving the VRPTW problem, and the underlying similarity measure. The experimental design used to test the algorithm is detailed in Section 5, and the analysis of results is presented in Section 6. Finally, some conclusions and ideas for future research in this area are provided in Section 7.

2. Problem Definitions

This section provides an overview of the two crucial topics relevant to this study: the formal description of the VRPTW and the technicalities of multi-objective optimization.

2.1. The vehicle routing problem with time windows

The VRPTW is a complex combinatorial optimization problem which is NP-hard [10]. The objective is to find a minimum-cost set of routes to deliver demand to customers, who have specific delivery time windows, by using a fleet of identical vehicles with limited capacity.

An instance of the VRPTW can be formally defined as follows. First, there is a set $\mathcal{V} = \{0, \ldots, N\}$ of vertices, and vertices in subset $\mathcal{V}^* = \mathcal{V} \setminus \{0\} = \{1, \ldots, N\}$ are called *customers*. Each customer $i \in \mathcal{V}^*$ is geographically located at coordinates (x_i, y_i), has a demand of goods $g_i > 0$, a time window $[b_i, e_i]$ during which it must be supplied, and a service time s_i required to unload its goods. The special vertex 0 is called the *depot*, from which the customers are serviced using a homogeneous fleet of vehicles with capacity $Q \geq \max\{g_i : i \in \mathcal{V}^*\}$. The depot is positioned at (x_0, y_0), has demand $g_0 = 0$, and time window $[0, e_0 \geq \max\{e_i : i \in \mathcal{V}^*\}]$.

The travel between vertices i and j has associated costs, such as the distance d_{ij} (relating to fuel cost) and travel time t_{ij} (relating to driver cost). For the benchmark problems to be considered later, unit velocity and direct travel are assumed, so the times and distances are both simply taken to be the Euclidean distances

$$t_{ij} = d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \ . \tag{1}$$

For real-world problems, however, the distances d_{ij} are unlikely to be Euclidean and the travel times t_{ij} are unlikely to be simply related to the distances. The following accommodates those possibilities.

The problem is to design a lowest-cost set of K routes $\mathcal{R} = \{r_1, \ldots, r_K\}$, such that each route begins and ends at the depot, and each customer is serviced by exactly one vehicle. So each vehicle is assigned a set of customers that it has to supply, with the sum of their demands not exceeding the vehicle capacity Q. If $r_k = \langle u(1,k), \ldots, u(N_k,k)\rangle$ specifies the sequence of N_k customers supplied in the k-th route, where $u(i,k)$ is the i-th customer to be visited in route k, then $\mathcal{V}_k^* = \{u(1,k), \ldots, u(N_k,k)\}$ is the set of customers serviced. The depot does not appear explicitly in this notation, but it has to be taken into account before the first and after the last customers when computing the costs, i.e. as $u(0,k) = u(N_k+1,k) = 0$. Then

$$D_k = \sum_{i\,=\,0}^{N_k} d_{u(i,k)u(i+1,k)} \tag{2}$$

is the total travel distance associated with route r_k.

In addition to defining the distances, the times are also required. Let $a(u(i,k))$ denote the arrival time of vehicle k at vertex i and $l(u(i,k))$ be the time it leaves, and have each vehicle k leave the depot at time 0, i.e. $l(u(0,k)) = 0$. The arrival time of vehicle k at the i-th customer is then

$$a(u(i,k)) = l(u(i-1,k)) + t_{u(i-1,k)u(i,k)} \ . \tag{3}$$

Arriving after the end of the customer's time window is not allowed, rendering the route invalid. However, arriving early *is* allowed, but then the vehicle will have to wait until the beginning of the customer time window to start unloading the goods, so there will be a waiting time

$$w(u(i,k)) = \begin{cases} 0 \text{ if } a(u(i,k)) \geq b_{u(i,k)} \\ b_{u(i,k)} - a(u(i,k)) \text{ otherwise} \end{cases} . \tag{4}$$

Thus, the leaving time from the i-th customer in route k is

$$l(u(i,k)) = a(u(i,k)) + w(u(i+1,k)) + s_{u(i,k)} \tag{5}$$

and the total time required to complete route r_k is the arrival time at the depot

$$T_k = \sum_{i\,=\,0}^{N_k} \left(t_{u(i,k)u(i+1,k)} + w(u(i+1,k)) + s_{u(i+1,k)} \right) \ . \tag{6}$$

Having defined the VRPTW, one can specify any number of relevant objective functions f_i to optimize. This paper will concentrate on the three key objectives, namely the number of routes or vehicles

$$f_1(\mathcal{R}) = |\mathcal{R}| = K \ , \tag{7}$$

the total travel distance given by summing the route distances

$$f_2(\mathcal{R}) = \sum_{k\,=\,1}^{K} D_k \ , \tag{8}$$

and the total travel time given by summing the arrival times back at the depot

$$f_3(\mathcal{R}) = \sum_{k\,=\,1}^{K} T_k \ , \tag{9}$$

subject to the demand associated with customers serviced in route r_k not exceeding the vehicle capacity

$$G_k = \sum_{i \in \mathcal{V}_k^*} g_i \leq Q \qquad \forall\, k = 1, \dots K \ , \tag{10}$$

and the arrival times being within the customer's constraints

$$b_{u(i,k)} \leq a(u(i,k)) \leq e_{u(i,k)}, \qquad \begin{array}{l} \forall\, k = 1, \dots K \\ 1 \leq i \leq N_k \end{array} \ . \tag{11}$$

2.2. Multi-objective optimization problems

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

$$\text{minimize } \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \dots, f_F(\mathbf{x})) \tag{12}$$

subject to constraints:

$$g_i(\mathbf{x}) \leq 0 \qquad \forall\, i = 1, \dots, p \tag{13}$$

$$h_j(\mathbf{x}) = 0 \qquad \forall\, j = 1, \dots, q \tag{14}$$

where $\mathbf{x} = (x_1, \dots, x_n) \in \mathcal{X}$ is the vector of decision variables, $\mathcal{X}$ is the parameter space, and $f_i : \mathbb{R}^n \to \mathbb{R}$, for $i = 1, \dots, F$, are the F objective functions. The constraint functions $g_i, h_j : \mathbb{R}^n \to \mathbb{R}$ in (13) and (14) restrict $\mathbf{x}$ so that only feasible solutions are considered.

A decision vector $\mathbf{x} \in \mathcal{X}$ is said to *cover* a decision vector $\mathbf{y} \in \mathcal{X}$ ($\mathbf{x} \preceq \mathbf{y}$) if $f_i(\mathbf{x}) \leq f_i(\mathbf{y})$, $\forall\, i = 1, \dots, F$. Vector $\mathbf{x}$ *dominates* $\mathbf{y}$ ($\mathbf{x} \prec \mathbf{y}$) if and only if $\mathbf{x} \preceq \mathbf{y}$ and $\exists\, j \in \{1, \dots, F\} : f_j(\mathbf{x}) < f_j(\mathbf{y})$. Similarly, one says 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. 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 $\mathcal{P}_f = \{\mathbf{f}(\mathbf{x}) \in \mathbb{R}^n \mid \mathbf{x} \in \mathcal{P}_s\}$.

For multi-objective problems, heuristics generally have two aims [11]: First, to minimize the distance of the generated solutions, called the *Pareto approximation*, from the true Pareto front, and second, to maximize the diversity of them, i.e. the coverage of the Pareto front. In the literature, there are already a number of evolutionary multi-objective optimizers that successfully address these aims, such as NSGA-II [12], SPEA2 [13], PAES [14] and IBEA [15], and this paper aims to introduce further improvements. However, the comparison of multi-objective optimizer performance is not easy. In contrast to single-objective problems, where one can straightforwardly compare the best solutions, or averages of them, from the various approaches studied, multi-objective problems have to compare whole sets of solutions, with at least the two aims just mentioned. For this reason, the definition and use of appropriate performance metrics is crucial. Fortunately, this issue has already been widely studied, and of the many proposed metrics [16, 17, 18, 19, 20], two are particularly applicable to the problem at hand. These, the *hypervolume* proposed by Zitzler and Thiele [16] and *coverage* from Zitzler et al. [17], are now defined.

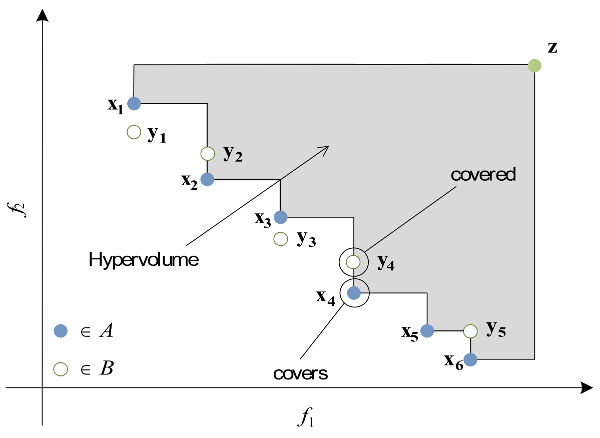


Figure 1: Graphical representation of the hypervolume $\mathsf{M_H}$ and coverage $\mathsf{M_C}$ metrics.

2.2.1. Hypervolume

This metric concerns the size of the objective space covered by a set $\mathcal{A}$ of solutions, which is limited by setting a suitable reference point. Figure 1 shows an example of this metric, where the solution points $\mathbf{x_1}, \dots, \mathbf{x_6} \in A$ cover the shaded region limited by the reference point $\mathbf{z}$. For maximization problems, it is common to take $\mathbf{z}$ to be the origin $(0, 0)$. For minimization problems, $\mathbf{z}$ is set to exceed the maximal values for each objective. Either way, when using this metric to compare the performance of two or more algorithms, the one providing solutions with the largest covered hypervolume is deemed to be the best.

Formally, for a two-dimensional objective space, each solution $\mathbf{x} \in \mathcal{A}$ covers a rectangle defined by its coordinates $(f_1(\mathbf{x}), f_2(\mathbf{x}))$ and the reference point $\mathbf{z} = (z_{f_1}, z_{f_2})$, and the size of the union of all such rectangles covered by the solutions is used as the measure. This concept can be extended to any number of dimensions D to give the general hypervolume metric [16]:

$$\mathsf{M_H}(\mathcal{A}) = \lambda \left(\bigcup_{\mathbf{x} \in \mathcal{A}} \{[f_1(\mathbf{x}), z_{f_1}] \times \cdots \times [f_D(\mathbf{x}), z_{f_D}]\} \right) \tag{15}$$

where $\lambda(\cdot)$ is the standard Lebesgue measure.

2.2.2. Coverage

This performance metric measures the extent to which one solution set $\mathcal{B}$ is covered by another solution set $\mathcal{A}$. It compares the number of solutions in $\mathcal{B}$ that are covered by solutions in $\mathcal{A}$ to the cardinality of $\mathcal{B}$. Formally, this ratio maps the ordered pair $(\mathcal{A}, \mathcal{B})$ to the interval [0,1] as the general coverage metric [17]:

$$\mathsf{M_C}(\mathcal{A}, \mathcal{B}) = \frac{|\{\mathbf{b} \in \mathcal{B} : \exists\, \mathbf{a} \in \mathcal{A}, \mathbf{a} \preceq \mathbf{b}\}|}{|\mathcal{B}|} \ . \tag{16}$$

The value $\mathsf{M_C}(\mathcal{A}, \mathcal{B}) = 1$ means that all solutions in $\mathcal{B}$ are covered by solutions in $\mathcal{A}$, while $\mathsf{M_C}(\mathcal{A}, \mathcal{B}) = 0$ indicates that none of the solutions in $\mathcal{B}$ are covered by those in $\mathcal{A}$.

Since $\mathsf{M_C}(\mathcal{A},\mathcal{B})$ does not necessarily equal $1 - \mathsf{M_C}(\mathcal{B},\mathcal{A})$, both $\mathsf{M}_{\mathcal{C}}(\mathcal{A},\mathcal{B})$ and $\mathsf{M_C}(\mathcal{B},\mathcal{A})$ need to be computed.

The idea here is that the algorithm with the best performance is the one which provides solutions with the largest coverage of the solutions from the others. In Figure 1 there are 6 filled circles $\mathbf{x_1},\ldots,\mathbf{x_6} \in A$ and 5 open circles $\mathbf{y_1},\ldots,\mathbf{y_5} \in B$. Three points in set B (y_2, y_4, and y_5) are covered by set A, and two points in A (x_1 and x_3) are covered by set B, so $\mathsf{M_C}(A,B) = 3/5$, and $\mathsf{M_C}(B,A) = 2/6$. Thus the algorithm providing solutions A is deemed better than that providing solutions B.

3. Previous Studies

This section provides a brief overview of the key past work of relevance to the current study.

Rahoual et al. [21] designed a Genetic Algorithm (GA) for the VRPTW based on the well-known NSGA [22] for minimizing the number of routes, the travel distance, and the penalties associated with violated constraints. Jung and Moon [23] then went on to propose a hybrid GA which used a 2D image of a solution to perform the crossover process, based on dividing the routes in the selected solutions into sequences of different types of curves drawn on the 2D space where customers are located, with a repair mechanism to re-connect the sequences. This algorithm continued with the use of three local optimization techniques. Later, Zhu [24] presented a GA that adapts the cross-over and mutation rates to the population dynamics, maintaining population diversity at user-defined levels, and thus preventing premature convergence. He considered a permutation based solution representation and a decoding mechanism which is $O(N^3)$, where N is the number of customers.

Jozefowiez et al. [25] addressed a VRP in which the total route length and the route imbalance are minimized simultaneously, and implemented an enhancement of the popular NSGA-II [12]. Murata and Itai [26] proposed a two-fold evolutionary multi-objective algorithm for solving a variant of the VRP in which customers have normal and high demands. They also proposed a similarity measure to show the importance of examining characteristics of solutions to both problems.

Le Bouthillier and Crainic [27] presented a parallel cooperative multi-search method for the 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 methods implemented a different meta-heuristic, an EA or a tabu search procedure [28].

Homberger and Gehring [29] proposed a two-phase hybrid meta-heuristic to solve the VRPTW, in which the first phase aims at minimizing the number of routes by means of a (μ,λ)-evolution strategy [30], and the second phase minimizes the total distance using a tabu search algorithm.

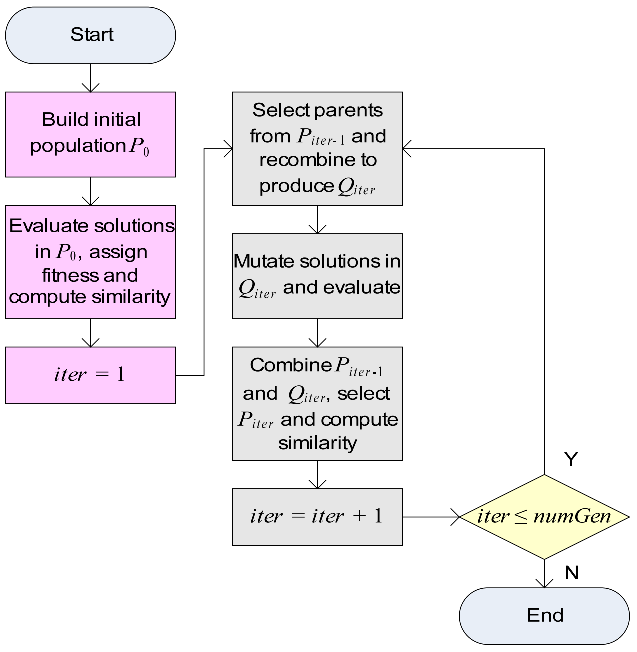
Most recently, Tan et al. [31] and Ombuki et al. [32] considered the VRPTW as a bi-objective optimization problem, minimizing the number of vehicles and the total travel distance, and used a GA for solving it. The former used the dominance rank scheme to assign fitness to individuals, designed a problem-specific crossover operator 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 that were applied every 50 generations. The latter proposed the genetic operators *best cost route crossover* and *constrained route reversal mutation*, which is an adaptation of the widely used inversion method.

Only a couple of these past studies explicitly considered using a method to measure the similarity of solutions, and just one has utilized this information with the aim of preserving population diversity. It is also worth noting that, although many widely-known and successful multi-objective evolutionary approaches, such as SPEA2 [33] and NSGA-II [12], incorporate population diversity preservation techniques, they 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 range of vehicle numbers [32].

In our own previous work [34], it became clear that the lack of population diversity was a crucial factor leading EAs to become stuck in suboptimal VRPTW solutions, so we proposed a method to restrict the number of clones. That algorithm eventually forced the population to have no clones at all, but the solutions were still not good enough. Consequently, we then designed a new Bi-objective EA (BiEA) [35] which incorporated a similarity measure, based on Jaccard's similarity coefficient (defined in Section 4.5), to select parents for the recombination process in a way that preserved a higher population diversity [36], and that enabled good solutions to be obtained for a set of publicly available benchmark instances. Unfortunately, it was not possible to compare the results against other approaches in a fully multi-objective manner, because there were no publicly available results showing values for all the objectives involved for each instance.

We then went on to test our BiEA with the widely-used Edit distance [37] replacing our Jaccard measure, and established that both measures resulted in similar performance, but the Edit distance required much longer execution times [38]. For that reason, this paper will use the Jaccard's similarity measure.

The remainder of this paper presents an improved version of our BiEA that addresses the limitations identified by the earlier work. The new algorithm has more effective recombination and mutation processes, a more extensive series of experiments are carried out, and there is an extended analysis of the results, including comparisons with those from NSGA-II using fully multi-objective performance metrics. It is also shown how the algorithm can easily optimize more than two objectives with equally successful results.



Start
Build initial population P_0
Evaluate solutions in P_0, assign fitness and compute similarity
$iter = 1$
Select parents from P_{iter-1} and recombine to produce Q_{iter}
Mutate solutions in Q_{iter} and evaluate
Combine P_{iter-1} and Q_{iter}, select P_{iter} and compute similarity
$iter = iter + 1$
$iter \leq numGen$
Y
N
End

Figure 2: General process of the proposed MOEA.

4. Improved Multi-Objective EA for solving the VRPTW

This section presents the proposed algorithm for solving the multi-objective VRPTW, and the crucial solution similarity measure that enables it to provide a better approximation to the full Pareto front.

4.1. The Multi-Objective EA (MOEA)

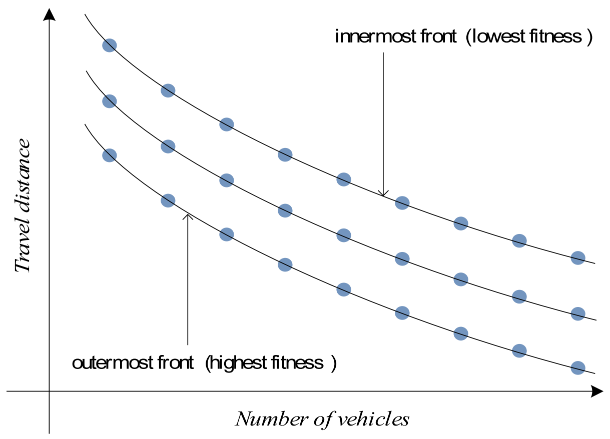
EAs are optimizers based on Darwin's theory of evolution, where the *fittest* individuals survive and produce offspring to populate the next generation [39]. A *population* of *individuals* is maintained, in which each individual is a problem solution, and *fitness* is some appropriate measure of how good an individual solution is. The operation of a particular EA is defined by a number of procedures or operators, and crucial to this is how the offspring are created from the parents. The general process of the proposed MOEA is presented in the flowchart in Figure 2, and the specific features are described below.

4.2. Solution encoding

Since the VRPTW solutions are lists of routes, which are themselves lists of customers, the appropriate encoding here is a list of lists. A solution encoding simply lists the customer identifiers in the order they are serviced.

4.3. Initial population

As is standard practice for EAs, the initial population is chosen randomly with the aim of covering the entire search space. Thus the MOEA here starts with a set of *popSize* solutions, each being a randomly generated feasible route constructed as follows: First, a customer is selected at random and placed as the first location to be visited on the first route. Then, a different random customer is chosen and, if the capacity and time constraints would be met, it is placed on the current route after the previous customer. If any of the constraints are not met, a new route is created and this customer is the first location to be visited on that route. This process is repeated until all customers have been assigned to a route.



innermost front (lowest fitness)
outermost front (highest fitness)
Travel distance
Number of vehicles

Figure 3: Fitness assignment to solutions.

4.4. Fitness assignment

At each generation of evolution, the objective functions are evaluated for every solution in the population, and each individual is assigned a fitness value which drives the natural selection process. When solving a single-objective problem, fitness is easily assigned to an individual according to its single objective function evaluation. In the multi-objective case, however, this assignment cannot be done straightforwardly because there is naturally more than one possible objective that can be used, and a whole set of solutions is required that reflects the trade-offs between them. One approach to assigning fitness to solutions, that satisfies the current requirements, is the non-dominance sorting criterion of Deb et al. [12]. In this, the population is divided into non-dominated fronts, and their depth specifies the fitness of the individuals belonging to them, as shown for two objectives in Figure fig:fitness. In this case, the lower the front, the fitter the solution.

4.5. Solution similarity measure

Maintaining population diversity is crucial for EAs, in that their success depends on the avoidance of premature convergence and the balancing of the trade-off between exploration and exploitation of the search space [39]. For multi-objective algorithms, it is also important that the final population contains solutions that represent the full Pareto front, rather than just a small portion of it. Diversity here not only refers to the number of distinct solutions in the population, but also to how different they are. It is usually easy to make sure that there are no duplicates in the population, but evaluating solution *spread* and using it to boost the diversity generally requires the development of encoding-specific tools.

To accomplish this with the VRPTW encoding, a similarity measure was designed based on Jaccard's similarity coefficient, which measures the similarity of two sets as the ratio of the cardinalities of the intersection and the union of those sets [35]. Formally, the similarity of sets A and B is

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} \; . \tag{17}$$

Thus, if both sets contain the same elements, the intersection equals the union, and the similarity is 1. If the two sets do not share any elements, the intersection is the empty set, and the similarity is 0.

The natural way to implement this measure for the VRPTW is to consider each solution $\mathcal{R}$ as the union of a set of segments or arcs $(u(i,k), u(i+1,k))$, so

$$\mathcal{R} = \bigcup_{k=1}^{K} \bigcup_{i\,=\,0}^{N_k} \{(u(i,k), u(i+1,k))\} \; . \tag{18}$$

Then the similarity of two solutions equals the ratio between the number of arcs that are common to both solutions and the total number of arcs used by them. Denoting $y_{ij\mathcal{R}} = 1$ if arc (i,j) is traversed by any vehicle in solution $\mathcal{R}$, and 0 otherwise, the similarity $\varsigma_{\mathcal{R}\mathcal{Q}}$ between solutions $\mathcal{R}$ and $\mathcal{Q}$ is then

$$\varsigma_{\mathcal{R}\mathcal{Q}} = \frac{\displaystyle\sum_{i,j\,\in\,\mathcal{V}} y_{ij\mathcal{R}} \cdot y_{ij\mathcal{Q}}}{\displaystyle\sum_{i,j\,\in\,\mathcal{V}} \text{sign}\,(y_{ij\mathcal{R}} + y_{ij\mathcal{Q}})} \; , \tag{19}$$

in which the term in the sum in the numerator will only equal 1 if arc (i,j) is used by both solutions, while that in the denominator will equal 1 if either solution uses it. Arcs (i,j) and (j,i) are considered to be different, even if their cost is the same. Thus, if solutions $\mathcal{R}$ and $\mathcal{Q}$ are the same, $\varsigma_{\mathcal{R}\mathcal{Q}} = 1$, while if they are two completely different solutions with no arc in common, $\varsigma_{\mathcal{R}\mathcal{Q}} = 0$. The algorithm to compute this Jaccard similarity is $O(N)$.

For the purposes of the proposed MOEA, a measure of how similar a given solution is to the rest of the evolutionary population is also required. If $\boldsymbol{P}$ is the population of solutions, and $|\boldsymbol{P}| = M$ is the population size, the similarity $\sigma_{\mathcal{R}}$ of solution $\mathcal{R} \in \boldsymbol{P}$ with the rest of the solutions in $\boldsymbol{P}$ will be given by the average similarity of $\mathcal{R}$ with every other solution $\mathcal{Q} \in \boldsymbol{P}$, that is

$$\sigma_{\mathcal{R}} = \frac{1}{M-1} \sum_{\mathcal{Q}\,\in\,\boldsymbol{P}\setminus\{\mathcal{R}\}} \varsigma_{\mathcal{R}\mathcal{Q}} \; . \tag{20}$$

The total complexity of the algorithm needed to compute this is $O(NM^2)$. Finally,

$$\delta = 1 - \frac{1}{M} \sum_{\mathcal{R}\,\in\,\boldsymbol{P}} \sigma_{\mathcal{R}} \tag{21}$$

i.e. one minus the average solution similarity, defines the diversity δ of the population of solutions $\boldsymbol{P}$.

4.6. Parent selection

The evolutionary process requires some stochastic function for selecting *parent* individuals from the population, according to their fitness, to undergo mating (or recombination) to create an *offspring*. The fittest individuals should be more likely to be selected, but low-fitness individuals should also be given a small chance, with the aim of not allowing the algorithm to be too greedy. A standard tournament method [40] achieves this by choosing T_{size} individuals randomly from the population and selecting the best individual from this group to be a parent.

A crucial difference of the MOEA proposed here to most EAs is that, in addition to using fitness to select good parents, it also uses the similarity measure to maintain population diversity. The first of two parents is chosen on the basis of fitness, and the second on the basis of similarity.

4.7. Recombination

Recombination is the process of generating one or more offspring from the selected parents, preferably in a manner that maintains and combines the desirable features from both parents. This is carried out with probability γ, otherwise the fittest individual is simply copied into the offspring population, or if both parents have the same fitness, the parent with lowest similarity is copied.

The MOEA here is designed to randomly select and preserve routes from both parents. First, a random number of routes are chosen from the first parent and copied into the offspring. Then all those routes from the second parent which are not in conflict with customers already copied from the first, are copied into the offspring. The recombination of two example parents is shown in Figure 4, with both routes on the left from the first parent selected to be copied into the offspring, and then only the route on the right can be copied from the second parent, as the other two contain customers already present in the offspring. Finally, if there remain any unassigned customers, these are allocated, in the order they appear in the second parent, to the route where the lowest travel distance is achieved, as in the example shown in Figure 4. If there is no way to insert such a remaining customer into the existing routes without violating a constraint, a new route is created.

4.8. Mutation

Once an offspring has been generated, a further stochastic change or *mutation* is applied with probability μ. This involves three basic functions and three operators. The three functions are:

- *selectRoute* which stochastically selects a route according to the ratio of the travel distance to the

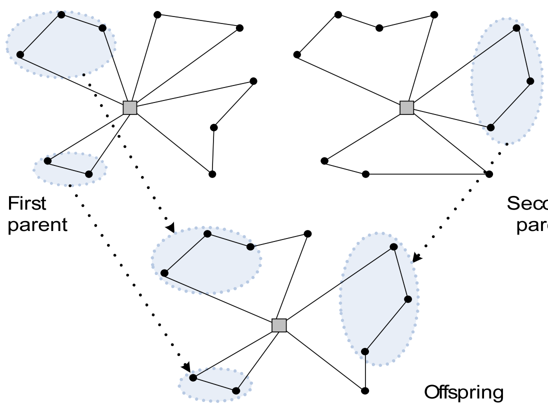


Figure 4: The recombination process.

number of customers, i.e. routes with a larger travel distance and fewer customers are more likely to be selected.

- *selectCustomer* which stochastically selects one customer from a specific route according to the average length of its inbound and outbound arcs, i.e. customers with longer associated travel distances are more likely to be chosen. A special case exists for the first and last customers in a route, where only the outbound and inbound arcs, respectively, are taken into account.

- *insertCustomers* which tries to insert, one at a time, a set of customers into a specific route where the lowest travel distance is obtained. If no route is specified, it tests all existing routes.

These functions are used by the mutation operators:

- *Reallocation* which takes a number of customers from a given route and allocates them to another. First, *selectCustomer* is used to choose two customers from the route. These are removed from the route, along with all those customers in between them. Then, *insertCustomers* attempts to reallocate the removed customers into any of the existing routes, including the one that they were removed from. This operation is illustrated in Figure 5.

- *Exchange* which swaps sequences of customers between two routes chosen by *selectRoute*. First, *selectCustomer* chooses two customers from each route. The sequences of customers between them are then removed from their route, and *insertCustomers* attempts to reallocate them into the other route. If one or more customers cannot be inserted into the other route, the original routes are preserved. This operation is illustrated in Figure 6.

- *Reposition* which uses *selectCustomer* and *insertCustomers* respectively to select one customer from a specific route and to reinsert it into the same route. Figure 7 illustrates this operation.

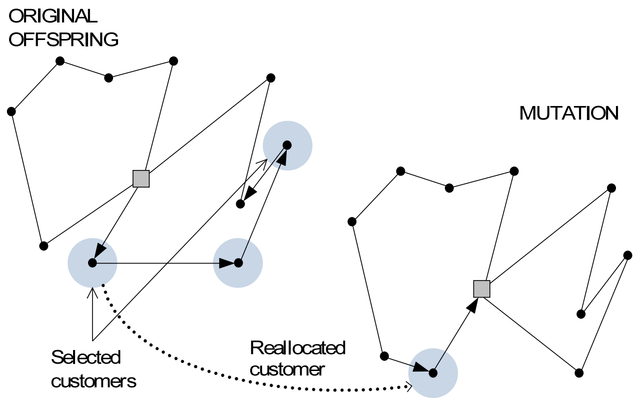


Figure 5: The reallocation mutation operator.

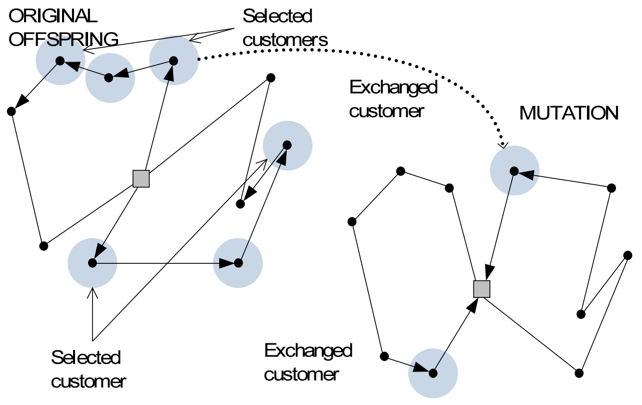


Figure 6: The exchange mutation operator.

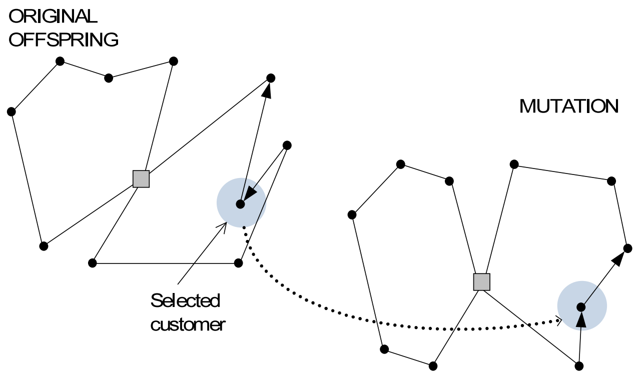


Figure 7: The reposition mutation operator.

The mutation process proceeds as follows: Two routes are chosen using *selectRoute*. If they are the same route, the *reallocation* operation is performed, otherwise the *exchange* operator is executed. Then *selectRoute* selects another route and the *reposition* operator is carried out.

4.9. Survival

The final stage of each evolutionary cycle is the selection of individuals to form the next generation. There are

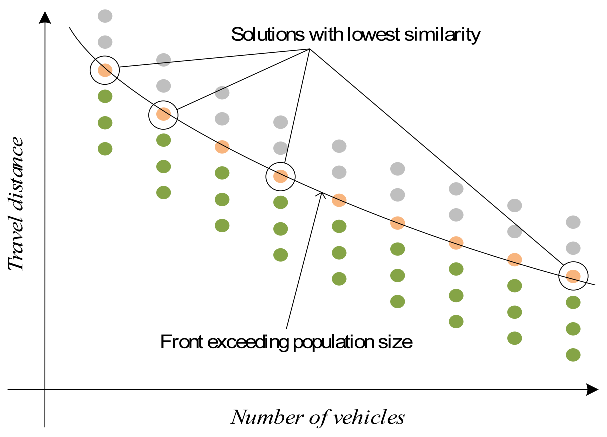

Solutions with lowest similarity
Travel distance
Front exceeding population size
Number of vehicles

Figure 8: Selection of best individuals for the next generation. The fourth front is in conflict with the population size, so the least similar solutions from it are chosen to join the first three fronts.

several obvious possibilities: the offspring population, a random selection from the combined parent and offspring populations, or the best individuals from the combined population. In the first two cases, good-quality individuals are likely to be lost, so the third approach is used here. The offspring and parent populations are combined and individual fitnesses determined as described in Section 4.4. Those solutions having the highest fitness, i.e. falling in the outermost fronts, are taken to survive and form the next generation. When the population size is exceeded in the last selected front, similarity is computed for the solutions in that front, and the least similar are chosen, as shown in Figure 8.

4.10. Repetition

The whole process of parent selection and offspring generation is repeated for a fixed number of generations (*numGen*).

5. Experimental design

This section describes the VRPTW benchmark instances used to test the proposed MOEA algorithm, and specifies the associated experimental design.

5.1. Solomon's benchmark instances

The proposed algorithm was tested on the standard public benchmark set of Solomon [41], which includes 56 instances of size $N = 100$ available from Solomon's web site[1]. These instances are categorised as: C1 and C2, where customers are located in geographical clusters, R1 and R2, where customers are randomly distributed, and RC1 and RC2, which have a mix of random locations and clusters. Moreover, instances in sets C1, R1 and RC1 have a short scheduling horizon, i.e. the time constraint acts as a capacity constraint which, together with the vehicle capacity constraint, allows only a few customers to be serviced by the same vehicle. In contrast, instances in sets C2, R2 and RC2 have a long scheduling horizon, which, coupled with large vehicle capacities, permits many customers to be serviced by the same vehicle [41].

These benchmark instances have been previously studied in detail, and a recent analysis by Tan et al. [31] 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, whereas many of the instances in categories R1, R2, RC1 and RC2 were found to have conflicting objectives.

5.2. Experimental set-up

The MOEA as described above was implemented in Java and tested on a computer cluster with 384 dual-processor dual-core 64-bit 2.6 GHz AMD Opteron 2218 nodes running Scientific Linux 5.2. The evolutionary parameters were set to the following suitable values determined by preliminary testing:

$$\begin{array}{ll} popSize = 100 & \gamma = 1.0 \\ numGen = 500 & \mu = 0.1 \\ T_{size} = 2 & \end{array}$$

To provide reliable statistics, each version of the algorithm was run 30 times, with different random number seeds, for each benchmark instance. The population diversity and the solutions in the first front were recorded at the end of every evolutionary generation for later analysis.

6. Analysis of results

The results from the VRPTW experiments were analyzed from four different perspectives: First, to investigate the importance of considering the similarity of solutions for finding diverse and good solutions. Second, to provide an indication of the number of instances which really do have conflicting objectives. Third, to show that similar or better results are obtained when multiple objectives are minimized simultaneously, rather than when only one objective is minimized. Finally, to compare the results obtained by our MOEA with those from previous studies. In particular, the multi-objective performance metrics are used to compare the MOEA solutions with those from NSGA-II [12], which has proved extremely successful for many applications in the past, and involves features that provide a useful contrast to the new MOEA.

Initially the MOEA is set up to minimize two objectives, the number of routes and travel distance as in previous studies, with the results presented in Sections 6.1 to 6.5. Then, to illustrate its use on further objectives, Section 6.6 presents results for other combinations of two objectives, and for three objectives.

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

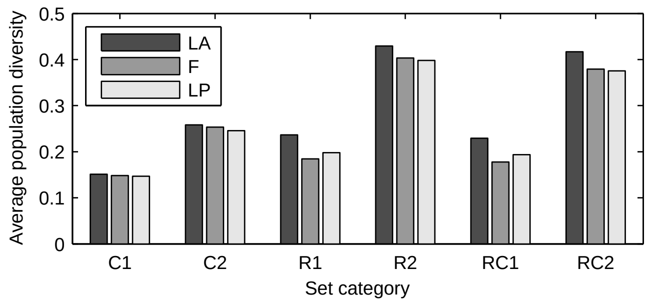


Figure 9: Average final population diversity, grouped by instance category, preserved by methods LA, F and LP.

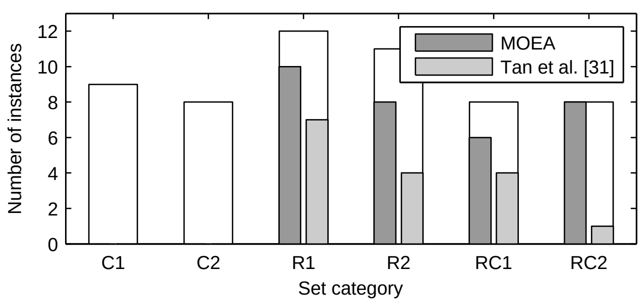


Figure 10: Number of instances with conflicting objectives (shaded bars), found by MOEA and Tan et al. [31], out of the total number of instances in each category (clear full bars).

6.1. How important is the similarity measure?

As explained earlier, two criteria are used in the MOEA recombination process: fitness to select the first parent, and similarity to select the second. To verify the importance of using similarity, a control case and two variations were tested. These had the second parent chosen according to fitness (referred to as method F), or least similarity on average (method LA), or least similarity to the first parent (method LP). To measure the contribution of the recombination process to the population diversity in each case, the mutation probability μ was temporarily set to zero to prevent it interfering. Figure 9 shows the average final population diversity (defined by equation 20) resulting from the three methods for each instance category. All three methods end up with similar population diversity for categories C1 and C2 because the customers in these instances are geographically clustered, which renders diverse solutions of little advantage. For all the other categories, method LA preserves significantly higher population diversity. This is because recombination with the least similar individual in the population forces the algorithm to explore wider regions of the search space.

The aim of maintaining diversity was to obtain better Pareto approximations, and this needed to be tested explicitly. Table 1 presents the Pareto approximation results using F, LP and LA. For each instance, all the solutions in the resulting non-dominated set are taken from all repetitions, and the average for each objective is computed. Then, these are averaged over each instance category. For each method and category is shown the average number of routes (upper), the average travel distance (middle), and the average execution time in seconds for a run of 500 generations (lower). The last column presents the total accumulated sum, indicating the average total number of routes, the average total travel distance, and the average total execution time for all 56 instances (which is the standard measure in the literature when comparing results for this problem). The last two rows show the percentage difference between the results from method LA and those from the method that obtained the lowest value for each objective. Method LA achieved the smallest number of routes and shortest travel distance for categories C1, R1, R2, RC1 and RC2, and accumulated. Method LP obtained solutions with the lowest number of routes and travel distance for category C2, but the percentage improvements over method LA in these cases was very small.

It is clear that including the similarity measure in the recombination phase of the MOEA is accomplishing its objective. Method LA preserves a higher population diversity, and the resulting quality of the solutions is better, with fewer routes and shorter travel distances. This approach will therefore be adopted for the remainder of this study.

6.2. Is VRPTW a multi-objective problem?

The next issue is whether the VRPTW really does need to be treated as a multi-objective problem. That is, are the problem instances such that there are trade-offs that result in more than one solution in the Pareto front, and does the MOEA have more than one solution in its Pareto approximation sets?

Figure 10 shows the number of instances for which the MOEA found approximation sets with multiple solutions (i.e. instances with conflicting objectives), out of the total number of instances in each category. For comparison, the corresponding numbers for the algorithm of Tan et al. [31] are also shown, which uses local search heuristics to improve its performance, rather than involving a similarity measure to maintain diversity. For all categories, MOEA found more approximation sets with multiple solutions than the other algorithm, except for categories C1 and C2 which both approaches agreed did not include any instances with conflicting objectives. In total, 29 instances out of 56 were found by MOEA to have incompatible objectives, and only two of those are dominated by the solutions found by the algorithm of Tan et al. [31].

Figure 11 presents bar graphs showing the number of instances for different sizes of Pareto approximation, for MOEA and for the best-known solutions for each instance. The 29 multiple-solution instances found by MOEA have 2, 3, 4 and 5 solutions in their Pareto approximation. If the best-known solutions are considered along with those from MOEA, the number of instances with conflicting objectives increases up to 34, with the number of instances with one and two solutions decreasing, and those with three, four

Method	C1	C2	R1	R2	RC1	RC2	Accumulated
	10.00	3.02	13.08	3.51	13.04	4.12	447.07
F	898.93	615.79	1270.39	1003.81	1449.74	1179.44	60336.79
	72.91	74.98	98.25	118.80	91.61	123.30	5461.10
	10.00	3.01	12.89	3.50	12.73	4.11	441.98
LP	866.37	608.20	1237.75	985.97	1411.00	1159.13	58922.65
	110.41	90.82	159.39	151.99	145.83	156.14	7720.60
	10.00	3.02	12.87	3.48	12.60	4.04	439.97
LA	851.71	609.72	1221.34	981.41	1394.76	1154.47	58388.60
	82.51	76.40	149.84	137.06	126.73	142.92	6816.84
% difference number of routes	0.00	0.12	0.00	0.00	0.00	0.00	0.00
% difference travel distance	0.00	0.25	0.00	0.00	0.00	0.00	0.00

Table 1: Mean number of routes, travel distances and execution times, averaged over instance categories, for solutions in the Pareto approximations obtained with methods F, LP, and LA.

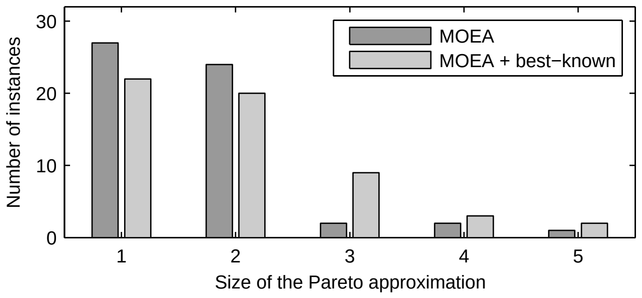


Figure 11: Number of instances for each number of solutions in the Pareto approximations found by MOEA, and considering the best-known solutions.

and five increasing. This confirms the multi-objective nature of the VRPTW, and indicates the extent to which MOEA is finding the best Pareto approximations.

6.3. Single- versus bi-objective optimization

The next question to consider is how well the MOEA performs compared to single-objective optimizers. To test this fairly, the single objective EA used was MOEA itself, but set to minimize only one objective. Table 2 presents the results from MOEA, along with those from the single-objective EA versions, in the same format as Table 1 excluding the execution times. The EA minimizing only the number of routes is labeled EAR, and the one minimizing only the travel distance is EAD. For each instance is taken, from each of the 30 repetitions, the solution with the smallest number of routes and the one with the shortest travel distance, and from these are computed the mean value for each objective, and the averages of these over each category. The numbers of problem instances for which there were significant improvements (shown by a *t-test* at 0.05 significance level) over the corresponding multi- or single-objective case are shown in brackets. The last two rows show for each category the percentage difference between the number of routes from MOEA and EAR, and the travel distances from MOEA and EAD, respectively.

The MOEA finds the solutions with the overall smallest number of routes for all categories and accumulated, with percentage improvements over EAR ranging from nearly 2% up to more than 5%. EAR has significantly better performance on only 4 problem instances, compared with 22 for MOEA. It seems that since EAR focuses on only one of the two objectives, it does not benefit from the improvement that can be achieved by simultaneously minimizing the travel distance as MOEA does. The results achieved indicate that the single-objective algorithm here has a higher tendency to become stuck in sub-optimal solutions.

Regarding the travel distance, the differences between the results from MOEA and EAD are smaller and less consistent. For problem categories C1, C2, R1 and RC1, there is not much difference between MOEA and EAD, but for categories R2 and RC2 the single objective EAD is significantly better than MOEA on 13 problems instances, and significantly worse on none. In this case, the generation of a good Pareto approximation by MOEA does not help, and can actually lead to worse performance. Of course, it would be trivial to run both EAD and MOEA and incorporate any better solution from EAD into the Pareto approximation obtained by MOEA, so one can easily have the best of both approaches.

6.4. Comparison with previous single-objective results

In previous VRPTW studies, the travel distance has been used as the standard benchmark for comparing the performance in single-objective optimization. Hence, this section will follow that and take *best* to be the solution with lowest travel distance.

Table 3 presents the solutions with the lowest travel distances from our MOEA, as well as those from six previous studies, in the same format as as Table 1. For each instance, the solution with the shortest travel distance is taken from the 30 runs, and the corresponding number of routes and travel distance is averaged over each instance category. For categories C1 and C2, MOEA achieved similar results to the previously published studies, except for the travel distance in C2, where it is only 0.08% higher. The lowest number of routes for the other categories, as

Algorithm	C1	C2	R1	R2	RC1	RC2	Accumulated
EAr	10.51	3.15	13.02	3.18	12.61	3.63	440.88
	(0)	(0)	(1)	(0)	(3)	(0)	(4)
	1732.13	932.61	1557.11	1464.75	1702.98	1759.59	85548.13
EAd	10.00	3.00	13.14	3.94	12.98	4.67	456.18
	834.37	591.62	1204.32	916.35	1372.12	1063.30	56257.32
	(0)	(0)	(1)	(8)	(1)	(5)	(15)
MOEA	10.00	3.00	12.64	3.09	12.36	3.54	426.85
	(4)	(3)	(8)	(1)	(5)	(1)	(22)
	832.55	591.74	1205.04	926.17	1372.96	1076.72	56472.66
	(1)	(0)	(0)	(0)	(1)	(0)	(2)
% difference number of routes	5.11	5.14	3.73	2.24	2.50	2.53	3.29
% difference travel distance	0.22	−0.02	−0.06	−1.03	−0.08	−1.14	−0.38

Table 2: Mean smallest number of routes and shortest travel distances, averaged over instance categories, for solutions obtained with algorithms EAr, EAd and MOEA. The numbers in brackets indicate the number of problem instances for which there is significant improvement over the corresponding multi- or single-objective algorithm.

well as the accumulated, was obtained by Le Bouthillier and Crainic [27], Homberger and Gehring [29], and Pisinger and Ropke [42], but MOEA found solutions with lower travel distances than them. Solutions from the hybrid GA of Jung and Moon [23] have the lowest travel distances for categories R1, R2, RC1 and RC2, and accumulated, where results from MOEA are 0.62%, 2.22%, 0.34%, 3.23%, and 1.09% higher respectively, though MOEA has smaller numbers of routes.

Looking at the best known results for the individual instances, MOEA found solutions for three instances that have lower travel distance than the best-known. Moreover, it achieved competitive solutions for another 17 instances for which the travel distance is only slightly ($< 2\%$) above the best-known, with the number of routes lower in six of them and equal in the rest. MOEA also found 17 solutions that are equal to the best-known for instances in categories C1 and C2. These results show that, overall, the single objective performance of our MOEA is comparable to the best of the previously published algorithms.

Finally in this section, the MOEA solutions are compared against the optimal solutions found by exact methods as presented on Solomon's web site[2]. Here the total travel distance is the primary objective and the distances are truncated to one decimal place [42]. Table 4 presents a summary of the MOEA results, taking into consideration the truncation criteria, and also covering further benchmark instances with smaller numbers of customers ($N = 25$ and 50). The second column shows the number of instances out of 56 that have been solved to optimality, and the third column gives the number of optimal solutions found by MOEA. The next two columns show the percent average difference between the optimal and the results from MOEA, first over all instances, and then only over the instances for which MOEA did not find the optima. The last column shows the average execution time in seconds for a run of 500 generations. These results indicate

N	Opt.	MOEA	% Diff	% Sub.	Time
25	56	56	0.00	0.00	11.47
50	53	39	0.23	0.88	29.76
100	37	17	0.81	1.50	116.73

Table 4: Comparison of results from MOEA against optima obtained by exact methods, for 56 instances of size $N = 25, 50$ and 100.

the ability of MOEA to find optimal solutions for instances of different sizes, and show that the gap between the optimal results and those from MOEA is narrow ($\leq 1.5\%$) even for the largest instances.

6.5. *Multi-objective performance comparisons*

To evaluate the performance of our MOEA as a multi-objective approach, NSGA-II [12] was implemented for comparison purposes, and the multi-objective performance metrics $\mathsf{M_H}$ and $\mathsf{M_C}$ (defined earlier in equations 15 and 16) were computed. The NSGA-II implementation used the same solution representation, with the same crossover and mutation operators, as the MOEA. The difference lies in the way selection is carried out in the mating and survival processes. MOEA selects one parent according to fitness and the other according to the similarity, in contrast to NSGA-II which uses fitness as the only criteria for the selection of both parents. MOEA considers similarity to identify which solutions are taken to the next generation, while NSGA-II utilizes the *crowding distance* which does not involve any routing information at all.

Computing the hypervolume metric $\mathsf{M_H}$ requires an appropriate reference points $\mathbf{z}$ to be set. Since each instance has an obvious maximal solution, with largest number of routes equal to the number of customers N, and longest travel distance $D^{\max}$ equal to twice the sum of the distances of all customers from the depot, the reference point for each instance was set at $\mathbf{z} = (N, D^{\max})$. For each problem instance, there is a set of 30 $\mathsf{M_H}$ values from the 30 runs performed, with each value computed using the non-

[2]`http://w.cba.neu.edu/~msolomon/problems.htm`

Author	C1	C2	R1	R2	RC1	RC2	Accumulated
Jung and Moon [23]	10.00	3.00	13.25	5.36	13.00	6.25	486.00
	828.38	589.86	1179.95	878.41	1343.65	1004.21	54779.02
Le Bouthillier and Crainic [27]	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
Homberger and Gehring [29]	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
Tan et al. [31]	10.00	3.00	12.92	3.55	12.38	4.25	441.00
	828.91	590.81	1187.35	951.74	1355.37	1068.26	56293.06
Ombuki et al. [32]	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
Pisinger and Ropke [42]	10.00	3.00	11.92	2.73	11.50	3.25	405.00
	828.38	589.86	1212.39	957.72	1385.78	1123.49	57332.00
Garcia-Najera and Bullinaria [35]	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
MOEA	10.00	3.00	13.08	4.00	12.63	5.38	459.00
	828.38	589.86	1187.32	897.95	1348.22	1036.65	55378.61
% difference number of routes	0.00	0.00	9.85	46.52	9.78	65.38	13.33
% difference travel distance	0.00	0.08	0.62	2.22	0.34	3.23	1.09

Table 3: Travel distance and number of routes, averaged over categories, for the best solutions found in previous studies and by MOEA.

Algorithm	C1	C2	R1	R2	RC1	RC2
NSGA-II	0.77	0.87	0.66	0.79	0.69	0.81
	(0)	(0)	(0)	(0)	(0)	(0)
MOEA	0.81	0.74	0.71	0.76	0.75	0.81
	(1)	(0)	(10)	(3)	(8)	(2)

Table 5: Hypervolume metric values, averaged over instance categories, for solutions obtained with NSGA-II and MOEA. Shown in brackets are the number of instances for which the result is significantly better than the other approach.

Algorithm	C1	C2	R1	R2	RC1	RC2
NSGA-II	0.81	0.91	0.13	0.36	0.14	0.33
	(0)	(1)	(0)	(1)	(0)	(0)
MOEA	0.89	0.90	0.74	0.67	0.72	0.60
	(4)	(1)	(12)	(10)	(8)	(8)

Table 6: Coverage metric values, averaged over instance categories, for solutions obtained with NSGA-II and MOEA. Shown in brackets are the number of instances for which the result is significantly better than the other approach.

dominated sets achieved by MOEA and NSGA-II starting from the same initial populations. Table 5 shows the hypervolume metric averages for each category and the numbers of instances with significant improvements over the other approach. For categories C1 and C2 there is little difference between the two algorithms. For categories R1 and RC1 the MOEA exhibits significant improvement over NSGA-II for most problem instances, and for categories R2 and RC2 the MOEA has significant improvement for some problem instances. There are no problem instances for which NSGA-II performs significantly better than the MOEA.

To apply the coverage metric, for each problem instance the coverage values $\mathsf{M}_\mathsf{C}(\mathrm{MOEA}_i, \mathrm{NSGA\text{-}II}_j)$ and $\mathsf{M}_\mathsf{C}(\mathrm{NSGA\text{-}II}_j, \mathrm{MOEA}_i)$ were computed for all pairs of runs $(i, j = 1, \ldots, 30)$, that is 900 M_C values each. Table 6 presents the averages of the $\mathsf{M}_\mathsf{C}(\mathrm{X}_i, \mathrm{Y}_j)$ values over all the instances within each problem category, and the numbers of instances for which there was significant improvement over the other approach. Again there is little difference in performance for categories C1 and C2, but here the MOEA shows significant improvement over NSGA-II for the large majority of instances in all the other categories.

To understand the improvements, it is instructive to look at the population diversity for both algorithms. Figure 12 presents six plots showing the mean population diversity and variance on the vertical axis for each instance category, as a function of the first 500 generations on the horizontal axis. It is clear that the MOEA preserves a higher diversity for categories R1, R2, RC1 and RC2 for which improvements over NSGA-II were observed, but not for the non-multi-objective categories C1 and C2 for which there were not. The different behavior for R1 and RC1 to that of R2 and RC2 is also consistent with that distinction in the hypervolume results and the increased numbers of solutions in the MOEA Pareto approximations compared to NSGA-II. Moreover, the MOEA diversities present a more gentle drop in all categories except C2, suggesting that MOEA performs a wider exploration of the search space before settling on its final solutions.

As an illustration of the consequences of this higher diversity, Figure 13 shows the solutions in the non-dominated set found by MOEA for the typical problem instance R201. The routes forming each solution are displayed in separate boxes to enable a clearer comparison. Although some routes contain similar patterns, no route is repeated across solutions. One can also see that the second and third routes in solution 1 contain patterns from at least two

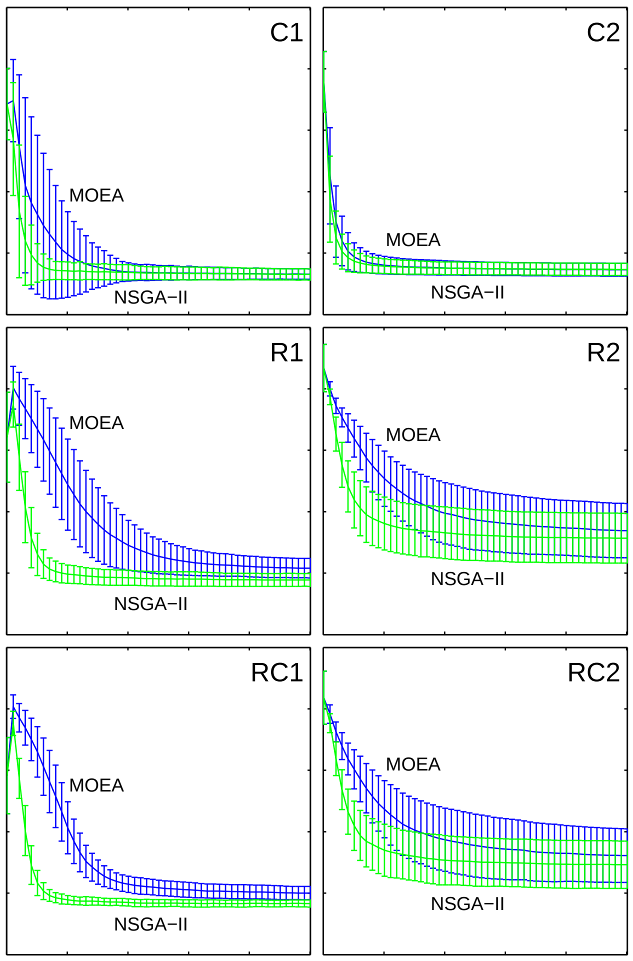


Figure 12: Average population diversity for each instance category as a function of generation, for MOEA and NSGA-II.

routes from each of the other solutions, meaning that, as well as having routes with different customers forming the solutions, the sequences of customers are also diverse.

In the past, fully multi-objective comparisons between algorithms have been hampered by a lack of published solution details. To remedy this for future algorithms, Table 7 presents the full details of the number of routes (columns R) and travel distances (columns D) for the solutions in the Pareto approximations obtained by our MOEA for the 26 instances in which the two objectives are in conflict.

6.6. Tri-objective performance comparisons

This section illustrates the application of our MOEA to other objectives (in particular, the delivery time) and for optimizing more than two objectives at once. The number of routes (R), travel distance (D) and delivery time (T) were minimized in pairs, giving three objective settings (RD, RT and DT), and with all three of them together (RDT). The results from each of these four settings were compared using the coverage performance metric $\mathsf{M_C}$ as in the previous section (but using all three objectives).

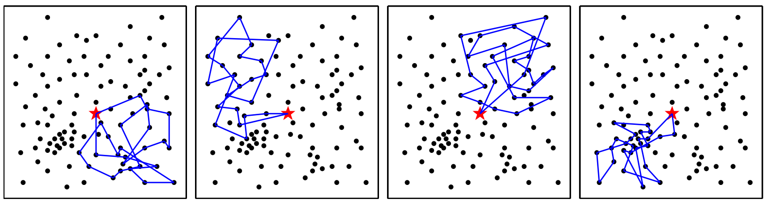
The comparison results presented in Table 8 lead to the following observations: For categories C1 and C2 there are

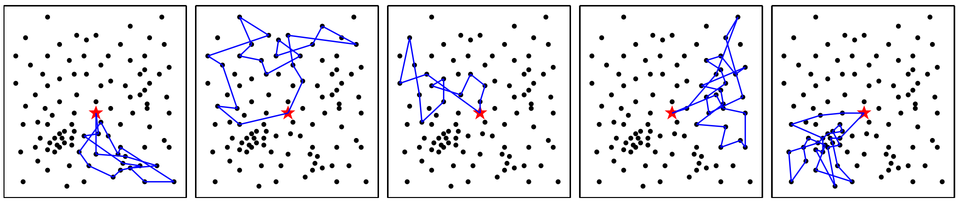
Instance	R	D	R	D
R101	20	1643.18	19	1650.80
R102	18	1473.73	17	1487.31
R103	14	1219.10	13	1299.18
R104	11	995.87	10	999.82
R105	15	1364.88	14	1377.11
R106	13	1240.41	12	1263.21
R107	11	1080.64	10	1164.14
R109	13	1155.73	12	1156.05
R201	7	1179.22	6	1185.03
	5	1194.78	4	1253.23
R202	5	1055.66	4	1081.60
R203	4	910.55	3	959.75
R205	4	959.74	3	1030.92
R206	4	890.90	3	924.64
R208	3	715.37	2	736.47
R209	4	876.69	3	921.37
R210	4	933.84	3	969.61
RC102	14	1466.97	13	1492.70
RC105	15	1519.44	14	1540.18
RC106	13	1391.76	12	1394.43
RC107	12	1215.94	11	1235.37
RC108	11	1133.69	10	1166.03
RC201	9	1297.29	7	1304.09
	6	1319.16	5	1335.31
	4	1415.00		
RC202	6	1123.12	5	1132.57
	4	1162.54		
RC203	4	957.08	3	1058.33
RC204	4	796.11	3	801.90
RC205	7	1226.27	6	1228.09
	5	1251.58	4	1304.93
RC206	4	1091.42	3	1257.15
RC207	5	994.00	4	1001.85
	3	1104.95		
RC208	4	807.92	3	834.88

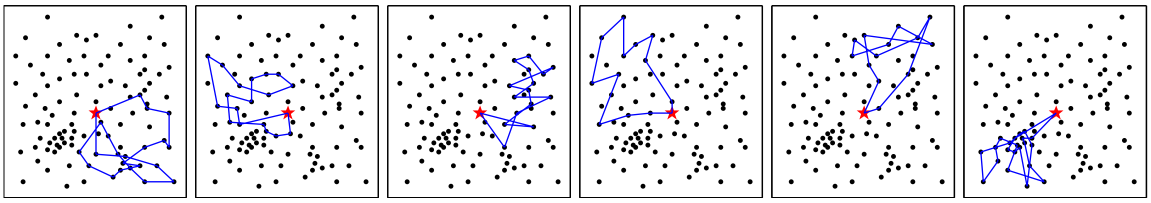
Table 7: Number of routes (R) and travel distance (D) for the instances where both objectives are in conflict, corresponding to the solutions in the Pareto approximations obtained by MOEA.

mixed results, with all settings having a high coverage of each other. Otherwise, the coverage of RT, DT and RDT by RD is low ($\leq 14\%$), and the coverage of RD, DT and RDT by RT is nearly zero. The most interesting cases are settings DT and RDT, as their coverage of RD and RT is much higher. Between them, the coverage of DT by RDT is larger than the coverage of RDT by DT. These results indicate that setting MOEA to minimize all three objectives does lead to better non-dominated solutions.

Results from our MOEA and from NSGA-II for the tri-objective RDT case are compared in Table 9. Again both algorithms show similar coverage of each other for instances in categories C1 and C2. For all instances in categories R1 and RC1, solutions in the non-dominated sets found by MOEA have significantly higher coverage of


(a) Solution 1: 4 routes


(b) Solution 2: 5 routes


(c) Solution 3: 6 routes

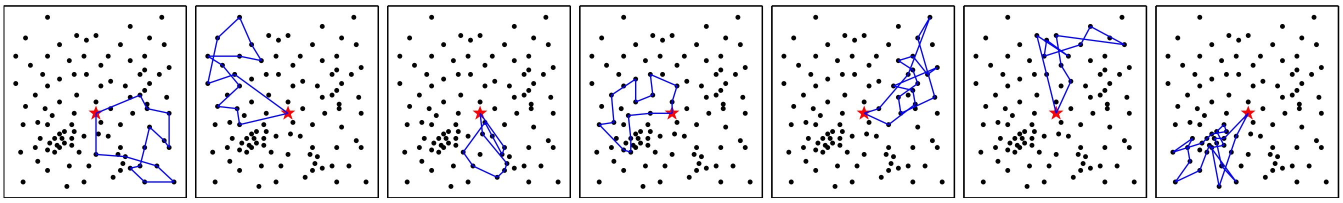

(d) Solution 4: 7 routes

Figure 13: Solutions in the non-dominated set found by MOEA for typical problem instance R201. For clarity, each route for each solution is displayed in a separate box.

those obtained by NSGA-II. For category R2, MOEA has a significant higher coverage in five instances, and NSGA-II has a significant higher coverage in four. Finally, for category RC2, MOEA has a significant higher coverage in seven instances, and NSGA-II in none. These results are reflected in increased numbers of solutions in the MOEA Pareto approximations over NSGA-II. Overall then, it can be concluded that the new MOEA has significantly better performance than NSGA-II.

7. Conclusions

This paper has proposed an improved Multi-Objective Evolutionary Algorithm (MOEA) for solving the Vehicle Routing Problem with Time Windows (VRPTW). It can simultaneously minimize any number of objectives, and has been tested here on the number of routes, travel distance and delivery time. The key improvement is the introduction of a measure of similarity between solutions which is primarily used to select the second parent for the recombination process. Since the first parent is selected according to fitness, as usual in evolutionary algorithms, the resulting offspring inherits the good quality from this parent, while exploring non-common areas of the search space from the second parent. Consequently, the solutions that emerge are more diverse, tending to cover more than one value in the number of routes dimension, and better approximating the whole Pareto front.

The new MOEA was tested using a popular benchmark set of 56 VRPTW problem instances, approximately half of which have conflicting objectives. An initial series of experiments were carried out to explore the effect the similarity measure had on population diversity and solution quality. Three methods for selecting the second parent were compared, including simple fitness, and this demonstrated the high importance that the similarity measure had in maintaining diversity and arriving at better solutions. This established the best approach for the remainder of the study; and the implication that it would work best on problems with conflicting objectives (problem categories R1, R2, RC1 and RC2), and not offer much improvement on others (C1 and C2), was later confirmed.

Obj.	Covers	C1	C2	R1	R2	RC1	RC2
RD	RT	0.87 (6)	0.64 (3)	0.04 (6)	0.01 (1)	0.05 (4)	0.02 (2)
	DT	0.82 (1)	0.72 (0)	0.08 (1)	0.11 (4)	0.14 (0)	0.08 (0)
	RDT	0.82 (1)	0.72 (1)	0.08 (1)	0.11 (3)	0.13 (0)	0.08 (1)
RT	RD	0.68 (0)	0.55 (1)	0.01 (2)	0 (0)	0.01 (2)	0.01 (0)
	DT	0.68 (0)	0.63 (0)	0.03 (0)	0.04 (0)	0.06 (0)	0.02 (0)
	RDT	0.68 (0)	0.62 (0)	0.04 (0)	0.03 (0)	0.07 (0)	0.03 (0)
DT	RD	0.91 (2)	0.9 (3)	0.31 (11)	0.14 (5)	0.36 (8)	0.21 (6)
	RT	0.97 (5)	0.92 (4)	0.49 (12)	0.42 (11)	0.46 (8)	0.48 (8)
	RDT	0.91 (2)	0.88 (2)	0.43 (4)	0.4 (4)	0.42 (2)	0.42 (3)
RDT	RD	0.89 (3)	0.91 (3)	0.32 (11)	0.16 (6)	0.38 (8)	0.23 (7)
	RT	0.97 (6)	0.93 (3)	0.52 (12)	0.44 (11)	0.49 (8)	0.44 (8)
	DT	0.89 (2)	0.89 (1)	0.44 (5)	0.43 (5)	0.46 (6)	0.41 (3)

Table 8: Coverage metric values, averaged over instance categories, for MOEA solutions obtained with objective sets RD, RT, DT and RDT. In brackets are the numbers of instances for which the result is significantly better than the reverse case.

Algorithm	C1	C2	R1	R2	RC1	RC2
NSGA-II	0.81 (0)	0.87 (2)	0.14 (0)	0.37 (4)	0.13 (0)	0.35 (0)
MOEA	0.88 (4)	0.87 (2)	0.66 (12)	0.55 (5)	0.63 (8)	0.49 (7)

Table 9: Coverage metric values, averaged over instance categories, for solutions obtained with NSGA-II and MOEA optimizing all three objectives (RDT). In brackets are the numbers of instances for which the result is significantly better than the other approach.

The performance of the algorithm was evaluated from three different perspectives: First the MOEA results were compared with those from single-objective EAs, showing that MOEA was able to find solutions with smaller numbers of routes and similar or shorter travel distances. Second, the MOEA results were compared with those from previously published single- and bi-objective algorithms, showing that MOEA found three solutions better than the best-known, and others that were highly competitive in the sense that the travel distance was no more than 2% higher, but the number of routes was the same or smaller. Finally, and perhaps most importantly, the new MOEA was evaluated using two multi-objective performance metrics, hypervolume and coverage, showing significantly better results than the well-known evolutionary multi-objective optimizer NSGA-II for both the bi-objective and tri-objective cases.

There remains considerable scope for further development of the MOEA approach proposed here. Although the similarity measure used by the MOEA has been shown in this paper to result in improved performance, it still only considers the arcs used in the routes in a solution and not the sequence of them. Consequently, further improvements might be possible by developing more refined similarity measures, that provide performance improvements that justify the inevitable increased computational costs. There is also scope to explore the extension of the MOEA to the optimization of further objectives, such as the route balance [25]. Finally, there remains the need to test further the scalability of MOEA to larger VRPTW problem instances, and different variants of the routing problem.

References

[1] P. Toth, D. Vigo, The vehicle routing problem, SIAM, Philadelphia, PA, USA, 2001.

[2] G. B. Dantzig, J. H. Ramser, The truck dispatching problem, Manage. Sci. 6 (1) (1956) 80–91.

[3] N. Jozefowicz, F. Semet, E.-G. Talbi, Multi-objective vehicle routing problems, Eur. J. Oper. Res. 189 (2008) 293–309.

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

[5] J.-F. Cordeau, G. Desaulniers, J. Desrosiers, M. M. Solomon, F. Soumis, VRP with Time Windows, in: P. Toth, D. Vigo (Eds.), The vehicle routing problem, SIAM, Philadelphia, PA, USA, 157–193, 2001.

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

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

[8] T. Yu, L. Davis, An Introduction to Evolutionary Computation in Practice, in: T. Yu, L. Davis, C. M. Baydar, R. Roy (Eds.), Evolutionary Computation in Practice, vol. 88 of *Studies in Computational Intelligence*, Springer, 1–8, 2008.

[9] O. Bräysy, W. Dullaert, M. Gendreau, Evolutionary Algorithms for the Vehicle Routing Problem with Time Windows, J. Heuristics 10 (6) (2004) 587–611.

[10] J. K. Lenstra, A. H. G. R. Kan, Complexity of Vehicle Routing and Scheduling Problems, Networks 11 (1981) 221–227.

[11] E. Zitzler, M. Laumanns, S. Bleuler, A Tutorial on Evolutionary Multiobjective Optimization, in: X. Gandibleux, M. Sevaux, K. Sörensen, V. T'kindt (Eds.), Metaheuristics for Multiobjective Optimisation, Springer, 3–38, 2004.

[12] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE T. on Evolut. Comput. 6 (2) (2002) 182–197.

[13] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: Improving the Strength Pareto Evolutionary Algorithm, Tech. Rep. 103, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Zurich, Switzerland, 2001.

[14] J. D. Knowles, D. W. Corne, Approximating the nondominated front using the Pareto Archived Evolution Strategy, Evol. Comput. 8 (2) (2000) 149–172.

[15] E. Zitzler, S. Künzli, Indicator-Based Selection in Multiobjective Search, in: Parallel Problem Solving from Nature VIII, Springer, 832–842, 2004.

[16] E. Zitzler, L. Thiele, Multiobjective Optimization Using Evolutionary Algorithms – A Comparative Case Study, in: 5th Inter-

national Conference on Parallel Problem Solving from Nature, Springer-Verlag, London, UK, 292–304, 1998.

[17] E. Zitzler, K. Deb, L. Thiele, Comparison of Multiobjective Evolutionary Algorithms: Empirical Results, Evol. Comput. 8 (2) (2000) 173–195.

[18] K. Deb, S. Jain, Running Performance Metrics for Evolutionary Multi-Objective Optimization, in: L. Wang, K. C. Tan, T. Furuhashi, J.-H. Kim, X. Yao (Eds.), 4th Asia-Pacific Conference on Simulated Evolution and Learning, Nanyang Technical University, Singapore, 13–20, 2002.

[19] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, V. G. da Fonseca, Performance assesment of multiobjective optimizers: An analysis and review, IEEE T. Evolut. Comput. 7 (2) (2003) 117–132.

[20] J. Knowles, L. Thiele, E. Zitzler, A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers, TIK Report 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, 2006.

[21] M. Rahoual, B. Kitoun, M. hakim Mabed, V. Bachelet, F. Benameur, Multicriteria Genetic Algorithms for the Vehicle Routing Problem With Time Windows, in: 4th Metaheuristics International Conference, 527–532, 2001.

[22] N. Srinivas, K. Deb, Muiltiobjective optimization using nondominated sorting in genetic algorithms, Evol. Comput. 2 (3) (1994) 221–248.

[23] S. Jung, B. R. Moon, A Hybrid Genetic Algorithm For The Vehicle Routing Problem With Time Windows, in: 2002 Genetic and Evolutionary Computation Conference, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1309–1316, 2002.

[24] K. Q. Zhu, A Diversity-Controlling Adaptive Genetic Algorithm for the Vehicle Routing Problem with Time Windows, in: 15th IEEE International Conference on Tools with Artificial Intelligence, IEEE, 176–183, 2003.

[25] N. Jozefowiez, F. Semet, E.-G. Talbi, Enhancements of NSGA II and Its Application to the Vehicle Routing Problem with Route Balancing, in: Artificial Evolution, 131–142, 2005.

[26] T. Murata, R. Itai, Multi-objective vehicle routing problems using two-fold EMO algorithms to enhance solution similarity on non-dominated solutions, in: C. A. Coello Coello, A. Hernandez, E. Zitzler (Eds.), 3rd International Conference on Evolutionary Multi-Criteria Optimization, vol. LNCS 3410, Springer, Guanajuato, Mexico, 885–896, 2005.

[27] A. Le Bouthillier, T. G. Crainic, A cooperative parallel metaheuristic for vehicle routing with time windows, Comput. Oper. Res. 32 (2005) 1685–1708.

[28] F. W. Glover, M. Laguna, Tabu Search, Springer, 1998.

[29] J. Homberger, H. Gehring, A two-phase hybrid metaheuristic for the vehicle routing problem with time windows, Eur. J. Oper. Res. 162 (2005) 220–238.

[30] H.-G. Beyer, H.-P. Schwefel, Evolution strategies – A comprehensive introduction, Nat. Comp. 1 (1) (2002) 3–52.

[31] K. C. Tan, Y. H. Chew, L. H. Lee, A Hybrid Multiobjective Evolutionary Algorithm for Solving Vehicle Routing Problem with Time Windows, Comput. Optim. and Appl. 34 (1) (2006) 115–151.

[32] B. Ombuki, B. J. Ross, F. Hanshar, Multi-Objective Genetic Algorithms for Vehicle Routing Problem with Time Windows, Appl. Intell. 24 (1) (2006) 17–30.

[33] E. Zitzler, M. Laumanns, L. Thiele, SPEA 2: Improving the Strength Pareto Evolutionary Algorithm for multiobjective optimization, in: K. Giannakoglou, D. Tsahalis, J. Periaux, K. Papailiou, T. Fogarty (Eds.), Evolutionary Methods for Design, Optimisation and Control, CIMNE, Barcelona, Spain, 19–26, 2002.

[34] A. Garcia-Najera, J. A. Bullinaria, A Multi-Objective Density Restricted Genetic Algorithm for the Vehicle Routing Problem with Time Windows, in: 2008 UK Workshop on Computational Intelligence, 2008.

[35] A. Garcia-Najera, J. A. Bullinaria, Bi-objective Optimization for the Vehicle Routing Problem with Time Windows: Using Route Similarity to Enhance Performance, in: M. Ehrgott, C. Fonseca, X. Gandibleux, J. K. Hao, M. Sevaux (Eds.), 5th International Conference on Evolutionary Multi-Criterion Optimization, vol. LNCS 5467, Springer-Verlag, 275–289, 2009.

[36] A. Garcia-Najera, Preserving Population Diversity for the Multi-Objective Vehicle Routing Problem with Time Windows, in: GECCO 2009 Graduate Student Workshop, ACM, 2689–2692, 2009.

[37] K. Sörensen, Distance measures based on the edit distance for permutation-type representations, J. Heuristics 13 (1) (2007) 35–47.

[38] A. Garcia-Najera, J. A. Bullinaria, Comparison of Similarity Measures for the Multi-Objective Vehicle Routing Problem with Time Windows, in: 2009 Genetic and Evolutionary Computation Conference, ACM, 579–586, 2009.

[39] A. E. Eiben, J. E. Smith, Introduction to Evolutionary Computing, Springer, 2003.

[40] D. E. Goldberg, K. Deb, A comparative analysis of selection schemes used in genetic algorithms, in: Foundations of Genetic Algorithms, Morgan Kaufmann, 69–93, 1991.

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

[42] D. Pisinger, S. Ropke, A general heuristic for vehicle routing problems, Comput. Oper. Res. 34 (8) (2007) 2403–2435.