



An Efficient Application of Goal Programming to Tackle Multiobjective Problems with Recurring Fitness Landscapes

Rodrigo Lankaites Pinheiro^{1,2(✉)}, Dario Landa-Silva², Wasakorn Laesanklang³,
and Ademir Aparecido Constantino⁴

¹ Webroster Ltd., Peterborough PE1 5NB, UK
rodrigo.pinheiro@webroster.com

² ASAP Research Group, School of Computer Science, University of Nottingham,
Nottingham, UK
dario.landasilva@nottingham.ac.uk

³ Department of Mathematics, Faculty of Science, Mahidol University,
Nakhon Pathom, Thailand
wasakorn.lae@mahidol.ac.th

⁴ Departamento de Informática, Universidade Estadual de Maringá, Maringá, Brazil
aaconstantino@uem.br

Abstract. Many real-world applications require decision-makers to assess the quality of solutions while considering multiple conflicting objectives. Obtaining good approximation sets for highly constrained many-objective problems is often a difficult task even for modern multiobjective algorithms. In some cases, multiple instances of the problem scenario present similarities in their fitness landscapes. That is, there are recurring features in the fitness landscapes when searching for solutions to different problem instances. We propose a methodology to exploit this characteristic by solving one instance of a given problem scenario using computationally expensive multiobjective algorithms to obtain a good approximation set and then using Goal Programming with efficient single-objective algorithms to solve other instances of the same problem scenario. We use three goal-based objective functions and show that on benchmark instances of the multiobjective vehicle routing problem with time windows, the methodology is able to produce good results in short computation time. The methodology allows to combine the effectiveness of state-of-the-art multiobjective algorithms with the efficiency of goal programming to find good compromise solutions in problem scenarios where instances have similar fitness landscapes.

Keywords: Multi-criteria decision making · Goal programming · Pareto optimisation · Multiobjective vehicle routing

1 Introduction

Tackling highly-constrained optimisation problems with many objectives is difficult even with modern multiobjective algorithms [1]. In real-world scenarios, decision-makers often benefit from having a set of solutions representing a compromise between the multiple objectives so that they can choose the preferred solution(s). It is often useful to use problem domain knowledge during the optimisation in order to obtain better sets of compromise solutions. For example, in the context of continuous multiobjective optimisation problems, [2] estimated Pareto fronts to then obtain values for the decision variables of interesting solutions. Their technique allows to focus the search in sub-regions of the objective space. Another example is the work by [3] using a Bayesian model to learn computationally expensive objective functions to then use the estimation model to explore the search space more quickly.

The multiobjective vehicle routing problem with time windows (MOVRPTW) is a well-know difficult combinatorial optimisation problem that arises in many real-world logistic scenarios [4]. This problem refers to creating a plan for a fleet of identical vehicles to take goods from a depot and deliver them to customers at various locations. Each customer has certain demand level that needs to be satisfied within a specified time window. Objectives usually considered in the MOVRPTW include among others, the minimisation of number of vehicles and the minimisation of total travel distance by all vehicles.

Due to the high number of constraints and objectives in MOVRPTW scenarios, even state-of-the-art multiobjective algorithms struggle to find good approximations to the Pareto optimal front within reasonable computation time. In logistic scenarios where problems like MOVRPTW arise, it is often the case that problem instances corresponding to a different planning periods share parts of the same data. For example, the same or very similar set of vehicles might be available in each planning period. Also, there might be a set of recurring customer orders that need to be satisfied in the different planning periods. This results in the different problem instances presenting recurring features in their fitness landscapes. Other problems like timetabling and personnel scheduling may also have instances with recurring features resulting in similar fitness landscapes (η -dimensional surface representing the Pareto front, where η is the number of objectives).

Previous work proposed a technique to analyse and visualise complex objective relationships and fitness landscapes in multiobjective problems [5,6]. Later, [7] introduced a methodology to exploit the recurring similarity between instances of a multiobjective workforce scheduling and routing optimisation problem, in order to solve instances of the same problem scenario more efficiently. In this methodology, a *pilot* problem instance is solved first using some effective (but not necessarily computationally efficient) multiobjective algorithm to produce an approximation to the Pareto optimal set. Such approximation set is given to the decision-maker so that *target* solutions representing the desired trade-off between the multiple objectives are identified. Then, goal programming is applied with a computationally efficient single-objective solving method, in order to find solutions for

other problem instances. In this paper, this methodology is applied to tackle the MOVRPTW in order to further investigate its performance for solving multiobjective problem instances with recurring features. The methodology can be very valuable to facilitate informed decision-making when searching solutions to multiobjective problems. Experiments in this paper are conducted on a set of benchmark instances of the MOVRPTW provided by [8].

Section 2 outlines the multiobjective vehicle routing problem with time windows considered here while Sect. 3 outlines goal programming, Section 4 describes the proposed methodology and Sect. 5 presents the experimental configuration. Sections 6 and 7 present and discuss the results. Section 8 concludes the paper and suggests related future research.

2 Multiobjective Vehicle Routing Problem with Time Windows

A Multiobjective Vehicle Routing Problem with Time Windows (MOVRPTW) is defined on a graph $G = (V, E)$ where V is the set of vertices representing the depot (vertex 0) and the customers (vertices $1 \dots n$) where each customer has a demand p_i ($i = 1, \dots, n$). There are h identical vehicles available, each one with capacity Q . In this MOVRPTW, h is considered large enough so that as many vehicles as needed are available to create the routing plan. A set of routes served by the set of vehicles should be created in order to satisfy all demands from all customers. All routes must start and end in vertex 0. The edge set E denotes all possible connections between all vertices. Each edge from vertex i to vertex j has an associated cost, denoted by c_{ij} , that represents distance or time for a vehicle to travel between vertices i and j . Each customer i must be served during their corresponding time window $[a_i, b_i]$. A waiting time is incurred if a vehicle arrives at time $t < a_i$ and hence it must wait until the start of the time window to serve the customer. A delay time is incurred if a vehicle arrives at time $t > a_i$ and hence it must start serving the customer immediately. Once the vehicle starts serving the customer, it stays there for s time until the delivery is completed, this is known as the service time.

[8] proposed a benchmark set of instances for the MOVRPTW with five minimisation objectives: number of vehicles (Z_1), total travel distance by all vehicles (Z_2), makespan or travel time of the longest route (Z_3), total waiting time for all vehicles (Z_4), and total delay time for all vehicles (Z_5). They designed their instances based on different characteristics of the problem and each instance is a combination of these features. The features that constitute a problem instance in these benchmarks are:

- **Number of Customers:** 50, 150 and 250 customers.
- **Time Window:** five different profiles ($tw0, tw1, tw2, tw3, tw4$) of time windows across a planning period of eight hours. These profiles are defined in terms of minutes from the start of the planning period $0 = 8:00$ am, $480 = 4:00$ pm, etc.). These five time-window profiles are defined as follows:

- $tw0$: $[0,480]$, all customers can be served at any time in the day.
 - $tw1$: $[0,160]$, $[160,320]$, $[320,480]$, refers to three types of customers (morning, midday and late).
 - $tw2$: $[0,130]$, $[175,305]$, $[350,480]$, also refers to three types of customers as in profile $tw1$ but with shorter time windows.
 - $tw3$: $[0,100]$, $[190,290]$, $[350,480]$, also refers to three types of customers as in profile $tw1$ but with longer time windows.
 - $tw4$: includes all time-windows from $tw0$, $tw1$, $tw2$ and $tw3$, each customer has one of the 10 time window types in the previous profiles.
- **Demand Types:** three types of demand (10, 20, 30) uniformly distributed.
 - **Vehicle Capacity:** the capacity of the vehicles is calculated according to a δ parameter such that $Q = \underline{D} + \delta/100(\overline{D} - \underline{D})$ where \underline{D} is the maximum single demand among all customers and \overline{D} is the sum of all customer demands. The dataset considers three δ values ($\delta_0 = 60$, $\delta_1 = 20$, $\delta_2 = 5$).
 - **Service Time:** three values of service time (10, 20, 30) uniformly distributed.

For more details of the MOVRPTW described above and a comprehensive study on the multiobjective nature of the problem, please refer to [9]. There are 45 problem instances and a generator available from <https://github.com/psxjpc/MOVRPTW-Generator>. The technique to analyse objective relationships described in [6] was applied to these problem instances and results indicate that indeed they have similar fitness landscapes. This is the case even for instances that have different time window profiles, vehicle capacity and the number of customers. However, in this work, we split the 45 problem instances into three datasets according to the number of customers. This decision was taken because even though the fitness landscapes are similar, the scale of the objective values vary considerably according to the number of customers. Therefore, we have 3 datasets each with 15 problem instances, the set VRP-50 with 50 customers, the set VRP-150 with 150 customers and the set of VRP-250 with 25 customers.

3 Goal Programming

Without loss of generality, a multiobjective optimisation problem can be written as minimise $F(x) = (f_1(x), f_2(x), \dots, f_n(x))$ subject to $x \in S$, where x is a solution, S is the set of feasible solutions, n is the number of objectives in the problem, $F(x)$ is the image of x in the k -objective space and each $f_i(x)$ is the value of objective i in solution x . For two solutions x and y , it is said that x dominates y , if $\forall i : f_i(x) \leq f_i(y)$ and $\exists j : f_j(x) < f_j(y)$. Moreover, x is said to be *Pareto Optimal* if it is not dominated by any other feasible solution. Then, the aim is to find the set of *Pareto Optimal* solutions usually called *Pareto Set*. This set contains a number of *non-dominated points in the objective space* creating the *Pareto Front*.

Goal programming is one of the earliest proposed approaches to tackle optimisation problems with multiple objective [10]. Basically, goal programming consists of establishing a specific numeric goal for each of the objectives considered in the problem. Then, search is conducted for a solution in which the weighted sum of deviations in the objective values with respect to the goals is minimised.

In other words, goal programming is about establishing a target for each objective and then searching for a solution with objective values as close as possible to those targets. There are three types of goals in goal programming [11]:

- Lower bound: defines a lower value for an objective such that solutions that fall below the lower value are penalised.
- Upper bound: defines an upper value for an objective such that solutions that present higher values than the upper value are penalised. This is the type of goals in the optimisation problem considered here, due to the minimisation nature of all objectives.
- Strict bound: defines a specific target value such that solutions that present values above or below are penalised. This is applicable when obtaining a solution with a specific target value for a given objective is essential. For example, in the case that solutions using exactly h number of vehicles were required in the MOVRPTW.

Once the goals for each objective are set, goal programming techniques derive problem models (LP, MIP, etc.) to find solutions that reach (or are close enough to) the target goals. Several strategies, or goal programming variants, have been presented in the literature. We briefly review the three most widely employed variants [12]:

- **Weighted GP** [13]: used when the decision maker is able to assign an *importance* weight to each goal. The objective function for the problem is then a weighted sum of the deviations from the goals.
- **Lexicographic GP**: when weighting the goals is difficult, but the decision maker is able to prioritise them, the lexicographic GP technique is commonly applied [14]. The deviations to the target goals are minimised according to defined priority levels such that deviations from a higher level goal are considered infinitely more important than deviations from a lower level goal.
- **Chebyshev GP** [15]: consists of minimising the maximum weighted normalised deviation from all the goals, hence promoting solutions that are well-balanced regarding the achievement of the target values.

The weighting and lexicographic methods are considered ‘a priori’ approaches in the sense that the decision maker should set a ranking between the objectives before conducting the search for solutions. This is not the case in the Chebyshev method which is an ‘a posteriori’ method because it seeks solutions that are well-balanced in the attainment of all goals so that the decision maker can choose afterwards. In this paper, it is assumed that the decision maker is able to choose a preferred solution from a set of trade-off solutions, instead of being able to establish weights or ranking between the multiple objectives. Hence, only the Chebyshev technique is used later in this work.

A potential issue with goal programming is that it may produce solutions that are not Pareto efficient [16]. This is especially true when the goals are ‘pessimistic’ and the objectives can be easily achieved. Several methods are proposed to address the issue. Most methods rely on extra information from the decision maker in order to promote the further improvement of certain objectives [17]. Other methods involve extending the search after the solution is found by the goal programming in order to find dominating solutions [18].

Works in the literature usually describe the application of goal programming using exact methods [12, 16]. However, many works exist where metaheuristics are employed to solve goal programming models. [19] presents a simulated annealing approach to tackle several test problems of preemptive goal programming. [20] employ a fast converging simulated annealing algorithm to solve a machine-tool selection and operation allocations problem with fuzzy variables. [21] propose a genetic algorithm to tackle a goal programming model for the vehicle routing problem with time windows and [22] presents a genetic algorithm to tackle a goal programming model for a transportation planning problem with three objectives. Goal programming is a sound approach to tackle the MOVRPTW considered here because this technique has been successfully applied to related scheduling and routing problems. For example, it has been applied to nurse scheduling [23, 24] and to a version of the vehicle routing problem with soft time-windows [25].

4 The Efficient GP Methodology

Figure 1 shows the overall concept of the methodology which was originally proposed in [7]. Each of the steps is explained below in reference to the MOVRPTW tackled in this paper. The overall idea is to find a set of compromise solutions for a representative instance of the multiobjective problem in hand. The decision maker then selects from this set a solution that exhibits the desirable qualities in respect of the various objectives, without the need to set weights or priorities for the objectives. The objective values in the selected solutions are set as the targets for goal programming when searching for solutions to the other problem instances (e.g. routing plans for other days in the same problem scenario).

1. A *pilot instance* from the dataset with recurring fitness landscape is selected by the decision-maker and solved using multiobjective algorithms to obtain the best possible non-dominated approximation set.
2. The decision-maker chooses a preferred solution t from the obtained non-dominated set. This chosen solution is known as the *target solution* and its objective-vector is denoted by

$$\mathbf{Z}^t = (Z_1^t, Z_2^t, Z_3^t, Z_4^t, Z_5^t)$$

3. Each other instance in the dataset can now be solved with a faster single-objective algorithm using a modified objective function (goal programming variant) aiming to reach the target objective vector.

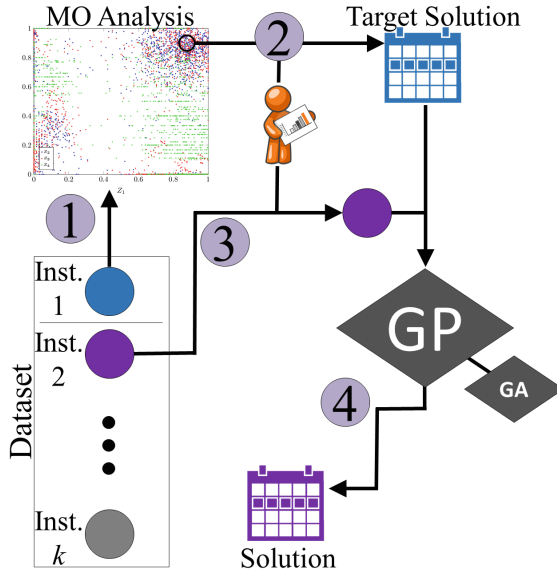


Fig. 1. Overview of the methodology as in [7]. The numbered steps are explained in the main text.

4. The final solution obtained in Step 3 is presented to the decision maker. The overall advantage of this approach is that Step 1, which is typically computationally expensive, needs to be executed only once for a given representative instance in the problem scenario. Then, other problem instances can be solved faster after the target solution is chosen.

The modified objective function of Step 3 has an important role in the methodology as it establishes the way in which the search will aim to attain the goals. Three approaches are used here for determining the objective function. The first one is the well known Chebyshev approach. The second one is to derive a weight-vector from the target solution and the approximation set of the pilot instance. The third approach minimises the Euclidean distances to the target objective-vector.

4.1 Chebyshev Goal Programming

Chebyshev goal programming aims to obtain a balanced solution by minimising the gap to the target of the objective that presents the highest gap, i.e. it seeks to minimise the largest gap to the goals [15]. Hence, if the target goals for the objectives are similarly difficult to attain, this technique can obtain a balanced solution. However, if at least one target objective value is more difficult to achieve (i.e. the target goal is too optimistic), the quality of that objective can be a bottleneck for the other objectives because the search will solely focus

on improving that objective. We define the Chebyshev objective function for the MOVRPTW as follows:

$$\text{Minimise } \lambda \tag{1}$$

Subject to

$$\frac{Z_1}{Z_1^t} \leq \lambda \tag{2}$$

$$\frac{Z_2}{Z_2^t} \leq \lambda \tag{3}$$

$$\frac{Z_3}{Z_3^t} \leq \lambda \tag{4}$$

$$\frac{Z_4}{Z_4^t} \leq \lambda \tag{5}$$

$$\frac{Z_5}{Z_5^t} \leq \lambda \tag{6}$$

The Chebyshev objective function given by Eq. (1) is used as the objective function for the MOVRPTW. The main objective is now to minimise λ , thus finding a well-balanced solution regarding reaching the target values. If all targets are reached, λ can assume fractional values and a solution that shows balanced improvements on all objectives may be obtained.

4.2 Derived Weight Vector

One problem with the Chebyshev approach is that it does not guarantee Pareto efficiency. However, the optimal solution for a weighted sum objective function (where weights are not simultaneously null) is always Pareto efficient. To derive a weight vector from the target solution, we first convert the approximation set of the pilot instance into a system of linear inequalities. Considering that the approximation set is composed of k objective-vectors (Z^1, Z^2, \dots, Z^5), the linear inequalities system can be defined as follows where the aim is to determine the values of $w = (w_1, w_2, w_3, w_4, w_5)$:

$$\begin{cases} wZ^t \leq wZ^1 \\ wZ^t \leq wZ^2 \\ \vdots \\ wZ^t \leq wZ^k \end{cases} \tag{7}$$

There is no guarantee that the system of linear inequalities has a solution if the fitness landscape is non-convex, i.e. if no set of weights can be set to achieve some points in the Pareto optimal front. Therefore, instead of finding a solution for the system, we aim to find a weight vector w that satisfies the largest number

of inequalities. Hence, we define the problem of finding the best weight vector as the following MIP (mixed-integer programming) minimisation problem.

$$\text{Minimise } \sum_{j=1}^k \bar{x}_j \tag{8}$$

Subject to

$$\mathbf{w}\mathbf{Z}^t - \mathbf{w}\mathbf{Z}^j \leq M\bar{x}_j \quad j = 1, \dots, k \tag{9}$$

$$w_i \in (0, 1], \bar{x}_j \text{ binary} \quad \begin{cases} i = 1, \dots, 5 \\ j = 1, \dots, k \end{cases} \tag{10}$$

The objective function in Eq. (8) aims to find a weight vector \mathbf{w} that minimises the number of linear inequalities in (7) which do not fulfill the condition $\mathbf{w}\mathbf{Z}^t \leq \mathbf{w}\mathbf{Z}^i$ expressed by constraint (9), M is a large constant. Constraint (10) guarantees that zero cannot be chosen as a weight-value (to avoid criteria being removed).

Finally, the weight vector \mathbf{w} obtained from the MIP model is used in the objective function for the MOVRPTW as given by Eq. (11).

$$\text{Minimise } \sum_{i=1}^5 w_i Z_i \tag{11}$$

4.3 Euclidean Distances

We propose an alternative based on the Euclidean distances to the target vector. In essence, this is a method that considers all objectives as equally important. Hence, minimising the Euclidean distances alone does not guarantee Pareto efficiency. In order to mitigate this issue, the proposed method consists of minimising the distances to the target vector for the objectives that are worse than the target. If the current distance for the objectives that are worse than the target vector is small ($< \epsilon$), then the aim is to maximise the distances of the objectives that are better than the target vector.

Henceforth, the objective function in Eq. (12) becomes the objective function for the optimisation problem in hand.

$$\text{Minimize } \begin{cases} z & \text{if } z > \epsilon \\ -z' & \text{otherwise} \end{cases} \tag{12}$$

where

$$z = \sqrt{\sum_{i=1}^5 z_i} \tag{13}$$

$$z' = \sqrt{\sum_{i=1}^5 z'_i} \tag{14}$$

$$z_i = \begin{cases} (Z_i - Z_i^t)^2 & \text{if } Z_i > Z_i^t \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

$$z'_i = \begin{cases} (Z_i - Z_i^t)^2 & \text{if } Z_i \leq Z_i^t \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

In summary, when the Euclidean distances of the objectives that are worse than the target vector are larger than the given parameter ϵ , the objective function consists of minimising the Euclidean distances (z). Otherwise, when $z \leq \epsilon$, the objective consists of maximising the distances for the objectives that are better than the target solution (z'). Thus, if the solution has not reached the target, the objective function attempts to close the gap to the target. If the solution is close or better than the target, the objective function attempts to further improve it.

5 Experimental Configuration

We applied the proposed methodology to the MOVRPTW datasets. The instances with $\delta 0$ and $tw4$ ($50\text{-}\delta 0\text{-}tw4$, $150\text{-}\delta 0\text{-}tw4$, $250\text{-}\delta 0\text{-}tw4$) were arbitrarily selected as pilot instances (Step 1 of methodology). Once the Pareto approximation sets were obtained, $k = 15$ target vectors were randomly selected (uniformly distributed) from each approximation set and the same target vectors were used for the Derived Weight Vector (WV) objective function, the Euclidean Distances (ED) objective function, and the Chebyshev (CV) objective function.

Multiobjective algorithms often struggle to find good approximation sets for combinatorial problems with many objectives (more than three) [1]. Hence, we resort to a tailored procedure to obtain an improved approximation set. [2] state that most multiobjective algorithms can be classified as either Pareto-based or decomposition-based. This study utilises NSGA-II [26] as the Pareto-based algorithm and MOEA/D [27] as the decomposition-based one. Thus, **for each problem instance** the approximation set was obtained (Step 1 of methodology) as described below. The number of solution vectors obtained for each pilot instance was 168 for $50\text{-}\delta 0\text{-}tw4$, 215 for $150\text{-}\delta 0\text{-}tw4$ and 206 for $250\text{-}\delta 0\text{-}tw4$.

1. run both the NSGA-II and MOEA/D for one million objective evaluations on each possible bi-objective vector (Z_1, Z_2) , (Z_1, Z_3) , \dots (Z_4, Z_5) ;

2. run both the NSGA-II and MOEA/D for one million objective evaluations on each possible three-objective vector $(Z_1, Z_2, Z_3), (Z_1, Z_2, Z_4), \dots (Z_3, Z_4, Z_5)$;
3. run both the NSGA-II and MOEA/D for one million objective evaluations on each possible four-objective vector $(Z_1, Z_2, Z_3, Z_4), (Z_1, Z_2, Z_3, Z_5), \dots (Z_2, Z_3, Z_4, Z_5)$;
4. create an archive composed of the non-dominated solutions found in the previous three steps;
5. generate a population of individuals where half of the elements are randomly generated and the other half are randomly drawn from the archive built in the previous step;
6. run both the NSGA-II and MOEA/D four times each, for two million objective evaluations, using the initial population generated in the previous step and the five-objective vector; and
7. compile an approximation set with all non-dominated solutions found in all steps.

[28] survey the literature on vehicle routing problem with time windows and show that genetic algorithms are well suited for that problem. Also, our early experiments showed that these algorithms present good enough solutions on these datasets and are simple enough to allow easy replication by other researchers. Hence, for Step 3 of the methodology, the other instances of the MOVRPTW are tackled with a straightforward genetic algorithm (GA) using a direct integer encoding of solutions, uniform crossover, 500 individuals population with a 5% elite being kept across generations and a tournament of two individuals employed for the selection mechanism.

6 Experimental Results

First, we show the effectiveness of the derived weight vector obtained from the MIP model in Eqs. (8)–(10). The effectiveness of a weight vector \mathbf{w} is given by the percentage of solutions (in the approximation set for the pilot instance) in which $\mathbf{w}\mathbf{Z}^t \leq \mathbf{w}\mathbf{Z}^i$, $i = 1, \dots, 5$. Hence, if the effectiveness is 100%, it means that the MIP model found a solution for the inequalities system in Eq. (7).

Figure 2 presents the results of the effectiveness analysis. As it was the case in [7] for another problem, the overall effectiveness of the obtained weight vectors here surpassed 90%. Pilot instance 50- $\delta 0$ -tw4 presented the best average value of 96% and 250- $\delta 0$ -tw4 presented the worst with 91.3%. Hence, in all cases, the MIP model provided good weight vectors to be used by the WV objective function.

Next, we show the results for each group of instances (for 50, 150 and 250 customers) in three charts. The *target achievement* chart displays the percentage of solutions, in the given dataset, that achieved the target value in each objective. The *gap to target* chart contains the average gap to the target solutions for the solutions that did not reach the target. Finally, the *overall comparison* chart displays the average quality of solutions where positive values indicate that, on

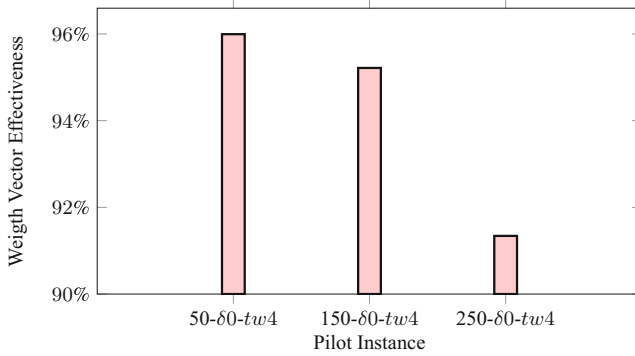


Fig. 2. Average percentage of the solutions in the approximation set of each pilot instance in the MOVRPTW datasets such that $wZ^t \leq wZ^i$.

average, the solutions found are better than the target solution and negative values indicate that the solutions are worse than the target solution.

Figures 3, 4 and 5 display the results of applying the implemented GA with all three objective functions (WV, ED and CV) to the other instances of dataset VRP-50. Results comprise the average values of eight runs for each target vector of each problem instance for each objective function.

Figure 3 shows that for Z_1 and Z_2 the target achievement is close to 100% on all three objective functions. On Z_3 the WV objective function noticeably presents the worst results, with only 63% achievement while the ED and CV objective functions both present similar results with near 80% achievement. Finally, on Z_4 and Z_5 the ED objective function presents a small advantage and the CV objective function is clearly the worst for Z_5 .

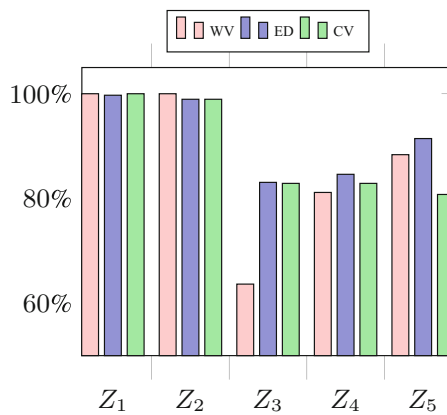


Fig. 3. Dataset VRP-50 – target achievement.

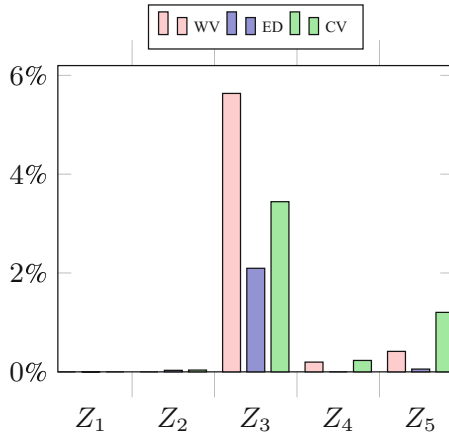


Fig. 4. Dataset VRP-50 – gap to the target.

Figure 4 reflects the findings of the previous figure where Z_3 shown the lowest overall target achievement. Still, on that objective, the overall gap is below 6% for the three objective functions, hence when the target was not met, the gap still was small. Noticeably, the ED objective function presents the lowest gaps. Moreover, Fig. 5 shows that except for WV on Z_3 , all objective functions on all objectives present improvements over the target solution, noticeably on Z_1 , Z_2 and Z_4 where the solutions found are up to 58% better than the target.

Figures 6, 7 and 8 present the results for the larger set VRP-150. On Fig. 6, we see that while on dataset VRP-50 the objective Z_3 presents the worst results, in this dataset the worst results appear on Z_4 with an average of roughly 75%

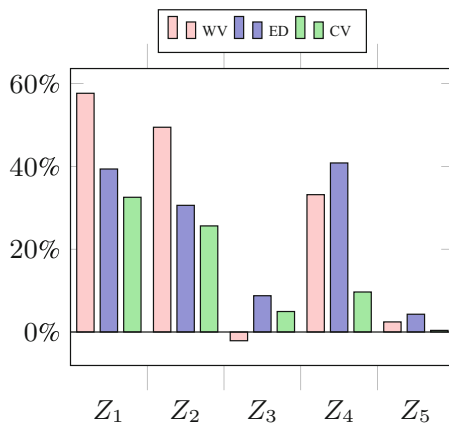


Fig. 5. Dataset VRP-50 – overall comparison.

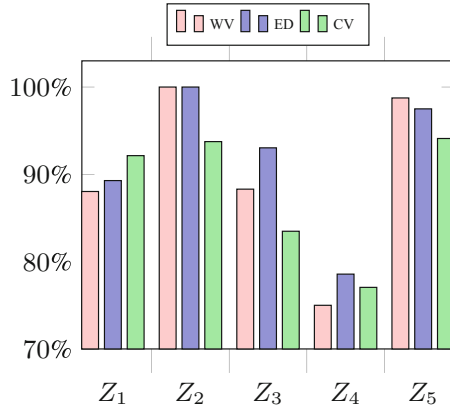


Fig. 6. Dataset VRP-150 – target achievement.

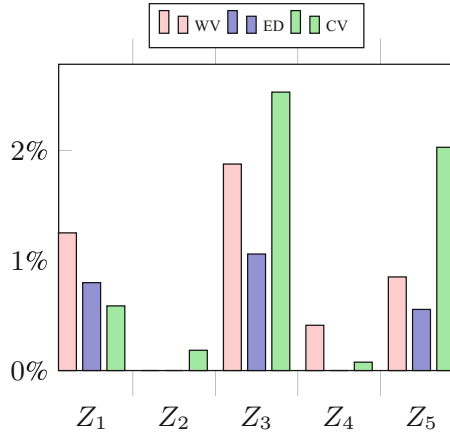


Fig. 7. Dataset VRP-150 – gap to the target.

achievement and, again, the WV objective function presents the worst results. On the other objectives, all objective functions present competitive results.

Figure 7 shows that the gap to the target on solutions that have not met the target is very small – only on Z_2 the gap is larger than 2% and only for the CV objective function.

Figure 8 displays the overall quality of solutions. On average, the quality is better on this dataset than on the previous one. With one or more objective functions, on every objective, the overall quality is more than 20% better than the target. This number increases to nearly 40% for the WV objective function on Z_1 and Z_2 .

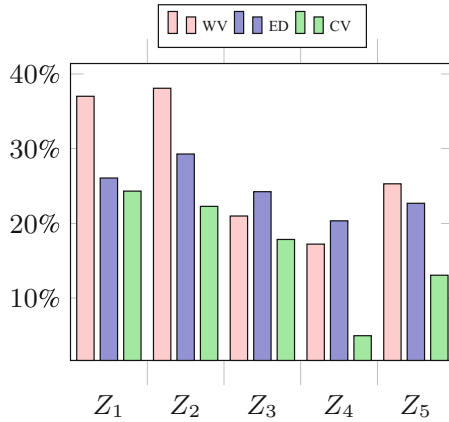


Fig. 8. Dataset VRP-150 – overall comparison.

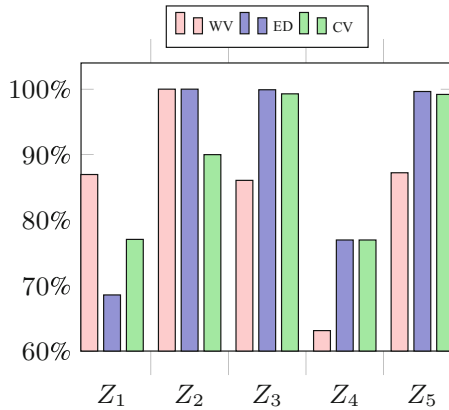


Fig. 9. Dataset VRP-250 – target achievement.

Finally, Figs. 9, 10 and 11 present the results for the largest dataset VRP-250. Figure 9 presents the target achievement. It can be seen that there is a trend, as the size of the datasets increases, the target achievement of Z_1 decreases. In this dataset, the objectives Z_1 and Z_4 presents the worst results. Regarding the objective functions, WV presents the best results for Z_1 . On the remaining objectives, the ED objective function presents the most competitive results.

Figure 10 shows the overall gaps to the target solutions. Clearly, the WV approach gets the worst results, even though the gaps were always below 4.2%. Noticeably, the CV objective function presents gaps always smaller than 1%.

Lastly, Fig. 11 shows the overall comparison of solutions with their targets. Again, the results show that all objective functions achieved improved results, with the ED edging Z_3 , Z_4 and Z_5 and the WV edging Z_1 and Z_2 .

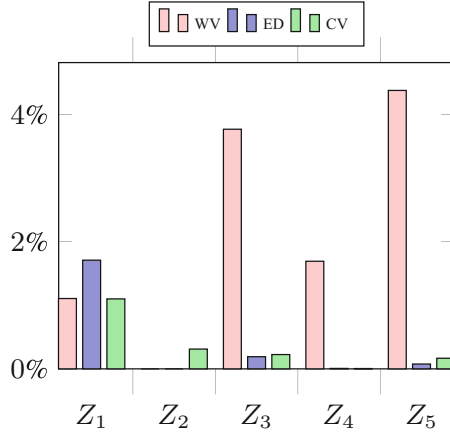


Fig. 10. Dataset VRP-250 – gap to the target.

7 Discussion

While the WSRP datasets tackled in [7] arise from real-world scenarios, the MOVRPTW datasets considered here were fabricated for benchmarking purposes. Also, even the largest MOVRPTW scenario is considerably smaller than a medium-sized WSRP. The target achievement for the MOVRPTW here was larger than in the WSRP overall. The best results obtained here were for the smaller MOVRPTW scenarios, while for the WSRP this happened in the larger instances. We speculate that a reason for this is that the largest MOVRPTW datasets are not large enough for the multiobjective algorithms to struggle in finding good approximation sets (as it happened in the larger WSRP datasets). Therefore, as the performance gap between single-objective algorithms and multiobjective algorithms is considerably smaller in the MOVRPTW problem instances, the difficulties of reaching the target vector becomes more evident.

However, the gaps to the targets of objectives that did not meet their targets were considerably lower here than on the WSRP. Also, the CV objective function, while clearly producing the worst results on the WSRP, it achieves competitive results on the MOVRPTW. This could be a consequence of the quality of the target solutions. The multiobjective algorithms were able to obtain approximation sets with fitness landscape closer the fitness landscape of the optimal Pareto front. Also, there is a higher uniformity of the fitness landscapes across instances for these datasets [6]. All this means that the identified target solutions were realistic, so they could be achieved on every instance. Hence, the CV objective function, which benefits from that, presented good results.

On the MOVRPTW datasets, except for a few exceptions, all objective functions were able to not only reach the target but also to substantially improve all objectives – also a reflection of the quality of the approximation set obtained by the multiobjective algorithms.

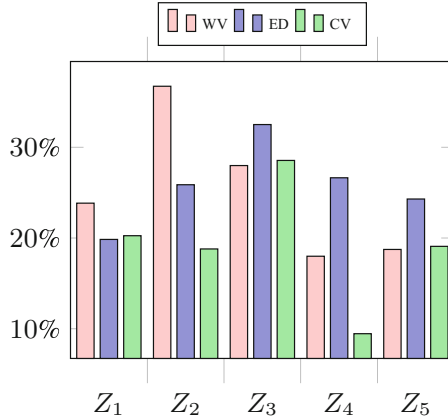


Fig. 11. Dataset VRP-250 – overall comparison.

Nonetheless, it is clear that estimating the Pareto front for problem instances that have similar fitness landscape to the pilot instance, is an effective way to tackle the problem. While the multiobjective algorithms required up to four hours to obtain the approximation set for the pilot instance of a dataset, the GA managed to find competitive solutions in minutes. For the majority of the experiments, targets were achieved and the overall quality of results was high.

8 Conclusion

In this work, we applied a methodology based on goal programming to use efficient single-objective algorithms to solve a multiobjective vehicle routing problem with time windows. The methodology was first presented in [7] and it consists of: (1) solving a pilot instance of the problem using multiobjective algorithms (which are typically computationally expensive) to obtain a good approximation set, (2) having the decision-maker to choose preferred target compromise solutions, and then (3) employing goal programming to solve other instances of the same dataset using the selected solutions in (2) as the target. Three different objective functions were used to guide the search for the target solutions with goal programming. One is the Chebyshev approach that seeks to achieve a solution balanced on all the objective targets. Another one is minimising a weighted function derived from the target solution. The third approach is to use the Euclidean distance to drive the search guided by the target solution.

This methodology was first applied by [7] to real-world instances of a Workforce Scheduling and Routing Problem (WSRP) in the home healthcare sector. In the present paper, the methodology has been further tested by applying it to a different multiobjective problem arising in logistic operational scenarios, the Multiobjective Vehicle Routing Problem with Time Windows (MOVRPTW). In both of these problem scenarios, instances usually arise from different planning periods and hence they present similarities in the fitness landscapes.

This is because usually in this type of real-world problems, instances have the same partial data (e.g. same fleet of vehicles or same set of workers). This paper has shown that the proposed technique is an effective and efficient approach to tackle real-world multiobjective highly-constrained combinatorial optimisation problems, by combining the effectiveness (but often computationally expensive) of state-of-the-art multiobjective algorithms with the efficiency of well-targeted single-objective optimisation through goal programming. For this, the multi-objective analysis technique proposed by [5, 6] offers an effective tool to analyse the relationships between objectives in multiobjective optimisation problems and determine the degree of similarity in the fitness landscape of different problem instances.

For future research, it would be interesting to investigate if other approaches besides the Chebyshev, derived weighted function and Euclidean distance approaches, would be more effective across different multiobjective problems. Perhaps an even more interesting but also more challenging future research would be to develop adaptive objective functions that change the search direction as the search progresses and in reaction to the fitness landscape features.

References

1. Giakiozis, I., Fleming, P.: Methods for many-objective optimization: an analysis. Research Report No. 1030 (2012)
2. Giakiozis, I., Fleming, P.: Pareto front estimation for decision making. *Evol. Comput.* **22**, 651–678 (2014)
3. Feliot, P., Bect, J., Vazquez, E.: A Bayesian approach to constrained single- and multi-objective optimization. *J. Glob. Optim.* **67**(1–2), 97–133 (2017). <https://link.springer.com/article/10.1007/s10898-016-0427-3>
4. Toth, P., Vigo, D.: The Vehicle Routing Problem. Monographs on Discrete Mathematics and Applications. Society for Industrial and Applied Mathematics (2002)
5. Pinheiro, R.L., Landa-Silva, D., Atkin, J.: Analysis of objectives relationships in multiobjective problems using trade-off region maps. In: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, GECCO 2015, pp. 735–742. ACM, New York (2015)
6. Pinheiro, R.L., Landa-Silva, D., Atkin, J.: A technique based on trade-off maps to visualise and analyse relationships between objectives in optimisation problems. *J. Multi-Criteria Decis. Anal.* **24**, 37–56 (2017)
7. Pinheiro, R.L., Landa-Silva, D., Laesanklang, W., Constantino, A.A.: Using goal programming on estimated pareto fronts to solve multiobjective problems. In: Proceedings of the 7th International Conference on Operations Research and Enterprise Systems - Volume 1: ICORES, INSTICC, pp. 132–143. SciTePress (2018)
8. Castro-Gutierrez, J., Landa-Silva, D., Moreno, P.J.: Nature of real-world multi-objective vehicle routing with evolutionary algorithms. In: IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 257–264 (2011)
9. Castro-Gutierrez, J., Landa-Silva, D., Moreno Perez, J.: Movrptw dataset (2015). <https://github.com/psxjpc/> Accessed 24 Apr 2018
10. Charnes, A., Cooper, W.: Goal programming and multiple objective optimizations. *Eur. J. Oper. Res.* **1**, 39–54 (1977)
11. Kornbluth, J.: A survey of goal programming. *Omega* **1**, 193–205 (1973)

12. Jones, D., Tamiz, M.: A review of goal programming. In: Greco, S., Ehrgott, M., Figueira, J. (eds.) *Multiple Criteria Decision Analysis*. International Series in Operations Research & Management Science, pp. 903–926. Springer, New York (2016). https://doi.org/10.1007/978-1-4939-3094-4_21
13. Romero, C.: *Titles of related interest*. In: *Handbook of Critical Issues in Goal Programming*. Pergamon, Amsterdam (1991)
14. Tamiz, M., Jones, D.F., El-Darzi, E.: A review of goal programming and its applications. *Ann. Oper. Res.* **58**, 39–53 (1995)
15. Flavell, R.B.: A new goal programming formulation. *Omega* **4**, 731–732 (1976)
16. Jones, D., Tamiz, M.: Goal programming variants. In: *Practical Goal Programming*, pp. 11–22. Springer, Boston (2010). https://doi.org/10.1007/978-1-4419-5771-9_2
17. Tamiz, M., Mirrazavi, S., Jones, D.: Extensions of pareto efficiency analysis to integer goal programming. *Omega* **27**, 179–188 (1999)
18. Hannan, E.: Nondominance in goal programming. *INFOR Inf. Syst. Oper. Res.* **18**, 300–309 (1980)
19. Baykasoglu, A.: Preemptive goal programming using simulated annealing. *Eng. Optim.* **37**, 49–63 (2005)
20. Mishra, S., Prakash, Tiwari, M.K., Lashkari, R.S.: A fuzzy goal-programming model of machine-tool selection and operation allocation problem in FMS: a quick converging simulated annealing-based approach. *Int. J. Prod. Res.* **44**, 43–76 (2006)
21. Ghoseiri, K., Ghannadpour, S.F.: Multi-objective vehicle routing problem with time windows using goal programming and genetic algorithm. *Appl. Soft Comput.* **10**, 1096–1107 (2010)
22. Leung, S.C.H.: A non-linear goal programming model and solution method for the multi-objective trip distribution problem in transportation engineering. *Optim. Eng.* **8**, 277–298 (2007)
23. Azaiez, M.N., Al Sharif, S.S.: A 0-1 goal programming model for nurse scheduling. *Comput. Oper. Res.* **32**, 491–507 (2005)
24. Musa, A.A., Saxena, U.: Scheduling nurses using goal-programming techniques. *IIE Trans.* **16**, 216–221 (1984)
25. Calvete, H.I., Galé, C., Oliveros, M.J., Sánchez-Valverde, B.: A goal programming approach to vehicle routing problems with soft time windows. *Eur. J. Oper. Res.* **177**, 1720–1733 (2007)
26. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **6**, 182–197 (2002)
27. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. Evol. Comput.* **11**, 712–731 (2007)
28. Bräysy, O., Gendreau, M.: Vehicle routing problem with time windows, Part II: metaheuristics. *Transp. Sci.* **39**, 119–139 (2005)