Applied Intelligence 24, 17–30, 2006

Multi-Objective Genetic Algorithms for Vehicle Routing Problem with Time Windows

BEATRICE OMBUKI, BRIAN J. ROSS AND FRANKLIN HANSHAR
Department of Computer Science, Brock University, St. Catharines, ON, Canada L2S 3A1
bombuki@brocku.ca
bross@brocku.ca
fh01ab@brocku.ca

Abstract. The Vehicle Routing Problem with Time windows (VRPTW) is an extension of the capacity constrained Vehicle Routing Problem (VRP). The VRPTW is NP-Complete and instances with 100 customers or more are very hard to solve optimally. We represent the VRPTW as a multi-objective problem and present a genetic algorithm solution using the Pareto ranking technique. We use a direct interpretation of the VRPTW as a multi-objective problem, in which the two objective dimensions are number of vehicles and total cost (distance). An advantage of this approach is that it is unnecessary to derive weights for a weighted sum scoring formula. This prevents the introduction of solution bias towards either of the problem dimensions. We argue that the VRPTW is most naturally viewed as a multi-objective problem, in which both vehicles and cost are of equal value, depending on the needs of the user. A result of our research is that the multi-objective optimization genetic algorithm returns a set of solutions that fairly consider both of these dimensions. Our approach is quite effective, as it provides solutions competitive with the best known in the literature, as well as new solutions that are not biased toward the number of vehicles. A set of well-known benchmark data are used to compare the effectiveness of the proposed method for solving the VRPTW.

Keywords: vehicle routing problem with time windows (VRPTW), genetic algorithm, multi-objective optimization, Pareto ranking

1. Introduction

Scheduling and routing problems are a subject of active research in the optimization community for a number of reasons. Firstly, they usually define challenging search problems, and are good for exercising new heuristic search techniques to their limits. They are easily cast into formal specifications, and standardized data sets of varying complexity are often made for them. This permits well-defined problem instances to be shared amongst researchers, thus making for effective comparisons of methodologies. Finally, they often have many practical real-world applications: the results are of genuine use to industry and others.

Vehicle Routing Problems (VRPs) are well known combinatorial optimization problems arising in transportation logistics that usually involve scheduling in constrained environments. In transportation management, there is a requirement to provide goods and/or services from a supply point to various geographically dispersed points with significant economic implications. VRPs have received much attention in recent years due to their wide applicability and economic importance in determining efficient distribution strategies to reduce operational costs in distribution systems. As a result, variants of VRP have been studied extensively in the literature (for detailed reviews, see [1–4]).

A typical VRP can be stated as follows: design least-cost routes from a central depot to a set of geographically dispersed points (customers, stores, schools, cities, warehouses, etc.) with various demands. Each customer is to be serviced exactly once by only one vehicle, and each vehicle has a limited capacity. The Vehicle Routing Problem with Time Windows (VRPTW) is an extension of the VRP; here a *time window* is associated with each customer. That is, in addition to the vehicle capacity constraint, each customer provides a time frame within which a particular service or task must be completed, such as loading or unloading a vehicle. A vehicle may arrive early, but it must wait until start of service time is possible. Some VRPTW models (*soft time window models*) allow for early or late window service, but with some form of penalty. However, most researchers have focused on the *hard time window models*, as does this paper. The objective of the VRPTW is to minimize the number of vehicles and total distance traveled to service the customers without violating the capacity and time window constraints. Capacity constraint is violated if the total sum of the customer demands in a given route exceeds the vehicle capacity. The VRPTW has received much attention due to applicability of time window constraints in real-world situations. Examples of practical applications of the VRPTW include school-bus and and taxi scheduling, courier and mail delivery/pickup, airline and railway fleet scheduling, and industrial refuse collection.

The VRPTW is a classic example of a NP-complete [5, 6] multi-objective optimization problem. The combinatorial explosion is obvious, and obtaining exact optimal solutions for this type of NP-hard problems is computationally intractable. Thus we can rarely accomplish optimal route schedules within reasonable time for large problem instances [6]. No polynomial algorithms have been developed for this type of problem, and their non-existence is generally believed [5]. Various researchers have investigated the VRPTW using exact and approximation techniques. Kohl's work [7] is one of the most efficient exact methods for the VRPTW; it succeeded in solving various 100-customer size instances. However, no algorithm has been developed to date that can solve to optimality all VRPTW with 100 customers or more. It should be noted that exact methods are more efficient in the situations where the solution space is restricted by narrow time windows, since there are fewer combinations of customers to define feasible routes [8].

Research on combinatorial optimization based on meta-heuristics has gained popularity especially since the 90s. These approaches seek approximate solutions in polynomial time instead of exact solutions which would be at intolerably high cost. Meta-heuristics, such as genetic algorithms (GA) [9–15, 17, 18], evolution strategies [16], simulated annealing [19], tabu search [21–25], and ant colony optimization [8] have been proposed for the VRPTW. Meta-heuristics are well suited to solving complex problems that may be too difficult or time-consuming to solve by traditional techniques. Other heuristics that have been applied to the VRPTW include constraint programming and local search [26, 27].

One of the most efficient techniques for the VRPTW has been the development of two-phased hybrid algorithms which divide the search into two stages: the minimization of (i) the number of routes and (ii) travel costs. A two-phased approach for the VRPTW is usually geared towards the design of algorithms tailored towards each sub-optimization. The two-phased algorithm normally uses two distinct local search procedures to exploit the minimization of routes which is followed by the minimization of travel costs. Gehring and Homberger [28] introduced a two-stage hybrid search which first minimizes the number of vehicles using an evolution strategy and then the total distance is minimized using a tabu search algorithm. In the two-phased tabu search, introduced by Potvin et al. [29], the first phase moved customers out of routes to reduce the total number of vehicles, and in the second phase inter- and intra-customer exchanges are done to reduce travel costs. Chiang and Russell [30] introduced a hybrid search based on simulated annealing and tabu search. A cluster-first, route-second method using genetic algorithms and local search optimization process was done by Thangiah [31]. Comparative studies of the performance of GA, tabu search and simulated annealing for the VRPTW is given in [31, 32].

Luca Maria Gambardella et al. [8] studied a type of multi-objective implementation of the VRPTW by minimizing a hierarchical objective function, where the first objective minimized the number of vehicles and the second minimized the total travel time. This was achieved by adapting the ant colony system (ACS) [33].

In previous work [11] we applied a hybrid search based on GA and tabu search to the soft VRPTW. While good results were obtained, the approach was two-phased: a GA was first used to set the number of vehicles, and then a local tabu search was employed to minimize the total cost of the distance traveled. In essence, the multi-objective VRPTW problem was transformed into a single-objective optimization.

It should be noted that all the above VRPTW work is biased towards the number of vehicles. Whenever a weighted sum fitness measure is undertaken, the vehicle and distance dimensions are essentially evaluated as a unified score. The advantage of this is that single solutions are obtained as a result. However, these solutions are biased by the single-objective transformation of the problem, and in all the previous work, this bias always prioritizes the number of vehicles. This can be seen by the fact that reported solutions tacitly assume that the vehicle count is first minimized, and then the distance is minimized with respect to this vehicle value.

This paper studies the VRPTW as a multi-objective optimization problem (MOP), as implemented within a GA. Specifically, the two dimensions of the problem to be optimized—the number of vehicles and the total distance traveled—are considered to be separate dimensions of a multi-objective search space. Although MOP's and genetic algorithms have been applied to the VRPTW before, this interpretation of the VRPTW as a MOP using GAs is uncommon. As with all MOP's, one immediate advantage is that it is not necessary to numerically reconcile these problem characteristics with each another. In other words, we do not specify that either the number of vehicles or the total distance traveled take priority. Using the Pareto ranking procedure, each of these problem characteristics is kept separate, and there is no attempt to unify them.

There are a number of advantages in using this literal MOP formulation of the VRPTW. First, by treating the number of vehicles and total distance as separate entities, search bias is not introduced. Second, there is a strong philosophical case to be made for treating the VRPTW as a MOP. The VRPTW specification requires a minimization of both the number of vehicles and total distance traveled. From a theoretical point of view, this may be impossible to realize, because instances of the VRPTW may have many non-dominated solutions. Some solutions may minimize the number of vehicles at the expense of distance, and others minimize distance while necessarily increasing the vehicle count. If one scans the literature, however, most researchers clearly place priority on minimizing the number of vehicles. Although this might be reasonable in some instances, it is not inherently preferable over minimizing distance. Minimizing the number of vehicles affects vehicle and labour costs, while minimizing distance affects time and fuel resources. Therefore, the VRPTW is intrinsically a MOP in nature, and our MOP formulation recognizes these alternative solutions.

Finally, our MOP formulation of the VRPTW is computationally advantageous. As will be shown, the performance and results obtained with the MOP with Pareto ranking are competitive with those found elsewhere.

Section 2 provides the VRPTW specification, and an overview of multi-objective optimization search. Experimental details are presented in Section 3 while Section 4 reports our results and gives comparisons with related work. A general discussion concludes the paper in Section 5.

2. Background

2.1. *Description of VRPTW*

The VRPTW is represented by a set of identical vehicles denoted by V, and a directed graph $G = (C, A)$, which consist of a set of customers, C. The nodes 0 and $n + 1$ represent the depot, i.e., exiting depot, and returning depot respectively. The set of n vertices denoting customers is denoted N. The arc set A denotes all possible connections between the nodes (including node denoting depot). No arc terminates at node 0 and no arc originates at node $n + 1$ and all routes start at 0 and end at $n + 1$. We associate a *cost* C_{ij} and a *time* t_{ij} with each arc $(i, j) \in A$ of the routing network. The travel time $t_{i,j}$ may include service time at customer i. Each vehicle has a capacity limit q and each customer i, has a demand d_i, $i \in C$. Each customer i has a *time window*, $[a_i, b_i]$, where a_i and b_i are the respective opening time and closing times of i. A vehicle may arrive before the beginning of the time window (i.e., a_i) meaning incur waiting time until service is possible. However, no vehicle may arrive past the closure of a given time interval, b_i. Vehicles must also leave the depot within the depot time window $[a_0, b_0]$ and must return before or at time b_{n+1}. Assuming waiting time is permitted at no cost, we may assume that $a_0 = b_0 = 0$; that is, all routes start at time 0.

The model has two types of decision variables x and s. For each arc (i, j), where $i \neq j, i \neq n + 1, j \neq 0$, and each vehicle k, the decision variable x_{ijk} is equal to 1 if vehicle k drives from vertex i to vertex j, and 0 otherwise. The decision variable s_{ik} denotes the time vehicle k, $k \in V$ starts to service customer i, $i \in C$. If vehicle k does not service customer i, then s_{ik} has no meaning. We may assume that $a_0 = 0$ and therefore $s_{0k} = 0$, $\forall k$. The objective of the VRPTW is to service all the C customers using the V vehicles such that the following objectives are met and the following constraints are satisfied.

Objectives

- Minimize the total number of vehicles used to service the customers.
- Minimize the distance traveled by the vehicles.

Constraints

- Vehicle capacity constraint is observed.
- Time window constraint should be observed.
- Each customer is serviced exactly once.
- Each vehicle route starts at vertex 0 and ends at vertex $n + 1$.

Figure 1 shows a simple graphical model of the VRPTW and its solution. In this example, there are two routes, route 1 with 4 customers and route 2 with 5 customers.

The VRPTW model can be mathematically formulated as shown below:

$$\min \quad \sum_{k\in V}\sum_{i\in N}\sum_{j\in N} c_{ij}x_{ijk} \qquad \text{such that;} \tag{1}$$

$$\sum_{k\in V}\sum_{j\in N} x_{ijk} = 1 \quad \forall i \in C \tag{2}$$

$$\sum_{i\in C} d_i \sum_{j\in N} x_{ijk} \leq q \quad \forall k \in V \tag{3}$$

$$\sum_{j\in N} x_{0jk} = 1 \quad \forall k \in V \tag{4}$$

$$\sum_{i\in N} x_{ihk} - \sum_{j\in N} x_{hjk} = 0 \quad \forall h \in C,$$

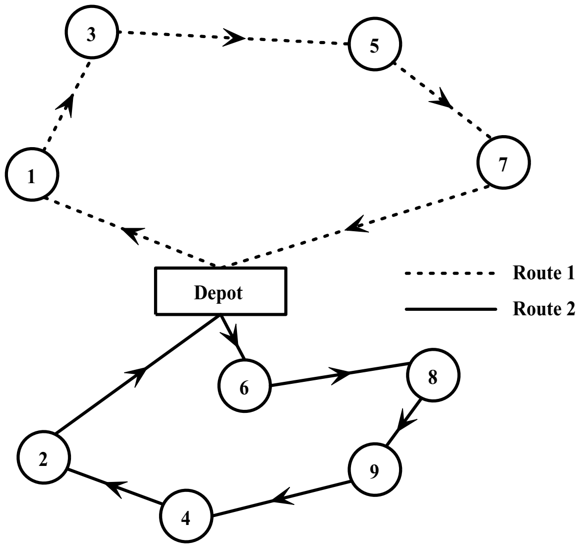

Figure 1. Example of a routing solution for VRPTW.

$$\forall k \in V \qquad (5)$$

$$\sum_{i \in N} x_{i,n+1,k} = 1 \qquad \forall k \in V \qquad (6)$$

$$s_{ik} + t_{ij} - K(1 - x_{ijk}) \leq s_{jk}$$
$$\forall i \in N, \forall j \in N, \forall k \in V \qquad (7)$$

$$ot_i \leq s_{ik} \leq ct_i \quad \forall i \in N, \forall k \in V \qquad (8)$$

$$x_{ijk} \in \{0, 1\} \quad \forall i \in N, j \in N, \forall k \in V \qquad (9)$$

$V = \{1, 2, \cdots, k\}$ vehicles

$C = \{1, 2, \cdots, n\}$ customer size

$0, n + 1$ depot

$N = \{0, 1, \ldots, n, n + 1\}$ node size

d_i client i demand

a_i client i open time

b_i client i close time

q_k vehicle k capacity

t_{ij} client i from j time

s_{ik} client i take k service time

The objective function (1) states that costs should be minimized. The constraint set (2) states that each customer must be visited exactly once by one vehicle, and constraint set (3) states that the vehicle capacity should not be exceeded. The next set of constraints (4), (5) and (6) give the flow constraints that ensure that each vehicle leaves depot 0, departs from a customer it visited and finally returns to the depot, given by node $n + 1$. The nonlinear inequality (7) (which can be easily linearized, see [1]) states that a vehicle K cannot arrive at j before $s_{ik} + t_{ij}$ if it travels from from i to j. Constraint (8) ensures that time windows are observed and (9) gives the set of integrality constraints.

2.2. *Multi-Objective Optimization and Pareto Ranking*

A multi-objective optimization problem (MOP) is one in which two or more objectives or parameters contribute to the overall result. These objectives often affect one another in complex, nonlinear ways. The challenge is to find a set of values for them which yields an optimization of the overall problem at hand. Evolutionary computation has been widely applied to MOP's [34–37]. Their success resides in the general applicability of evolutionary algorithms in finding good solutions to problems with appropriate structure, and the adaptability of genetic representation and fitness evaluation towards problems in the MOP field.

The Pareto ranking scheme has often been used in MOP applications of genetic algorithms [9]. It is easily incorporated into the fitness evaluation process within a genetic algorithm, by replacing the raw fitness scores with Pareto ranks. These ranks, to be defined below, stratify the population into preference categories. With it, lower ranks are preferable, and the individuals within rank 1 are the best in the current population.

The idea of Pareto ranking is to preserve the independence of individual objectives. This is done by treating the current candidate solutions as stratified sets or ranks of possible solutions. The individuals in each rank set represent solutions that are in some sense incomparable with one another. Pareto ranking will only differentiate individuals that are clearly superior to others in all dimensions of the problem. This contrasts with a pure genetic algorithm's attempt to assign a single fitness score to a MOP, perhaps as a weighted sum. Doing so essentially recasts the MOP as a single-objective problem. The difficulty with this is that the weighted sum necessitates the introduction of bias into both search performance and quality of solutions obtained. For many MOP's, finding an effective weighting for the multiple dimensions is difficult and *ad hoc*, and often results in unsatisfactory performance and solutions.

The following is based on a discussion in [36]. We assume that the MOP is a minimization problem, in which lower scores are preferred.

Definition. Given a problem defined by a vector of objectives $\vec{f} = (f_1, \ldots, f_k)$ subject to appropriate problem constraints. Then vector $\vec{u}$ **dominates** $\vec{v}$ iff

$$\forall i \in (1, \ldots, k) : u_i \leq v_i \wedge \exists i \in (1, \ldots, k) : u_i < v_i$$

This is denoted as $\vec{u} \preceq \vec{v}$.

The above definition says that a vector is dominated if and only if another vector exists which is better in at least 1 objective, and at least as good in the remaining objectives.

Definition. A solution $\vec{v}$ is **Pareto optimal** if there is no other vector $\vec{u}$ in the search space that dominates $\vec{v}$.

Definition. For a given MOP, the **Pareto optimal set** $\mathcal{P}^*$ is the set of vectors $\vec{v}_i$ such that $\forall v_i : \neg\exists\vec{u} : \vec{u} \preceq \vec{v}_i$.

Definition. For a given MOP, the **Pareto front** is a subset of the Pareto optimal set.

Many MOP's will have a multitude of solutions in its Pareto optimal set. Therefore, in a successful run of a genetic algorithm, the Pareto front will be the set of solutions obtained.

As mentioned earlier, a Pareto ranking scheme is incorporated into a genetic algorithm by replacing the chromosome fitnesses with *Pareto ranks*. These ranks are sequential integer values that represent the layers of stratification in the population obtained via dominance testing. Vectors assigned rank 1 are non-dominated, and inductively, those of rank $i + 1$ are dominated by all vectors of ranks 1 through i. Figure 2 shows how a Pareto ranking can be computed for a set of vectors. First, the set of non-dominated vectors in the population are assigned rank 1. These vectors

```
Curr_Rank := 1
N := (population size)
m := N
While $N \neq 0$ {     /* process entire population */
    For i := 1 to m {     /* find members in current rank */
        If $\vec{v}_i$ is non-dominated {
            rank($\vec{v}_i$) := Curr_Rank
        }
    }
    For i := 1 to m {     /* remove ranked members from population */
        if rank($\vec{v}_i$) = Curr_Rank {
            Remove $\vec{v}_i$ from population
            N := N-1
        }
    }
    Curr_Rank := Curr_Rank + 1
    m := N
}
```

Figure 2. Pareto ranking algorithm.

are removed, and the remaining non-dominated vectors are assigned rank 2. This is repeated until the entire population is ranked. Evolution then proceeds as usual, using the rank values as fitness scores. Note that Pareto ranks are relative measurements, and there is no concept of "best solution" using a rank score. Therefore, every generation in a run will have a rank 1 set. In order to determine whether an actual solution has been found, and that the run should be terminated, the raw fitness measurements need to be inspected.

Recently, more advanced approaches to solving MOP's using evolutionary algorithms have been proposed, such as SPEA [38], PAES [39], NSGA II [40], and others [37]. An important feature of advanced MOP strategies is the ability to maintain population diversity. Pareto ranks by themselves often result in strongly converged populations. This is a product of Pareto ranking, in which a population with a rich variety of fitness vectors is recast into discrete integral rank values. The result is usually the convergence of the population into a few strong niches within the Pareto set. The resulting lack of diversity inhibits effective evolutionary progress. Note that, although convergence also occurs with single-objective genetic algorithms, it is usually less acute than what arises in pure Pareto ranking.

Although it is possible that some of the new MOP approaches mentioned above might be effective for the VRPTW problem, it is not a goal of this research to find the most effective evolutionary MOP approach for solving the VRPTW. We leave this for future considerations. The nature of the VRPTW fitness space, however, would likely make some of these new approaches ineffective. Algorithms such as SPEA and NSGA II require the definition of niche spaces, which are used to evaluate diversity characteristics of the population. The VRPTW uses relatively low-valued discrete vehicle numbers as one of its dimensions. This aspect of the fitness space makes the definition of niche areas problematic, since most good solutions to the VRPTW reside in a very small portion of the vehicle number dimension of the problem. In fact, preliminary experiments with the VRPTW used a parameterless diversity strategy from [41], which is an enhancement of the NSGA II strategy [40]. We quickly realized that the VRPTW fitness space prevented the definition of effective fitness distances required by the diversity heuristic, and hence diversity was not enhanced whatsoever during the runs.

3. Multi-Objective Genetic Search for the VRPTW

This section provides the details of the VRPTW representation, fitness evaluation, Pareto strategy and other GA parameters used. In the GA, each chromosome in the population pool is transformed into a cluster of routes. The chromosomes are then subjected to an iterative evolutionary process until a minimum possible number of route clusters is attained or the termination condition is met. The transformation process is achieved by our routing scheme described in Section 3.7. The evolutionary part is carried out as in ordinary GAs using crossover and selection operations on chromosomes. Tournament selection with elite retention is used to perform fitness-based selection of individuals for further evolutionary reproduction. A problem-specific crossover operator that ensures solutions generated through genetic evolution are all feasible is also proposed. Hence, both checking of the constraints and repair mechanism can be avoided, thus resulting in increased efficiency. Figure 3 outlines the genetic routing system.

3.1. *Chromosome Representation and Initial Population Creation*

In order to apply the GA to a particular problem, we need to select an internal string (chromosome) representation for the solution space. The choice of this component is one of the critical aspects to the success/failure of the GA for a problem of interest. In our approach, a chromosome representing a network configuration is given by an integer string of length N, where N is the number of customers in a particular problem instance. A gene in a given chromosome indicates the original node number assigned to a customer, whilst the sequence of genes in the chromosome string dictates the order of visitation of customers. An example of a chromosome resulting in a solution for the network given in Figure 1 is as follows:

$$2 \quad 5 \quad 1 \quad 4 \quad 7 \quad 8 \quad 6 \quad 3 \quad 9$$

```
Read problem instance data
Set GA parameters
Generate randomly an initial population, POP
for gen := 1 to maxGen {
    Transform each individual into feasible network
    Evaluate fitness of the individuals of POP
    Rank the individuals of POP and select new population
    Apply GA operators (crossover and mutation)
};
```

Figure 3. An outline of the genetic routing system.

A chromosome string contains a sequence of routes, but no delimiter is used to indicate the beginning or end of a respective route in a given chromosome. However, the correspondence between a chromosome and the routes is further explained in Section 3.6. To generate the initial population, 90 percent of the population is created by random permutations of N customers nodes. The remaining 10 percentage is generated by a greedy procedure as follows:

1. Given a set of customers C of size N;
2. Initialize an empty chromosome string l;
3. Randomly remove a customer $c_i \in C$;
4. Add customer node c_i to the chromosome string l;
5. Within an empirically decided Euclidean radius centered around c_i, choose the nearest customer c_j, where $c_j \notin l$; else if c_j does not exist, then goto 3.
6. Append c_j to l, and remove c_j from C;
7. Let $c_i = c_j$ and goto 5.
8. If chromosome length $= N$, terminate, else goto 5;

3.2. *Pareto Fitness Evaluation and Other Evaluation Strategies*

Once each chromosome has been transformed into a possible feasible network topology using the route clustering scheme given in Section 3.7, the fitness of each chromosome is determined. The chromosome fitness is evaluated according to two approaches: (1) weighted sum fitness function and (2) rank based upon Pareto ranking technique.

3.2.1. Weighted Sum Method. This method requires adding the problem objective functions together using weighted coefficients for each individual objective. That is, our multi-objective VRPTW is transformed into a single-objective optimization problem where the fitness of an individual $F(x)$ is returned as:

$$\text{Fitness} = \alpha \cdot |V| + \beta \cdot \sum_{k \in V} D_k \tag{10}$$

$$D_k = \sum_{i \in N} \sum_{j \in N} t_{ij} x_{ijk} \tag{11}$$

α and β are weight parameters associated with the number of vehicles and the total distance traveled by vehicles respectively. The weight values of the parameters used in this function were established empirically and set at $\alpha = 100$ and $\beta = 0.001$.

3.3. *Pareto Ranking Procedure*

A straight-forward MOP interpretation of the VRPTW is adopted. The two objectives are the number of vehicles and the total cost. They define two independent dimensions in a multi-objective fitness space. Thus, using the characterization of Section 2.1, each candidate VRPTW solution in the population has associated with it a vector $\vec{v} = (n, c)$, where n is the number of vehicles for that candidate solution, and c is the total cost. Unlike the weighted sum above, these two dimensions are retained as independent values, to be eventually used by the Pareto ranking procedure.

```
Create initial population.
Repeat until max. generation reached {
  For each chromosome $i$ in population:
      Evaluate route, determining number of vehicles and cost.
⇒ $(n_i, c_i)$
      Determine Pareto rank.
        ⇒ $rank_i$
  Apply GA to population, using $rank_i$ as fitness value.
        ⇒ new population
}
```

Figure 4. GA with Pareto ranking.

Figure 4 shows how the Pareto ranking scheme of Figure 2 is incorporated with the genetic algorithm. Pareto ranking is applied to the (n, c) vectors of the population, essentially creating for the population a set of integral ranks ≥ 1. These ranks are then used by the GA as fitnesses for generating the next population. Note that the ranks themselves do not convey the quality of solutions, nor whether an optimal solution has been discovered. Each population, including the randomized initial population, is guaranteed to have a rank 1 set. This is not a disadvantage for general instances of the VRPTW anyway, since there is no efficient means of knowing whether a candidate VRPTW solution is truly optimal.

3.4. *Fitness-Based Selection*

At every generation stage, we need to select parents for mating and reproduction. The tournament selection strategy with elite retaining model [9] is used to generate a new population. The tournament selection strategy is a fitness-based selection scheme that works as follows. A set of K individuals are randomly selected from the population. This is known as the tournament set. In this paper, the set size is taken to be 4. We also select a random number r, between 0 and 1. If r is less than 0.8 (0.8 is set empirically by trying values 0.6, 0.7, 0.8, 0.9 and 1.0), the fittest individual in the tournament set is then chosen as the one to be used for reproduction. Otherwise, any chromosome is chosen for reproduction from the tournament set.

An elite model is incorporated to ensure that the best individual is carried on into the next generation. The advantage of the elitist method over traditional probabilistic reproduction is that it ensures that the current best solution from the previous generation is copied unaltered to the next generation. This means that the best solution produced by the overall best chromosome can never deteriorate from one generation to the next. In our GA, although the preceding fittest individual is passed unaltered to the next generation, it is forced to compete with the new fittest individual.

3.5. Recombination Phase

One of the unique and important aspects of the techniques involving genetic algorithms is the important role that recombination (traditionally, in the form of crossover operator) plays. In [11] we carried experiments where we established that two standard crossover operators: Uniform Order Crossover (UOX) [42] and Partially Mapped Crossover (PMX) [42] are not suitable for hard VRPTW. We then introduced Route Crossover (RC) which is an improvement of the UOX. Experimental details showed that the RC outperforms UOX and PMX (see details of RC in [11]). Thus, in this work, we initially employed RC. While we established that the RC is better suited for VRPTW than well-known crossover operators, a weakness of the RC is that it is more suited for soft VRPTW where some conditions are relaxed. For example, applying the RC occasionally results in some customers not being assigned to any vehicle. In this case, the chromosome resulting in unserviced customer(s) was simply penalized during the fitness evaluation stage. In this paper, we are dealing with hard VRPTW, where all the constraints should be satisfied, hence we need a crossover operator that does not result in some customers being unserviced. This paper employs a problem-specific crossover (Best Cost Route Crossover, BCRC) which aims at minimizing the number of vehicles and cost simultaneously while checking feasibility constraints. The dynamics of the proposed Best Cost Route Crossover are shown in Figure 5.

Figure 5 illustrates the creation of two offspring, C1 and C2, from two parents, P1 and P2, using an arbitrary problem instance of customer size 9 for explanation purposes. RP1 and RP2 give corresponding sets of routes associated with P1 and P2, respectively, at the current generation. For examples, P1 has three routes (R1–R3) with associated customers, i.e., R1: 3 1 7, R2: 5 6 and R3: 4 2 8 9. As shown in Step *a*, from each parent, a route is chosen randomly. In this case, for P1, route R2 with customers 5 and 6 is chosen, while for P2, route R3 with customers 7 and 3 is picked. Next, for a given parent, the customers in the chosen route from the opposite parent are removed. For example in Step *a*, for parent P1, customers 7 and 3 (which belonged to the randomly selected route in P2) are removed from P1 resulting in the upcoming child C1. Likewise customers 5 and 6 which belonged to a route in P1 are removed from the routes in P2, resulting in the upcoming C2.

Since each chromosome should contain all the customer numbers (for a given VRPTW problem instance), the next step is to locate the best possible locations for the removed customers in the corresponding children. As shown in Step *b*, the algorithm needs to re-insert customers 7 and 3 in child C1 and customers 5 and 6 in child C2, respectively. Note that the choice of which customer to insert first is done randomly, i.e., in creating C1, for example, the order of insertion of 7 and 3 is done arbitrarily. In this case, customer 3 was first inserted in the best location found in C1 (as shown in Step *b*) before 7 was inserted as shown in Step *c*.

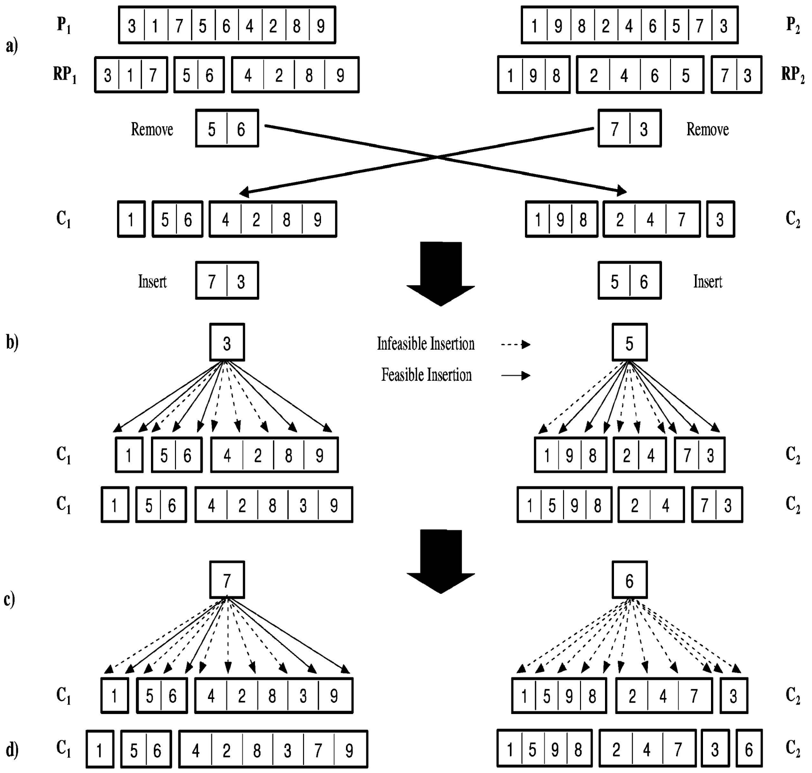

Figure 5. Example: Best cost route crossover (BCRC) operator.

An insertion point is said to be infeasible if it results in the routes either not meeting the vehicle capacity or time window constraints. The best insertion location is one that results in total minimum cost routes. In this example, customers 3 and 7 were both found to fit into route 3 of P1 as shown in Step *c*. Occasionally, no feasible insertion point is found and a new route is started. For example, in creating C2, customer 6 could not be inserted in the current routes for P2, hence a new route was created.

3.6. Constrained Route Reversal Mutation

Mutation aids a genetic algorithm to break free from fixation at any given point in the search space, and is used here in VRPTW for that very reason. Since mutation can be highly destructive of good schemas, each chromosome has a low probability of being mutated; in this research each chromosome has a 0.10 probability of being chosen for mutation. When utilizing mutation it may be best to introduce the smallest possible change in the chromosome, especially in the VRPTW, where the time windows can easily be violated. We propose a *constrained route reversal mutation*, which is an adaptation of a simple widely used mutation, usually referred to as inversion [42]. In a simple use of "inversion" mutation applied to the TSP, where the chromosome representation is simply a permutation of the order in which to visit each locale, two cut points are selected in the chromosome, and the genetic material between these two cuts points is reversed, for example given the TSP chromosome:

9 5 1 7 8 2 4 3

Two cut points are generated:

9 5 |1 7 8 2 4| 3

And the genetic material is reversed between the cut points giving:

9 5 4 2 8 7 1 3

In this paper mutation is carried out only in one randomly chosen route so as to minimize total route disruption. Maintaining the route time window imposes major constraints since the violation of individual customer time windows can segment that route into multiple routes. Thus we employ a constrained reversal, which is limited in length to 2–3 customers. Since a reversal of 2-customers is the smallest change one can make in a route (since it changes only two edges of the graph) we employ this type of mutation to aid search away from converging at local optima.

3.7. Routing Scheme

It is quite common among research for the VRPTW to route exhaustively in relation to vehicle capacity; that is, vehicles are filled with customers until capacity constraints disallow the addition of another customer. A worthy exception [32] attempts all feasible routing schemes, and chooses the routing scheme with the best cost. Our work used a two-phase routing scheme that transforms each of the chromosomes into a cluster of routes. In Phase 1, a vehicle must depart from the depot and the first gene of a chromosome indicates the first customer the vehicle is to service. A customer is appended to the current route in the order that he/she appears on the chromosome. The routing procedure takes into consideration that the vehicle capacity and time window constraints are not violated before adding a customer to the current route. A new route is initiated every time a customer is encountered that cannot be appended to the current route due to constraints violation. This process is continued until each customer has been assigned to exactly one route.

In Phase 2, the last customer of each route r_i, is relocated to become the first customer to route r_{i+1} . If this removal and insertion maintains feasibility for route r_{i+1}, and the sum of costs of r_1 and r_{i+1} at Phase 2 is less than sum of costs of $r_i + r_{i+1}$ at Phase 1, the routing configuration at Phase 2 is accepted, otherwise the network topology before Phase 2 (i.e., at Phase 1) is maintained.

4. Experimental Results and Comparisons

This section describes computational experiments carried out to investigate the performance of the proposed GA. In particular the experimental results shown here aim at showing two types of simulations: (i) where the VRPTW simulations consider only a single objective where minimizing the number of vehicles is given more weight over minimizing travel costs, and (ii) where VRPTW is considered as a multi-objective problem (MOP), hence concurrently minimizing both number of vehicles and travel costs without bias. The algorithm was coded in Java and run on an Intel Pentium IV 1.6 MHz PC with 512 MB memory. Our experimental results use the standard Solomon's VRPTW benchmark problem instances available at [43]. Solomon's data is clustered into six classes; C1, C2, R1, R2, RC1 and RC2. Problems in the C category means the problem is clustered, that is, customers are clustered either geographically or according to time windows. Problems in category *R* mean the customer locations are uniformly distributed whereas those in category RC imply hybrid problems with mixed characteristics from both C and *R*. Furthermore, for C1, R1 and RC1 problem sets, the time window is narrow for the depot, hence only a few customers can be served by one vehicle. Conversely, the remaining problem sets have wider time windows hence many customers can be served by main vehicles. See [44] for further descriptions of Solomon's problem sets. Unless otherwise stated, the results presented below are based on the following set of GA parameters:

- population size = 300
- generation span = 350
- crossover rate = 0.80
- mutation rate = 0.10

4.1. Comparisons with Best Published Results

In Figs. 6–8, we illustrate some of the network topologies obtained after running the GA for 350 generations. Figure 6 represents a data set where customers are clustered together and have a small time window. Figure 7 shows a data set where customers are also clustered but have a wider time window; hence as expected, the network topology shows that one vehicle can serve more customers as opposed to Fig. 8. On the other hand, Fig. 8 shows customers that also have a small time window but the locations of customers are uniformly distributed. It should be noted that nodes in the R1 category are much harder to solve than in the C category. Due to space limitations we show only three network topologies here, however, the general behavior is representative of the respective data sets.

Tables 1 and 2 present a summary of our results and compare them with the published solutions. Route costs are measured by average Euclidian distance. The column labeled *Best Known* gives the best known published solutions, column *wGA* gives the best solution in 10 runs, where the VRPTW was interpreted as a single objective problem by using a weighted sum fitness evaluation criterion. The columns labeled *pGA vehicles* and *pGA cost* show the best solutions obtained in 10 runs, when the VRPTW was interpreted as multi-objective optimization problem and the Pareto procedure was incorporated into our GA for fitness evaluation. The reported Pareto solutions in Tables 1 and 2 are two examples from the Pareto rank 1 set at the end of a run. *pGA vehicles* solution is the rank 1 solution that has the minimal number of vehicles, while the *pGA cost* value is the rank 1 solution with the minimal cost. Note that in experiments in which the difference in the number of vehicles in these two columns is ≥2, there are even further rank 1 solutions between these two extremes. We report only these two instances, however, to give an idea of the range of possible solutions returned by the Pareto MOP approach. Bolded figures in Tables 1 and 2 indicate an improvement on the best currently known results from literature (when considering either number of vehicles or cost). A tick, on the other hand, indicates that the solution we obtained is the same as the best known. The results obtained by our GA are quite good as compared to the best published results found in literature.

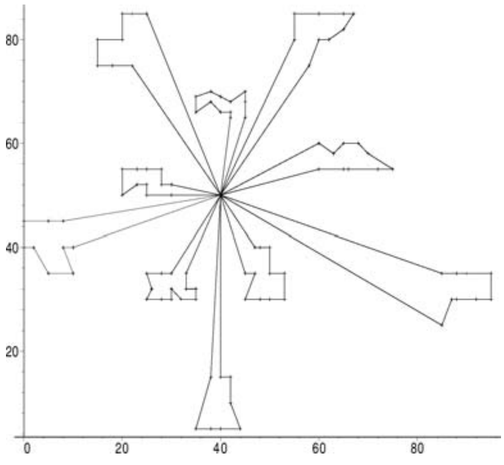

Figure 6. Network topology for 100 geographically clustered customers with a narrow time window. Test problem *c*101: [43].

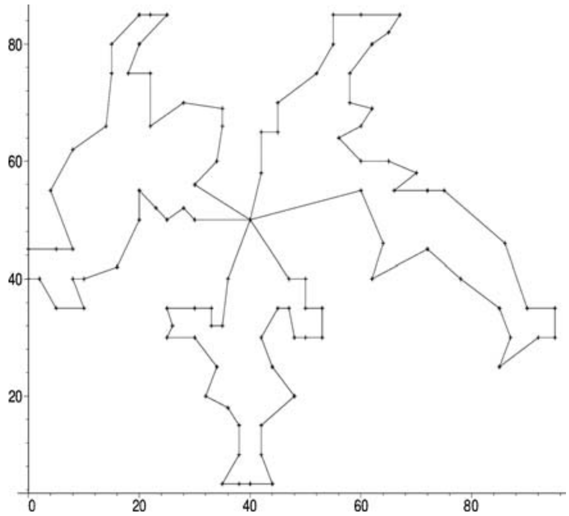

Figure 7. Network topology for 100 geographically clustered customers with a wide time window. Test problem *c*201: [43].

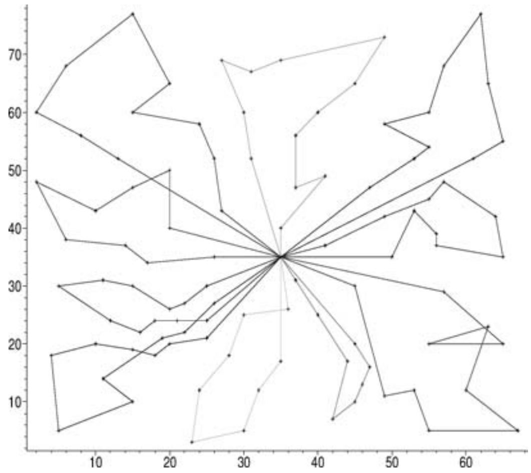

Figure 8. Network topology for 100 uniformly distributed customers with a narrow time window. Test problem *r*108: [43].

The advantages of the efforts of interpreting the VRPTW as a MOP using Pareto ranking as opposed to the single objective using weighted sum can be established from

Table 1. Solomon Benchmarks with narrow time windows: comparison of our GAs with best published results.

Instance data	Best known	Ref.	wGA	pGA vehicles	pGA cost
c101	10 828.94	[21]	√	√	√
c102	10 828.94	[21]	√	√	√
c103	10 828.06	[21]	√	√	√
c104	10 824.78	[21]	√	√ 825.65	√ 825.65
c105	10 828.94	[13]	√	√	√
c106	10 828.94	[21]	√	√	√
c107	10 828.94	[21]	√	√	√
c108	10 828.94	[21]	√	√	√
c109	10 828.94	[13]	√	√	√
r101	19 1650.8	[21]	√ 1685.27	√ 1690.28	20 1664.13
r102	17 1486.12	[21]	18 1523.10	√ 1513.74	18 1487.07
r103	13 1292.68	[27]	√ 1348.28	14 **1237.05**	14 **1237.05**
r104	9 1007.31	[27]	10 1010.36	10 1020.87	11 1010.24
r105	14 1377.11	[27]	15 1427.72	√ 1415.13	15 1390.12
r106	12 1252.03	[21]	√ 1273.62	13 1254.22	13 1254.22
r107	10 1104.66	[27]	**11 1100.97**	11 **1100.52**	11 **1100.52**
r108	9 963.99	[27]	10 **960.26**	10 975.34	10 975.34
r109	11 1194.73	[16]	12 1211.81	12 **1169.85**	13 **1166.09**
r110	10 1124.4	[20]	11 1146.11	11 **1112.21**	11 **1112.21**
r111	10 1096.72	[20]	11 1132.51	11 **1084.76**	12 **1079.82**
r112	9 982.14	[8]	10 985.99	10 **976.99**	10 **976.99**
rc101	14 1696.94	[23]	15 **1675.86**	15 **1636.92**	15 **1636.92**
rc102	12 1554.75	[23]	13 **1536.04**	14 **1488.36**	14 **1488.36**
rc103	11 1261.67	[27]	12 1309.59	12 1306.42	12 1306.42
rc104	10 1135.48	[27]	√ 1154.18	√ 1140.45	√ 1140.45
rc105	13 1633.72	[20]	14 **1623.33**	14 **1616.56**	16 **1590.25**
rc106	11 1427.13	[25]	12 1441.46	12 1454.61	13 **1408.70**
rc107	11 1230.48	[27]	√ 1271.59	12 1254.26	12 1254.26
rc108	10 1142.66	[23]	√	√ **1141.34**	12 1254.26

the solution quality. When using the Pareto ranking, one has a choice of two (or more) solutions, depending on whether the user wants the best number of vehicles or best travel costs solutions. In some experiments, for example c101 in Table 1, there is a single Pareto solution that is optimal to the best known in both vehicle and distance dimensions. Other solutions, such as rc102, reduce the distance significantly, but at the expense of adding extra vehicles. In rc105, there are two reported solutions in the table that have better distance scores than the best known solution. However, both these solutions use 14 and 16 vehicles, which add 1 and 3 vehicles respectively to the 13 vehicles used in the best known solution. Should distance (and hence time) be critical, then these alternate Pareto solutions are clearly preferable. Note that another Pareto solution with 15 vehicles may likely exist in rc105 as well, but it was not reported in the table. With a weighted sum approach, there is only one solution which does not necessarily effectively serve the purpose of both objectives.

Table 3 gives the average performance of the GA. The columns labeled *wGA* gives the average solutions (for vehicles and distance respectively) for 10 runs. For the pareto experiments, the columns labeled *pGA* gives the averages of all the rank 1 solutions (for vehicles and distance respectively) in 10 runs including duplicates. By comparing Table 3 with the solutions in Tables 1 and 2, we see that overally, the average GA performance is good.

Tables 4 and 5 give comparisons of our results with published results using GA (or hybrid GAs) in terms of average number of vehicles employed and average costs respectively (boldface are best solutions) for each problem set. Table 4 illustrates that our GAs (given by *wGA* and *pGA*) obtained better or similar average number of vehicles as compared to some of the well known published GA based methods for VRPTW. Except for [31] which is a hybrid strategy (incorporating GA, tabu search and simulated annealing), our *wGA* solution quality is better or very competitive with most of the other published work. The results of our multiple

Table 2. Solomon Benchmarks with wide time windows: comparison of our GAs with best published results.

Instance data	Best known	Ref.	wGA sum	pGA vehicles	pGA cost
c201	3 591.56	[13]	√	√	√
c202	3 591.56	[13]	√	√	√
c203	3 591.17	[21]	√	√	√
c204	3 590.60	[13]	√ 596.55	√ 596.55	√ 596.55
c205	3 588.88	[13]	√	√	√
c206	3 588.49	[13]	√	√	√
c207	3 588.29	[21]	√	√	√
c208	3 588.32	[21]	√	√	√
r201	4 1252.37	[16]	√ 1276.2	√ 1268.44	7**1173.75**
r202	3 1191.7	[20]	4 **1087.52**	4 **1112.59**	5 **1046.16**
r203	3 942.64	[16]	√ 952.52	√ 989.11	5 **890.50**
r204	2 849.62	[25]	3 **766.92**	3 **760.82**	3 **760.82**
r205	3 994.42	[20]	√ 1036.08	√ 1084.34	5 **954.16**
r206	3 912.97	[21]	√ 921.32	√ 919.73	4 **889.39**
r207	2 914.39	[22]	3 **821.32**	3 **825.07**	4 **822.90**
r208	2 726.823	[8]	3 738.41	√ 773.13	3 **719.17**
r209	3 909.86	[20]	√ 928.93	√ 971.70	5 **874.95**
r210	3 939.37	D[26]	√ 983.77	√ 985.38	5 **930.42**
r211	2 910.09	[16]	3 **786.23**	3 **833.76**	4 **761.10**
rc201	4 1406.94	[25]	√ 1438.43	√ 1423.73	7 **1306.34**
rc202	3 1377.089	[8]	4 **1181.99**	4 **1183.88**	8 **1118.05**
rc203	3 1060.45	[16]	√ 1078.38	√ 1131.78	5 **951.08**
rc204	3 798.46	[8]	√ 810.15	√ 806.44	4 **796.14**
rc205	4 1302.42	[16]	√ 1334.83	√ 1352.39	7 **1181.86**
rc206	3 1153.93	[20]	√ 1203.7	4 1269.64	5 **1080.50**
rc207	3 1062.05	[25]	√ 1093.25	√ 1140.23	5 **982.58**
rc208	3 829.69	[20]	√ 912.76	√ 881.20	4 **785.93**

objective GA (given by *pGA* in Table 4) are equally competitive with other published work. Also in Table 5, it is shown that our multi-objective GA gives the best distances in almost all instances (see bold figures in Table 5) when considering costs. The corresponding average number of vehicles for the respective distances of the multi-objective GA is given in brackets under column *pGA cost* in Table 5. The corresponding average number of vehicles for the other published distances in Table 5 can be inferred from Table 4.

5. Concluding Remarks

This paper presented a multi-objective genetic algorithm approach to the vehicle routing problem with time windows. The solution quality of our GA is competitive with the best solutions reported for the VRPTW by other researchers. However, the most significant contribution of this paper is our interpretation of the VRP as a MOP. Our simple translation of the VRPTW into a MOP was surprisingly effective. Firstly, its performance was very good. Our results are competitive with other vehicle-biased results in the literature. Secondly, the Pareto scoring procedure precludes the need to experiment with weights as required in a weighted-sum approach. Poorly chosen weights result in unsatisfactory solutions, and only after considerably experimentation can effective weights be obtained for a specific instance of a VRPTW.

Perhaps most significantly, our MOP interpretation of the VRPTW represents a philosophically different view of the problem as the whole. When the VRPTW is viewed without bias towards number of vehicles or total cost, we are afforded with a more natural multi-objective perspective for this application problem. No unnecessary bias is introduced into the search. This is in stark contrast to most other work in VRP's, in which the number of vehicles is given implicit priority, and consequently the scoring procedure must prioritize this dimension of the problem. We claim that there is no theoretical nor practical advantage to giving priority to the number of vehicles, perhaps other than having a common framework from which to compare different researcher's results. Admittedly, there is an associated cost to having

Table 3. Solomon Benchmarks: average performance of our GA.

Instance	wGA	pGA	Instance	wGA	pGA
C101	√	√	C201	√	√
C102	√ 844.90	√	C202	√	√
C103	√ 835.20	√ 830.74	C203	√ 592.98	√
C104	√ 830.26	√ 827.84	C204	√ 601.79	√ 599.39
C105	√ 841.57	√	C205	√	√
C106	√ 837.78	√	C206	√	√
C107	√ 858.87	√	C207	√	√
C108	√ 854.53	√	C208	√	√
C109	√ 860.99	√	R201	√ 1313.23	5.2 1211.76
R101	19.6 1693.23	19.7 1690.32	R202	√ 1114.77	5.0 1071.85
R102	18.4 1534.15	18.2 1525.46	R203	√ 974.51	4.2 935.33
R103	13.8 1281.32	14.7 1258.65	R204	3 777.37	3.3 775.63
R104	√ 1035.10	10.9 1038.24	R205	√ 1070.66	4.4 1005.00
R105	15 1452.62	14.9 1430.86	R206	√ 949.25	3.7 913.34
R106	12.8 1298.27	13.5 1283.99	R207	3 848.30	3.4 842.36
R107	11 1115.87	11.1 1106.57	R208	3 747.98	2.9 752.66
R108	10 990.39	10.5 990.20	R209	√ 955.46	4.0 913.67
R109	12.5 1244.87	12.9 1228.71	R210	√ 999.02	4.5 952.58
R110	11.9 1146.11	11.95 1136.38	R211	3 823.34	3.9 791.52
R111	11 1132.51	11.6 1094.29	RC201	√ 1492.67	5.8 1362.34
R112	10.3 1022.51	10.5 1008.50	RC202	√ 1212.49	5.5 1150.80
RC101	15.5 1686.72	15.4 1668.52	RC203	√ 1152.64	4.4 985.90
RC102	13.8 1536.04	14.4 1512.34	RC204	√ 826.19	3.2 819.88
RC103	12 1350.15	12.1 1329.85	RC205	√ 1378.44	6.1 1264.95
RC104	10.4 1184.29	10.8 1188.13	RC206	3.3 1164.33	4.4 1128.15
RC105	15 1618.63	15.8 1612.76	RC207	3.7 1052.13	4.6 1066.68
RC106	12.8 1450.30	12.95 1427.89	RC208	3 938.24	3.7 861.79

Table 4. Comparison of average number of routes on the Solomon Benchmarks with other GA based published results.

Set	[18] (95)	[13] (96)	[31] (99)	[14] (99)	[12] (01)	[17] (01)	[32] (01)	[12] (01)	[10] (03)	wGA	pGA (V)
C1	**10.0**	**10.0**	**10.0**	**10.0**	**10.0**	10.1	10.1	**10.0**	**10.0**	**10.0**	**10.0**
C2	**3.0**	**3.0**	**3.0**	**3.0**	**3.0**	3.3	3.3	**3.0**	**3.0**	**3.0**	**3.0**
R1	12.8	12.6	**12.3**	12.6	12.6	13.2	14.4	12.6	12.8	12.7	12.7
R2	3.2	**3.0**	**3.0**	3.1	3.2	5.0	5.6	3.2	**3.0**	3.2	3.1
RC1	12.5	12.1	**12.0**	12.1	12.8	13.5	14.6	12.8	13.0	12.3	12.5
RC2	**3.4**	**3.4**	**3.4**	**3.4**	3.8	5.0	7.0	3.8	3.7	**3.4**	3.5

more vehicles, and the associated manpower to drive them. However, there is also an associated cost to the additional fuel and time used in using fewer vehicles at longer distances to service clients. Furthermore, vehicle counts can be less important when vehicle and manpower costs are low if using, for example, bicycle couriers. By considering minimal cost (distance), we reduce energy consumption. Such ecological considerations are arguably of growing concern in a world of greenhouse gases and a depleted ozone layer. In any case, the VRPTW is naturally multi-objective, and neither dimension is fundamentally more important than the other from a theoretical perspective and even from a practical aspect, it is arguably debatable as to whether the optimization search should be biased towards minimizing the number of vehicles deployed as most current research work on VRPTW tends to do. Hence, as can be seen with our results, the MOP approach generates a set of equally valid VRPTW solutions. These solutions represent a range of possible answers, with different numbers of vehicles and costs. We leave it to the user to decide which kind of solution is preferable.

Table 5. Relative average cost on the Solomon Benchmarks with other GA based published results.

Set	[18] (95)	[13] (96)	[31] (99)	[14] (99)	[15] (01)	[17] (01)	[32] (01)	[12] (01)	[10] (03)	pGA cost
C1	892.11	838.11	830.89	857.64	867.36	861	860.62	833.32	828.9	**828.48** (10.0)
C2	749.13	590.00	640.86	624.31	625.40	619	623.47	593.00	**589.9**	590.60 (3.0)
R1	1300.25	1296.83	1227.42	1272.34	1369.97	1227	1314.79	**1203.32**	1242.7	1204.48 (13.1)
R2	1124.28	1117.64	1005.00	1053.65	1193.6	980	1093.37	951.17	1016.4	**893.03** (4.5)
RC1	1474.13	1446.25	1391.13	1417.05	1577.64	1427	1512.94	1382.06	1412.0	**1370.79** (13)
RC2	1411.13	1368.13	1173.38	1256.80	1377.86	1123	1282.47	1132.79	1201.2	**1025.31** (5.6)

Acknowledgment

This research is supported by the Natural Sciences and Engineering Research Council of Canada Grant No. 249891-02. This support is greatfully acknowledged. The authors wish to thank three anonymous referees and Mario Ventresca for their valuable comments.

References

1. J. Desrosier, Y. Dumas, M.M. Solomon, and F. Soumis, "Time constraint routing and scheduling," in *Handbooks in Operations Research and Management Science*, Vol. 8, *Network Routing*, edited by M.O. Ball, T/L Magnanti, C.L Monma, G.L. Nemhauser (eds.). Elsevier Science Publishers: Amsterdam, pp. 35–139, 1995.
2. J.F. Cordeau, G. Desaulniers, J. Desrosiers, M.M. Solomon, and F. Soumis, "The VRP with time windows," *to appear in the Vehicle Routing Problem*, Chapter 7, edited by P. Toth and D. Vigo, SIAM Monographs on Discrete Mathematics and Applications, 2001.
3. O. Braysy and M. Gendreau, "Vehicle routing problem with time windows, Part 1: Route construction and local search algorithms," SINTEF Applied Mathematics Report, Department of Optimization, Norway, 2001.
4. O. Braysy and M. Gendreau, "Vehicle routing problem with time windows, Part II: Metaheuristics," *SINTEF Applied Mathematics Report, Department of Optimization*, Norway, 2001.
5. M.R. Garey and D.S. Johnson, *Computers and Intractability, A Guide to The Theory of NP-Completeness*, W. H. Freeman and Company, 1979.
6. J.K. Lenstra and A.H.G. Rinnooy Kan, "Complexity of vehicle routing problem with time windows," *Networks*, vol. 11, pp. 221–227, 1981.
7. N. Kohl. "Exact methods for time constrained routing and related scheduling problems," PhD Thesis, Department of Mathematical Modeling, Technical University of Denmark, 1995.
8. L.M. Gambardella, E. Taillard, and G. Agazzi, "MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows," in edited by David Corne, Marco Dorigo, and Fred Glover, New Ideas in Optimization, , McGraw-Hill: London, 1999, pp. 63–76.
9. D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison Wesley, 1989.
10. K.Q. Zhu. "A diversity-controlling adaptive genetic algorithm for the vehicle routing problem with time windows," in *Proceedings of the 15th IEEE International Conference on Tools for Artificial Intelligence (ICTAI 2003)*, 2003, pp. 176–183.
11. B. Ombuki, M. Nakamura, and O. Maeda, "A hybrid search based on genetic algorithms and tabu search for vehicle routing," in *6th IASTED Intl. Conf. On Artificial Intelligence and Soft Computing (ASC 2002)*, edited by A.B. Banff, H Leung, ACTA Press, pp. 176–181, July 2002.
12. H. Wee Kit, J. Chin, and A. Lim, "A hybrid genetic algorithm for the vehicle routing problem," *International Journal on Artificial Intelligence Tools* (to appear).
13. J.Y. Potvin and S. Bengio, "The vehicle routing problem with time windows—Part II: Genetic search," *INFORMS Journal of Computing*, vol. 8, pp. 165–172, 1996.
14. O. Braysy, "A new genetic algorithms for vehicle routing problem with time windows based on hybridization of a genetic algorithm and route construction heuristics," in *Proceedings of the University of Vaasa, Research Papers*, p. 227, 1999.
15. K.Q. Zhu, "A new genetic algorithm for VRPTW," in *IC-AI 2000*, Las Vegas, USA.
16. J. Homberger and H. Gehring. "Two evolutionary meta-heuristics for the vehicle routing problem with time windows," *INFORMS Journal on Computing*, vol. 37, no. 3, pp. 297–318, 1999.
17. K.C. Tan, L.L. Hay, and O. Ke. "A hybrid genetic algorithm for solving vehicle routing problems with time window constraints," *Asia-Pacific Journal of Operational Research*, vol. 18, no. 1, pp. 121–130, 2001.
18. S. Thangiah. "Vehicle routing with time windows using genetic algorithms," in *Applications Handbook of Genetic Algorithms: New Frontiers*, Vol. II, CRC Press, Boca Raton, 1995, pp. 253–277.
19. W.C. Chiang and Russell, "Simulated annealing metaheuristic for the vehicle routing problem with time windows," *Annals of Operations Research*, vol. 63, pp. 3–27, 1996.
20. L.M. Rousseau, M. Gendreau, and G. Peasant. "Using constraint-based operators to solve the vehicle routing problem with time windows," *Journal of Heuristics* (forthcoming).
21. Y. Rochat and E.D. Taillard. "Probabilistic diversification and intensification in local search for vehicle routing," *Journal of Heuristics*, vol. 1, pp. 147–167, 1995
22. W.C. Chiang and Russell. "A reactive tabu search metaheuristic for the vehicle routing problem with time windows," *INFORMS Journal on Computing*, vol. 9, pp. 417–430, 1997.
23. E.D. Taillard, P. Badeau, M. Gendreau, F. Gueritin, and J.-Y Potvi, "A tabu search heuristic for the vehicle routing problem with soft time windows," *Transportation Science*, vol. 31, pp. 170–186, 1997.
24. P. Badeau, M. Gendreau, F. Guertin, J.-Y. Potvin, and E. D. Taillard, "A parallel tabu search heuristic for the vehicle routing problem with time windows," *Transportation Research-C*, vol. 5, pp. 109–122, 1997.
25. J.F. Cordeau, G. Larporte, and A. Mercier, "A unified tabu search heuristic for vehicle routing problems with time windows," *Journal of the Operational Research Society*, vol. 52, pp. 928–936, 2001.
26. B. De Backer, V. Furnon, P. Shaw, P. Kilby, and P. Prosser, "Solving vehicle routing problems using constraint programming and metaheuristics," *Journal of Heuristics*, vol. 6, pp. 501–523, 2000.
27. P. Shaw, "Using constraint programming and local search methods to solve vehicle routing problems," in *Proceedings of the Fourth International Conference on Principles and Practice of Constraint Programming (CP'98)*, edited by M. Maher and J.-F. Puget, Springer-Verlag, pp. 417–431, 1998.
28. H. Gehring and J. Homberger, "Parallelization of a two-phased metaheuristic for routing problems with time windows," *Asia- Pacific Journal of Operational Research*, vol. 18, pp. 35–47, 2001.
29. J.Y. Potvin and J.M. Rousseau, "A parallel route building algorithm for the vehicle routing and scheduling problem with time windows," *European Journal of Operational Research*, vol. 66, pp. 331–340, 1993.

30. W.C. Chiang and R. Russell, "Hybrid heuristics for the vehicle routing problem with time windows," *Transportation Science*, vol. 29, no. 2, 1995.
31. S.R. Thangiah, "A hybrid genetic algorithms, simulated annealing and tabu search heuristic for vehicle routing problems with time windows," in *Practical Handbook of Genetic Algorithms, Volume III: Complex Structures*, edited by L. Chambers, CRC Press, pp. 347–381, 1999.
32. K. C. Tan, L. H. Lee, Q. L. Zhu, and K. Ou. "Heuristic methods for vehicle routing problem with time windows," *Artificial Intelligent in Engineering*, pp. 281–295, 2001.
33. M. Dorigo and L. M. Gambardella, "Ant Colony System: A cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 53–66, 1997.
34. C.A. Coello Coello, D.A. Van Veldhuizen, and G.B. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Kluwer Academic Publishers, 2002.
35. C. M. Fonseca and P. J. Fleming. "An overview of evolutionary algorithms in multiobjective optimization," *Evolutionary Computation*, vol. 3, no. 1, pp. 1–16, 1995.
36. D. A. Van Veldhuizen and G. B. Lamont, "Multiobjective evolutionary algorithms: Analyzing the state-of-the-art," *Evolutionary Computation*, vol. 8, no. 2, pp. 125–147, 2000.
37. K. Deb, *Multi-objective Optimization using Evolutionary Algorithms*, John Wiley, 2001.
38. E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.
39. J. Knowles and D. Corne, "The Pareto archived evolution strategy: A new baseline algorithm for Pareto multiobjective optimisation: NSGA-II," in *Proceedings CEC*, 1999, pp. 98–105.
40. K. Deb, S. Agrawa, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," in *Proceedings PPSN VI*, Springer-Verlag, 2000, pp. 849–858.
41. B. J. Ross and H. Zhu, "Procedural texture evolution using multi-objective optimization," *New Generation Computing*, vol. 22, no. 3, pp. 271–293, 2004.
42. Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs* (3rd ed.), Springer-Verlag: New York, 1998.
43. http://w.cba.neu.edu/ msolomon/problems.htm.
44. M.M. Solomon. "Algorithms for the vehicle routing and scheduling problems with time window constraints," *Operations Research*, vol. 35, no. 2, pp. 254–265, 1987.