Genetic Algorithms for the Vehicle Routing Problem with Time Windows

Olli Bräysy
Dept. of Mathematics and Statistics, University of Vaasa
P.O. Box 700, FIN-65101 Vaasa
FINLAND

Abstract

A typical vehicle routing problem can be described as the problem of designing least cost routes from one depot to a set of geographically scattered points (cities, stores, warehouses, schools, customers etc). The routes must be designed in such a way that each point is visited only once by exactly one vehicle, all routes start and end at the depot, and the total demands of all points on one particular route must not exceed the capacity of the vehicle. The vehicle routing problem with time windows is a generalization of the standard vehicle routing problem involving the added complexity that every customer should be served within a given time window. In this paper we review shortly the developed genetic algorithm based approaches for solving the vehicle routing problem with time windows and compare their performance with the best recent metaheuristic algorithms. The findings indicate that the results obtained with pure genetic algorithms are not competitive with the best published results, though the differences are not overwhelming.

1 Introduction

Vehicle Routing Problems (VRP) are all around us in the sense that many consumer products such as soft drinks, beer, bread, snack foods, gasoline and pharmaceuticals are delivered to retail outlets by a fleet of trucks whose operation fits the vehicle routing model. In practice, the VRP has been recognized as one of the great success stories of operations research and it has been studied widely since the late fifties. Public services can also take advantage of these systems in order to improve their logistics chain. Garbage collection, or town cleaning, takes an ever increasing part of the budget of local authorities.

A typical vehicle routing problem can be described as the problem of designing least cost routes from one depot to a set of geographically scattered points (cities, stores, warehouses, schools, customers etc). The routes must be designed in such a way that each point is visited only once by exactly one vehicle, all routes start and end at the depot, and the total demands of all points on one route must not exceed the capacity of the vehicle.

The Vehicle Routing Problem with Time Windows (VRPTW) is a generalization of the VRP involving the added complexity that every customer should be served within a given time window. Additional complexities encountered in the VRPTW are length of route constraint arising from depot time windows and cost of waiting time, which is incurred when a vehicle arrives too early at a customer location. Specific examples of problems with time windows include bank deliveries, postal deliveries, industrial refuse collection, school-bus routing and situations where the customer must provide access, verification, or payment upon delivery of the product or service [Solomon and Desrosiers, 1988].

Besides being one of the most important problems of operations research in practical terms, the vehicle routing problem is also one of the most difficult problems to solve. It is quite close to one of the most famous combinatorial optimization problems, the Traveling Salesperson Problem (TSP), where only one person has to visit all the customers. The TSP is an NP-hard problem. It is believed that one may never find a computational technique that will guarantee optimal solutions to larger instances for such problems. The vehicle routing problem is even more complicated. Even for small fleet sizes and a moderate number of transportation requests, the planning task is highly complex. Hence, it is not surprising that human planners soon get overwhelmed, and must turn to simple, local rules for vehicle routing. Next we will describe basic principles of genetic algorithms and some applications for vehicle routing problem with time windows.

2 General principles of genetic algorithms

The Genetic Algorithm (GA) is an adaptive heuristic search method based on population genetics. The basic concepts are developed by [Holland, 1975], while the practicality of using the GA to solve complex problems is demonstrated in [De Jong, 1975] and [Goldberg, 1989]. References and details about genetic algorithms can also be found for example in [Alander, 2000] and [Mühlenbein, 1997] respectively.

The creation of a new generation of individuals involves primarily four major steps or phases: representation, selection, recombination and mutation. The representation of the solution space consists of encoding significant features of a solution as a chromosome, defining an individual member of a population. Typically pictured by a bit string, a chromosome is made up of a sequence of genes, which capture the basic characteristics of a solution.

The recombination or reproduction process makes use of genes of selected parents to produce offspring that will form the next generation. It combines characteristics of chromosomes to potentially create offspring with better fitness. As for mutation, it consists of randomly modifying gene(s) of a single individual at a time to further explore the solution space and ensure, or preserve, genetic diversity. The occurrence of mutation is generally associated with low probability. A new generation is created by repeating the selection, reproduction and mutation processes until all chromosomes in the new population replace those from the old one. A proper balance between genetic quality and diversity is therefore required within the population in order to support efficient search.

Although theoretical results that characterize the behavior of the GA have been obtained for bit-string chromosomes, not all problems lend themselves easily to this representation. This is the case, in particular, for sequencing problems, like vehicle routing problem, where an integer representation is more often appropriate. We are aware of only one approach by [Thangiah, 1995] that uses bit string representation in vehicle routing context. In all other approaches for vehicle routing problem with time windows the encoding issue is disregarded.

Next we describe the basic principles of the genetic algorithms used to solve vehicle routing problem with time windows. One must note that in addition to algorithms discussed below, for example [Thangiah *et al*., 1994] use genetic algorithm to create initial solutions for the hybrid consisting of Simulated Annealing, Tabu Search and well-known λ-exchange route improvement procedure by [Osman, 1993]. Details about Tabu Search and Simulated Annealing can be found for example in [Hertz *et al*., 1997], and [Aarts *et al*., 1997] respectively.

3 Applications for vehicle routing problem with time windows

[Thangiah, 1995] describes a method called GIDEON that assigns customers to vehicles by partitioning the customers into sectors by genetic algorithm and customers within each formed sector are routed using the cheapest insertion method of [Golden and Stewart, 1985]. In the next step the routes are improved using λ-exchanges introduced by [Osman, 1993]. The two processes are run iteratively a finite number of times to improve the solution quality. The search begins by clustering customers either according to their polar coordinate angle or randomly. The proposed search strategy accepts also infeasibilities during the search against certain penalty factors. In the GIDEON system each chromosome represents a set of possible clustering schemes and the fitness values are based on corresponding routing costs. The crossover operator exchanges a randomly selected portion of the bit string between the chromosomes and mutation is used with a very low probability to randomly change the bit values.

[Potvin and Bengio, 1996] propose a genetic algorithm GENEROUS that directly applies genetic operators to solutions, thus avoiding the coding issues. The initial population is created with cheapest insertion heuristic of [Solomon, 1987] and the fitness values of the proposed approach are based on the number of vehicles and total route time. The selection process is stochastic and biased toward the best solutions. For this purpose a linear ranking scheme is used. During the recombination phase, two parent solutions are merged into a single one, so as to guarantee the feasibility of the new solution. Two types of crossover operators are used to modify a randomly selected route or to insert a route into the other parent solution. A special repair operator is then applied to the offspring to generate a new feasible solution. Mutation operators are aimed at reducing the number of routes. Finally, in order to locally optimize the solution, mutation operator based on Or-opt exchanges [Or, 1976] is included.

[Berger *et al*., 1998] propose a method based on the hybridization of a genetic algorithm with well-known construction heuristics. The authors omit the coding issues and represent a solution by a set of feasible routes. The initial population is created with nearest neighbor heuristic inspired from [Solomon, 1987]. The fitness values of the individuals are based on the number of routes and total distance of the corresponding solution and for selection purposes the authors use the so called roulette-wheel scheme. In this scheme the probability of selecting an individual is proportional to its fitness; for details see [Goldberg, 1989]. The proposed crossover operator combines iteratively various routes r_1 of parent solution P_1 with a subset of customers, formed by r_2 nearest-neighbor routes from parent solution P_2. A removal procedure is first carried out to remove some key customer nodes from r_1. Then an insertion heuristic inspired from [Solomon, 1987] coupled to a random customer acceptance procedure is locally applied to build a feasible route, considering the partial route r_1 as an initial solution. The mutation operators are aimed at reducing the number of routes of solutions having only a few customers and locally reordering the routes.

[Bräysy, 1999a] and [Bräysy, 1999b] extended the work of [Berger *et al*., 1998] by proposing several new crossover and mutation operators, testing different forms of genetic algorithms, selection schemes, scaling schemes and the significance of the initial solutions. When it comes to recombination, an approach where customers within randomly generated segments in parent solution P_1 are replaced with some other customers on the near routes of parent solution P_2 is found to perform best. The best-performing mutation operator selects randomly one of the shortest routes and tries to eliminate it by inserting the customers into other longer routes. Regarding different forms of genetic algorithms it is concluded that it is important to create many new offspring each generation and it is enough to maintain only one population. For selection purposes so-called tournament selection is found to perform best. In the first phase two individuals are selected with a random procedure that is biased towards better fitness scores. In the second phase, the individual with better fitness is selected. However the differences between different schemes were minor. A new scaling scheme based on a weighted combination of number of routes, total distance and waiting time is found to perform particularly well. Finally to create the initial population, several strategies, such us heuristics of [Solomon, 1987] and randomly created routes were tried and it was concluded that the best strategy is to create a diverse initial population that also contains some individuals with better fitness scores.

[Homberger and Gehring, 1999] propose two evolutionary metaheuristics for VRPTW. The proposed algorithms are based on the class of evolutionary algorithms called Evolution Strategies. Differences to GA exist with regard to the superior role of mutation compared to the recombination operators. Here the individual representation also includes a vector of so-called strategy parameters in addition to the solution vector and both components are evolved by means of recombination and mutation operators. In the proposed application for VRPTW these strategy parameters refer to how often a randomly selected local search operator is applied and to binary parameter used to alternate the search between minimizing the number of vehicles and total distance.

Selection of the parents is done randomly and only one offspring is created through the recombination of parents. This way a number $\lambda > \mu$ offspring is created, where μ is the population size. At the end fitness values are used to select μ offspring to the next population. Because the parents are not involved in the selection process, deteriorations during the search are permitted. The first out of the two proposed metaheuristics, evolution strategy ES1, skips the recombination phase. The second evolution strategy ES2 uses uniform order-based crossover to modify the initially randomly created mutation codes of the two parents and tries to improve the solution vector of a third randomly selected parent using the modified code. The mutation code is used to control a set of removal and insertion operators performed by Or-opt operator [Or, 1976]. The fitness values are based on number of routes, total travel distance and on a criterion that determines how easily the shortest route of the solution in terms of the number of customers on the route can be eliminated. The individuals of a starting population are generated by means of a stochastic approach, which is based on the savings algorithm of [Clarke and Wright, 1964].

[Bräysy *et al*., 2000] describe a two-phase hybrid evolutionary algorithm based on the hybridization of a genetic algorithm and an evolutionary algorithm consisting of several local search and route construction heuristics inspired from the studies of [Solomon, 1987] and [Taillard *et al*., 1997]. In the first phase a genetic algorithm based on the studies [Berger et al., 1998] and [Bräysy, 1999a] is used to obtain a feasible solution. The evolutionary algorithm used in the second phase picks every pair of routes in random order and applies randomly one out of the four local search operators or route construction heuristics. Finally, offspring routes generated by these crossover operators are mutated according to a user-defined probability by selecting randomly one out of two operators. Selecting each possible pair of routes, mating and mutation operators are repeatedly applied for a certain number of generations and finally a feasible solution is returned. To escape from local minimum, arcs longer than average are penalized, if they appear frequently during the search.

4 Comparison of the experimental results

In this section we compare the results obtained with the genetic and evolutionary approaches described above with the results of Ant Colony Optimization algorithm by [Gambardella *et al*., 1999] and Tabu Search algorithm by [Cordeau *et al*., 2000]. For comparison purposes we used the most well-known 100-customer benchmark problem set by [Solomon, 1987]. In these problems, the travel times are equal to the corresponding Euclidean distances. The geographical data were either randomly generated using a uniform distribution (problem sets R1 and R2), clustered (problem sets C1 and C2) or mixed with both randomly distributed and clustered customers (problem sets RC1 and RC2). Problem sets R1, C1 and RC1 have a narrow scheduling horizon. Hence only a few customers can be served by the same vehicle. Conversely, problem sets R2, C2 and RC2 have a large scheduling horizon, and more customers can be served by the same vehicle. The results are depicted in Table 1.

All algorithms in Table 1 are stochastic and they are implemented using C-language. A hierarchic objective function is used in every case. The number of routes is considered as the primary objective and for the same number of routes, the secondary objective is to minimize the total traveled distance. An exception is found in PB, where the second objective is to minimize the total duration of routes.

As can be seen from Table 1, the best results are always produced by HG1, GTA or CLM. HG1 performs best in problem group R1, GTA gives best output for problem groups R2 and RC2 and finally CLM seems to outperform the other methods in problem group RC1. In clustered problem groups C1 and C2 the methods perform equally well. However, if computational burden is considered, HG1 seems to be most efficient approach. This is especially due to the fact that the results of GTA and CLM are the best ones obtained during the whole computational study, thus requiring clearly more computational resources. Moreover, HG1 and HG2 are the best two methods regarding cumulative number of vehicles. Since the primary minimization objective in all methods is the number of vehicles, the CNV can be considered as a good measure of solution quality and robustness. The differences between HG1 and HG2 are quite small, except for problem group R2, for which HG1 is about 6% better in total distance. In addition HG2 seems to be clearly more time consuming.

When it comes to other approaches, the clearly worst results are produced by T. Only T, PB, BSB and B can be considered as pure genetic algorithms. The best performing approach within the pure genetic algorithms seems to be B, though the differences are often insignificant. It seems that pure genetic algorithms are not competitive with the other approaches, such as Tabu Search or Evolutionary Algorithms. Generally the difference in number of vehicles is about 6% and if T is not considered the difference is approx. 4%. Problem group RC2 seems to be the most problematic regarding the total distance. The differences between T and HG1 and B and HG1 are about 23% and 10% respectively which can hardly be justified in practical settings. Considering the information available, we conclude that the HG1 is the best method.

PROB.	T	PB	BSB	B	HG 1	HG 2	BBB	GTA	CLM
R1	**12.75**	**12.58**	**12.58**	**12.58**	**11.92**	**12.00**	**12.42**	**12.00**	**12.08**
	1300.25	1296.83	1261.58	1272.34	1228.06	1226.38	1213.86	1217.73	1210.14
R2	**3.18**	**3.00**	**3.09**	**3.09**	**2.73**	**2.73**	**3.09**	**2.73**	**2.73**
	1124.28	1117.64	1030.01	1053.65	969.95	1033.58	978.00	967.75	969.57
C1	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**	**10.00**
	892.11	838.11	834.61	857.64	828.38	828.38	828.75	828.38	828.38
C2	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**
	749.13	590.00	594.25	624.31	589.86	589.86	591.81	589.86	589.86
RC1	**12.50**	**12.13**	**12.13**	**12.13**	**11.63**	**11.50**	**12.13**	**11.63**	**11.50**
	1474.13	1446.25	1441.35	1417.05	1392.57	1406.58	1372.20	1382.42	1389.78
RC2	**3.38**	**3.38**	**3.50**	**3.38**	**3.25**	**3.25**	**3.38**	**3.25**	**3.25**
	1411.13	1368.13	1284.25	1256.80	1144.43	1175.98	1170.23	1129.19	1134.52
CNV	**429**	**422**	**424**	**423**	**406**	**406**	**421**	**407**	**407**

Table 1. Comparison of evolutionary and genetic algorithms with the best known approaches. The notations in the leftmost column stand for the 6 problem groups introduced by [Solomon, 1987] and CNV is the cumulative number of vehicles required to serve the customers in all 56 test problems. For each method the average number of vehicles and total traveled distance with respect to corresponding problem group are given. One must note that the results are the best ones obtained with the described algorithms.

The following list provides information concerning author, computer used, number of runs and average time consumption in minutes for a single run. The reader must note that we rounded the time consumption to nearest minute.

T: [Thangiah, 1995], Solbourne 5/802, number of runs not reported, 2 minutes.
PB: [Potvin and Bengio, 1996], Sun Sparc 10, number of runs not reported, 25 minutes.
BSB: [Berger et al., 1998], Sun Sparc 10, number of runs not reported, 1−10 minutes.
B: [Bräysy, 1999b], Sun Ultra Enterprise 450 (300 MHz), 5 runs, 17 minutes.
HG1: [Homberger and Gehring, 1999], Pentium 200 MHz, 10 runs, 13 minutes.
HG2: [Homberger and Gehring, 1999], Pentium 200 MHz, 10 runs, 19 minutes.
BBB: [Bräysy et al., 2000], Pentium Celeron 366 MHz, 5 runs, 15 minutes.
GTA: [Gambardella et al., 1999], number of runs and time consumption not reported.
CLM: [Cordeau et al., 2000], number of runs and time consumption not reported.

5 Conclusions

The purpose of this study was to review well-known genetic algorithms for solving the vehicle routing problem with time windows. In addition three evolutionary algorithms were described. The basic principles of the algorithms were depicted and the best results obtained with these algorithms were compared with the best results obtained with the best other metaheuristic approaches. The first of the two major conclusions was that the evolutionary algorithm by [Homberger and Gehring, 1999] is currently the most efficient approach for solving the vehicle routing problem with time windows, producing to each tested problem type good average solution quality using a reasonable amount of time. The other conclusion was that the results obtained with pure genetic algorithms are not competitive with the best published results, though the differences are not overwhelming.

References

[Aarts *et al.*, 1997] Emile Aarts, Jan H. M. Korst and Peter J. M. van Laarhaven. Simulated annealing. In *Local Search in Combinatorial Optimization*, pages 91--120, John Wiley & Sons, Chichester, 1997.

[Alander, 2000] Jarmo T. Alander. An indexed bibliography of genetic algorithms in operations research. Technical Report series No. 94-1-OR, University of Vaasa, Vaasa, Finland, 2000. Available via anonymous `ftp:` site `ftp.uwasa.fi` directory `cs/report94-1` file `gaORbib.ps.Z.`

[Berger *et al.*, 1998] Jean Berger, Martin Salois and Regent Begin. A hybrid genetic algorithm for the vehicle routing problem with time windows. In *Proceedings of the 12th Bienneal Conference of the Canadian Society for Computational Studies of Intelligence*, pages 114—127, Springer-Verlag, Berlin 1998.

[Bräysy, 1999a] Olli Bräysy. A hybrid genetic algorithm for the vehicle routing problem with time windows, Unpublished, Vaasa University Library, Vaasa, Finland, 1999.

[Bräysy, 1999b] Olli Bräysy. A new algorithm for the vehicle routing problem with time windows based on the hybridization of a genetic algorithm and route construction heuristics. *Proceedings of the University of Vaasa*, *Research papers* 227, Vaasa, Finland, 1999.

[Bräysy *et al.*, 2000] Olli Bräysy, Jean Berger and Mohamed Barkaoui. A new hybrid evolutionary algorithm for the vehicle routing problem with time windows. Presented at the Route 2000-Workshop, Skodsborg, Denmark, August 2000.

[Clarke and Wright, 1964] G. Clarke and J. W. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12:568--581, 1964.

[Cordeau *et al.*, 2000] Jean-Francois Cordeau, Gilbert Laporte and Alain Mercier. Unified tabu search heuristic for vehicle routing problems with time windows. Publication CRT-2000-03, University of Montreal, Montreal, Canada, 2000.

[De Jong, 1975] Kenneth A. De Jong. An analysis of the behavior of a class of genetic adaptive systems. Ph.D. Dissertation, University of Michigan, U.S.A, 1975.

[Gambardella et al., 1999] Luca M. Gambardella, Eric Taillard and Giovanni Agazzi. MACS-VRPTW: a multiple ant colony system for vehicle routing problems with time windows. In *New Ideas in Optimization*, pages 63--76, McGraw-Hill, London, 1999.

[Goldberg, 1989] David Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley Publishing Company Inc, New York, 1989.

[Golden and Stewart, 1985] Bruce L. Golden and William R. Stewart. Empirical analysis of heuristics. In *The Traveling Salesman Problem*, pages 207--249, John Wiley & Sons, Chichester, 1985.

[Hertz *et al.*, 1997] Alain Hertz, Eric Taillard and Dominique de Werra. Tabu search. In *Local Search in Combinatorial Optimization*, pages 121--136, John Wiley & Sons, Chichester, 1997.

[Holland, 1975] John H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, 1975.

[Homberger and Gehring, 1999] Jörg Homberger and Hermann Gehring. Two evolutionary meta-heuristics for the vehicle routing problem with time windows. *INFORMS Journal on Computing*, 37(3): 297--318, 1999.

[Mühlenbein, 1997] Heinz Mühlenbein. Genetic algorithms. In *Local Search in Combinatorial Optimization*, pages 137--172, John Wiley & Sons, Chichester, 1997.

[Or, 1976] I. Or. Traveling salesman-type combinatorial problems and their relation to the logistics of regional blood banking. Ph.D. Thesis, Northwestern University, Evanston, Illinois, 1976.

[Osman, 1993] Ibrahim H. Osman. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problems. *Annals of Operations Research*, 41: 421--452, 1993.

[Potvin and Bengio, 1996] Jean-Yves Potvin and Samy Bengio. The vehicle routing problem with time windows part II: genetic search. *Journal on Computing*, 8(2): 165--172, 1996.

[Solomon, 1987] Marius M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2): 254--265, 1987.

[Solomon and Desrosiers, 1988] Marius M. Solomon, and Jacques Desrosiers. Time window constrained routing and scheduling problems. *Transportation Science*, 22(1): 1--13, 1988.

[Taillard *et al*., 1997] Eric Taillard, Philippe Badeau, Michel Gendreau, Francois Guertin and Jean-Yves Potvin. A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation Science*, 31(2): 170--186, 1997.

[Thangiah *et al*., 1994] Sam Thangiah, Ibrahim Osman and Tong Sun. Hybrid genetic algorithm, simulated annealing and tabu search methods for vehicle routing problems with time windows. Working Paper UKC/IMS/OR94/4, University of Kent, Canterbury, 1994.

[Thangiah, 1995] Sam Thangiah. Vehicle routing with time windows using genetic algorithms. In *Application Handbook of Genetic Algorithms*: *New Frontiers*, *Volume II*, pages 253−277, CRC Press, Boca Raton, 1995.