International Journal of Computer & Information Technologies (IJOCIT)
www.ijocit.ir & www.ijocit.org
ISSN = 2345-3877

A Survey on Genetic Algorithms for the Vehicle Routing Problem

Ayda Khoramrouz
Department of Software, Ardabil Branch, Engineering, Islamic Azad University, Ardabil, Iran
Email: Ayda_khoramrouz@yahoo.com

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 met heuristic 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.

Keywords: Vehicle Routing, Warehouses, Genetic Algorithm, Time Windows.

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)[2]. 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 Defrosters, 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.[4] 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 [5].

2. Vehicle Routing Problem

Vehicle routing problem (VRP) is a general name given for a class of problems, in which a set of vehicles service a set of customers. This statement was first defined by [2,3]. VRP is a generalization of a traveling salesman problem (TSP), where only one traveler is taken into account. The TSP is defined as a set of cities, where a single traveler needs to visit all of them and return to the starting city. The objective of the TSP is to find the shortest route. The vehicle routing problem typically is described as a graph $G = (N, E)$ and a set of homogeneous vehicles V = {v1… vt}, where t is the number of vehicles.

The graph G consists of the nodes N = {n0, n1, ..., nk}, where n0 is a depot and $N \backslash \{n_0\}$ are k customers that need to be serviced, and edges $E = \{e_{ij}\}$, where $i \neq j, ۰ \leq i \leq k, ۰ \leq j \leq k, e_{ij} = (n_i, n_j)$

Each vehicle that services customers starts the travel from the depot and finishes it in the depot as well. The objective of the typical VRP is to find the solution, at first, minimizing the total vehicle number required, and secondly, minimizing the length of the total traveled path [7, 8]. For the set E, the cost matrix D is defined, where dij is the cost of the edge eij=(ni, nj), and dii = 0. Usually the VRP is treated as symmetric, where dij = dji. In the real world problem, the cost matrix is asymmetric and needs to be calculated from geographic data by using the shortest path algorithms. Moreover, if a vehicle set is not homogeneous, some roads can be forbidden for certain vehicles and allowed for others. The different shortest path can exist for a different vehicle type, so a different matrix needs to be calculated for all the different vehicle types.

3. General Principle of Genetic Algorithms

The Genetic Algorithm (GA) is an adaptive heuristic search method based on population genetics. The basic concepts are developed by [8], while the practicality of using the GA to solve complex problems is demonstrated in [9]. 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 [7] 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 [9] 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 [10].

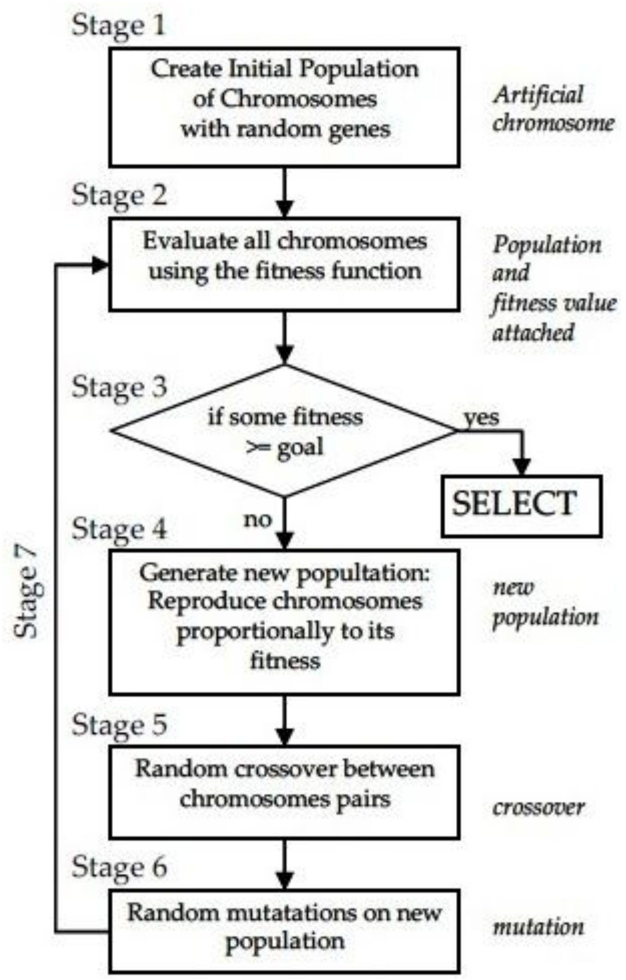
Stage 1
Create Initial Population of Chromosomes with random genes
Artificial chromosome
Stage 2
Evaluate all chromosomes using the fitness function
Population and fitness value attached
Stage 3
if some fitness >= goal
yes
SELECT
no
Stage 4
Generate new popultation: Reproduce chromosomes proportionally to its fitness
new population
Stage 5
Random crossover between chromosomes pairs
crossover
Stage 6
Random mutatations on new population
mutation
Stage 7

Figure 1. Schematic representation of the GA procedure

As already mentioned, VRP is a generalization of the TSP problem. VRP includes additional components, i.e. fleet of vehicles, and additional constraints. An additional component of the problem can affect computation and even require to design the problem specific genetic operators. Genetic algorithm approaches to solve the VRP can be categorized according to the following features:

- Representation. Solution in GA can be encoded as a chromosome (expressed as a literal string), or unencoded, where encoding of the solution within chromosome is not addressed.

- Feasibility handling. Genetic algorithm operators can be designed to preserve the feasibility of individuals within a population or allow the generation of infeasible individuals.

An example of VRP solution, where 3 routes are used to service customers expressed as a chromosome is as follows (Berger et al., 1998), where $\forall n_e$ belongs to one route $\forall n_f$ belongs to the second route and $\forall n_g$ belongs to the third route:

$$|n_{e1}, n_{e2} \dots | n_{f1}, n_{f2} \dots | n_{g1}, n_{g2} \dots |$$

The standard genetic operators can be applied to such a chromosome, however, such a representation does not hold any problem specific information and, depending on the encoding approach, the selected genetic algorithm can be ineffective. Different approaches for encoding the VRP solution can be found in the literature, i.e. in [10], a chromosome representation based on the angles of vectors starting from a depot node is proposed, where the VRP is treated as a planar graph problem. Researches can be found that compare crossover operators designed to work with the chromosome representation. When dealing with constraints, a stochastic approach to find optimal solutions can compute very long, until an acceptable solution has been found [12]. For a constrained problem, there exist feasible and infeasible search spaces $S_F(\forall x \in S_F$ does not violate any of the defined constraints) and $S_U(\forall x \in S_U$ does violate at least one defined constraint) Let us define the whole search space S, then $S_F \subseteq S, S_U \subseteq S, S_U \cup S_F = S, S_U \cap S_F = \emptyset$ the solution x belongs to the feasible search space $S_F, if\ F_c(x) = 0$ highly constrained problems are those, where the feasible search space is very small. Thus the probability to generate solutions in such a space for crossover and mutation operators can be adequately small [12]. Approaches, where a solution is represented as a chromosome or where solutions are allowed to be generated in the infeasible search space SU, require additional approaches for constraint handling. The following approaches are used to deal with the infeasibility in genetic algorithms:

- Applying penalty function.
- Treating problem as multi-objective.
- Repairing solution.
- Preserving feasibility in the genetic operators.

- **Penalty.** The penalty function p(x) transforms a constrained problem into an unconstrained one [12]. A penalty method is widely used in genetic algorithms for constrained problems. The main target is to add a significant value to the fitness value for the generated offspring's that violate constraints. In [13] the author discusses the advantages and disadvantages of having feasible and infeasible solutions in genetic algorithms and how they influence the results. The discussion is carried out on the issue how the feasible and infeasible solutions can be compared.

- **Multi-objective**. In a multi-objective approach, the constrained problem is transformed into a multi-objective problem. In [13,14], the Pareto ranking method is used to solve the VRPTW expressed as multi-objective, where Pareto ranking, similarly to the penalty approach, is used to adjust the ranking mechanism of the genetic algorithm and assign the relative strength of individuals in the population. The ranking mechanism assigns the smallest rank to non-dominated individuals and the dominated individuals are ranked according to the individuals in the population and the defined criteria. Pareto ranking attempts to assign a single fitness score to the solution of a multi-objective problem. In literature there can be found Pareto ranking in the genetic algorithm treated as equivalent to the penalty approach.
- **Repair**. The second approach for feasibility handling is a repair method. The repair method defines the transition function y = r(x), where y is the repaired version of x, such as $y \in S_F \ and \ x \in S_U$ The repair can be designed in two different ways:
  - An individual is repaired for evaluation only, where $f_u(x) = f_f(y)$ and y is a repaired (i.e. feasible) version of x. It is the so-called Lamarckian approach [14]. The weakness of such an approach is that it depends on the problem and a specific repair algorithm has to be designed.
  - An individual is repaired and the previous individual is replaced by its repaired version. It is called a Baldwin Ian approach [15]. This method has the same limitation as the previous one. The question of replacement is also widely considered. In some researches the fixed percent of the repaired individuals replace the previous one or this can be dependent on the

problem or even on the evolution process.

Preserving feasibility. The author in [12] discusses the possibility of having feasible solutions generated in crossover and mutation operations, where feasibility handling in a two point crossover where a set of crossovers with different boundary indices is considered. Probability function is defined to find a feasible crossover for a linearly constrained optimization problem. However, for a highly constrained problem where a feasible space is very small as compared to the full search space, only a half feasible crossover with a single boundary point is discussed. In order to handle feasibility in the mutation process, the proposed mutation operator is based on the crossover operator, where the selected individual is crossed with a randomly generated individual [16].

4. Genetic Algorithm for Vehicle Routing Problem

A genetic algorithm based on insertion heuristics without considering any additional local search methods for the improvement is proposed in this section. The definition "genetic algorithm" can describe either a general approach or a set of the specific genetic operators. In this thesis the proposed version of genetic algorithm for VRP with constraints will be called "new genetic algorithm" further on to distinguish it from other approaches. Genetic algorithms and insertion heuristics combine together their best characteristics to search for the optimal solution. It is generally accepted that any genetic algorithm for solving a problem should have basic components, such as a genetic representation of solutions, the way to create the initial solution, the evaluation function for ranking solutions, genetic operators, values of the parameters (i.e. population size, probabilities for applying genetic. In the proposed genetic algorithm crossover and mutation operators are defined in the "remove and reinsert" approach. The approach is similar to a single point relocation method, where the node is extracted and inserted into a different place. However, reinsertion of a single node in a different place can be unsuccessful, because the constructed routes have reached constraint limits and cannot be extended by an additional node. If a single node has been chosen for reinsertion, there is a large probability that the node will be inserted in the same place from which it has been removed. In order to enable the node reinsertion, multiple nodes have to be extracted. In the proposed algorithm the mutation operator is applied with probability MP = 0.1 and the crossover operator is applied to all individuals selected for mating. In the crossover operation new offspring's are generated from two parent solutions that are selected from population by using the ranking method. The new offspring's are added to the population and the worst individuals are removed from the population to keep the same population size in each iteration. The defined mutation operators are based on a random insertion and can produce individuals that will not survive. In order to increase the probability of the mutation operator to generate individuals that will survive, a second population is created. The success of the mutation operator depends on the generated solution in comparison to the solutions in the population. If the fitness value of generated solution is better than the average fitness value in the population, such solution will have a higher probability to be selected for reproduction. If the fitness value of generated solutions is similar to the worst fitness value in the population, there is higher probability that the solution will be removed from the population in next generations. There is no benefit if the solution generated in the mutation operator does not participate further in the reproduction.

5. Crossover Operators

In the genetic algorithm new crossover operators that are based on insertion heuristics are proposed. However, apart from the insertion heuristics, the proposed crossover operators handle most of the negative aspects of the reviewed crossover operators in Section 1.5 and include another intensification approach. The effectiveness of LNS depends on the degree of destruction, where, if only a small part is destroyed, LNS can have troubles in exploring the search space, or can be involved in the repeated re-optimization, if a very large space is destroyed (Pisinger and Ropke, 2009). It should be such destruction method which would explore the search space, where the global optimum is expected to be found. Thus, it means that the removed nodes can have a low probability to change their positions in the solution. Other crossovers (SBX, RBX, LRX) convey the union of the solutions, where some parts from both individuals are combined with the intention to find a better solution. Such crossovers explore only a small neighborhood and only in the cases, where additional unassigned nodes are left during the recombination of parents. Differently than the union crossover operators, the proposed crossovers are designed to preserve common parts of the two selected individuals. The common parts could be the nodes assigned to the same route, the nodes assigned to the route starting from the same depot, or the nodes that are related to the same type of cargo, etc. By removing the nodes that do not belong to the common parts of solutions, the common neighborhood of two solutions is identified. A size of the neighborhood is inversely proportional to the size of the common parts. If the initial individuals in a genetic algorithm are created in a stochastic way, by preserving the nodes that have common characteristic in both parents, the nodes will be preserved that more probably are optimally constructed than the other parts of the solution. Most probably, the nodes will be removed that prolong the overall path, where long paths can lead to a larger number of routes. The target of the proposed crossover operators is to identify the common parts in the parent individuals, preserve it in the intermediate solution and reconstruct it in the offspring individual. Three different crossovers (common nodes crossover, common arcs crossover and longest common sequence crossover) are defined to increase the probability of convergence to the global optimum where each crossover produces an offspring by focusing on a different information obtained from parents. Differently than the union crossover operators, the proposed crossovers are designed to preserve common parts of the two selected individuals. The common parts could be the nodes assigned to the same route, the nodes assigned to the route starting from the same depot, or the nodes that are related to the same type of cargo, etc. By removing the nodes that do not belong to the common parts of solutions, the common neighborhood of two solutions is identified. A

size of the neighborhood is inversely proportional to the size of the common parts. If the initial individuals in a genetic algorithm are created in a stochastic way, by preserving the nodes that have common characteristic in both parents, the nodes will be preserved that more probably are optimally constructed than the other parts of the solution. Most probably, the nodes will be removed that prolong the overall path, where long paths can lead to a larger number of routes.

6. Conclusion

In order to keep solutions in the feasible search space, we propose a genetic algorithm that is based on a random insertion heuristics. The random insertion heuristic is considered to preserve a stochastic characteristic of the genetic algorithm, and to generate solutions in the feasible space by checking compliance to the defined constraints in the insertion process. Pre-computation scheme is proposed for speed-up evaluation of constraints in insertion heuristic. Infeasibility is still allowed in the proposed algorithm because the random insertion approach can create infeasible initial solutions in a highly constrained problem. The defined GA individual includes feasible partial routes and a set of customers that were not serviced due to constraint violation. The novelty of the proposed approach is the usage of random insertion heuristics in combination with the proposed crossover and mutation operators. Differently from other genetic algorithms, the proposed crossover operators do not construct the offspring directly, but by evaluating information from previous generation, identify those parts of solutions that should be preserved for the next generation and weak parts that should be reconstructed. The crossover and mutation operators are defined to identify those weak parts of the solution. The second population is used in the mutation process, where the second population increases the probability that the solution, obtained in the mutation process, will survive in the first population, and increase the probability to find the global optimum. In contrast to other approaches, the proposed algorithm does not involve additional local search methods to improve the solution; therefore it does not depend on the local search limitations and can be easily extended with additional constraints.

References:

1. Thangiah, S., Nygard, K., Juell, P. (2011). GIDEON: A genetic algorithm system for vehicle routing with time windows. In 7th Conference on Artificial Intelligence Applications, 322–328.
2. Tommiska, M., Skytta, J. (2011), Dijkstra's Shortest Path Routing Algorithm in Reconfigurable Hardware., in G. J. Brebner., R. Woods, ed., Field-Programmable Logic and Applications, 11th International Conference, Springer, 2147, 653–657.
3. Toth, P., Vigo, D. (2011). Branch-and-bound algorithms for the capacitated VRP. Society for Industrial and Applied Mathematics, 29–51.
4. Toth, P., Vigo, D. (2012). The Vehicle Routing Problem, Society for Industrial and Applied Mathematics.
5. Vatsavai, R. R., Shekhar, S., Burk, T. E. and Lime, S. (2016), UMNMapServer: A High-Performance, Interoperable, and Open Source Web Mapping and Geo-spatial Analysis System., in M. Raubal; H. J. Miller; A. U. Frank and M. F. Goodchild, ed., GIScience, Springer, 400–417.
6. Vidal, T., Crainic, T. G., Gendreau, M., Prins, C. (2013). Heuristics for multiattribute vehicle routing problems: A survey and synthesis. European Journal of Operational Research, 231 (1), 1–21.
7. Von Lossow, M. (2007). A min-max version of Dijkstra's algorithm with application to perturbed optimal control problems. In Proceedings in Applied Mathematics and Mechanics ICIAM 2007/GAMM 2007, 7, Zurich, Switzerland.

8. Yang, I, Huang, C., Chao, K. (2015). A fast algorithm for computing a longest common increasing subsequence. Inf. Process. Lett., 93(5), 249–253.
9. Yeniay, O. (2015). Penalty function methods for constrained optimization with genetic algorithms. Mathematical and Computational Applications, 10(1), 45–56.
10. Yeun, L. C., Ismail, W. R., Omar, K., Zirour, M. (2008). Vehicle Routing Problem: Models and Solutions, Journal of Quality Measurement and Analysis (JQMA), 4(1), 205–218.
11. Zhang, J., Chung, H. S.-H., Hu, B.J. (2014). Adaptive probabilities of crossover and mutation in genetic algorithms based on clustering technique Evolutionary Computation. In Proceedings of IEEE Congress on Evolutionary Computation (CEC2004), 2, 2280–2287.
12. Zhang, J., Chung, H. S.-H., Lo, W.-L. (2007). Clustering-Based Adaptive Crossover and Mutation Probabilities for Genetic Algorithms. IEEE Transactions on Evolutionary Computation, 11(3), 326–335.
13. Zhong, J., Hu, X., Gu, M., Zhang, J. (2005). Comparison of Performance between Different Selection Strategies on Simple Genetic Algorithms. Proceeding of the International Conference on Computational Intelligence for Modelling, Control and automation, and International Conference of Intelligent Agents, Web Technologies and Internet Commerce (CIMCA/IAWTIC), IEEE Computer Society, 1115–1121
14. Zhu, K. Q. (2013). A Diversity-Controlling Adaptive Genetic Algorithm for the Vehicle Routing Problem with Time Windows. Proceedings of the 15th IEEE International Conference on Tools for Artificial Intelligence (ICTAI 2013), 176–183.
15. Žilinskas, A., Žilinskas, J. (2007). Parallel genetic algorithm: assessment of performance in multidimensional scaling. In proceedings of the 9th annual conference on Genetic and evolutionary computation (GECCO 2007), Association for Computing Machinery (ACM), 1492-1499.
16. Žilinskas, J. (2008). On dimensionality of embedding space in multidimensional scaling. Informatica, 19(3), 447-460.
17. Rizzoli, A. E., Montemanni, R., Lucibello, E., Gambardella, L. M. (2007). Ant colony optimization for real-world vehicle routing problems. Swarm Intelligence, 1(2), 135–151.

18. Romeijn, H. E., Smith, R. L. (2009). Parallel Algorithms for Solving Aggregated Shortest Path Problems. Computers & Operations Research, (26), 941–953.
19. Ropke, S., Pisinger, D. (2010). A unified heuristic for a large class of Vehicle Routing Problems with Backhauls. European Journal of Operational Research, 171 (3), 750–775.
20. Rosenkrantz, D. J., Stearns, R. E., Lewis II, P. M. (2007). An Analysis of Several Heuristics for the Traveling Salesman Problem. SIAM J. Comput., 6(3), 563–581.
21. Schensted, C. (2011), Longest increasing and decreasing subsequences. Canad. J. Math. 13, 179–191.
22. Schneider, M., Stenger, A., Goeke, D. (2012). The Electric Vehicle Routing Problem with Time Windows and Recharging Stations. In Tech. Report 02/2012, BISOR, TU Kaiserslautern.
23. Šešok, D. (2008). Topology optimization of truss structures using genetic algorithms. Doctoral dissertation, Vilnius Gediminas Technical University.
24. Solomon, M.M. (1987). Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. Operations Research, 35(2), 254–265.