Polish J. of Environ. Stud. Vol. 17, No. 4C (2008), 91-95

Evolutionary Algorithms with Intelligent Mutation Operator that Solve the Vehicle Routing Problem of Clustered Classification with Time Windows

O. Díaz-Parra, M. A. Cruz-Chávez

CIICAp, Universidad Autónoma del Estado de Morelos
Cuernavaca, Morelos, México

Abstract

This paper presents an evolutionary algorithm that uses the combination of the selection operator *„the best"* and the proposed operators, crossover *„crossover-k"* and intelligent mutation *„mutation-S."* The initial population (feasible individuals) is generated with the *k-means algorithm* of clustering (Data-Mining Techniques). The proposed algorithm solves the Vehicle Routing Problem with Time Windows, problem type C (clustered customers classification) with 100-node problems.

Keywords: genetic algorithm, clustering, mutation, optimization

Introduction

The Vehicle Routing Problem (VRP) is a problem based on the assignment of routes to vehicles in order to attend different clients [1]. A variant of the Vehicle Routing Problem is the Vehicle Routing Problem with Time Windows (VRPTW), which is characterized by the use of time intervals assigned to each client, known as time windows. The time window increases the number of restrictions in the problem which makes the solution search more difficult. Some solutions used in the literature are for genetic algorithms. For example, Thangiah [2] proposes a genetic algorithm that consists first of grouping by route and applying the global search strategy, and second by using a post-optimization local heuristic which changes and moves between routes to approach the solution. The clustering method of clients is used in genetic algorithms. The Thangiah solution works well for problems of type R and RC of VRPTW instances. Zhu [3] proposes a genetic algorithm that uses a clustering method called the *Push-Forward Insertion Heuristic* (PFIH). The PFIH is used by Thangiah and introduced by Solomon [4]. The PFIH guarantees the discovery of a feasible solution if one exists, and from the feasible solution it generates the initial population. The Zhu genetic algorithm uses a mutation based on a probabilistic scheme which has a mutation rate of 0.06. The Zhu algorithm generates solutions near the optimum reported for instances C, R and RC. Authors such as Gonzalez [5] and Olivera [6] mention the use of the cluster or applied clustering heuristic technique to routing problems and compare the cluster technique to a genetic algorithm. This paper proposes the combination of the k-means algorithm (clustering technique) with a genetic algorithm to solve the Vehicle Routing Problem with Time Windows (VRPTW). The proposed genetic algorithm consists of the combination of three operators. One operator is the selection operator called *the best*, which is used commonly in literature. The other two operators are those proposed by the authors of this paper, the crossover operator (*crossover-k*) and the mutation operator (*mutation-S*). The initial population is generated by the k-means algorithm (clustering technique) commonly used in Data-Mining Techniques. This work focuses its attention on the mutation-S operator, which minimizes the value of aptitude of each

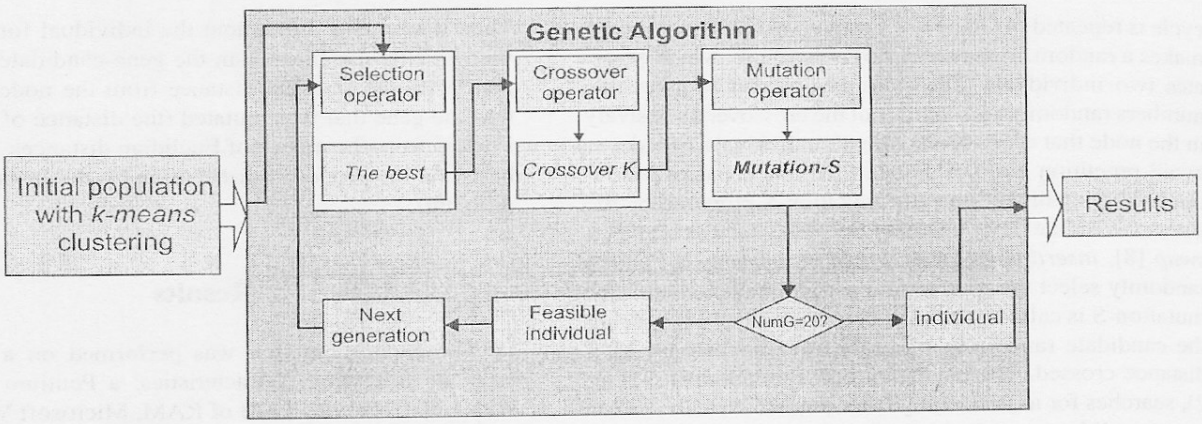


Fig. 1. Solution methodology.

individual. An individual consists of a set of genes grouped into chromosomes. An individual in VRPTW is a route made up of sub-routes (chromosomes), each sub-route is made up of nodes (genes). The mutation-S operator is called intelligent because it does not randomly make changes, instead it attempts to reduce the total distance travelled by only making changes that satisfy the time and capacity constraints. Later in the paper, the operation of the mutation-S operator will be explained in detail. Section *Methodology of Development* explains the methodology used to construct the solution, Section *Results* analyzes the obtained results and the experimentation showing the efficiency and efficacy of the algorithm in comparison to other algorithms reported in literature, and last section presents the conclusions.

Methodology of Development

Fig. 1, shows the methodology used to construct a solution to VRPTW. The methodology consists of a genetic algorithm that contains a selection operator called *the best*, a crossover operator called *crossover-k* and a mutation operator called *mutation-S* with an initial population generated with the *k-means* algorithm (clustering technique) commonly used in Data-Mining Techniques. In the development process it is important to emphasize the significance of the initial population. If feasible individuals or good individuals are used, as Tan [7] mentions, a better individual solution can be obtained. In the proposed genetic algorithm, the population is created by means of the *k-means* algorithm (clustering technique) in which the individual is grouped according to its geographic distribution as indicated by the instances of the Solomon benchmark.

This methodology starts with a feasible individual solution, which is not necessarily the best one, and attempts to find a good individual solution, minimizing the value of the fitness function. Starting with the individual solution, and using the intelligent mutation operator *mutation-S*, each gene is reorganized on its respective chromosome to arrive at a better individual solution. This means that the clustering of genes in the individual results in crossing a smaller total distance since the clustering will be between genes near to each other in the Euclidian space.

In order to obtain a solution, the authors of this paper propose a genetic algorithm that uses the combination of the selection operator „the best", and the proposed operators of crossover „crossover-k" and intelligent „mutation-S".

The description of the proposed genetic algorithm called GA-VRPTW follows:

```
0  Begin
1     Initial-Population ();       // k-means clustering
2     NumG=0;
3     while(NumG!=20)
4     {     Arrange-list ();
5           for(i=0 to 500){       // i individuals
6                 Selection-theBest ();}  // Selection operator
7           Crossover-K ();        // Crossover operator
8           Mutation-S ();         // Mutation Operator
9           NextGeneration();
10          Distances-T ();        // Compute the total distances
11          NumG=NumG+1;
12    }// end while
13 End
```

The temporal function that represents the crossover mechanism is a second order polynomial as in equation (1), with a computational complexity $O(n^2)$. Equation (2) is the temporal function of the *mutation-S* mechanism with a computational complexity of $O(n^3)$. Equation (3) is the global temporal function of the GA-VRPTW genetic algorithm.

$$T(n) = c_1 n^2 + c_2 n + c_3 \quad (1)$$

$$T(n) = c_1 n^3 + c_2 n^2 + c_3 n + c_4 \quad (2)$$

$$T(n) = c_1 n^4 + c_2 n^3 + c_3 n^2 + c_4 n + c_5 \quad (3)$$

The temporal function of the GA-VRPTW algorithm is a fourth order polynomial with a computational complexity $O(n^4)$.

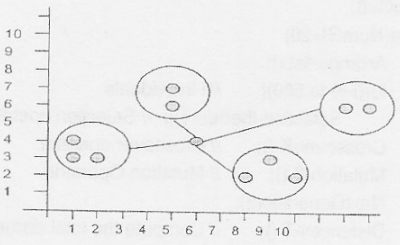
The genetic algorithm that uses the selection operator *the best*, consists of taking the best individual each time the

cycle is repeated in GA. The crossover operator *crossover-k* makes a random crossover of two individuals, which generates two individuals. The crossover procedure takes two numbers randomly and carries out the crossover exclusively in the node that corresponds to both individuals, in order to avoid repetition and not violate time and capacity restrictions. The intelligent operator mutation-S proposed by the authors is unlike other mutation operators such as *based on swap* [8], *insertion* [9], and *simple investment* [10] which randomly select the gene to mutate. The mutation operator mutation-S is called intelligent because instead of choosing the candidate randomly, it searches to minimize the total distance crossed. The intelligent operator *mutation*-S (Fig. 2), searches for a gene with greater implied distance, called gene-candidate, and looks for another gene, called gene-mutation. The gene-mutation must have a shorter distance than the gene-candidate. Once the operator *mutation*-S has identified these genes, the gene-mutation makes the change only if the time and capacity constraints are not violated. With the movements of genes in an individual, the operator *mutation*-S reduces the fitness.

The operator *mutation*-S is explained by an example of an individual with 10 genes. The individual must fulfill demand, time window and distance constraints. The process was initiated by taking an individual from the population. The operator searched for a gene with the greatest distance, this gene was the gene-candidate for mutation. Then it searched throughout the individual for the genes with smaller distances than the gene-candidate. Then the gene with the smallest distance from the node-candidate was the gene that was mutated (the distance of each gene is taken from the matrix of Euclidian distances, calculated based on the proposed distribution by the instance of the problem).

Results

The experimentation was performed on a computer with the following characteristics: a Pentium processor with 1.60 GHz with 1 GB of RAM, Microsoft Visual C++ 6.0 software, Heuristics Laboratory [11] software, and Windows XP operating system. The instances used were type C (Clustered Customers Classification) for 100-nodes taken from the instances by Solomon [12] benchmark for VRPTW. The input parameters for the experimentation in all the algorithms were made with an initial population of 1000 individuals and 20 generations. The obtained results are compared with results reported by authors who have worked with genetic algorithms applied to VRPTW [13-15]. Table 1 shows the results obtained by the genetic algorithm with intelligent mutation. *V* represents the vehicles number, *Best* represents the best result, *Average* represents the average of all execution, σ repre-



MATRIX OF EUCLIDEAN DISTANCES

	0	1	2	3	4	5	6	7	8	9	10
0	0	4.1	5.1	5.1	3.1	7.2	6.3	2.8	2.2	4.4	3.1
1	4.1	0	1.4	1	7	11.4	10.4	6.1	4.2	8.1	5
2	5.1	1.4	0	1	8.1	12.1	11.1	7.2	4.5	9.2	5
3	5.1	1	1	0	8	12.3	7.2	7.1	5	9.2	5.6
4	3.1	7	8.1	8	0	5	4.2	1.4	5	1.4	5.6
5	7.2	11.4	12.1	12.3	5	0	1	6.4	7	5	7.1
6	6.3	10.4	11.1	7.2	4.2	1	0	5.6	7	4.4	7.1
7	2.8	6.1	7.2	7.1	1.4	6.4	5.6	0	5	2	5.8
8	2.2	4.2	4.5	5	5	7	7	5	0	6.4	1
9	4.4	8.1	9.2	9.1	1.4	5	4.4	2	6.4	0	7.1
10	3.1	5	5	5.6	5.5	7.1	7.1	5.8	1	7.1	0

V1 VEHICLE1
V2 VEHICLE2
V3 VEHICLE3
CAPACITY OF VEHICLE=12

V1				V2				V3		
3	1	9	7	5	4	10	2	6	8	← INDIVIDUAL
51	1	8.1	2	7.2	5	5.6	5.1	11.1	8	← DISTANCE
3	2	1	5	5	5	2	4	2	5	← DEMAND
4	6	9	11	4	5	6	5	2	9	← TIME WINDOW

TOTAL DISTANCE = 57.2

V1				V2		MUTATION		V3		
3	1	9	7	5	4	6	2	10	8	← INDIVIDUAL
51	1	8.1	2	7.2	5	4.2	5.1	5	8	← DISTANCE
3	2	1	5	5	5	2	4	2	5	← DEMAND
4	6	9	11	4	5	7	5	2	9	← TIME WINDOW

TOTAL DISTANCE = 43.7

Fig. 2. Intelligent mutation operator *Mutation-S*.

Table 1. Results obtained by the genetic algorithm with the intelligent mutation operator.

Benchmark	V	Best	Average	σ	Op*	V*	RE
c101.100	10	**827.3985**	962.7439	107.8829	**827.3**	10	0.0119062
c102.100	10	828.2443	1027.1140	126.6793	827.3	10	0.1141423
c103.100	10	827.4851	986.2039	116.9508	826.3	10	0.1434304
c104.100	10	824.0308	1077.0705	124.9379	822.9	10	0.1374278
c105.100	10	**827.3167**	950.8616	99.4707	**827.3**	10	0.0020294
c116.100	10	827.4854	970.5487	109.2358	827.3	10	0.0224102
c107.100	10	827.8844	992.7809	122.4051	827.3	10	0.0706498
c108.100	10	**827.3336**	962.8849	105.4526	**827.3**	10	0.0040704
c109.100	10	827.6744	970.1983	106.5438	827.3	10	0.0452577

sents standard deviation, *Op** represents the optimum result reported in the literature, *V** represents the optimum vehicle number and *RE* represents the relative error. Table 1 shows that the GA-VRPTW algorithm found the optimal solution reported in literature for the C101.100, C105.100 and C108.100 instances, and for the other instances the GA-VRPTW is near the optimal solution with a small relative error.

Table 2 shows the results of the efficiency and effectiveness of the genetic algorithm with the intelligent mutation operator compared to algorithms of the heuristics laboratory [11] software, which contains genetic algorithms. The efficacy of the GA-VRPTW algorithm is high compared to the following algorithms: Genetic General Algorithm (GGA), Steady State Genetic Algorithm (SSGA) and Sexual Genetic Algorithm (SXGA) as reported by the heuristics laboratory software, because the GA-VRPTW algorithm generates results near the optimal solution in many cases, and finds the optimal solution for the instances C101.100 C105.100 and C108.100. When comparing efficiency, the heuristics laboratory algorithms are slightly faster for some instances of the GA-VRPTW algorithm. This may be because the GA-VRPTW uses the *mutation*-S mechanism which performs more operations while trying to diminish total distance crossed by conducting several searches among the genes of the individual. Even though the time function of this mechanism (see equation 2) is polynomial so the proposed operator can conserve a good efficiency for largest instances, the additional searches require increased algorithm run time. For the C101.100, C106.100, and C107.100 instances used in the GGA algorithm and for the C103.100 and C106.100 instances used in the SXGA algorithm, the run times of the GA-VRPTW algorithm are better.

Where: GGA= General Genetic Algorithm, SSGA= Steady State Genetic Algorithm, SXGA= Sexual Genetic Algorithm, UB= Upper Bound, t = time in seconds, *Op*= optimum.

The GA-VRPTW genetic algorithm proposed competes in efficacy and efficiency with the GGA and SXGA algorithms, for instances of type C (Clustered Customers Classification) for 100-nodes taken from the Solomon [12] instances for VRPTW. The k-means algorithm (clustering technique) when applied to the construction of the initial population in combination with the operator *mutation*-S generated good results for the instances that by definition are represented in the Euclidian space in clustered form (instances type C). Table 2 presents the comparative results of efficiency and efficacy of the GA-VRPTW as well as other genetic algorithms for VRPTW.

Table 2. Comparative results of efficiency and efficacy of the GA-VRPTW vs. other genetic algorithms for VRPTW.

Benchmark	**GA-VRPTW**		GGA		SSGA		SXGA		Op
	UB	t, sec	UB	t, sec	UB	t, sec	UB	t, sec	
C101.100	**827.3**	2223	931.0	7221	1099.9	23	1143.7	2100	827.3
C102.100	828.2	2734	1159.1	1740	1204.2	**24**	1159.8	2040	827.3
C103.100	827.4	2196	1204.8	1680	1184.8	24	1222.6	2220	826.3
C104.100	824.0	2515	1109.9	1740	1256.6	22	1101.9	1920	822.9
C105.100	**827.3**	2313	1047.1	2100	1038.1	24	938.9	1860	827.3
C106.100	827.4	2131	1108.0	2280	1163.4	25	1046.2	2160	827.3
C107.100	827.8	2048	1171.9	3180	1301.8	23	1064.8	1800	827.3
C108.100	**827.3**	2051	1008.9	1860	1224.7	24	1072.1	2220	827.3
C109.100	827.6	1985	1143.8	1800	1208.3	24	1164.7	2460	827.3

In this table it can be observed that the GA-VRPTW algorithm for the C101.100, C105.100 and C108.100 instances found the optimal solution reported, for the other instances the GA-VRPTW is very near to the optimal solution.

Conclusions

From the results obtained by the GA-VRPTW genetic algorithm using the operators of mutation (*mutation-S*) and crossover (*crossover-k*), the authors conclude that in terms of efficacy and efficiency, the GA-VRPTW algorithm competes with the GGA, SSGA and SXGA algorithms. The GA-VRPTW has better efficacy because none of the competing algorithms found the optimal solution. When comparing efficiency, the GA-VRPTW algorithm is competitive with the GGA and SXGA algorithms but there is a small difference between times; probably because of the *mutation-S* mechanism in GA-VRPTW algorithm, which involves several additional searches which consume time. The time function of the GA-VRPTW algorithm is of $O(n^4)$ polynomial grade. Another one of the mechanisms that the GA-VRPTW algorithm uses is the *k-means* algorithm (clustering technique), clustering individuals by the nearest node. For instances of C (clustered customer classification) it generates good results using only the coordinates (x, y) obtained from the instance. The individuals generated by the *k-means* become feasible by applying the genetic algorithm that verifies the time window and demand restrictions. Future work will improved the clustering technique and will test instances of type R (customers location generated uniformly randomly over a square) and RC (randomly and clustered customer classification) for 100-nodes of the instances of Solomon [12] for VRPTW. It is proven that the proposed algorithm is very competitive compared with other genetic algorithms and in some cases is even better.

References

1. TOTH P., VIGO D., The Vehicle Routing Problem, Society of Industrial and Applied Mathematics, Philadelphia: USA, pp 157-193, **2001**.
2. THANGIAH S. R., Vehicle Routing with Time Windows using Genetic Algorithms, Application Handbook of Genetic Algorithms: New Frontiers, Volume II, Lance Chambers (Ed.), CRC Press, 253-277, **1995**.
3. ZHU K. Q., A new genetic algorithm for vrptw, Proceedings of the International Conference on Artificial Intelligence, **2000**.
4. SOLOMON M. M., Algorithms for vehicle routing and scheduling problem with time windows constraints, Operations research **35** (2), **1987**.
5. GONZÁLEZ VARGAS G., GONZÁLEZ F., Meta heuristics applied to Vehicle Routing Problem: a case study. Part 2 genetic algorithm, comparison with a heuristic solution, Engineering and Investigation, April, Vol. 27, N. 001 National University of Colombia Bogotá, Colombia, pp. 149-157, **2007** [in Spanish].
6. OLIVERA A., Heuristics for Vehicle Routing Problem, Institute of computer science, faculty of computer science. University of the republic, Montevideo, Uruguay. **2004** [in Spanish].
7. TAN K. C., CHEW Y. H., LEE L. H., A Hybrid Multiobjective Evolutionary Algorithm for Solving Vehicle Routing Problem with Time Windows, Computational Optimization and Applications, Vol. **34** N.1, pp.115-151, ISSN:0926-6003 May **2006**.
8. BANZHAF W., The „molecular" travelling salesman, Biological Cybernetics **64**, pp: 7-14, **1990**.
9. FOGEL D., An evolutionary approach to the travelling salesman problem, Biological Cybernetics **60**, pp: 139-144, **1988**.
10. HOLLAND J. H., Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, **1975**.
11. WAGNER S., AFFENZELLER M., The Heuristic Lab Optimization Environment, Technical Report, Institute of Formal Models and Verification, Johannes Kepler University Linz, Austria. **2004**.
12. SOLOMON M. M., Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints, Operations Research **35** (2), 254-265, **1987**.
13. BERGER J., BARKAOUI M., BRÄYSY O., A Parallel Hybrid Genetic Algorithm for the Vehicle Routing Problem with Time Windows, Working paper, Defense Research Establishment Val Cartier: Canada, **2001**.
14. MESTER D., BRÄYSY O., DULLAERT W., A Multiparametric Evolution Strategies Algorithm for Vehicle Routing Problems, Working Paper, Institute of Evolution, University of Haifa: Israel, **2005**.
15. HOMBERGER J., GEHRING H., Two Evolutionary Metaheuristics for the Vehicle Routing Problem with Time Windows, INFOR, Vol. **37**, 297-318, **1999**.