American Journal of Applied Sciences 6 (8): 1586-1590, 2009
ISSN 1546-9239

Solving the Traveling Salesman Problem Using New Operators in Genetic Algorithms

[1]Naef Taher Al Rahedi and [2]Jalal Atoum
[1]Department of Electrical Engineering, Fern University in Hagen,
Control Systems Engineering Group, Germany
[2]Department of Computer Science, Princess Sumaya University of Technology,
PSUT, Amman, Jordan

Abstract: Problem statement: Genetic Algorithms (GAs) have been used as search algorithms to find near-optimal solutions for many NP problems. GAs require effective chromosome representations as well as carefully designed crossover and mutation operators to achieve an efficient search. The Traveling Salesman Problem (TSP), as an NP search problem, involves finding the shortest Hamiltonian Path or Cycle in a graph of N cities. The main objective of this study was to propose a new representation method of chromosomes using upper triangle binary matrices and a new crossover operator to be used as a heuristic method to find near-optimum solutions for the TSP. **Approach:** A proposed genetic algorithm, that employed these new methods of representation and crossover operator, had been implemented using DELPHI programming language on a personal computer. Also, for the purpose of comparisons, the genetic algorithm of Sneiw had been implemented using the same programming language on the same computer. **Results:** The outcomes obtained from running the proposed genetic algorithm on several TSP instances taken from the TSPLIB had showed that proposed methods found optimum solution of many TSP benchmark problems and near optimum of the others. **Conclusion:** Proposed chromosome representation minimized the memory space requirements and proposed genetic crossover operator improved the quality of the solutions in significantly less time in comparison with Sneiw's genetic algorithm.

Key words: Chromosomes representation, crossover operator, mutation operator, upper triangle binary matrix

INTRODUCTION

The traveling salesman problem: The Euclidean Traveling Salesman Problem (TSP) involves finding the shortest Hamiltonian Path or Cycle in a graph of N cities. The distance between the two cities is just the Euclidean distance between them. This problem is a classic example of NP problems and is therefore impossible to search for an optimal solution for realistic sizes of N. This motivated many researchers to develop heuristic search methods for searching the solution space. The TSP is probably the most-studied optimization problem of all time. Applications of TSP include Circuit board drilling applications with up to 17,000 cities[18], X-ray crystallography instances with up to 14,000 cities[18] and instances arising in VLSI fabrication have been reported with as many as 1.2 million cities[18]. Moreover, 5 h on a multi-million dollar computer for an optimal solution may not be cost-effective if one can get sub optimal solutions with acceptable error tolerance in seconds. Thus there remains a need for heuristics[18].

Since the salesman is interested in finding the shortest possible rout, this problem corresponds to finding the shortest Hamiltonian cycle in a complete graph G = (V, E) of an n nodes[10]. Thus the TSP consists of finding a permutation of the set $\{C_1, C_2, C_3, \ldots, C_N\}$ that minimize the quantity:

$$\text{Minimize}\left[\sum_{i=1}^{N-1} d\left(c_i, c_{i+1}\right) + d\left(c_N, c_1\right)\right]$$

where, $d(C_i, C_j)$ denotes the distance between city C_i and city C_j.

MATERIALS AND METHODS

Genetic algorithms: GAs are based on the biological evolution processes that can be founded in natural evolution. In a GA evolution, the individual species

Corresponding Author: Naef Taher Al Rahedi, Department of Electrical Engineering, Fern University in Hagen, Control Systems Engineering Group, Germany

compete with each other to survive (Darwinian selection). Also, GAs are considered to be search and optimization processes[5].

Genetic Algorithms (GAs) has several advantages; multiple directional searches, problem coding instead of decision variables and using stochastic transition rules[5]. It has therefore been widely used to solve Production And Operation Management (POM) problems such as supply chain and logistics[6,13], production scheduling[12], facility layout[7] and university course timetabling[1]. However, the GA applications on some POM problem areas such as transportation within logistics chain network[3], quality planning, short/long term forecasting and short-term capacity planning have rarely been found[4].

A GA presumes that a potential solution of any problem is an individual and can be presented by set of parameters. These parameters are regarded as the genes of chromosome and can be structured by a string of values in a binary form. The fitness value of an individual is used to reflect the degree of goodness of a solution (chromosome) for the problem[9]. When a constraint is violated a penalty is imposed on the individual timetable solution. The fitness of the solution depends on the penalties imposed by the constraints being violated[1].

A GA initially creates a population of solutions and applies genetic operators to evolve the solutions from one generation to the next generation until it finds an optimal or near to optimal solution, or terminates the execution under certain condition without finding any solution. The genetic operators that had been proposed by Holland to reproduce new solutions are[5,7]:

- Selection operator: Select chromosomes from the population according to their fitness values for recombination and call them parents
- Crossover operator: Produce new off springs by interchanging subparts between parents
- Mutation operator: Randomly flips some bits in a new offspring

Chromosome representation: The flexibility of chromosome representation is one of the major advantage strategies within the Genetic Algorithm (GA). For example, a single row chromosome representation is normally used for solving sequencing or scheduling problems whilst a single matrix-based chromosome representation is required for a candidate solution of a single stage transportation problem. Sun *et al*.[17] have applied a single matrix-based GA for solving the unit commitment problem, which plays an important role in the economic operation of power system. In their study, the repair mechanism is additionally embedded in the GA for dealing with the infeasible solutions generated.

Crossover operation: Genetic operations including crossover and mutation are the main stochastic search process within the GA. Crossover operation helps search strategy to explore the solution space whilst exploitation is conducted by the mutation mechanism. Fifteen crossover operations and eleven mutation techniques have been reviewed and investigated in literature[8,14].

Genetic algorithms for the TSP: Many researches had proposed various representations and genetic operators to solve the TSP with genetic algorithms. There are two approaches for these researches. The first approach represents the chromosome as string of integer numbers, this had resulted in more complex implementation of genetic operators and more time to execute. These researches combine the genetic algorithm with local search heuristics to improve the quality of solutions[10,11]. The second approach represents the chromosome as a Binary Matrix, this had resulted in faster execution of genetic operators of Seniw[16]. In this approach no heuristics where used to optimize solutions, in addition it needs more memory space requirement for implementation[18].

This study proposed a new representation of the chromosome as an Upper Triangle Matrix to save memory space and proposed a new crossover operator to be used as a heuristic to find near-optimum solution for the TSP.

Proposed tsp genetic algorithm:

Chromosome upper triangle matrix representation: The proposed algorithm represents a tour as an Upper Triangle Binary Matrix (UTBM) as in Fig. 1 which represents the tour (0, 2, 4, 1, 3, 5). Every gene is represented as binary bit, if the element (i, j) in the matrix is set to (1) it means that there is an edge (direct path) between city (i) and city (j) in the tour.

0	0	1	0	0	1
	1	0	1	1	0
		2	0	1	0
			3	0	1
				4	0

Fig. 1: The proposed representation of a chromosome

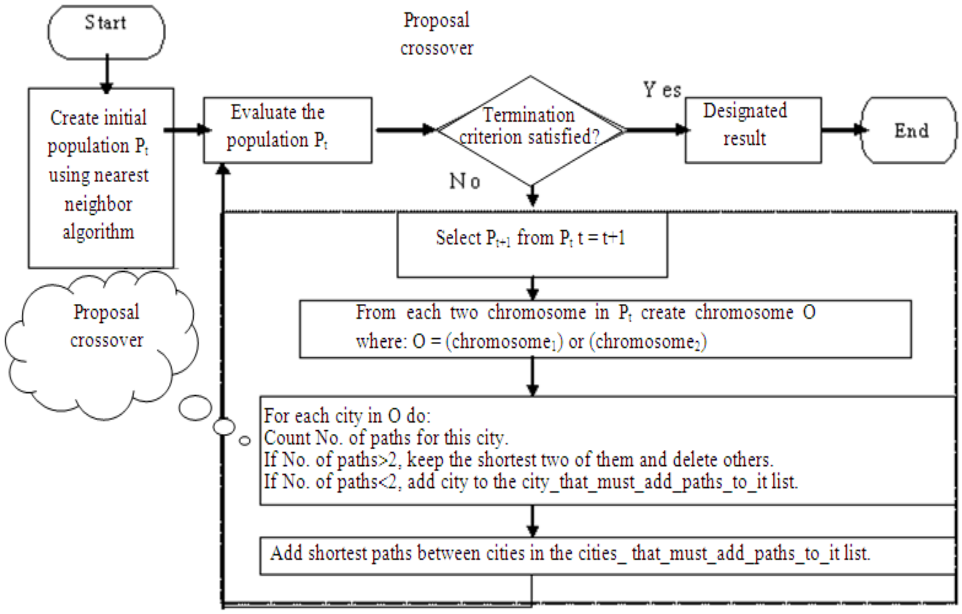

Fig. 2: The proposed TSP genetic algorithm

The matrix representation must satisfy the following conditions to satisfy a legal tour:

- The number of elements in the matrix that have the value (1) must equal to the number of the cities in the tour:

$$\sum_{i=0}^{N-2} \sum_{j=1}^{N-1} c_{ij} = N$$

- The number of matrix elements that have the value of (1) in each row and each column of the same city must be equal to two:

$$\mathrm{Col}_j = \sum_{i=0}^{N-2} c_{ij}$$
$$\mathrm{Row}_i = \sum_{j=1}^{N-1} c_{ij}$$
$$\forall (i = j) : \mathrm{Row}_i + \mathrm{Col}_j = 2$$

The process of the Proposed TSP genetic algorithm is illustrated Fig. 2.

Heuristic crossover operator: The proposed crossover operator construct an offspring from two parents as follows:

- Union the two parents upper triangle matrices into a single upper triangle matrix by executing the or operation
- If the resulted tour (upper triangle matrix) is an illegal tour (i.e., does not satisfy the two conditions above), then it must be repaired. The repairing is done by counting the number of elements that have the value of (1) in each row and column for the same city, if the number is greater than 2 edges then repeat deleting the longest edge from the resulted tour until the number of elements is equal to 2. However, if the number of elements in the resulted tour is less than 2 then add this city to the list:

List_of_Cities_that_Must_Add_Edge_To_It.
(LCMAETI)

- Adding the missing edges to the cities in the LCMAETI list is done through the greedy algorithm

RESULTS AND DISCUSSION

The repairing process of a tour in the crossover operator in the matrix representation consumes most of the time in execution[18]. To improve the efficiency of the proposed algorithm a path cost upper triangle matrix is created to store the cost of edges between any two cities. This upper triangle binary matrix can be implemented as a string of bits in order to save memory space and increase the efficiency of operators through reducing the calculation of path cost.

Table 1: Results Obtained from Proposed Algorithm for TSP Instances from TSPLIB

Problem	Time (hh: mm: sec)	No. of possible solutions	N. of solutions that the GA checked	Near-optimum solution (%)
Bayg29	<1 sec	$8.841762*10^{30}$	60	100.00
Att48	00:01:59	$1.2413916*10^{61}$	297000	95.61
Eil51	00:02:00	$1.5511188*10^{66}$	297000	99.97
Berlin52	00:00:12	$8.0758175*10^{67}$	26520	100.00
KorA100	00:09:09	$9.332622*10^{157}$	335250	96.73
D198	00:09:43	$1.98155243*10^{370}$	99000	89.47
A280	00:01:38	$1.67722778*10^{565}$	9000	88.18
Lin318	00:02:12	$2.07298525*10^{659}$	10000	84.51
Pcb442	00:04:34	$1.09740011*10^{979}$	4000	87.15
Rat783	00:05:25	$4.0634732*10^{1927}$	2000	83.55
Fl1400	00:1:45	$3.46062936*10^{3798}$	400	80.31
Fl1577	00:03:24	$1.2479901*10^{4360}$	600	87.05

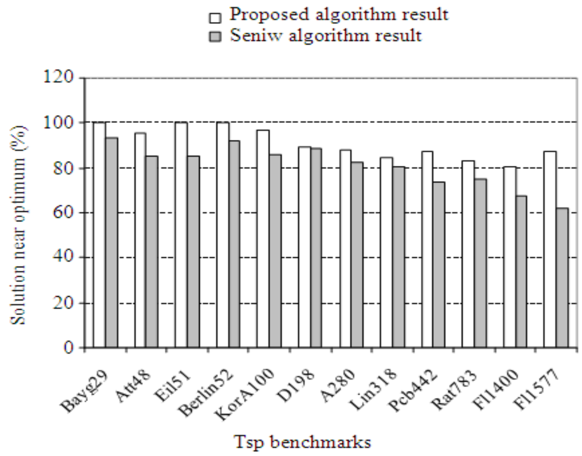

Fig. 3: Quality of solutions of proposed algorithm and Seniw algorithm

The time complexity of the proposed algorithm is:

$$O\left(I\lambda P\left(\left(\frac{N}{2}*(N-1)\right)+N^2+M^2\log M+P\right)\right)$$

Where:

I = The number of iterations
P = Population size
N = The number of cities in the tour
M = The number of cities in the List of Cities_Must_Add_Edges_To_It

Seniw Algorithm represents a tour as a binary matrix so that the number of bits to implement a tour that consist of N cities is 2N. For each tour there is more than one implementation depending on the start city of the tour. In the proposed algorithm the tour is represented as an UTBM so that the number of bits to implement tour that consist of N cities is (2N/2-N/2) and there is only one implementation for any tour though the start city may differ, this will result in a reduction of memory space requirement.

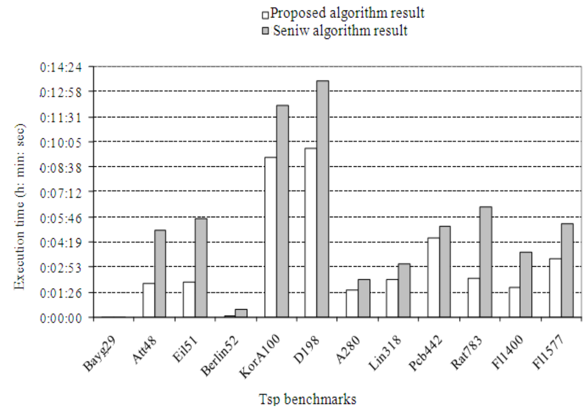

Fig. 4: Execution time of proposed algorithm and Seniw algorithm

Furthermore, in Sneiw algorithm an offspring is constructed by multiplying the two matrixes of the parents (AND operator) and then the missing edges are added randomly. In the proposed algorithm an offspring is constructed using the heuristic as discussed above, this will result in optimizing the solutions obtained form the proposed GA.

The proposed algorithm presented in this study have been implemented in DELPHI on a PC running at 2 GHz. Table 1 shows the results obtained for several TSP instances taken from the TSPLIB[15].

Figure 3 shows the comparisons of quality of solutions between the proposed algorithm and Seniw algorithm.

The comparison of execution times of the two algorithms is depicted in Fig. 4.

From Fig. 3 and 4 we can notice that the proposed algorithm have significantly outperformed Seniw algorithm in terms of quality of solutions and in execution time.

CONCLUSION

A new representation method of chromosomes had been proposed using an Upper Triangle Matrix. Also, a new crossover operator had been proposed as a heuristic method to find near-optimum solution for the TSP problem.

The results of comparisons of the proposed genetic algorithm and of Seniw algorithm had showed that the proposed representation minimizes the memory space requirement for binary representation and the combination of the proposed heuristic and genetic crossover improve the quality of the solutions.

REFERNCES

1. Adamidis, P. and P. Arapakis, 1999. Evolutionary algorithms in lecture timetabling. Proceeding of the Congress on Evolutionary Computation, Washington DC., USA., ISBN: 0-7803-5536-9.
2. Aggarrwal, Ch.C., J.B. Orlin and R.P. Tai, 1997. Optimized crossover for the independent set problem. Operat. Res., 45: 226-234.
3. Aytug, H., M. Knouja and F.E. Vergara, 2003. Use of genetic algorithms to solve production and operations management problems: A review. Int. J. Prod. Res., 41: 3955-4009.
4. Chaudhry, S.S. and W. Luo, 2005. Application of genetic algorithms in production and operation management: A review. Int. J. Prod. Res., 43: 4083-4101. DOI: 10.1080/00207540500143199
5. Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Massachusetts, Addison-Wesley, ISBN: 0201157675, pp: 412.
6. Gunnarsson, H., M. Ronnqvist and J.T. Lundgren, 2004. Supply chain modeling of forest fuel. Eur. J. Operat. Res., 158: 103-123.
7. Hicks, C., 2006. A genetic algorithm tool for optimising cellular or functional layouts in the capital goods industry. Int. J. Prod. Econ., 104: 598-614. DOI: 10.1016/j.ijpe.2005.03.010
8. Hong, T.P., H.S. Wang, W.Y. Lin and W.Y. Lee, 2002. Evolution of appropriate crossover and mutation operators in a genetic process. Applied Intel., 16: 7-17.
9. Man, K.F., K.S. Tang and S. Kwong, 1996. Genetic algorithms: Concept and applications. IEEE. Trans. Ind. Elect., 46: 519-533. DOI: 10.1109/41.538609
10. Merz, P. and B. Freisleben, 1997. Genetic local search for the TSP: New results. Proceedings of the IEEE International Conference on Evolutionary Computation, Apr. 13-16, IEEE Xplore Press, Indianapolis, USA., pp: 159-164. DOI: 10.1109/ICEC.1997.592288
11. Mühlenbein, H., 1992. Parallel Genetic Algorithm. In: Combinatorial Optimization, Computer Science and Operations Research, Balci, O., R. Sharda and S. Zenios (Eds.). Pergamon Press, New York, ISBN: 3-929037-16-5, pp: 441-456.
12. Pongcharoen, P., C. Hicks and P.M. Braiden, 2004. The development of genetic algorithms for the finite capacity scheduling of complex products, with multiple levels of product structure. Eur. J. Operat. Res., 152: 215-225. DOI: 10.1016/S0377-2217(02)00645-8
13. Pongcharoen, P., W. Chainate and P. Thapatsuwan, 2007. Exploration of genetic parameters and operators through travelling salesman problem. Sci. Asia, 33: 215-222. DOI: 10.2306/scienceasia1513-1874.2007.33.215.
14. Pongcharoen, P., D.J., Stewardson, C. Hicks and P.M. Braiden, 2001. Applying designed experiments to optimize the performance of genetic algorithms used for scheduling complex products in the capital goods industry. J. Applied Stat., 28: 441-455. DOI: 10.1080/02664760120034162
15. Reinelt, G., 1991. TSPLIB-A traeling salesman problem library. ORSA. J. Comput., 3: 376-384. DOI: 10.1287/ijoc.3.4.376
16. Seniw, D., 1991. A genetic algorithm for the traveling salesman problem, MSc Thesis, University of North Carolina, at Charlotte. http://www.heatonresearch.com/articles/65/page1.html
17. Sun, L., Y. Zhang and C. Jiang, 2006. A matrix real-coded genetic algorithm to the unit commitment problem. Elect. Power Syst. Res., 76: 716-728. DOI: 10.1016/J.EPSR.2005.10.005.
18. Zbigniew, M., 1996. Genetic Algorithms + Data Structures = Evolution Programs. 3rd Edn., Springer, New York, ISBN: 3540606769, pp: 387.