Comparison Of Various Mutation Operators Of Genetic Algorithm To Resolve Travelling Salesman Problem

Akshatha P. S.[1], Vasudha Vashisht[2], Tanupriya Choudhury[3]

Student, Computer Science Department, Lingaya's University, Faridabad, India[1]

Assistant Professor, Computer Science Department, Lingaya's University, Faridabad, India[2]

Assistant Professor, Computer Science Department, Lingaya's University, Faridabad, India[3]

Abstract— The Travelling Salesman Problem (TSP) is a classic combinatorial optimization problem, which is simple to state but very difficult t o solve. This problem is known to be NP-hard, and cannot be solved exactly in polynomial time. Genetic algorithms have been successfully utilized for solving NP-hard problems in the last 30 years. They can reduce the running times of NP-complete problems substantially. The main purpose of this paper is to give a brief introduction about genetic algorithm and travelling salesman problem. And we are going to discuss how genetic algorithm can be applied, especially its mutation operators to solve the TSP. In the end we will conclude that inversion mutation operator will always gives better results compared to other mutation operators.

Keywords— Genetic Algorithm, Travelling Salesman problem, Mutation Operators.

I. INTRODUCTION

The travelling salesman problem is finding a shortest possible cycle visiting every city in a map given the set of cities and pair wise distances between them. Modeling of this problem can also be done with an undirected weighted graph. The vertices of the graph are cities. The edges of the graph are distances. The travelling salesman problem is one the toughest and fundamental problems in real world application. Genetic Algorithms have proven efficient in solving the Travelling Salesman Problem in last 30 years or so. Travelling Salesman is an NP-Hard problem. In this paper we compared and concluded the solution of TSP using Exchange Mutation, Inversion Mutation and Scramble Mutation and we found that, the using of Inversion Mutation operator will always gives better solution based on our experiments.

II. GENETIC ALGORITHMS

.

Genetic Algorithms (or simply GAs) are powerful and widely applicable stochastic search and optimization methods based on the concepts of natural selection and natural evaluation. GAs work on a population of individuals represents candidate solutions to the optimization problem. These individual are consists of a strings (called chromosomes) of genes. The genes are a practical allele (gene could be a bit, an integer number, a real value or an alphabet character,…, etc depending on the nature of the problem). GAs applying the principles of survival of the fittest, selection, reproduction, crossover (recombining), and mutation on these individuals to get, hopefully, a new butter individuals (new solutions) .GAs are applied for those problems which either cannot be formulated in exact and accurate mathematical forms and may contain noisy or irregular data or it take so much time to solve or it is simply impossible to solve by the traditional computational methods.

How Genetic algorithms Work

The flowchart explains how genetic algorithms work is showing in the following figure.

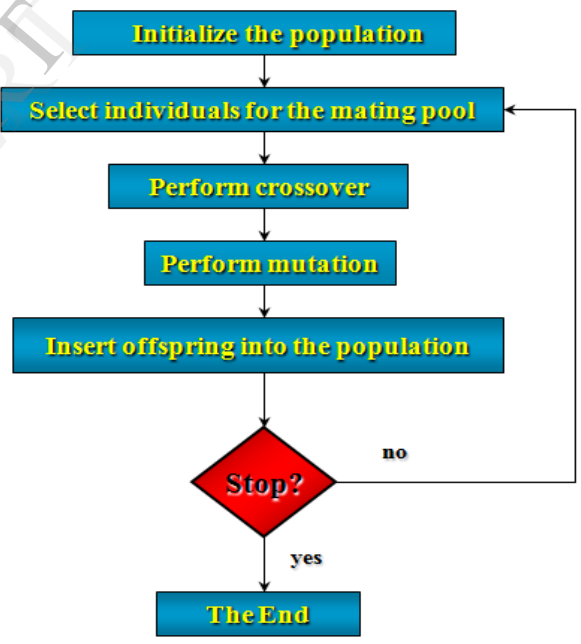


Figure (1) Flowchart explains the general structure of the genetic algorithms.

Genetic algorithm maintains a population of individuals, say **P(t)**,for generation **t**. Each individual represents a potential solution to the problem at hand. Each individual is evaluated to give some measure of its fitness. Some individuals undergo stochastic transformations by means of genetic operations to form new individuals. There are two type of transformation:-

1) Mutation, which creates new individuals by making changes in a single individual.

2) Crossover, which creates new individuals by combining parts from two individuals.
The new individuals, called offspring **C(t)**, are then evaluated. A new population is formed by selecting the more fit individuals from the parent population and offspring population.

The genetic algorithm process consists of the following:

1) Encoding: The process of representing the solution in the form of a string that conveys the necessary information. They are clssified in to 2 catageroies[26].

a. Binary Encoding

Chromosome A 1011001011001110010l

Chromosome B 1111111000000001111l

b. Permutation Encoding
Eg: TSP

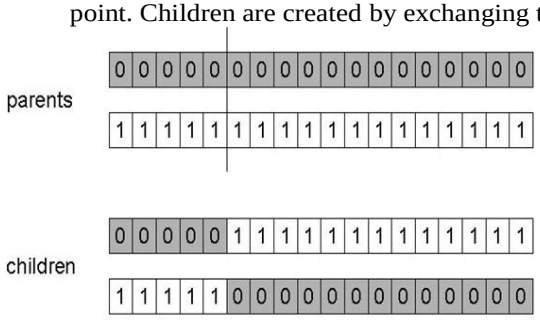
Chromosome A 1 5 3 2 6 4 7 9 8

Chromosome B 8 5 6 7 2 3 1 4 9

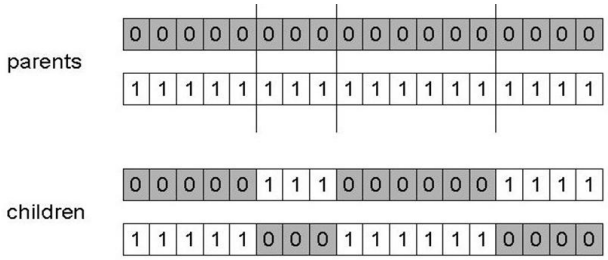
2) Selection: The best chromosomes will be selected based on the fitness function. A fitness value is assigned to each solution depending on how close it actually is to solving the problem.

*3) Crossover:*It is the process in which two chromosomes (strings) combine their genetic material (bits) to produce a new offspring which possesses both their characteristics. The following are some of the examples for Crossover Operator applied on binary encoding.

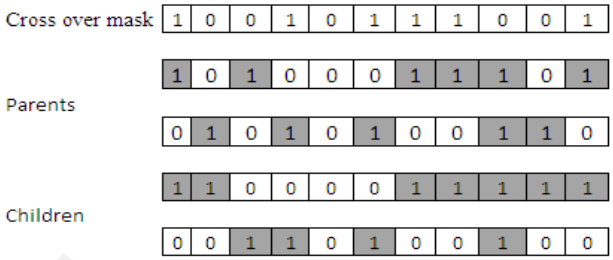
a) Single Point: A cross-point is selected randomly. The parents are split at this cross over point. Children are created by exchanging tails.



b) Multi Point Crossover: Multi cross over points are selected randomly.

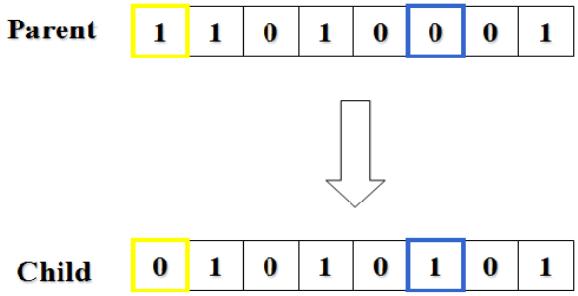


c) Uniform Crossover: In this form of cross over operator each bit from either parent is selected with a probability of 0.5 and then exchanged. A cross over mask is randomly generated. When there is 1 in the mask, the gene is copied from the first parent and if 0, the gene is copied from the second parent.

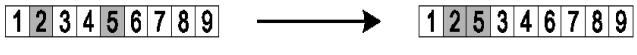


4) Mutation: The operation of mutation allow new individual to be created. It begins by selecting an individual from the population based on its fitness. A point along the string is selected at random and the character at that point is randomly changed, the alternate individual is then copied in to the next generation of the population. Following are the some of the examples for mutation operator.

a) Bit-Mutation:



b) Insert-Mutation:



5) Recombination: The process that determines which solutions are to be preserved and allowed to reproduce and which ones deserve to die out. The good

solution survive, while bad ones die. This process will be useful to generate new population.

These 5 steps of genetic algorithm will be repeated until it meets predefined iterations or until to get the near optimal solution.

III. TRAVELLING SALESMAN PROBLEM

The Travelling Salesman Problem (TSP) is considered to be an important combinatorial problem, because of its simplicity to define Though TSP is easy to define but difficult to solve because it is as an NP-hard optimization problem. It consists of n cities which must be visited by a salesman, starting from one of them passing through each city only once and returning to the first city. The cost is given for the journey, which is referred to as a tour. Finally, the minimum cost is required to solve this problem.

The Travelling Salesman Problem (TSP) is determined as follows: Given N cities, known as nodes, a distance matrix where, D = $[d_{ij}]$, consists of the distance between city i and city j. In an attempt to finding near optimal solutions for NP-hard problems; the Travelling Salesman Problem (TSP) is considered a standard benchmark problem for combinatorial methods [2]. It provides a standard optimization test bed, to find near optimum solutions to NP-hard problems [1, 4, 5, and 6]. It is to be known, that, the Travelling Salesman Problem (TSP) is called Symmetric TSP (Standard), if the cost between any two cities are equal in both directions, which means, the distance from city i-j is the same as the distance from city j-i. Two alternative approaches exist to solve the Travelling Salesman Problem (TSP). First, is to find its solution and try proving its optimality, this takes a long period of time, which might take years. The second, approach, is to find an approximate solution in a short period of time [3].

Practical Applications of TSP:

Much of the work on the TSP is motivated by its use as a platform for the study of general methods that can be applied to a wide range of discrete optimization problems. This is not to say, however, that the TSP does not find applications in many fields. Indeed, the numerous direct applications of the TSP bring life to the research area and help to direct future work.

1. *Genome Sequencing*
   Researchers at the National Institute of Health have used Concorde's TSP solver to construct radiation hybrid maps as part of their ongoing work in genome sequencing. The TSP provides a way to integrate local maps into a single radiation hybrid map for a genome; the cities are the local maps and the cost of travel is a measure of the likelihood that one local map immediately follows another.

2. *Starlight Interferometer Program*
   A team of engineers at Hernandez Engineering in Houston and at Brigham Young University have experimented with using Chained Lin-Kerninghan to optimize the sequence of celestial objects to be imaged in a proposed NASA *Starlight* space interferometer program. The goal of the study it to minimize the use of fuel in targeting and imaging maneuvers for the pair of satellites involved in the mission (the cities in the TSP are the celestial objects to be imaged, and the cost of travel is the amount of fuel needed to reposition the two satellites from one image to the next).

3. *Scan Chain Optimization*
   A semi-conductor manufacturer has used Concorde's implementation of the Chained Lin-Kernighan heuristic in experiments to optimize scan chains in integrated circuits. Scan chains are routes included on a chip for testing purposes and it is useful to minimize their length for both timing and power reasons.

4. *DNA Universal Strings*
   A group at AT&T used Concorde to compute DNA sequences in a genetic engineering research project. In the application, a collection of DNA strings, each of length *k*, were embedded in one universal string (that is, each of the target strings is contained as a substring in the universal string), with the goal of minimizing the length of the universal string. The cities of the TSP are the target strings, and the cost of travel is *k* minus the maximum overlap of the corresponding strings.

5. *Whizzkids '96 Vehicle Routing*
   A modified version of Concorde was used to solve the Whizzkids'96 vehicle routing problem, demonstrating that the winning solution in the 1996 competition was in fact optimal. The problem consists of finding the best collection of routes for 4 newsboys to deliver papers to their 120 customers. The team of David Applegate, William Cook, Sanjeeb Dash, and Andre Rohe received a 5,000 Gulden prize for their solution in February 2001 from the information technology firm CMG.

6. *Touring Airports*
   Concorde is currently being incorporated into the Worldwide Airport Path Finder web site to find shortest routes through selections of airports in the world. The author of the site writes that users of the path-finding tools are equally split between real pilots and those using flight simulators.

7. *Designing Sonet Rings:* An early version of Concorde's tour finding procedures was used in

a tool for designing fiber optical networks at Bell Communications Research (now Telcordia). The TSP aspect of the problem arises in the routing of sonet rings, which provide communications links through a set of sites organized in a ring. The ring structure provides a backup mechanism in case of a link failure, since traffic can be rerouted in the opposite direction on the ring.

IV. SOLVING TSP USING GA MUTATION OPERATOR: A PROPOSED WORK:

A genetic algorithm [7] can be used to find a solution in much less time. Although it might not find the best solution, it can find a near perfect solution for a 100 city tour in less than a minute.
To solve the traveling salesman problem, we need a list of city locations and distances, or cost, between each of them. The following are basic steps of our proposed work.

Step 1: **Encoding**

Permutation encoding will be used to solve TSP. We represented every city by an integer. For example, *Consider 6 Indian cities:* Mumbai, Nagpur, Calcutta, Delhi, Bangalore and Chennai and assign a number to each.

Mumbai	1
Nagpur	2
Calcutta	3
Delhi	4
Bangalore	5
Chennai	6

After representing the cities in the form of integer, we will initialize the population. i.e., for the above cities, in total 6! Different possible paths are available, out of which to choose the best one decreases as the number of cities increases; hence we consider few paths for TSP solution randomly.

Step2: **Distance Matrix/ Cost Matrix**

Distance Matrix is an NxN matrix of point to point distances/costs. For example, consider the following distance matrix for above 6 cities.

	1	2	3	4	5	6
1	0	863	1987	1407	998	1369
2	863	0	1124	1012	1049	1083
3	1987	1124	0	1461	1881	1676
4	1407	1012	1461	0	2061	2095
5	998	1049	1881	2061	0	331
6	1369	1083	1676	2095	331	0

Step 3: **Selection Based on Fitness Function**

The fitness function will be the total cost of the tour represented by each chromosome.This can be calculated as the sum of the distances traversed in each travel segment. *The Lesser The Sum, The Fitter The Solution* Represented By That Chromosome[7].

So, for a chromosome [4 1 3 2 5 6], the total cost of travel or fitness will be calculated as shown below
Fitness = 1407 + 1987 + 1124 + 1049 + 331+ 2095
= 7993 kms.
Since TSP deals with round trip, i.e., we are also supposed to add the distance between city 6 to city 4.
Therefore Fitness= 7993+2095
= 10088 kms
Since our objective is to minimize the distance, the lesser the total distance, the fitter the solution.

Step 3: Generating random numbers equal to population size.

Step 4: Dividing random numbers into intervals of 2.

Step 5: According to sequence of random numbers selecting routes from population size.

Step 6: Best of 2 routes will be chosen using Tournament selection to apply Mutation.

[***Note***: In our experiment we haven't considered the crossover operator due to the following reason.

Parent 1	1 2 3 4 5
Parent 2	3 5 2 1 4
Child 1	1 2 3 1 4
Child 2	3 5 2 4 5

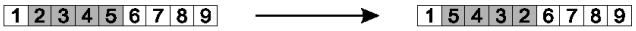
Problems here:

- City 1 repeated in Child 1
- City 5 repeated in Child 2

The above situations does not satisfy the rule of TSP. Though many researchers are invented new crossover techniques for permutation encoding, we are not considering it , we have considered only mutation.]

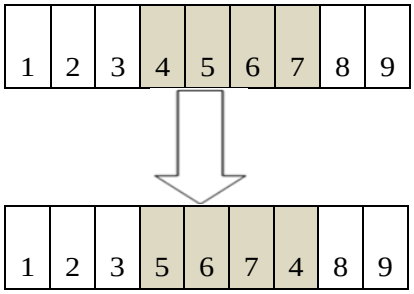
Step 7: Best route from each group of 2 will be chosen and applied mutation operators; they are, Inversion Mutation, Exchange Mutation and Scramble Mutation.

Inversion Mutation:



Exchange Mutation:

Scramble Mutation:



Step 8: Next generation of population size will be generated.

Step 9: Process will undergo predefined iterations.

Step 10: After the final iteration the smallest distance from the population size will be displayed as result.

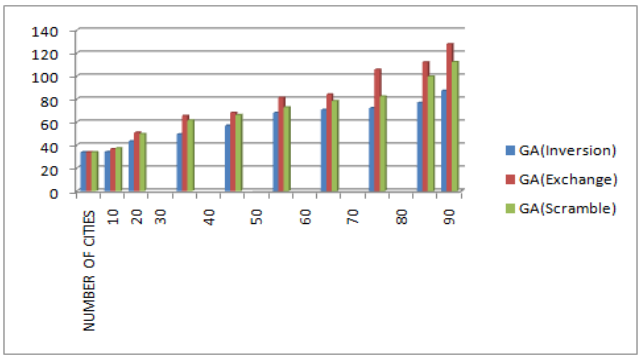
V. RESULTS

The following are the results obtained for each mutation operator we have used. Consider the following the following table which shows that Inversion mutation will always gives better results.

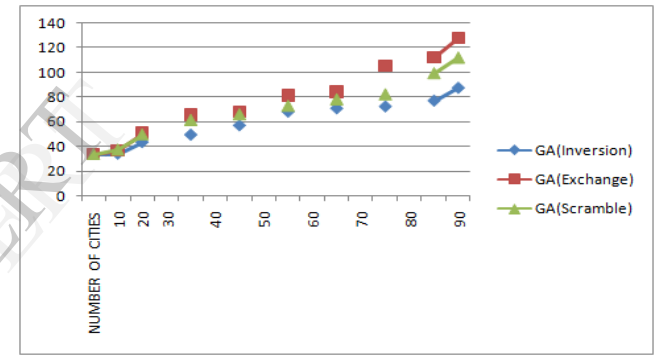
NUMBER OF CITIES	ITERATIONS	POP_SIZE	GA(Inversion)	GA(Exchange)	GA(Scramble)
10	2000	20	33.7464	33.7464	33.7464
20	4000	40	33.9979	36.5076	37.341
30	6000	60	43.3578	50.8658	49.7778
40	8000	80	49.3478	65.5511	61.5794
50	10000	100	56.902	68.1469	66.1501
60	12000	120	68.023	81.2718	72.9509
70	14000	140	70.6605	84.1625	78.3081
80	16000	160	72.2052	105.7127	82.3256
90	18000	180	76.8693	112.0468	99.599
100	20000	200	87.272	127.9681	112.3047

It is very clear from the above table that, as the number of cities increases, the inversion mutation operator will always gives the better results. Thus from the above experimental results we conclude that Inversion mutation operator is better than rest two and also Exchange mutation operator gives worst results than rest two.

The following is the column graph for the above table. The horizontal line refers number of cities and where as the vertical line refers total distance.



The following graph is a line with marker for the same table. The horizontal line refers number of cities and where as the vertical line refers total distance.



VI. CONCLUSION

Genetic algorithms appear to search good solutions for the traveling salesman problem, however it depends very much on the way the problem is encoded and which crossover and mutation methods are used. So our sincere efforts to decide which mutation operator will always gives result to tackle most of the real life applications of travelling salesman problem. As a future work we want continue the same procedure different other crossover and mutation operator to give better solution to a travelling salesman problem.

REFERENCES

[1] Beasley, D., D.R. Bull and R.R. Martin, 1993. An overview of genetic algorithms: Part 1, Research Topics. Univ. Comput.,15:58-69.
http://www.geocities.com/francorbusetti/gabeasley1.pdf

[2] Pullan, W., 2003. Adapting the genetic algorithm to the traveling salesman problem. Proceeding of the Congress on Evolutionary Computation, Dec. 8-12, IEEE Computer Society, Washington DC., USA., pp: 1029-1035. DOI: 10.1109/CEC.2003.1299781

[3] Yang, R., 1997. Solving large travelling salesman problems with small populations. Proceeding of the Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications, Sep. 2-4, IEEE Computer Society, Washington DC.,USA.,pp:157-162. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnum ber=681004

[4] White, C.M. and G. Yen, 2004. A hybrid evolutionary algorithm for travelling salesman problem. Proceeding of the Congress on Evolutionary Computation, June 19-23, IEEE Computer Society, Washington DC., USA., pp: 1473-1478. DOI: 10.1109/CEC.2004.1331070

[5] Reinelt, G., 1994. The travelling salesman: Computational solutions for TSP applications. Lecture Notes Comput. Sci., 840. DOI: 10.1007/3- 540-48661-5

[6] Fiechter, L., 1994. A parallel tabu search algorithm for large travelling salesman problems. Discrete Applied Math. Combinat. Operat. Res. Comput. Sci., 51: 243-267. DOI: 10.1016/0166- 218X(92)00033-I

[7] Holland, H.J., 1992. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence. 1st Edn., MIT Press, Cambridge, MA., USA., ISBN: 10: 0262581116, pp: 228.

[8] Zheng, Y. and S. Kiyooka, 1999. Genetic algorithm applications.www.me.uvic.ca/~zdong/ courses/mech620/GA_App.PDF

[9] Liao, Y. and C.T. Sun, 2001. An educational genetic algorithms learning tool. IEEE Trans. Educ., 44: 20. DOI: 10.1109/13.925863

[10] AL-Madi, A.N. and A.T. Khader, 2007. A socialbased model for genetic algorithm. Proceedings of the 3rd International Conference on Information Technology, (ICIT'07), AL-Zaytoonah University, Amman, Jordan.

[11]Dalton,J.,2007.GeneticAlgorithms. http://www.edc.ncl.ac.uk/highlight/rhjanuary2007.php/

[12] Bagheri, E. and H. Deldari, 2006. Dejong function optimization by means of a parallel approach to fuzzified genetic algorithm. Proceedings of the 11th IEEE Symposium on Computers and Communications, June 26-29, IEEE Computer Society, Washington DC., USA., pp: 675-680. DOI: 10.1109/ISCC.2006.57

[13] Back, T., 1996. Evolutionary Algorithms in theory and practice Evolution Strategies, Evolutionary Programming, Genetic Algorithms. 1st Edn., Oxford University Press, USA., ISBN: 0195099710, pp: 328

[14] Lamom, A., T. Thepchatri and W. Rivepiboon, 2008. Heuristic algorithm in optimal discrete structural designs. Am. J. Applied Sci., 5: 943-951. http://findarticles.com/p/articles/mi_7109/is_8_5/ai_n28552361 /pg_6

[15] Gorges-Schleuter, M., 1989. ASPARAGOS an asynchronous parallel genetic optimization strategy. Proceedings of the 3rd International Conference on Genetic Algorithms, 1989, Morgan Kaufmann Publishers Inc., San Francisco, CA., USA., pp: 422-427. http://portal.acm.org/citation.cfm?id=93126.94041 &coll=ACM&dl=ACM

[16] Back, T., D.B. Fogel and Z. Michalewicz, 1997. Handbook on Evolutionary Computation. Ringbound Edn., IOP Publishing Ltd. and Oxford University Press, ISBN: 10: 0750303921, pp: 988.

[17] Alba, E., M. Giacobini and M. Tomassini, 2002. Comparing synchronous and asynchronous cellular genetic algorithms. Lecture Notes Comput. Sci., 2439: 601-610. http://www.springerlink.com/content/bfnc36jllk6vxvvy/

[18] Ursem, R.K., 1999. Multinational evolutionary algorithms. Proceedings of the Congress of Evolutionary Computation, July 6-9, IEEE Computer Society, Washington DC., USA., pp: 1633-1640. DOI: 10.1109/CEC.1999.785470

[19] Zbigniew S. and K. De Jong, 2005. The influence of migration sizes and intervals on island models. Proceedings of the Conference on Genetic and Evolutionary Computation, June 25-29, ACM Press, New York, USA., pp: 1295-1302. http://portal.acm.org/citation.cfm?id=1068009.1068219

[20] Belal, M.A. and I.H. Khalifa, 2002. A comparative study between swarm intelligence and genetic algorithms. Egypt. Comput. Sci. J., 24. http://net.shams.edu.eg/ecs/jan_02_a2.htm

[21] Babbar, M., B. Minsker and D.E. Goldberg, 2002. A multiscale island injection genetic algorithm for optimal groundwater remediation design. Proceedings of the Conference on Water Resources Planning and Management, (CWRPM'02), American Society of Civil Engineers (ASCE) Environmental and Water Resources Institute (EWRI), Roanoke, VA.

[22] Wei, J. and D.T. Lee, 2004. A new approach to the travelling salesman problem using genetic algorithms with priority encoding. Proceeding of the Congress on Evolutionary Computation, June 19-23, IEEE Computer Society, Washington DC., USA., pp: 1457-1464. DOI: 10.1109/CEC.2004.1331068

[23] Nguyen, H.D., I. Yoshihara, K. Yamamori and M. Yasunaga, 2007. Implementation of an effective hybrid GA for large-scale travelling salesman problems. IEEE Trans. Syst. Man Cybernet Part B., 37: 92-99. DOI: 10.1109/TSMCB.2006.880136

[24] Xuan, W. and Y. Li, 2005. Solving travelling salesman problem by using a local evolutionary algorithm. Proceeding of the IEEE International Conference on Granular Computing, July 25-27, IEEE Computer Society, Washington DC., USA., pp: 318-321. DOI: 10.1109/GRC.2005.1547294

[25] Lee, Z.J., 2004. A hybrid algorithm applied to travelling salesman problem. Proceedings of the IEEE International Conference on Networking, Sensing and Control, Mar. 21-23, IEEE Computer Society, Washington DC., USA., pp: 237-242. DOI: 10.1109/ICNSC.2004.1297441

[26] http://www.obitko.com/tutorials/genetic algorithms/ encoding.php

[27] Smith, K., 1996. An argument for abandoning the travelling salesman problem as a neural-network benchmark. IEEE Trans. Neural Networks, 7: 1542-1544. DOI: 10.1109/72.548187

BIOGRAPHY

1st Author

Akshatha P S received her Bachelor's degree in Computer Science from VTU, Karnataka, India. Currently she is pursuing her Master's degree from Lingaya's University. She has around 5 years of teaching experience. She has authored 10 papers and her areas of interests include Genetic Algorithm, Data Mining, Artificial Intelligence, Mobile Ad-hoc networks etc. She is a student member of Computer Society of India.

2nd Author

Vasudha Vashisht received her Bachelor's and Master's degree in Computer Science from M.D. University, Haryana, India. Currently she is pursuing her doctoral degree in Computer Science & Engg. She has 7 years of experience in teaching. Currently, she is working as an Assistant Professor in the School of Computer Science at Lingaya's University, Faridabad, Haryana, India. She has authored 15 papers and her areas of interests include Artificial Intelligence, Cognitive Science, Brain Computer Interface, and Image & Signal Processing. She is a member of reputed bodies like IEEE, International Association of Engineers, International Neural Network Society, etc.

3rd Author

Tanupriya Choudhury received his Bachelor's degree in CSE from West Bengal University of Technology, Kolkata, India, and Master's Degree in CSE from Dr. M.G.R University, Chennai, India. Currently he is pursuing his doctoral degree in Computer Science & Engg. He has 3 years experience in teaching. Currently he is working as an Assistant Professor in School of Computer Science at Lingaya's University, Faridabad, Haryana, India. He is the author of 10+ Nat'l/Int'l publications. His areas of interests include Cloud Computing, Network Security, Data mining and Warehousing, Image processing etc.