IJITKM 0973-4414 Volume-9, Number-2 Jan-Jun 2016 pp. 1-8 **Impact Factor 2.8** available online at www.csjournalss.com

A Knowledge Augmented Order Crossover for Travelling Salesperson Problem

Poonam Punia, Surender Jangra
Ph.D Research Scholar, Deptt. of Computer Applications, Singhania University, Jhunjunu (Raj.)
Deptt. of Computer Applications, GTB College, Bhawanigarh (Sangrur),Punjab
poonamgill25@gmail.com, ssjangra20@rediffmail.com

Abstract: This paper proposes a modified Order crossover operator for genetic algorithm that generates high quality solutions to the Traveling Salesman Problem (TSP) effectively. As the main time consuming process of crossover operator and schemata preserving process in crossover is the hole filling done in Order Crossover operator, so if the crossover point are selected by augmenting some knowledge then it might be proven to find near optimum solutions in effective and efficient manner. The Modified Order Crossover operator constructs an off spring from a pair of parents using the existing Order Crossover operator with the augmentation of knowledge. The efficiency of the Modified Order Crossover is compared as against existing order crossover operators; Experimental results suggest that the new crossover operator enabled improved results compared OX .

Keywords: Crossover Operator, Genetic algorithm, NP-complete, Order Crossover, Traveling salesman problem.

I. INTRODUCTION

The Traveling Salesman problem (TSP) is one of the benchmark and old problems in Computer Science and Operations Research. It can be stated as: A network with „n nodes (or cities), with 1 node as main node and a travel cost (or distance, or travel time etc.,) matrix C= [Cij] of order n associated with ordered node pairs (i, j) is given. The problem is to find a least cost Hamiltonian cycle. On the basis of the structure of the cost matrix, the TSPs are classified into two groups – symmetric and asymmetric. The TSP is symmetric if Cij = Cji, ∀i,j and asymmetric otherwise. For an n-city asymmetric TSP, there are $(n-1)!$ possible solutions for all n > 2, one or more of which gives the minimum cost. For an n-city symmetric TSP, there are $(n-1)! / 2$ possible solutions for all n > 2 along with their reverse cyclic permutations having the same total cost. In either case the number of solutions becomes extremely large for even moderately large n so that an exhaustive search is impracticable. There are mainly three reasons why TSP has been attracted the attention of many researchers and remains an active research area. First, a large number of real-world problems can be modeled by TSP. Second, it was proved to be NP-Complete problem [1]. Third, NP-Complete problems are intractable in the sense that no one has found any really efficient way of solving them for large problem size. Also, NP-complete problems are known to be more or less equivalent to each other; if one knew how to solve one of them one could solve all of them. The TSP finds application in a variety of situations such as automatic drilling of printed circuit boards and threading of scan cells in a testable VLSI circuit [2], X-ray crystallography [3], etc. The methods that provide the exact optimal solution to the problem are called exact methods. An implicit way of solving the TSP is simply to list all the feasible solutions, evaluate their objective function values and pick out the best. However it is obvious that this "exhaustive search" is grossly inefficient and impracticable because of vast number of possible solutions to the TSP even for problem of moderate size. Since practical applications require solving larger problems,hence emphasis has shifted from the aim of finding exactly optimal solutions to TSP, to the aim of getting, heuristically, „good solutions in reasonable time and „establishing the degree of goodness. Genetic algorithm (GA) is one of the best heuristic algorithms that have been used widely to solve the TSP instances. In this paper, A Modified Order Crossover operator is proposed and accordingly a genetic algorithm based on Modified Order Crossover is developed for solving the TSP. This paper is organized as follows: Section 2 describes a genetic algorithm Methodology. Section 3 describes related work on the problem. Section 4 describes proposed genetic algorithm based on Modified Order Crossover operator. Section 5 describes computational experiments for Modified Order Crossover and some of the existing crossover operators. Finally, Section 6 presents conclusion and concluding remarks.

II. GENETIC ALGORITHM METHOLOGY

Genetic algorithms (GAs) are based essentially on mimicking the survival of the fittest among the species generated by random changes in the gene-structure of the chromosomes in the evolutionary biology [4]. In order to solve any real life problem by GA, two main requirements are to be satisfied: (a) a string can represent a solution of the solution space, and (b) an objective function and hence a fitness function which measures the goodness of a solution can be constructed / defined. A simple GA works by randomly generating an initial population of strings, which is referred as gene pool and then applying operators to create new, and hopefully, better populations as successive generations. The first operator is selection where strings are copied to the next generation with some probability based on their objective function value. The second operator is crossover where randomly selected pairs of strings are mated, creating new strings. The third operator, mutation, is the occasional random alteration of the value at a string position. The crossover operator is the most powerful process in the GA search. Mutation diversifies the search space and protects from loss of genetic material that can be caused by reproduction and crossover. So, the probability of applying mutation is set very low, whereas the probability of crossover is set very high.The search of the solution space is done by creating new chromosomes from old ones. The most important search process is crossover. Firstly, a pair of parents is randomly selected from the mating pool [5]. Secondly, a point, called crossover site, along their common length is randomly selected, and the information after the crossover site of the two parent strings are swapped, thus creating two new children. Of course, this basic crossover method is not applicable for the TSP as it may repeat some of the cities lying on different sides of parents and may ignore some of the cities also, so new crossover operators are designed for TSP problems.

III. RELATED WORK

Since the crossover operator plays a vital role in GA, so many crossover operators have been proposed for the TSP.Gold berg and Lingle [6] defined an operator called PMX (partially mapped crossover). This operator first randomly selects two cut points on both parents. In order to create an offspring, the substring between the two cut points in the first parent replaces the corresponding substring in the second parent. Then, the inverse replacement is applied outside of the cut points, in order to eliminate duplicates and recover all cities. Consider two possible codings of a tour of eight cities, A1 and A2.

A1 – 3 5 1 2 7 6 8 4

A2 – 1 8 5 4 3 6 2 7

Two positions are determined randomly along the A1 coding. The actual cities located between the same positions along A2. For example, if the positions three and five are chosen, the sub-coding alone A1 1-2-7 and the sub-coding along A2 is 5-4-3. Each of these cities is then exchanged, leading to the new tours A1 and A2.

A1 – 7 1 5 4 3 6 8 2

A2 – 5 8 1 2 7 6 4 3

This PMX operator was the first attempt to apply GA to the TSP, in which they found near-optimal solutions to a wellknown 33-node problem. The OX (ordered crossover) operator developed by Davis [7] builds offspring by choosing a subsequence of a tour from one parent and preserving the relative order of nodes from the other parent. For example, if two parents are selected as below for crossover:

Parent1: 4 3 6 2 5 1 9 7 8

Parent2: 6 4 7 1 5 2 9 8 3

and let crossover sites are chosen as 2 and 6. PMX produce following child's:

Child1 : 4 3 7 1 5 2 9 6 8

Child2 : 7 4 6 2 5 1 9 8 3

while OX produce following child's:

Child1 : 3 6 7 1 5 2 9 8 4

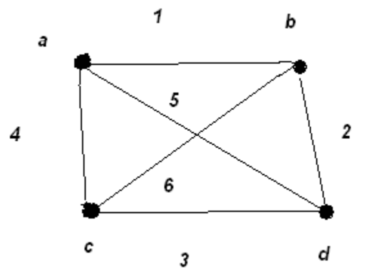
Child2 : 4 7 6 2 5 1 9 8 3

Another crossover operator, named CX (cycle crossover) operator was proposed by Oliver et al. [8], where offspring are built in such a way that each node (and its position) comes from one of the parents. Whitley et al. [9] proposed edge recombination crossover (ERX) operator that uses an „edge map to construct an offspring that inherits as much information as possible from the parent structures. This edge map stores all the connections from the two parents that lead into and out of a node. A crossover operator based on the conventional N-point crossover operator, named as generalized N-point crossover (GNX), was proposed by Radcliffe and Surry [10]. Poona and Carter [11] developed a tie break crossover (TBX), which was then modified by Choi et al. [12] by combining PMX and TBX operators. Moon et al. [13] proposed a new crossover operator named Moon Crossover (MX), which mimics the changes of the moon such as waxing moon → half moon → gibbous → full moon. As reported, performance of MX operator and OX operator is almost same, but OX never reached an optimal solution for all trials.

IV. PROPOSED GENETIC ALGORITHM

A. The path representation: As opposed to the ordinal representation, the path representation is a natural way to encode TSP tours. However, a single tour can be represented in 2N distinct ways, because any city can be placed at position 1, and the two orientations of the tour a r e the same for a symmetric problem. Of course, the factor N can be removed by fixing a particular city at position 1 in the chromosome.
The crossover operators based on this representation typically generate offspring that inherit either the relative order or the absolute position of the cities from the parent chromosomes. We will now describe these operators in turn. In each case, a single offspring is shown. However, a second offspring can be easily generated by inverting the roles of the parents.

B . Problem Representation: To find minimum path for traveling person the cities are represented as graph with N vertices. These are represented in computer memory using an array of NXN elements. An element in this array with subscripts I,J represent the distance between Ith and Jth node. There is another constraint on this specification that the distance between I and J is equal to distance between J and I if J not equal to I. The distance between I and J equal to 0 if I equal to J.



	a	b	c	d
a	0	1	4	5
b	1	0	6	2
c	4	6	0	3
d	5	2	3	0

Fig. 1. Path Matrix

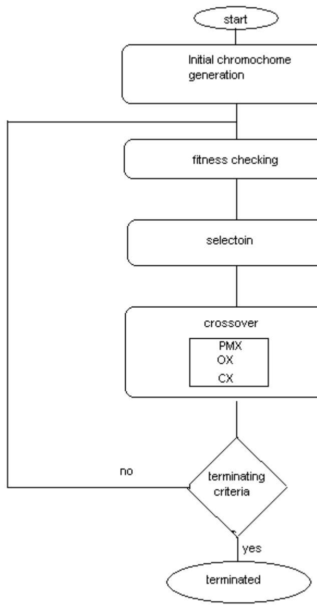


Fig.2 Simple Genetic Algorithm

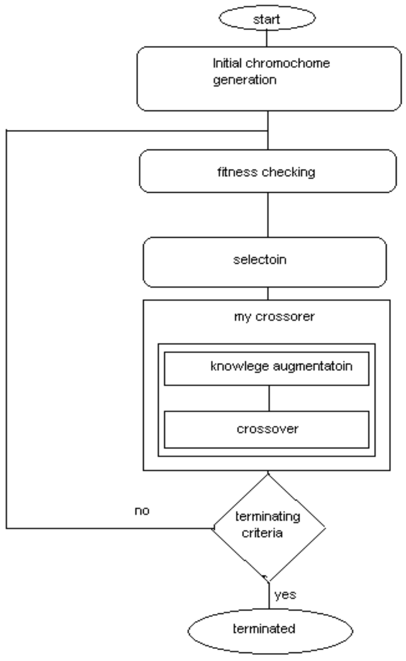


Fig. 3 Knowledge Augmented Genetic Algorithm

C. Order Crossover (OX) Davis (85), Oliver et al. (87): This crossover operator extends the modified crossover of Davis by allowing two cut points to be randomly chosen on the parent chromosomes. In order to create an offspring, the string between t h e two cut points in the first parent is first copied to the offspring. Then, the remaining positions are filled by considering the sequence of cities in the second parent, starting after the second cut point (when the end of the chromosome is reached, the sequence continues a t position 1).

In Figure 1, the substring 564 in parent 1 is first copied to the offspring (step 1). Then, the remaining positions are filled one by one after the second cut point, by considering the corresponding sequence of cities in parent 2, namely 57814236 (step 2). Hence, city 5 is first considered to occupy position 6, but it is discarded because it is already included in the offspring. City 7 is the next city to be considered, and it is inserted at position 6. Then, city 8 is inserted at position 7, city 1 is inserted at position 8, city 4 is discarded, city 2 is inserted at position 1, city 3 is inserted at position 2, and city 6 is discarded.

parent 1 : **1 2 I 5 6 4 I 3 8 7**
parent 2 : 1 4 I 2 3 6 I 5 7 8

---

offspring
(step 1) : - - **5 6 4** - - -
(step 2) : 2 3 **5 6 4** 7 8 1
Fig. 4 The order crossover.

Clearly, OX tries to preserve the relative order of the cities in parent 2, rather than their absolute position. In ox, the offspring does not preserve the position of any city in parent 2. However, city 7 still appears before city 8, and city 2 before city 3 in the resulting offspring.

D. Modified order crossover: In order to improve th efficiency of order crossover the minimum edge is detected from the second chromoshome.select first node of this edge as first crossover point and select 2nd crossover point after first randomly.the minimum edge is selected as it has higher probability to participate in result.
parent 1 : **1 2 5 6 4 3 8 7**
parent 2 : 1 4 2 3 6 5 7 8

Step 1 detect shortest edge from second parent
e.g 3,6 from second parent
Step 2 select second crossover point randomly after first for e.g 5
crossover points are before 3 and after 5 in second chromoshome
Step 3 apply order crossover
offspring
Step 3.1 : - - - 3 6 5 - -
Step 3.2 : 1 2 5 3 6 5 8 7
Step3.3: 1 2 4 3 6 5 8 7
Fig. 5 Knowledge augmented order crossover

E. Sample Dataset: To take result the Ga with ox and cyclic mox are tested on different size data set. An adjency matrix is used to store path length between two cities results and sample dataset are given below.
{0, 3, 4, 5, 1, 6, 8, 3, 8, 9, 2, 6, 6, 4, 6, 2, 8, 3, 8, 9, }
{3, 0, 1, 4, 9, 5, 7, 1, 4, 2, 8, 9, 3, 5, 7, 8, 3, 1, 4, 6, }
{4, 1, 0, 7, 8, 1, 5, 8, 9, 1, 4, 2, 7, 4, 8, 2, 5, 7, 1, 4, }
{5, 4, 7, 0, 2, 4, 5, 2, 6, 7, 3, 6, 2, 5, 2, 4, 1, 5, 7, 8, }
{1, 9, 8, 2, 0, 2, 4, 6, 7, 2, 4, 5, 7, 9, 1, 3, 5, 4, 8, 2, }
{6, 5, 1, 4, 2, 0, 5, 8, 2, 6, 9, 4, 5, 3, 5, 4, 7, 5, 8, 1, }
{8, 7, 5, 5, 4, 5, 0, 5, 7, 8, 2, 4, 6, 7, 8, 2, 4, 9, 1, 6, }
{3, 1, 8, 2, 6, 8, 5, 0, 3, 6, 2, 7, 2, 7, 1, 5, 8, 1, 5, 6, }
{8, 4, 9, 6, 7, 2, 7, 3, 0, 8, 2, 5, 3, 7, 1, 8, 9, 2, 5, 3, }
{9, 2, 1, 7, 2, 6, 8, 6, 8, 0, 6, 7, 8, 2, 6, 4, 8, 2, 3, 6, }
{2, 8, 4, 3, 4, 9, 2, 2, 2, 6, 0, 1, 4, 6, 8, 8, 2, 6, 7, 8, }
{6, 9, 2, 6, 5, 4, 4, 7, 5, 7, 1, 0, 9, 3, 2, 5, 6, 3, 7, 8, }

{6, 3, 7, 2, 7, 5, 6, 2, 3, 8, 4, 9, 0, 2, 6, 9, 3, 6, 8, 4, }
{4, 5, 4, 5, 9, 3, 7, 7, 7, 2, 6, 3, 2, 0, 3, 2, 5, 6, 7, 8, }
{6, 7, 8, 2, 1, 5, 8, 1, 1, 6, 8, 2, 6, 3, 0, 1, 1, 4, 4, 5, }
{2, 8, 2, 4, 3, 4, 2, 5, 8, 4, 8, 5, 9, 2, 1, 0, 2, 6, 7, 9, }
{8, 3, 5, 1, 5, 7, 4, 8, 9, 8, 2, 6, 3, 5, 1, 2, 0, 2, 4, 5, }
{3, 1, 7, 5, 4, 5, 9, 1, 2, 2, 6, 3, 6, 6, 4, 6, 2, 0, 6, 7, }
{8, 4, 1, 7, 8, 8, 1, 5, 5, 3, 7, 7, 8, 7, 4, 7, 4, 6, 0, 5, }
{9, 6, 4, 8, 2, 1, 6, 6, 3, 6, 8, 8, 4, 8, 5, 9, 5, 7, 5, 0, }

*F. Implementatiom issues:*To apply a GA for any optimization problem, one has to think a way for encoding solutions as feasible chromosomes so that the crossovers of feasible chromosomes result in feasible chromosomes. The techniques for encoding solutions vary by problem and, involve a certain amount of art. For the TSP, a solution is typically represented by chromosome of length as the number of nodes in the problem.

i. Each gene of a chromosome takes a label of node such that no node can appear twice in the same chromosome. Thereare mainly two representation methods for representing tour of the TSP: adjacency representation and path representation. adjency representation which simply lists the label of nodes
sidered in this model
.
ii. Fitness function: The GA is usually used for optimization problems and TSP is an optimization problem; a function f(x) which calculates cost of the tour represented by the chromosome „x is considered as fitness function for that chromosome.

iii. Selection operator: In selection process, chromosomes are copied into next generation mating pool with a probability associated with value. By assigning to next generation a higher portion of the highly fit chromosomes, selection mimics the Darwinian survival-of-the-fittest in the natural world.
Iv .In natural population, fitness is determined by a creatures ability to survive predators, pestilence, and other obstacles to adulthood and subsequent reproduction. In this phase no new chromosome is produced. The commonly used reproduction operator is the proportionate reproduction operator, where a string is selected for the mating pool with a probability proportional to its fitness value. Roulette Wheel selection method is used as selection operator in this study.

v. Mutation operator: The mutation operator randomly selects a position in the chromosome and changes the corresponding allele, thereby modifying information. The need for mutation comes from the fact that as the less fit members of successive generations are discarded; some aspects of genetic material could be lost forever. By performing occasional random changes in the chromosomes, GA ensure that new parts of the search space are reached, which selection and crossover alone couldnt fully guarantee. In doing so, mutation ensures that no important features are prematurely lost, thus maintaining the mating pool diversity. For the TSP, the classical mutation operator does not work. For this investigation, the reciprocal exchange mutation that selects two nodes randomly and swaps them has been considered.

vi. Replacement: After performing crossover and mutation operation replacement method is used for selecting next generation population. The replacement of GA considers two kinds of chromosomes for the next generation: (1) parents in current population of size m, and (2) offspring that are generated by crossover of size m. The (μ,λ) replacement method that replace chromosomes in (1) by (2) completely is considered. In this case, all the μ parents in the present generation are replaced with next generation.

vii. Control parameters: There are a lot of parameters that govern the GA search process. Some of them are:
(a) Population size: - It determines how many chromosomes and thereafter, how much genetic material is available for use during the search. If there is too little, the search has no chance to adequately cover the space. If there is too much, the GA wastes time evaluating chromosomes.
(b) Crossover probability: - It specifies the probability of crossover occurring between two chromosomes.
(c) Mutation probability: - It specifies the probability of doing mutation.
(d) Termination criteria: - It specifies when to terminate the genetic search.

V. COMPUTATIONAL EXPERIMENTS

Table No.1: Result of ox Crossover and Myox crossover

Sample no.	**ox, No. of iteration**	**Shortest path**	**Myox, No of iteration**	**Shortest path**
1	1	86	1	87
2	1	348	3	339
3	1	1727	1	2225
4	1	605	2	610
5	2	2432	2	2388

VI. CONCLUSION

Table No.2 : Result Analysis

Type of Crossover	**Simple**		**Knowledge Augmented**		**Result Analysis**	
	Shortest Path average	No. of iteration average	Shortest Path average	No. of iteration average	Shortest Path	Convergence Performance
Cx	880	2.2	685	4.7	Improved	Improved

A modified OX crossover operator (MOX) for a genetic algorithm for the Traveling Salesman Problem (TSP) has been proposed. In terms of quality of the solution, for all the instances MOX found to be better in terms of solution quality and the overall time taken by the algorithm. Among all the operators, experimental results show that Modified OX crossover operator (MOX) is better than the OX in terms of quality of solutions as well as execution time.Any local search technique is not used to improve the solution quality. So an incorporation of good local search technique to the algorithm may solve exactly more instances, which is under consideration.

REFERENCES

[1] Papadimitriou C.H. and Steglitz K. (1997), *"Combinatorial Optimization: Algorithms and Complexity"*. Prentice Hall of India Private Limited, India.

[2] Ravikumar C.P. (1992) *"Solving Large-scale Travelling Salesperson Problems on Parallel Machines"*. Microprocessors and Microsystems 16(3), pp. 149-158.

[3] Bland R.G. and Shallcross D.F. (1989), *"Large Travelling Salesman Problems arising form Experiments in X-ray Crystallography: A Preliminary Report on Computation"*. Operations Research Letters 8, pp. 125-128.

[4] Goldberg D.E. (1989), *"Genetic Algorithms in Search, Optimization, and Machine Learning"*. Addison-Wesley, New York.

[5] Deb K. (1995) *"Optimization For Engineering Design: Algorithms And Examples"*. Prentice Hall Of India Pvt. Ltd., New Delhi, India.

[6] Goldberg D.E. and Lingle R. (1985), *"Alleles, Loci and the Travelling Salesman Problem"*. In J.J.Grefenstette (ed.) Proceedings of the 1st International Conference on Genetic Algorithms and Their Applications. Lawrence Erlbaum Associates, Hilladale, NJ.

[7] Davis L. (1985), *"Job-shop Scheduling with Genetic Algorithms"*. Proceedings of an International Conference on Genetic Algorithms and Their Applications, pp. 136-140, 1985.

[8] Oliver I.M., Smith D. J. and Holland J.R.C (1987), *"A Study of Permutation Crossover Operators on the Travelling Salesman Problem"*. In J.J.

[9] Whitley D., Starkweather T. and Shaner D. (1991), *“The Traveling Salesman and Sequence Scheduling:Quality Solutions using Genetic Edge Recombination”*. In L. Davis (Ed.) Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York, pp. 350-372.

[10] Radcliffe N.J. and Surry P.D. (1995), *“Formae and variance of fitness”*. In D. Whitley and M. Vose (Eds.) Foundations of Genetic Algorithms Morgan Kaufmann, San Mateo, CA, pp. 51-72.

[11] Poon P. and Carter J. (1995), *“Genetic algorithm crossover operations for ordering applications”*. Computers and Operations Research Grefenstette (ed.). Genetic Algorithms and Their Applications: Proceedings of the 2nd International Conference on Genetic Algorithms. Lawrence Erlbaum Associates, Hilladale, NJ, 1987.22, pp.135–47.

[12] Choi I., Kim S. and Kim H. (2003), *"A genetic algorithm with a mixed region search for the asymmetric traveling salesman problem"*. Computers & Operations Research 30, pp. 773 – 786.

[13] Moon C., Kim J., Choi G. and Seo Y (2002), *"An efficient genetic algorithm for the traveling salesman problem with precedence constraints"*. European Journal of Operational Research 140, pp. 606-617.

[14] TSPLIB, http://www.iwr.uni-heidelberg.de/iwr/comopt/software/TSPLIB95/