A Study of the Genetic Algorithm Parameters for solving Multi-Objective Travelling Salesman Problem

Romit S Beed
Department of Computer Sc.
St. Xavier's College
(Autonomous), Kolkata
rbeed@yahoo.com

SunitaSarkar
Department of Computer Sc.
Assam University,
Silchar
sunitasarkar@rediffmail.com

Arindam Roy
Department of Computer Sc.
Assam University
Silchar
arindam_roy74@rediffmail.com

Shubham Chatterjee
Department of Computer Sc.
St. Xavier's College
(Autonomous), Kolkata
shubham.chatterjee94@gmail.com

Abstract—The objective of this work is to present a solution to a multiple-objective optimization problem using genetic algorithms (GA). Generally the objectives (minimizing cost, maximizing performance, reducing carbon footprints, maximizing profit) are conflicting for multiple-objective problems, hindering concurrent optimization of each objective. A bi-objective traditional combinatorial optimization of Travelling Salesman Problem is undertaken named as the Multi-Objective Travelling Salesman Problem (MTSP). The two objectives are minimization of the distance travelled by the salesman and minimization of the time taken to travel. The purpose of this paper is to model the problem as a single objective optimization problem using the weighted sum method of modeling the objective function and using a Genetic Algorithm to see how the distance and time values change with the changes in weights assigned to the two objectives. The probability of mutation, the initial population and the number of generations have been varied to study its effect on the fitness value.

Keywords—Genetic Algorithms (GA); Travelling Salesman Problem (TSP); Tournament Selection, Initial Population, Crossover, Mutation, Fitness.

I. INTRODUCTION

Genetic Algorithms are computer algorithms that obtain favorable results to a problem within a huge set of likely possible results [1]. Introduced in the 1960s by John Holland and his team at the University of Michigan, genetic algorithm alters a population of distinct objects, each having a relevant fitness value, into a new generation of the population using the Darwinian evolution principle and survival of the fittest logic; involving genetic operations like crossover and mutation.GA tries to obtain the most optimal or the best solution to the problem under restrictions through genetic refinement across generations. The various applications of GAs include optimization, automatic programming, machine learning, data analysis, strategy planning, Travelling Salesman Problem, henceforth referred to as TSP and sequence scheduling and designing social systems [2]. Due to operative ease and extensive applicability, GA plays an important role in computational optimization and operations research.

A Travelling Salesman Problem is an 'NP-hard' problem. It is defined as follows: Given a set of nodes, and distances between each pair of nodes, the salesman must find a shortest possible tour that visits each node exactly once and that minimizes the total distance travelled. The mathematical model of TSP is described as follows: Given a set of nodes=$\{C_1,C_2,\ldots.C_n\}$, the distance of each pair of nodes is $(C_{i,j})$, the problem is to find a route $[C_1,C_2,\ldots.C_n]$ that visits each node exactly once and makes $f(x) = \min \Sigma d_{ij}x_{ij}$ to have minimum value where f(x) is a fitness function which represents the quality of the chromosome, d_{ij} is the distance between vertices i and j, and the x_{ij}'s are the decision variables [3]. Fitness function is then used for assessing the population and selecting the best quality of individuals for generating offspring. Presently TSP is gaining importance in artificial intelligence, computational mathematics and optimization. Multi-objective models are gaining popularity these days in various multifaceted engineering optimization situations. These objectives may at times conflict with each other, and optimizing a problem with respect to one objective may give undesired outputs with respect to the other objectives. A better approach to such a multi-objective problem is to explore a number of solutions, each of which obeys the objectives at a certain permissible level without being subjugated by any other solution.

Typical examples of multiple-objectives are minimization of cost, maximization of performance, maximization of reliability, minimization of time, minimization of travel distance etc. There are two basic techniques to multiple-objective optimization. The first one is to merge the different objective functions into a single merged function. This merged objective function can be solved by utility theory method or the weighted sum method. There is however a problem - the accuracy in selecting the weights or utility functions to portray the correct situation. The other approach is to obtain a complete Pareto optimal solution set which generates a set of values that are non-dominated with respect to each other.

Genetic Algorithms are appropriate to solve multi-objective optimization problems since it is based on a population concept. A broad single-objective GA can be transformed to find a set of multiple non-dominated solutions in a single run. The capability of GA to concurrently explore different areas of a solution space allows the system to obtain a varied set of solutions for hard problems with non-convex, disjointed and multi-modal solutions spaces.

978-1-5386-2924-6/17 $31.00 © 2017 IEEE
DOI 10.1109/ICIT.2017.49

The weighted sum method is the most widely used traditional method of transforming a number of objectives into a single objective by assigning weights. The weight of an objective is generally selected in proportion to an objective's relative importance in the equation. Objectives need to scaled appropriately to bring them to similar magnitude. Once the objectives are normalized, a composite objective function can be generated by summing the weighted normalized objectives and is reduced to a single-objective optimization problem [4] as follows:

$$\min z = w_1 z_1'(x) + w_2 z_2'(x) + \cdots + w_k z_k'(x) \qquad (1)$$

where $z_i'(x)$ is the normalized objective function $z_i(x)$ and $\sum w_i = 1$. This is a priori approach since the weights are expected to be provided by the users. The biggest positive feature of the weighted sum approach is its straightforward operation. As a solitary objective is used in fitness allotment, a single objective GA can be used with minor changes. Secondly, this procedure efficiently finds the optimal solution However, the negative feature of using weighted sum procedure is that not all Pareto-optimal results can be inspected when the true Pareto front is non-convex. The two objectives that are considered here are minimization of the distance and reduction of the time.

The road map of the paper is as follows: Section II is on Literature Survey. Section III expounds Multi objective Optimization Model. Experiments and results are presented in Section IV and Section V concludes the paper

II. LITERATURE REVIEW

Heinrich Braun presented a genetic algorithm for solving the traveling salesman problem. He established that the genetic algorithm will finally converge. He proposed the island model, where numerous populations each secluded on an island are optimized by the GA until they degenerate. The degeneration is then neutralized by reorganizing the population on each such island by including new entities of neighboring island leading to evolution [5]. It has been proposed by Akshatha et al. [6] that GAs aim to provide quality results for the traveling salesman problem, though the quality depends on the problem encoding and the choice of the crossover and mutation operators. The procedures based on heuristics for encoding edges of the tour provide the most optimal results. However the major challenge that persists is the difficulty to retain the structure between the parent chromosomes and the child chromosomes and generate a valid tour.

K. Bryant [7] mentioned that GAs are an optimization procedure built on natural evolution focusing on the survival of the fittest idea mapped into a exploration algorithm which offers a procedure of exploring, though not all possible results, to get a respectable result. Initially starting off by a guess, the process slowly produces the fittest solutions to generate the next generation solutions superior than earlier generation. Gerard Reinelt suggested that Greedy Algorithms [8] are a favorable method to obtain feasible solution to TSPs. The solutions may not always be the best. The Nearest Neighbor algorithm picks a source city and then reaches the nearest city without generating a cycle. This is repeated for all the nodes. Although the process is simple and efficient, it may not always choose the shorter routes due to its "greedy" nature. Dwivedi et al. [9] proposed a new crossover method called Sequential Constructive Crossover (SCX) for use in GA in TSP domain which uses the most favorable edges of the parent's structure to generate the new offspring. Omar et al. suggested an improved GA [10] using a new crossover operator (Swapped Inverted Crossover), population reformulation operator, multi-mutation operator, partial local optimal mutation operator and rearrangement operator were used to solve the TSP and produce a better performance.

Chetan et al. provided a comparative study on varied parent selection methods: Roulette Wheel, Elitism and Tournament selection [11]. The roulette wheel selection method chooses the individual proportional to its fitness value; tournament selection pairs every individual with another individual in the population randomly and chooses the better chromosome among the two. In Elitism method, the individuals were selected based on their fitness value. The paper concluded that all the three selection methods gave similar solution when the population size is small but when the population size is large Elitism method gave the better result. Multi objective optimization problems have two types of solution: Nash equilibrium and Tchebycheff compromise solution [12]. The first solution focuses on constituents of the vector-valued condition function competing with each other whereas the latter pertains to situations of cooperation, based on the notion of Pareto-dominance. Presently, traffic congestion is a cause of concern. The road network can be considered as a graph. Finding the most optimal path between the origin-destination pair is a big challenge. Popular algorithms like Dijikstra's, Bellman Ford, etc. were generally used for finding shortest route in a road network. However, the shortest route would comprise of several measures like travelling time, distance and congestion [13]. Hence there exist multiple objectives for the shortest path problem. Thus the problem can be defined as multi objective shortest path problem.

III. MULTI OBJECTIVE OPTIMIZATION MODEL

The main purpose of this model is to devise a Genetic Algorithm to solve a Multi-Objective Travelling Salesman Problem. There are two objective functions which one wishes to minimize - Distance and Time - the influence of two factors on the solution of the TSP. Hence, it transforms from the traditional single-objective TSP to a multi-objective TSP problem. The major components of GAs are: populations of chromosomes, fitness based selection, crossover to generate new offspring and random mutation of these offspring. The chromosomes in GAs represent the space of candidate solutions. The fitness function assigns a score to each chromosome in the current population. Selection of the higher value (fitter) chromosomes is based on the fitness values; these are then chosen to reproduce. To improve the quality, the selection process is improved by elitism - a method, which first copies the top chromosome of one generation to the new population and then includes the rest of the population. Crossover is the procedure of uniting the bits of one chromosome with those of another to generate an offspring that inherits characteristics of both parents. Mutation is performed after crossover to maintain genetic diversity from one generation of a population of chromosomes to the next. Each iteration is called a generation. Through the algorithm, GA regulates which chromosomes should be alive, which should reproduce, and which should perish.

So, the basic steps of GA are as follows:

1. Initial Population: Randomly produce a population of N chromosomes.
2. Fitness: Calculate the fitness value of all chromosomes.
3. Create a new population:
   a. Selection: Choose 2 chromosomes from the population.
   b. Crossover: Perform crossover on the 2 chromosomes selected.
   c. Mutation: Perform mutation on the chromosomes.
4. Interchange: Interchange the current population with the new population.
5. Termination condition Test: Test whether the end condition is met. If yes, stop else go to Step 2.

3.1 Initial Population

The initial population is a collection of an ordered list of artificial chromosomes where every chromosome represents a sequence of nodes. Each gene of a chromosome takes a label of node such that no node can appear twice in the same chromosome. There are mainly two representation methods for representing tour of the TSP – adjacency representation and path representation; here path representation is selected for a tour. Let {1, 2, 3, 4, 5, 6, 7, 8, 9} be the labels of nodes where 1 represents the starting point, then a tour {1→3→8→2→5→7→4→6→9→1} may be represented as

1	3	8	2	5	7	4	6	9	1

Figure 1: Chromosome

Each chromosome, a string of non-repeating nodes, is a solution for the TSP where the traveller travels the nodes in the sequence. The distance between two nodes has been calculated using the Euclidean Distance formula. The distance between two nodes with coordinates (x_1, y_1) and $x_2, y_2)$ is given by

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (2)$$

3.2 *Fitness*

The weighted sum method has been used to formulate the fitness function of the Genetic Algorithm. Solving a problem with the objective function stated earlier (equation 1) for a given weight vector w = {w_1, w_2,…,w_k} yields a single solution. For multiple solutions, the problem must be solved multiple times with different weight combinations. For the two parameters, distance and time, the weights have been varied in steps of 0.1 ranging from 0.1 to 0.9. This variation of weights has been plotted against distance, time and the overall fitness for a particular population. This is repeated for various values of the GA parameters such as population size, number of generations and mutation rate and graphs have been plotted using MATLAB to analyze the results.

Normalization of Objectives - There is a need to normalize the objectives while formulating the composite objective function using the weighted sum method. Since distance is in the range of kilometers, time is in the range of hours there is this need to normalize them to a common range [0, 1] for optimization. The maximum value among the set of values for a particular attribute is ‘b’ and the minimum value is ‘a’, in order to normalize the value ‘x’, the following function is used:

$$f(x) = \frac{x-a}{b-a} \quad (3)$$

3.3.1 Selection

Tournament Selection method has been implemented to select the chromosomes. Two tournaments have been run, one each to select the two parents for crossover and mutation. Tournament selection involves running several “tournaments” among a few selected chromosomes chosen at random from the population. The winner of each tournament (the one with the best fitness) is selected for crossover. A set of random 2 routes are selected for participation in one tournament and another set of 2 are selected for participation in the other. The fitness value is found. The route of minimum fitness value wins in the tournament and is selected as the parent.

3.3.2 Crossover (Minimal Weight Variable Order Selection Crossover)

One crossover method that is able to produce a valid route is ordered crossover. In this crossover method, one selects a subset from the first parent, and then adds that subset to the offspring. Any missing values are then adding to the offspring from the second parent in order that they are found. As seen from Figure 2, a subset of the route is taken from the first parent (6, 7, 8) shaded gray and added to the offspring's route. Next, the missing route locations are added in order from the second parent. The first location in the second parent's route is 9 which is not in the offspring's route so it is added in the first available position. The next position in the parent’s route is 8 which is in the offspring's route so it is skipped. This process continues until the offspring has no remaining empty values.

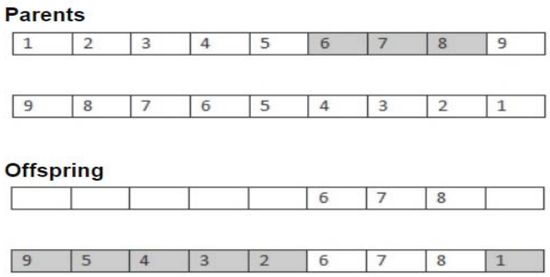

Figure 2: Ordered Crossover

If implemented correctly, the end result should be a route which contains all the positions of the parents. An improvement over the ordered crossover is the Minimal Weight Variable Order Selection Crossover [14]. Originally a fixed number of varying swaths with successive alleles in definite locations are taken for next step from parent 1. The weight calculation is implemented for every designated swath and the top swath with least weight (fitness) is chosen and shifted to the offspring. Two replicas of the child are produced and the residual locations are completed from parent 2 in onward and backward directions. Fitness of both children are obtained separately and the child with the least weight is retained.

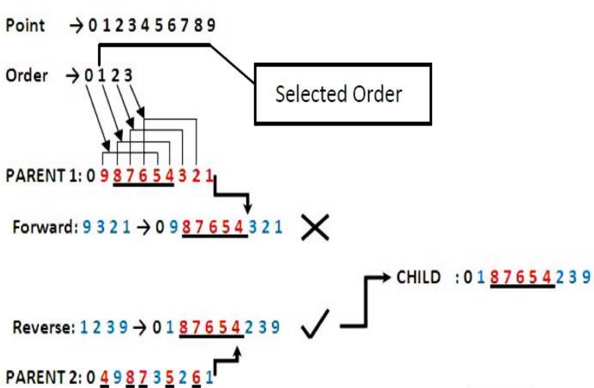

Figure 3: Variable Weight Minimal Order Crossover [14]

Comparing MWVOSX with OX, this gives better performance for smaller sized instances, but with growing size, it becomes slower.

3.3.3 Mutation

The mutation process simply shuffles the route, it does not add or delete a position from the route, otherwise it would risk generating an invalid solution. The mutation method used is swap mutation. Two locations in the route are selected at random and then their positions are simply swapped. As seen from Figure 4, positions 3 and 8 were switched creating a new list with the same values, just in a different order. Because swap mutation is only swapping pre-existing values, it will never create a list which has missing or duplicate values when compared to the original.

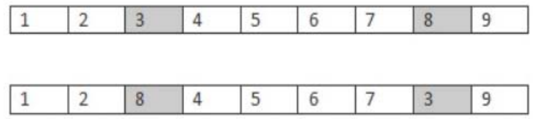

Figure 4: Swap Mutation

IV. Experiments and Results

In this paper, experiments were conducted to study how distance, time and fitness values change against the varying weights for the following three parameters: (a) number of generations (b) population size and (c) mutation rate or probability of mutation (pm). JAVA has been used to write and implement the program. To study the results and infer, normalized fitness values have been plotted for varying number of generations, population size and mutation rate or probability of mutation(pm).

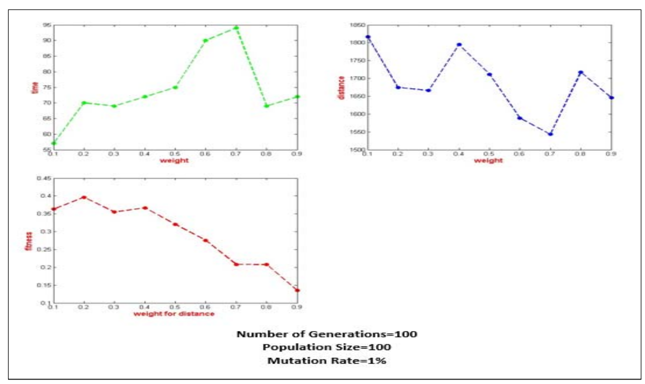

Figure 5: Plot of time, distance and fitness against weight(No of Generations-100, Population Size-100, Mutation Rate-1%.)

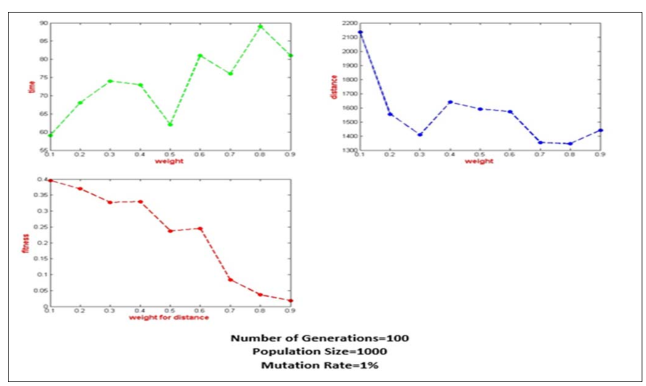

Figure 6: Plot of time, distance and fitness against weight (No of Generations-100, Population Size-1000, Mutation Rate -1%.)

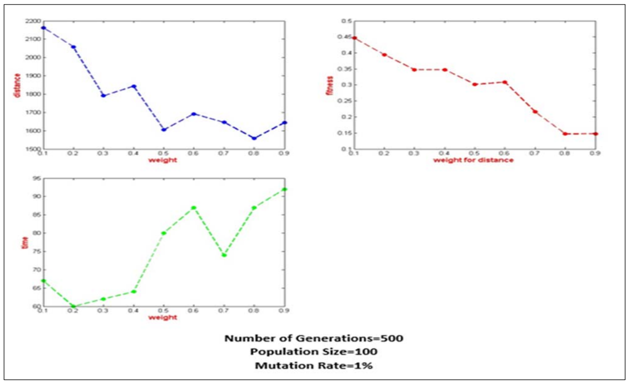

Figure 7: Plot of time, distance and fitness against weight (No of Generations-500, Population Size-100, Mutation Rate-1%.)

It is observed from figs. 5, 6 and 7 keeping the number of generations=100, mutation rate or pm fixed at 1%, on increasing the population size from 100 to 1000 (figs. 5 and 6), the fitness value seems to converge to a lesser value with increasing weights, indicating the desirability of a greater initial population. Increasing the number of generations from 100 to 500 (for fixed population size=100 and mutation rate=1%) seems to have no significant effect on fitness with increasing weights (figs 5 and 7).

To study the effect of the number of generations on the fitness value under different mutation rates with changing weights of distance and time, the following graphs (figs. 8, 9 and 10) were plotted.

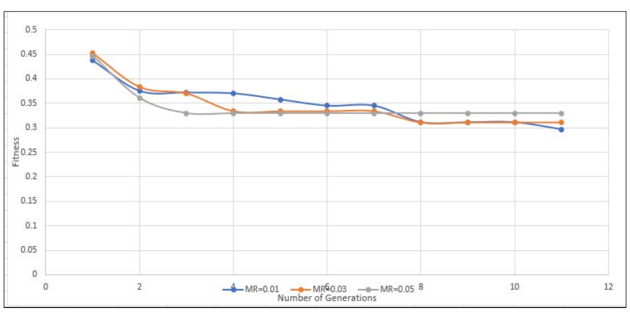

Figure 8: Plot of fitness vs. no of generations for w_1=0.1 for distance and w_2=0.9 for time. [Mutation rates=1%, 3% and 5%]

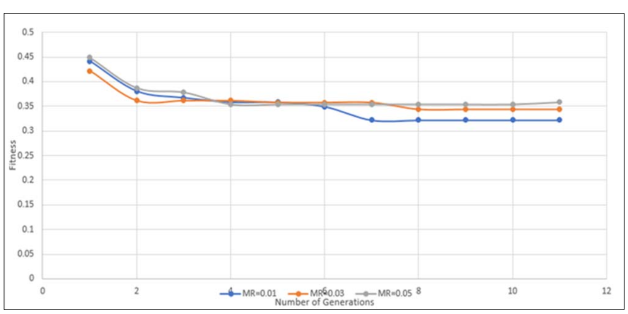

Figure 9: Plot of fitness vs. no of generations for w_1=0.5 for distance and w_2=0.5 for time. [Mutation rates=1%, 3% and 5%]

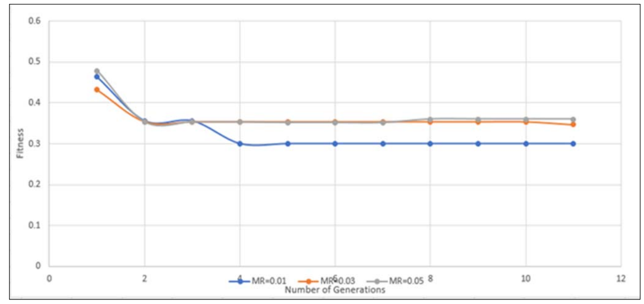

Figure 10: Plot of fitness vs. no of generations for w_1=0.9 for distance and w_2=0.1 for time. [Mutation rates=1%, 3% and 5%]

It is observed from the figs 8, 9 and 10 that, when the weights corresponding to distance were increased from 0.1 to 0.9 and simultaneously that corresponding to time were decreased from 0.9 to 0.1, it was observed that for lower mutation rate (here 1%), the decrease in fitness value is consistently the least. This indicates that lower mutation rates are expected to perform better with respect to fitness value with increasing number of generations under varying weighing patterns of distance and time. Table 1 provides the fitness values (approximated upto 5 places of decimal) for four different mutation rates [1%, 3%, 5% and 10%] for the first 51 generations in steps of 2. A steady decrease is seen in all cases.

Table 1: Fitness values for varying mutation rates.

Gen. No	rate=1%	rate=3%	rate=5%	rate=10%
1	0.46587	0.47600	0.45260	0.46887
3	0.30293	0.31567	0.33640	0.35167
5	0.28507	0.31567	0.33640	0.35167
7	0.28507	0.31567	0.33640	0.34233
9	0.28507	0.28260	0.33640	0.33933
11	0.28507	0.28260	0.32593	0.32707
13	0.28507	0.28260	0.32120	0.32707
15	0.28507	0.28260	0.32120	0.31080
17	0.28507	0.28260	0.32120	0.30000
19	0.28507	0.28260	0.31253	0.30000
21	0.28507	0.28260	0.31253	0.30000
23	0.28507	0.28260	0.31253	0.30000
25	0.28507	0.28260	0.31253	0.30000
27	0.28507	0.28260	0.28420	0.30000
29	0.28507	0.28260	0.27833	0.30000
31	0.28507	0.28260	0.27833	0.30000
33	0.28507	0.28260	0.27833	0.30000
35	0.28060	0.28260	0.27833	0.30000
37	0.28060	0.28260	0.27833	0.30000
39	0.25687	0.28260	0.27833	0.28953
41	0.25460	0.26053	0.27833	0.28953
43	0.25460	0.26053	0.27833	0.28953
45	0.25460	0.26053	0.26960	0.28953
47	0.24813	0.26053	0.26960	0.28953
49	0.24813	0.26053	0.26960	0.28953
51	0.24813	0.26053	0.26960	0.28887

Variations of fitness values over generations were observed with varying initial population (figs. 11, 12 and 13). It can be concluded from the plots that the interval within which the fitness values converge for varying mutation rates is lowered for higher initial population.

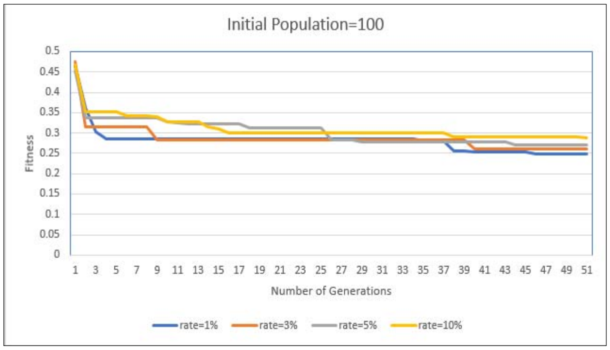

Figure 11: Plot of fitness vs. no of generations for initial population of 100. [Mutation rates=1%, 3% , 5% and 10%]

Figure 12: Plot of fitness vs. no of generations for initial population of 500. [Mutation rates=1%, 3%, 5% and 10%]

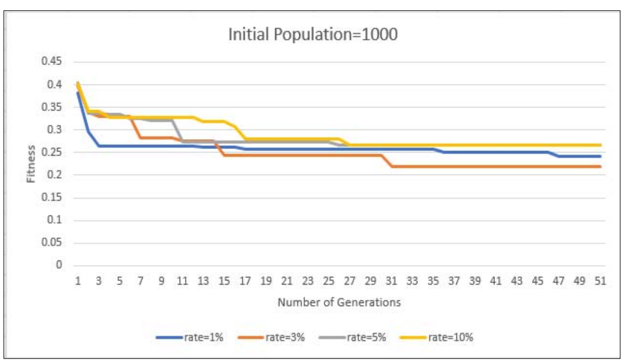

Figure 13: Plot of fitness vs. no of generations for initial population of 1000. [Mutation rates=1%, 3% , 5% and 10%]

Table 2 provides the fitness values (approximated upto 5 places of decimal) for three different initial population sizes for the first 51 generations in steps of 2. A steady decrease is seen in all cases like in the previous table.

Table 2: Fitness values for varying initial population.

Gen No	size=100	size=500	size=1000
1	0.50400	0.41280	0.39333
3	0.33607	0.30220	0.29913
5	0.33607	0.30220	0.29913
7	0.33287	0.30220	0.29913
9	0.33287	0.30220	0.29320
11	0.32520	0.30220	0.29260
13	0.32520	0.28620	0.29180
15	0.30700	0.28620	0.28400
17	0.30700	0.28540	0.28400
19	0.30700	0.28540	0.28300
21	0.30700	0.27373	0.28080
23	0.30700	0.27373	0.27020
25	0.29847	0.27373	0.27020
27	0.29847	0.26533	0.27020
29	0.29847	0.26533	0.27020
31	0.29847	0.26533	0.27020
33	0.29460	0.26533	0.27020
35	0.29460	0.26533	0.27020
37	0.29460	0.26533	0.27020
39	0.29460	0.26533	0.27020
41	0.29460	0.26533	0.27020
43	0.29460	0.26533	0.27020
45	0.29460	0.26533	0.24873
47	0.29460	0.26533	0.24873
49	0.27407	0.26533	0.24873
51	0.27407	0.26533	0.24873

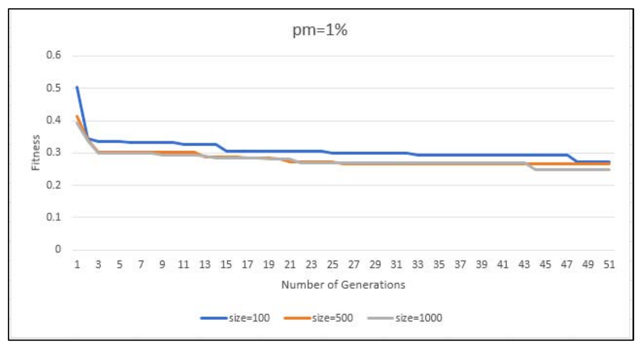

Figure 14: Plot of fitness vs. no of generations for mutation rate/ pm=1%. [Initial Population=100, 500 and 1000]

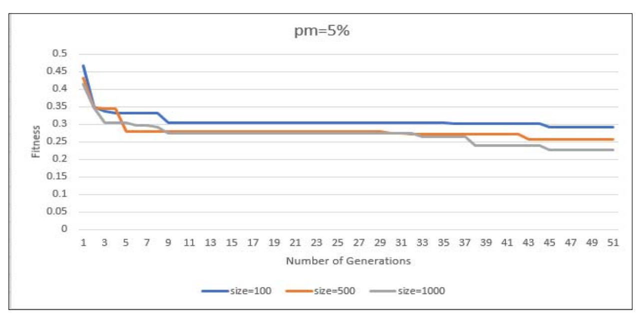

Figure 15: Plot of fitness vs. no of generations for mutation rate/ pm=5%. [Initial Population=100, 500 and 1000]

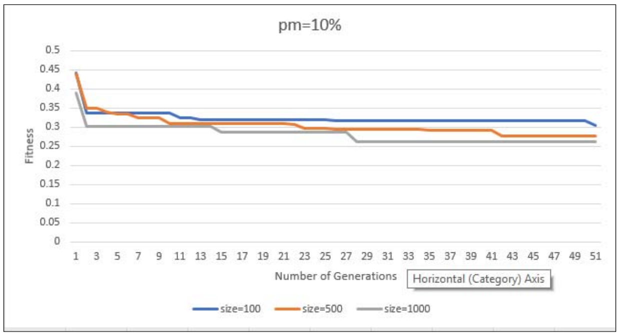

Figure 16: Plot of fitness vs. no of generations for mutation rate/ pm=10%. [Initial Population=100, 500 and 1000]

The nature of convergence of fitness values over generations for increasing initial population sizes were studied for varying mutation rates (figs 14, 15 and 16). It is clear from the plots that irrespective of the rates of mutation, the fitness values corresponding to higher initial population converge to a lower value thus indicating better performance with respect to fitness under larger initial population size.

V. CONCLUSION AND FUTURE WORK

GAs being a population based technique, are ideal for solving multi-objective optimization problems. A basic single-objective GA can be improved to obtain a collection of numerous non-dominated results in one run. The capacity of GAs to parallel examine diverse sections of a solution space makes it likely to obtain a varied set of results for complex problems with non-convex, irregular, and multi-modal solutions spaces.

From the experiments, it can be inferred that lower the mutation rate, the system performs better with respect to fitness values with increasing number of generations under varying weighing patterns of distance and time. Secondly by increasing the population size, the fitness value converges to a lesser value with increasing weights, indicating the desirability of a greater initial population. Thirdly increasing the number of generations beyond a certain value does not have significant effect on fitness with increasing weights.

The approach adopted here is the Weighted Sum Method to solve the Multi-Objective Travelling Salesman Problem using Genetic Algorithm. However, the power of the Evolutionary Algorithms lies in the fact that they can work with a population of solutions and iteratively improve them in successive generations. Hence, there is a need to extend this work to obtain Pareto-Optimal Solutions. Secondly, the Euclidean Distance has been used to calculate the distance between two cities. This work can be extended to consider actual distances to obtain better results. Lastly, Global positioning system could also be used to obtain the coordinates of locations and simulation could be done to implement the model using city road maps.

REFERENCES

[1] J. H. Holland, Adaptation in natural and artificial systems. Ann Arbor, MI: University of Michigan Press, 1975.
[2] M. Pelikan, "Genetic Algorithms," MEDAL. Report No.2010007, 2010.
[3] A. Rexhepi et al., "Analysis of the impact of parameters valueson the Genetic Algorithm for TSP," International Journal of Computer Science Issues, Vol. 10, no.1/3, pp.158-164, Jan. 2013.
[4] A. Konak et al., "Multi-objective optimization using genetic algorithms: A tutorial," Reliability Engineering and System Safety, vol. 91, no.9, pp.992–1007, Sep.2006.
[5] H. Braun, "On solving travelling salesman problems by genetic algorithms," in International Conference on Parallel Problem Solving from Nature, Dortmund, Germany, 1990, pp.129-133
[6] Akshatha P.S. et al.,"Open Loop Travelling Salesman Problem using Genetic Algorithm," International Journal of Innovative Research in Computer and Communication Engineering Vol. 1, no. 1, pp. 112-116, Mar. 2013.
[7] K. Bryant, "Genetic algorithms and the traveling salesman problem." Master's thesis, Harvey Mudd College, Claremont, United States, 2000.
[8] G. Reinelt, The Traveling Salesman: Computational Solutions for TSP Applications. Springer, 1994.
[9] V. Dwivedi et al., "Travelling Salesman Problem using Genetic Algorithm," in International Journal of Computer Applications (IJCA): *Proc. of the* National Conference on Development of Reliable Information Systems, Techniques and Related Issues (DRISTI), Ghaziabad, India, 2012, pp.25-30.
[10] O. M. Sallabi and Y. EI-Haddad, "An Improved Genetic Algorithm to Solve the Traveling Salesman Problem", World Academy of Science, Engineering and Technology, Vol. 52, no.3, pp.471-474 2009.
[11] C. Chudasama, S. M. Shah and M. Panchal, "Comparison of Parents Selection Methods of Genetic Algorithm for TSP", in International Journal of Computer Applications (IJCA): *Proc. of the* International Conference on Computer Communication and Networks CSI-COMNET-2011, pp 85-87.
[12] I. Siwale, "Practical Multi-objective Programming[rev 6]," Apex Research Ltd, Lomdon, Technical Report RD-14-2013.
[13] Vaibhav and S. K. Harit, "Multi-Objective Optimization for Traffic Route Selection," IJCSC, Vol. 7, No. 2, pp. 148-152, 2016.
[14] M. Sudharson, "Genetic Algorithm for the Traveling Salesman Problem using Minimal Weight Variable Order Selection Crossover Operator (MWVOSX)" [Online] Available: http://www.academia.edu/6143584/ [last accessed on 21.5.17]