© 2018 IJSRSET | Volume 4 | Issue 4 | Print ISSN: 2395-1990 | Online ISSN : 2394-4099
Themed Section : Engineering and Technology

Travelling Salesman Problem Solved using Genetic Algorithm Combined Data Perturbation

J Kanimozhi*, R Subramanian
Department of Computer Science, Pondicherry University, Puducherry, Tamil Nadu, India

ABSTRACT

Traveling Salesman Problem (TSP) is a challenging problem in combinatorial optimization. The important of this problem is due to the fact that it is used in many fields such as transportation, logistics, semiconductor industry, problem of routing etc. In this paper Travelling Salesman Problem (TSP) is solved using Genetic Algorithm (GA) combined with data perturbation (DP) and the algorithm named as Perturbed GA. DP is a technique used to avoid local optima and to increase the diversity property of the problem. Efficiency of the algorithm is calculated in terms of fitness, convergence and error rate and from the result analysis, it's proved that Perturbed GA outperforming GA to solve TSP.

Keywords: Travelling Salesman Problem, Genetic algorithm, Data perturbation, Cost function

I. INTRODUCTION

The Travelling Salesman Problem (TSP) is a NP-hard problem [1] - [4]. Travelling salesman problem states that given a number of cities N and the distance or time to travel between the cities, the traveler has to travel through all the given cities exactly once and return to the same city from where he started and also the cost of the path is minimized [1] - [7]. This path is called as the tour and the path length or travel time is the cost of the path. The TSP mathematical model follows [3] [8] [9]:

$$\min Z = \sum_{i=1}^{N}\sum_{j=1}^{N} X_{ij} C_{ij}$$

Subject to

$\sum_{i=1}^{N} X_{ij} = 1$ j=1, 2, 3…... N

$\sum_{j=1}^{N} X_{ij} = 1$ i=1, 2, 3…... N

Let Cij be the cost for traveling from i-th city to j-th city. Where Xij = 1 if the salesman travels from city-i to city-j, otherwise Xij = 0. Z is the minimum distance of the optimal path.

Genetic algorithm is a technique used for estimating computer models based on methods adapted from the field of genetics in biology [3] [4]. To use this technique, one encodes possible model behaviours into ''genes". After each generation, the current models are rated and allowed to mate and breed based on their fitness. In the process of mating, the genes are exchanged, crossovers and mutations can occur. The current population is discarded and its offspring forms the next generation. Also, Genetic Algorithm describes a variety of modelling or optimization techniques that claim to mimic some aspects of biological modelling in choosing an optimum. Then a large number of candidate models are generated and tested against the current data. Each model is scored and the "best" models are retained for the next generation. If the model is constructed so that they have "genes," the winners can "mate" to produce the next generation. And finally these best off-springs replace the existing individuals in the population if it dominates the existing solutions and the process repeats. When the termination condition exists then the process will stop.

This paper presents A Perturbed GA Algorithm to solve TSP. The framework of Perturbed GA combines GA and Data Perturbation. The paper is organized as follows: Section 2 gives the literature review of Genetic algorithm to solve TSP and data perturbation techniques to avoid local optima. Section 3 describes perturbed GA algorithm and its components. Section 4 gives the Experimental result and section 5 concludes the work.

II. RELATED WORK

The following section A gives the literature review of the GA to solve MOTSP and B gives the data perturbation concept and two methods of doing the data perturbation.

A. Literature Review of GA to Solve TSP

The GA steps to solve TSP and the chromosome representation are given in [1] [3]. A variable neighbourhood search is used in [2] to find the optimal solution for TSP. Convex Partition and Gradual Insertion (CPGI) Algorithm is implemented [4] using convex hull as tool to find the shortest cycle to a TSP. In GA the crossover and mutation techniques used will be a special method to produce the best result. But in [10] no special crossovers are used to generate the new trails and with some inverse permutation method, new generations are generated and solved TSP. It is observed that this method outperforms PMX in convergence rate by a factor which can be as high as 11.1 times, on a cost of obtaining slightly worse solutions on average. In [13] TSP is solve using exact algorithms and genetic algorithm and the performance is compared. GA out performs exact algorithm which gave the base to solve TSP using GA. Paper [8] presents a new approach based a genetic algorithm to solve selective travelling salesman problem, which is concerned with finding a path between a given set of control points, among which a start and an end point are the same, and to maximize the total gain collected subject to a prescribed cost constraint. The results of the application of this approach give us the best of the objective function and the number of iterations for algorithm is reduced.

GA can solve problems with non-parametrical problems, multi-dimensional, non-continuous and non-differential optimization problems. But Genetic Algorithm has weaker local search ability [14]. During the later period of Hierarchical Genetic Algorithm, the fitness converges, and less superior individuals are produced. However, Hybrid Simulated Annealing Algorithm can make it jump out of the erroneous zone of local optimum. Due to the compatibility of Genetic Algorithm, it is feasible to combine Simulated Annealing Algorithm and Hierarchical Genetic Algorithm to form the modified Simulated Annealing Genetic Algorithm [15]. The modified Simulated Annealing Genetic Algorithm can sooner achieve a better global optimum solution. The reasons are: the suffix structure design of chromosome reduced the space of the solution, the self-adaptive genetic operator and double crossover and mutation improved 'premature convergence problem'; the introduction of Simulated Annealing Algorithm stretched the fitness and enhanced the local search ability.

Still there is problem with GA in solving TSP. GA give local optimal solutions in a minimal time [16]. So to avoid the local optima and to get global optima data perturbation is used. The need of DP, method of doing DP and its literature survey are given in the next section A.

B. Data Perturbation (DP)

In Evolutionary Algorithms, the initial population size, diversity and convergence property of initial population influences more on the optimal solution [9]. "Data Perturbation" (DP) technique, originally proposed by Codenotti et al. for the single-objective TSP and has been introduced in MO optimization by Lust and Teghem [17] [18]. A perturbation move is a technique used to escape from local optimum which is an issue found in MOGA. Instead of modifying the starting solution DP suggests to modify input data [9].

The perturbation is a double bridge move [14] that cuts the current tour at four appropriately chosen edges into four sub-tours and reconnects these in a different order to yield a new starting tour for the local search.

There are two methods to do the DP which are given in [17] - [21]. The first DP method in [17] [19] [20] is started with the input parameters number K of iterations, three parameters that determine the perturbation scheme (the fixed scale SF, the varying scale SV and the local perturbation LP) and the cost matrices $C_{i,j}^{k}$ of the MOTSP. During an iteration k, first compute a weight set λ by following a linear scheme; K uniformly distributed weight sets are thus generated. We then create a new cost matrix C^{λ}. Then, we slightly perturb each cost $C^{\lambda}(i, j)$ of the matrix C^{λ} to find new potentially efficient solutions. Increase in the number K of iterations gives an important improvement of the indicators and allows reaching excellent results, since the number of potentially efficient solutions |PE| is increasing while the distance D1 is decreasing. The number of iterations for the number of perturbation steps is equal to the number of cities N minus 50.

The second DP method in [18] is done with a single parameter d, whereas the above one needs three parameters. Higher the d value, the larger the perturbation is. It adds a random noise to the cost function and so the search direction is given in all the way. The value of d is set from 3 to 20 percent variations. The better results are given with 5 %. So for optimal result use d=5%.

III. PERTURBED GENETIC ALGORITHM

From the survey of GA and Data perturbation techniques, it is efficient to used data perturbation with GA to solve TSP. In existing from [17] – [21] Data perturbation technique is used to solve bi-objective TSP and MOTSP. In [18], the second method of DP is used in all other the first technique is used. From the literature survey is proved that DP is giving better solution than the algorithm without DP, by increasing the diversity property of the problem. The second DP method with single parameter d is used in our work since the execution time is less compared with the first method. It is appropriate to used DP, when the number of cities is more. This section gives the framework of Perturbed GA, characteristics and flow chart. Characteristics of Perturbed GA follow:

- Initialization of Perturbed GA starts with the data perturbation of cost matrix. The initial cost matrix is computed using data set and using d parameter cost matrix is modified.
- Apply GA operator to generate new solutions.
- Update the optimal solutions in population and continue the same process till termination condition met.

Figure 1 shows the framework of Perturbed GA to solve TSP. Initial population is given random and using weight vector cost evaluation is done. Data perturbation is applied on to the cost function using d parameter, and a new perturbed cost of the population is calculated. With the perturbed cost GA process continues.

The following are the notations used in the description of Perturbed GA.

- A unique weight $\lambda^k = (\lambda_1^k, \ldots \lambda_j^k)$
- $C^k(e) = \sum_{j=1}^{p} \lambda_j^k \, C_j(e)$ is the initial cost function
- $P = \{x^1, \ldots\ldots x^n\}$
- N: Population Size
- T: Maximum number of generations

The Perturbed GA algorithm is given below in the Algorithm 1.

Algorithm 1: Perturbed GA

Input: a TSP problem, a stopping criterion, data perturbation parameter d, Initial Population P
Output: set A of potentially efficient solution

Begin

Let C^k be the cost matrix related to TSP
Data perturbation (C^k, d)
Foreach e ∈ E do
v ← U (1-d, 1+d)
$C^k(e) \leftarrow v \times \sum_{j=1}^{p} \lambda_j^k \; C_j(e)$
end
Randomly initialize the population P = {x^1,……x^n}
For i=1:T
GA Operator ($C^k(e)$, P)
Generate a new solution y with crossover and mutation
Update the population P
end
Return the Best Individual

End

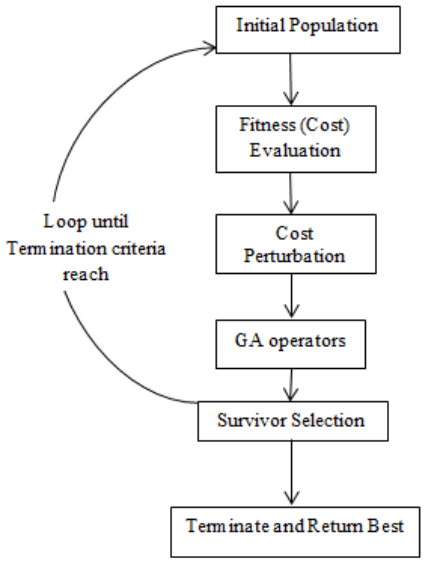

Figure 1. Framework of Perturbed GA

The framework starts with the Cost matrix (C^k) and continued with the data perturbation. The DP adds noise into the multi objective cost function. The data perturbation parameter $d \geq 0$, which controls the maximum variation of the noise added to the cost. The value of v is a real number calculated using d value varies with uniform distribution of (1-d) to (1+d). So the perturbed cost matrix is $C^k(e) \leftarrow v \times \sum_{j=1}^{p} \lambda_j^k \; C_j(e)$. With $C^k(e)$ the GA operator continues to generate new solutions. The best individuals are added in Population (P) and the process end when the termination condition met. With experimental analysis d value must be equal to 5% to produce an optimal solution which is show in the result analysis section.

IV. RESULT ANALYSIS

The significance of the proposed Perturbed GA is demonstrated using the travelling salesman problem (TSP). The TSP is one of the most important optimization problems for the researcher has been chosen as the testbed. Naturally, the TSP belong the class of NP-hard problem and act as tested for new optimization algorithm. Hence, the instances eil51, st70, kroA100, kroC100, eil101 and tsp225 are chosen

as the test dataset which is obtained from the standard library of TSPLIB.

The algorithm is implemented in Matlab and the result analysis is done using fitness function, convergence and error rate. Therefore, metrics used to evaluate the performance of the proposed algorithm is given below:

Fitness Function:

$$Fit = \min\left\{\left(\sum_{j=1}^{P} dist(C_i, C_{i+1})\right) + dist(C_p, C_1)\right\}$$

Whereas,

- P refers to the number of cities in the individual,
- $dist(C_i)$ is the combined fitness of two objectives,
- $dist(C_i, C_{i+1})$ Refers the distance between two cities C_i and C_{i+1},
- $dist(C_p, C_1)$ Refers to the distance between last city and first city during return after the tour.

The fitness value is one of the important assessment criteria which give the tangible result of the optimal solution. Each algorithm was run on each instance 25 times and hence the best among the 25 runs are considered for analysis and validation purposes.

Convergence rate: The convergence rate indicates the quality of the optimal solution generated from the population. Convergence rate of an individual in the population is defined as the percentage of fitness value obtained by the individual in accordance with the optimal fitness value. It can be formulated as follows:

$$convergencerate\,(\%) = 1 - \frac{Fitness - optimalfitness}{optimalfitness} \times 100$$

Average convergence: Average convergence rate can be defined as the average percentage of fitness value of the individual w.r.t to the optimal fitness value. It can be expressed as follows:

$$Avg.conv.(\%) = 1 - \frac{averagefitness - optimalfitness}{optimalfitness} \times 100$$

Error rate: The evaluation of the proposed algorithm in terms of error rate is important for the analysis. The best error rate indicates how far the best individual convergence rate deviates from the optimal fitness value while the worst error rate demonstrates the difference between the convergence rate of worst individual from the population and the optimal solution.

Error rate of an individual is defined as the percentage of difference between fitness obtained by the individual and optimal fitness value. It can be given as:

$$Errorrate\,(\%) = \frac{Fitness - optimalfitness}{optimalfitness} \times 100$$

Table 1. Computational Results Of Ga, Perturbed Ga (5, 10, 20%)

S. No	TSP Instance	Technique	Optimum value	Fitness		Convergence rate (%)	Error rate (%)	Average Convergence (%)
				Best	Average			
1	eil51	GA	426	439.45	455.89	96.8	3.1	92.5
		P-GA (5%)		436.23	441.76	97.6	2.3	95.6
		P-GA (10%)		439.83	452.62	96.2	3.1	93.4
		P-GA (20%)		439.26	454.19	92.9	3.1	92.3
2	St70	GA	675	727.27	727	92.25	7.7	87.4
		P-GA (5%)		710.81	710	94.69	5.3	90.51
		P-GA (10%)		725.36	725	92.53	7.4	88.19

		P-GA (20%)		723.99	723	92.71	7.2	88.0
3	kroA100	GA	21282	21783.6	22441.78	97.6	2.3	92.5
		P-GA (5%)		21330.07	21320.53	99.7	0.5	95.26
		P-GA (10%)		21745.43	21735.76	97.8	2.17	93.2
		P-GA (20%)		21733.03	21723	97.7	21	93.2
4	kroC100	GA	20749	21314.06	22534.78	97.2	2.7	91.45
		P-GA (5%)		21000.54	21018.53	98.8	1.25	95.68
		P-GA (10%)		21314.99	21324.99	97.6	2.8	93.28
		P-GA (20%)		21141.49	21121.91	97.1	2.3	92.17
5	Lin105	GA	14379	14707.69	15166.18	94.45	2.28	91.45
		P-GA (5%)		14614.3	14988	98.36	1.6	95.68
		P-GA (10%)		14782.5	15067	97.28	2.8	93.28
		P-GA (20%)		14719.3	15225	97.63	2.3	92.83
6	Tsp225	GA	3919	4401.67	4401	87.67	9.3	90.74
		P-GA (5%)		4209.32	4209	92.57	4.4	96.03
		P-GA (10%)		4408.78	4508	87.50	8.4	92.45
		P-GA (20%)		4407.54	4627	87.53	7.2	92.83

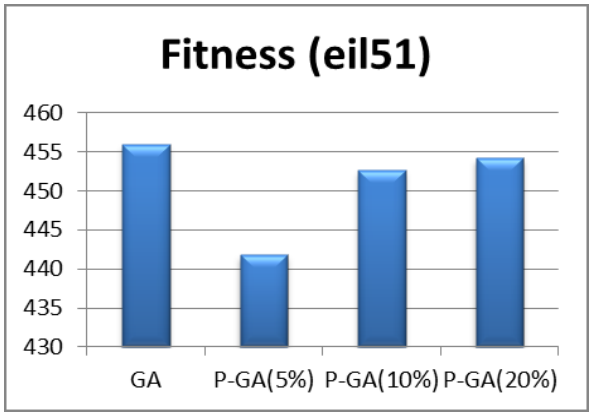

Figure 2(a)

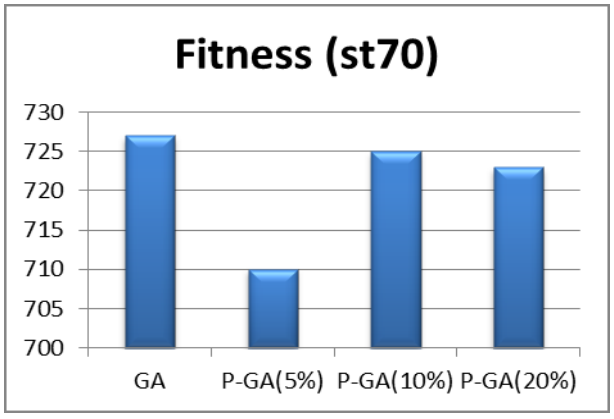

Figure 2(b)

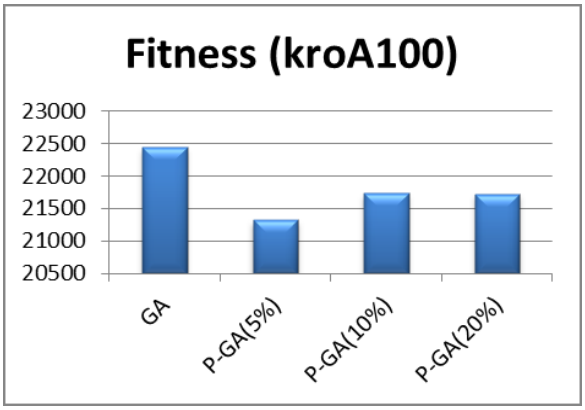

Figure 2(c)

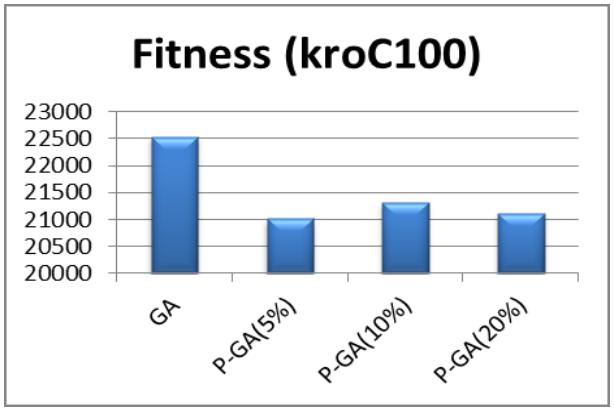

Figure 2(d)

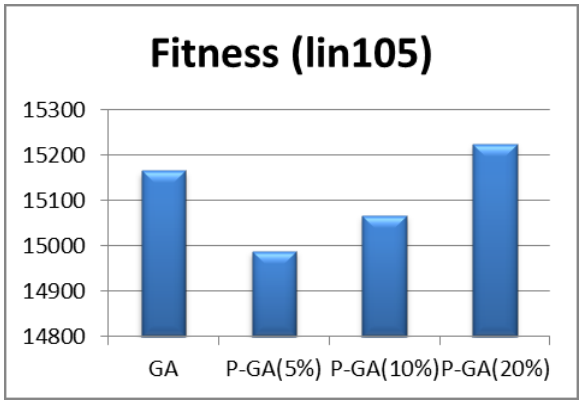

Figure 2(e)

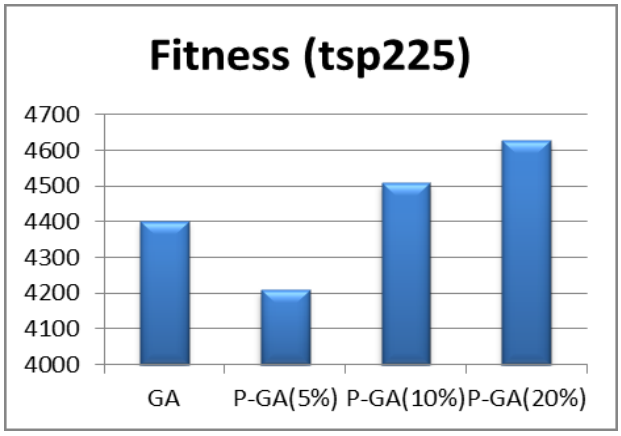

Figure 2(f)

Figure 2. (a) - 2(f). Analyses the performance based on Fitness Value

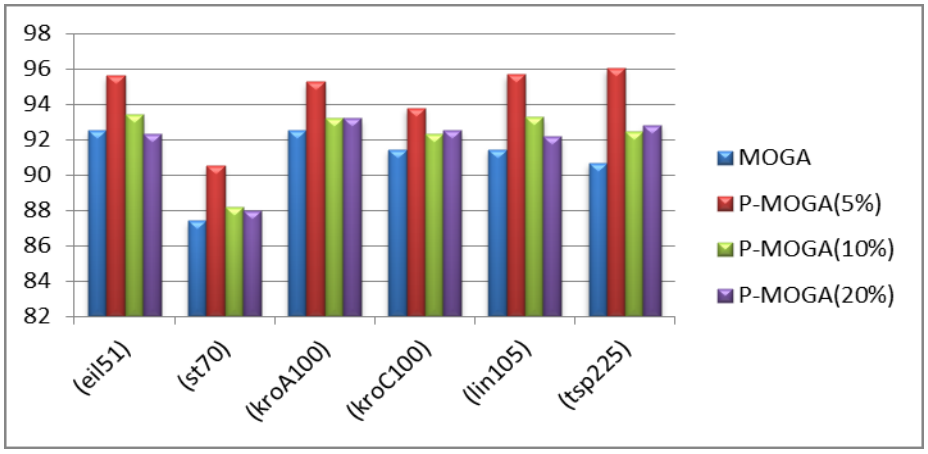

Figure 3. Analyses the performance based Average Convergence rate

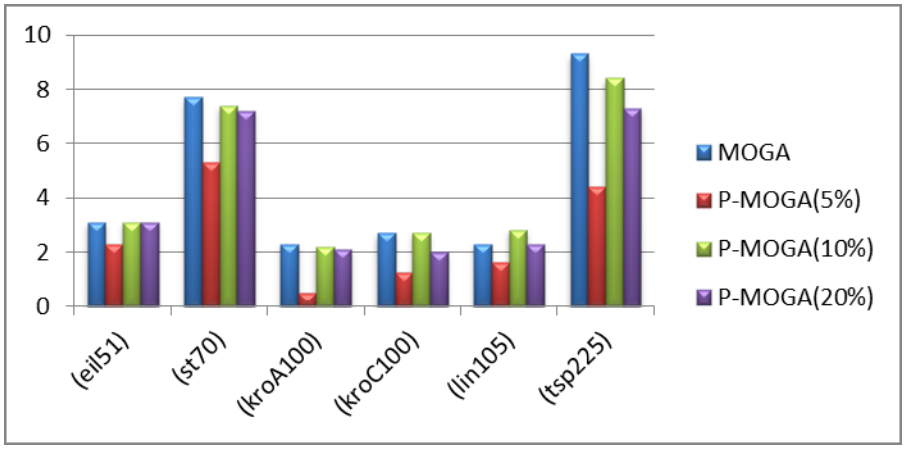

Figure 4. Analyses the performance based Error rate

Table 1 show the fitness, convergence, average convergence and error rate of the algorithms MOGA, Perturbed MOGA 5%, 10%, 20%. From the computational results shown in table 1, it is shown that perturbed MOGA (P-MOGA) outperforms MOGA on the above TSP instances. The fitness of the solutions is shown in the Figure 3(a) – 3(f), where P-MOGA gives the better result. The convergence and error rate of the algorithm is also compared and plotted in Figure 4 and Figure 5 to show the performance of the algorithm. From eil51 and other data sets, it is know that when the number of cities is more, then it is appropriate to solve the problem with Perturbed MOGA.

V. CONCLUSION

In the present work, we introduced a new efficient component in GA algorithm: the data perturbation. We showed that by using data perturbation combined with GA operators work well to solve the TSP. Perturbed GA is implemented with d parameter varies from 5 to 20% and compared with general GA. The optimal solution is given when the d parameter is kept 5%. The robustness of the Perturbed Genetic Algorithm has been clearly illustrated with respect to fitness value, convergence rate, error rate and average convergence rate.

VI. REFERENCE

[1] Kylie Bryant, A. Benjamin, "Genetic Algorithms and the Traveling Salesman Problem", Department of mathematic Harvey Mudd college, December, 2000.

[2] X. Meng, J. Li, S. Member, X. Dai, and J. Dou, "Variable Neighborhood Search for a Colored Traveling Salesman Problem," IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, vol. 19, no. 4, pp. 1018–1026, 2018.

[3] Philip, "A Genetic Algorithm for Solving Travelling Salesman Problem," International Journal of Advanced Computer Science and Applications,vol. 2, no. 1, pp. 26–29, 2011.

[4] A. Kai, "A simple algorithm for solving travelling salesman problem", 2012 Second IEEE Conference on Instrumentation & Measurement, Computer, Communication and Control, pp. 1–5, 2012.

[5] S. G. E. Brucal and E. P. Dadios, "Comparative Analysis of Solving Traveling Salesman Problem using Artificial Intelligence Algorithms," IEEE, 2017.

[6] R. Takahashi, "Quantitative Evaluation of Iterative Extended Changing Crossover Operators to Solve the Traveling Salesman Problem", 10th International Conference on Natural Computation, pp. 235–244, 2014.

[7] A. Al-dallal, "Using Genetic Algorithm with Combinational Crossover to Solve Travelling Salesman Problem." Ahlia University, (n.a).

[8] L. Settat, "A new approach based a genetic algorithm for the Selective Travelling Salesman Problem", IEEE, no. 2, pp. 785–788, 2016.

[9] S. Lianshuan, "An Improved Pareto Genetic Algorithm for Multi-Objective TSP", Fifth International Conference on Natural Computation, pp. 585–588, 2009, pp. 585–588, 2009.

[10] GOKTURK UCOLUK, "Genetic Algorithm Solution of the TSP Avoiding Special Crossover and Mutation." Middle East Technical University, (n.a).

[11] A. Hussain, "Genetic Algorithm for Traveling Salesman Problem with Modified Cycle Crossover Operator", Computational Intelligence and Neuroscience, Volume 2017 (2017).

[12] C. R. Lopes, "Using Genetic Algorithms to minimize the distance and balance the routes for the multiple Traveling Salesman Problem", IEEE, pp. 3171–3178, 2015.

[13] Jean Yves, "Genetic Algorithm to solve TSP", Annals of Operations Research 63(1996)339-370, (1996).

[14] Y. Expressway, G. Noida, G. Nagar, and U. Pradesh, "A Survey : Swarm Intelligence vs . Genetic Algorithm", International Journal of Science and Research, vol. 3, no. 5, pp. 231–235, 2014.

[15] Xu, M., Li, S., & Guo, J. (2017). Optimization of Multiple Traveling Salesman Problem Based on Simulated Annealing Genetic Algorithm, 25.

[16] Y. Expressway, G. Noida, G. Nagar, and U. Pradesh, "A Survey : Swarm Intelligence vs . Genetic Algorithm", International Journal of Science and Research, vol. 3, no. 5, pp. 231–235, 2014.

[17] T. Lust and J. Teghem, "Two-phase Pareto local search for the biobjective traveling salesman problem", Springer Science, pp. 475–510, 2010, 2009.

[18] M. Cornu, T. Cazenave , D. Vanderpooten, "Perturbed Decomposition Algorithm applied to the multi-objective Traveling Salesman Problem. Computers & Operations Research, 79, 1–17. https://doi.org/10.1016/j.cor.2016.04.025, 2016.

[19] T. Lust and A. Jaszkiewicz, "Speed-up techniques for solving large-scale biobjective TSP," Comput. Oper. Res., vol. 37, no. 3, pp. 521–533, 2010.

[20] T. Lust and A. FNRS, "New metaheuristics for solving {MOCO} problems : application to the knapsack problem, the traveling salesman problem and {IMRT} optimization," 2010.

[21] L. Paquete and T. Stutzle, "A two-phase local search for the biobjective traveling salesman problem," Evol. Multi-Criterion Optim. Proc., vol. 2632, pp. 479–493, 2003.