Comparative Analysis of Meta-Heuristic Algorithms for Solving Optimization Problems

Muhammad Kashif[1,a] ,Gao Shang[1, b *] , Qaisar Sohail[1] , Abdul Mannan Masood[1] , Muhammad Atif[2], Usman Ashraf[3] , Aleena Akhtar[3] and Shahid Ali[3]

[1] School of Computer Science and Technology, Jiangsu University of Science and Technol ogy, Zhenjiang 212003, China

[2]School of Design and Manufacturing Engineering, National University of Science and Technology, Pakistan

[3]School of Electronics and Information Engineering. Jiangsu University of Science and Technology, China

[a] kashif.ciitwah@gmail.com, [b] gao_shang@just.edu.cn

* The corresponding author

Keywords: Ant Colony Algorithm (ACO), Genetic Algorithm (GA), Simulated Annealing Algorithm (SAA), Traveling salesman Problem (TSP), Meta-Heuristics, Optimization

Abstract. Combinatorial Problems (NP hard Problem) have always been a hard task to be solved to optimal level but for the efficiency and finding the best possible solution in a certain span of time it has been solved to suboptimal level. During the study for solving the combinatorial problems to suboptimal level different heuristic algorithms has been used for acquiring results from the TSPLIB Instances. Different Suboptimal level has been achieved through different heuristics like Ant Colony Algorithm, Genetic Algorithm and Simulated Annealing Algorithm. The perimeters were tuned to different levels of all heuristics to find suboptimal level of the instances of TSPLIB. The paper will also present the effects of perimeters tuning to achieve the suboptimal results.

Introduction

Optimization Problems has always been a prime focus in every field of Engineering. Optimization Problems are gaining more importance and it is shifting its attention to utilization of less resources and efficiency. As well now the focus is to solve problems with efficiency as well as accuracy is the basic ingredient and it has been achieved to better levels and with the passage of time it is getting better and better. But as the science is getting advanced we are looking for solution with less processing time and more accuracy.

This sort of problems can be found in many fields like vehicle routing problem, physical mapping [1], integrated circuit designing [2], scheduling problem [3] etc. Traveling Salesman Problem (TSP) is known NP-hard problem which is extendedly studied for the research work in field of computer science and especially Artificial Intelligence. In the problem of TSP, a salesman always wants to visit set of cities which is defined in terms of coordinates in TSP Libraries exactly once and return to the initial city with minimum distance travelled and hence achieve optimality [4]. TSP has been one of most used topic for researchers in Computer Science [5] since the mid of 20th century which in result produced a wide number of techniques to solve this combinatorial problem. TSP is generally interpreted by the complete edge weighted graph G=(V,E). Where V is absolute number set of number of cities or nodes while $E \subseteq V \times V$ which represents the set of edges. Each arc $(i, j) \in E$ assign value dij to the cities from city i to city j where i and j belongs to V. The set of TSP data can either be symmetric or asymmetric. The difference between the symmetric and asymmetric is that the distance between two nodes remains constant from i to j and j to i presented $d_{ij} = d_{ji}$ while in asymmetric the i to j and j to i is never same which can be interpreted as

$d_{ij} \neq d_{ji}$. The ultimate purpose of TSP is to find the least possible Hamiltonian Circuit of the graph where the circuit is the locked path in which all the cities (nodes) n of the graph (G) are visited exactly once [6]. So optimal solution for TSP is the permutation (π) of the cities indices from {1 to n} such that the $f(\pi)$ is the possible minimum which can be represented mathematically by[7];

$$f(\pi) = \sum_{i=1}^{n-1} d\pi(i)\pi(i+1) + d\pi(n)\pi(1) \tag{1}$$

TSP has always been the test bed for the Non-Polynomial Hard Problems while several methods used are used to solve these problems. The first conventional methods that were used to solve TSP problems were Branch and Bound but that was a hard task to operate and non-feasible so as the mathematicians and computer science advanced the heuristics were introduced to solve these NP Hard problems. There are several constraints in TSP for instance take the constraint of subtour and its elimination. For detail study of constraints and its elimination refer to the document mentioned in the reference [8]. While certain algorithms have also been explained for solving optimization problem in [9]. The TSP source of origination and its name is somehow ambiguous but it still looks like it has been worked on by the mathematician in different ages and for many years. Amazingly the result has been derived in history also by mathematical literature Leonard Euler 1st represented the tours and circuits in 1757 in his paper which concerns with solution of Knight's tour problem in Chess. The problem was to find the moves order of knight that will start from the start from the chess board 1st square to the last without repetitions of any square and comes to starting square [10][11]. Irish Mathematicians handled mathematical computation in 1800's which was related to TSP. The German handbook went through 47 cities which was considered to be of the good quality and maybe optimal as mentioned by the author [12]. Sir William Hamilton in 1859 contributed to the further growth of graph theory by creating the Icosian game that requires a player to complete a tour using only specified paths throughout the total 20 points of the game. [13]. There is no exact source for the name of TSP but according to [14] it has been first introduced in 1934 during a seminar at Princeton University. Reinelt composed and published TSPLIB a library which contains many of the test problems studied for more than 50 years and still used as a Test bed for optimization problem [15].

Simulated Annealing Algorithm

Simulated Annealing Algorithm is a meta heuristic algorithm which process only one solution and in order for SA to work we should define the neighborhood of that solution. The neighborhood of present solution should be defined as such that all the possible solutions can be reached from the present solution by precisely one application of swap operator (mutation operator of the GA is known as Standard Swap Operator). Simulated Annealing Algorithm is successfully improved to give approximate solution to the TSPLIB Instances. SAA algorithm basically work on the basis of randomized local search algorithms allowing the movement with the negative gain. Simulated Annealing working procedure is as such; In every iteration of SAA a solution from neighborhood of the present solution is arbitrarily chosen. Then if the new solution is better than the present solution, the present solution is replaced by the new solution. And if the new solution is not better than the present solution then it is substituted by the new one solution with the certain probability. Probability is a function of the number of iterations which has been passed since the start of the optimization procedure and of the variance in functional objective values between the two measured solutions. The probability of accepting worse solution decreases as the algorithm proceeds. At the beginning of optimization SAA behave more like a local random search algorithm but then later it becomes bias and only accepts the better solution. And at the last stages of the algorithms the SAA behave more like a greedy search algorithm. Better results can be obtained by increasing the run time of the SA algorithm. The basic implementation of SAA for TSP is being given in [11]. Fig. 1 describes the working procedure of SAA. Simulated Annealing Algorithm is a very famous meta-heuristic search algorithm which is being used to solve combinatorial optimization problems This algorithm is basically the upgradation of hill climbing algorithm which has ability to escape from the local optimum values in the search space. Although it gives good solution but still it is very slow as compared to the original Hill Climbing Algorithm. Simulated Annealing is the term derived from the Annealing of solid materials where it has been tried to slowly decrease the energy of the system until the atoms are being stabilized. The slow cooling procedure allows the atom to form a line and shape in crystalline structure which will have low energy and high density. While Annealing schedule control the initial and reducing temperature for the process of Annealing. In 1983 Simulated Annealing by first presented by Kirk Patrick [16] which was being applied to combinatorial problem of optimization from statistical mechanics algorithm names Metropolis Algorithm [17]. MA provides a simplification and generalization of iterative improvement in statistical mechanics where controlled Uphill's movements; which usually doesn't decrease energy level of the system, are accepted in the finding of a better escaping from local optimum and organization of atoms.

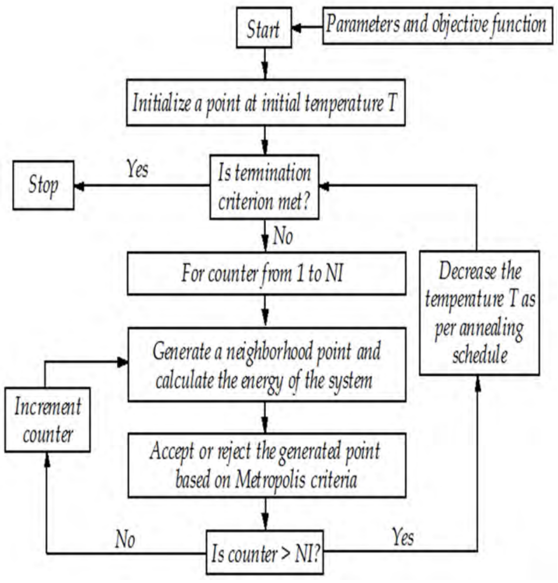
Start
Parameters and objective function
Initialize a point at initial temperature T
Stop
Yes
Is termination criterion met?
No
For counter from 1 to NI
Decrease the temperature T as per annealing schedule
Generate a neighborhood point and calculate the energy of the system
Increment counter
Accept or reject the generated point based on Metropolis criteria
No
Is counter > NI?
Yes

Figure 1. Finite Flow Chart SAA

In every step of the SA, an atom is given a little displacement and if it results in the decrease of system's energy then it is accepted and used as initial point for the next step. If the energy of the system is not decreased by the displacement then it is accepted with the probability of exp-E/KbT where E shows the energy change due to the displacement while T represents the current temperature of the system while kb is Boltzman Constant. Depending on the value returned by exp-E/KbT the new change is accepted or the system is retained to old state. For any given temperature enough iteration numbers lead the system to thermal equilibrium state. Using the Markov Chain's theory, it can be shown that Simulated Annealing possess a formal proof of convergence [18][19].

Ant Colony Algorithm

Ant Colony Optimization (ACO) is a meta-heuristic algorithm which is being inspired by the behaviour of real ants. This behaviour was 1st established by Dorigo in 1991 [20]. Real ants find their food source by laying a pheromone trail along the way of their movement from nest to food source. New ants follow the path with the high pheromone trails to the food source this it forms the shortest path to the food source which even deposit more pheromone trail on the path and hence a shorter path is established to the food source. The process of communication followed by the ants is known as Stigmergy [21]. Stigmergy is the concept in which the positive feedbacks are searched for the possible path to the source of food on the basis of the experience of previous ants. The ACO algorithm flow chart is shown in Fig. 26 [22].

The ACO is being designed keeping in view the behaviour of real ants which are very capable to find the shortest path to food source from their nests without using any visual signals. To know how the real ANTS work to find the short path; an example has been taken from [10]. In Ant Colony Optimization there are artificial ants which were being taken as inspired from real ants' behaviour of searching for food and bringing to nest. Now Suppose we have n number of cities with a distance of dij. All the artificial ants are distributed among all the cities(n) randomly. Every Ant will choose to visit the next according to the pheromone trail being laid on the paths just as mentioned in the example given above. However, there are two main difference between real ants and artificial ants.

(1) Artificial ants have memory and they are able to remember the cities they have visited and cities already visited will not be selected again.

(2) Artificial Ants are not fully blind. They have been given the distance and they know the distance between the two cities and tend to choose the next nearby city from its current city.

Therefore, mathematically we can represent the probability of selection of city j by ant k by the equation

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{\tau_{ij}^{\alpha}(t) \cdot \eta_{ij}^{\beta}}{\sum\limits_{s \in allowed_k} \tau_{is}^{\alpha}(t) \cdot \eta_{is}^{\beta}} & if\ j \in allowed_k \\ 0 & otherwise \end{cases} \tag{2}$$

In the equation above τij is the intensity of pheromone trail laid between the city i and city j. α is the parameter which regulate the influence of τij. ηij is the visibility of city j from the current city i. β is the parameter which regulates the influence of the parameter of ηij and allowedk the set of cities which are not yet being visited by the ants. At the starting of the K ants are placed randomly in the n number of cities. Then every ant decides which city it should visit next according the probability given in the above equation. After the n number of iterations every ant completes a tour and the shortest tour completed by any ant will leave a denser trail of the pheromone than the longer tours. The trail levels of pheromone are updated on every tour. The quantity of pheromone is given by Q/Lk where Q is the constant while Lk is the length of tour respectively. While the pheromone trail is evaporated with time and should be updated and it is given by the equation below.

$$\tau_{ij}(t+n) = \rho\tau_{ij}(t) + \Delta\tau_{ij} \tag{3}$$

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{4}$$

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{Q}{L_k} & \text{if k - th ant uses edge (i, j) in its tour} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

In above equation t is counter for iteration counting and ρ ϵ (i, j) is the parameter to regulate the decrease of τij and Δ τij is the total increase of pheromone trail which is being on the edge from i to j by the ant k. After the trail updates the next iteration t+1 will start and will work. ACO algorithms is specifically dedicated to solve the TSP problems [23][24][25]. Despite of many shortcoming in the ACO such as Performance which is extremely dependent on previous round. It is also easy to converge and can get stagnant this require a long processing time. This hence introduce challenges like searching space and computation time [25].

Genetic Algorithm

Genetic Algorithm has been a very hot topic for the field of computer science and Artificial Intelligence for the Optimization Problem. In every iteration GA processes not only, solution but a whole set of population of solutions which is known as Generation. The main component of GA is the following Operators: Selection Operator and Mutation Operator.

Genetic algorithms start with a set of randomly selected chromosomes as the initial population that encrypts a set of possible solutions. In GAs, variables of a problem are represented as genes of a chromosome, and the chromosomes are evaluated according to their costs using some measures of profit that is needed to be optimized. The recombination typically involves last two operators mentioned above. The genetic operators alter the composition of genes to give new chromosomes which is known as offspring. While the selection operator is an artificial way of natural selection, a Darwinian survival of the fittest among populations, to create populations from generation to generation, and chromosomes with better cost have higher probabilities of being selected in the next generation. After several generations, GA can converge to the best solution. Recently, GAs with local search method has been considered as good alternatives for solving optimization problem of TSP.
Genetic algorithm is a meta-heuristic evolutionary algorithm which is being taken from the inspiration of Darwin theory of Evolution and natural selection of organism [26] which tracks the classification as generating the initial population, evaluation, selection, crossover, mutation, and followed by regeneration as can be noticed in the algorithm flow chart [27]. The history of biological evolution, natural selection and its simulation draw back to 1950's. Alex Fraser the early pioneer in this area publish his research in 1957 [28][29]. However, John Holland firstly established the theoretical foundation of Genetic Algorithm in 1975 [30], with which Genetic Algorithm became famous as an Intelligent Optimization Algorithm and can be adopted to solved many hard problems. on the technical side GA is capable of finding the global optimal value and is very well adapted to the problems [31]. While on the contrary Genetic algorithm can easy be trapped into premature convergence which will make it impossible to achieve the best results while GA also require a huge amount of computation time for a big data set of TSP [32][33].

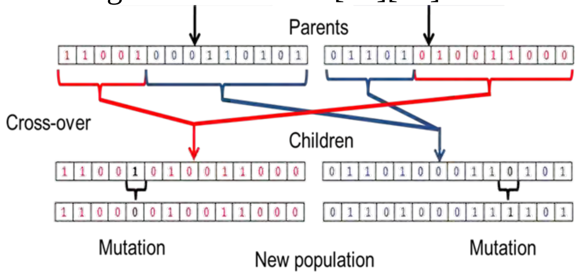

Figure 2. Finite Flow Chart GA Operations

Conclusion

All the meta-heuristics algorithms i.e. Simulated Annealing Algorithm, Genetic Algorithm and ANT Colony Optimization which have discussed above has been run for 10 times with the best parameters known as per initial testing. After obtaining the best parameters then the algorithms have been run for 10 times on the five Benchmark of TSPLIB varying from 318 cities to 1655 cities. The Benchmarks which has been used as test bed are D1655, KroA150, Lin318, Rat575 and Rat 783.

The simulated annealing algorithm has been run for the above-mentioned Benchmarks for 10 times and the results obtained are being categorized Optimum Results which is already known from the TSPLIB website, Best solution has been taken from the 10 test runs and mean has been taken as average results from all the 10 test runs. The parameters which has been set to get the Sub-Optimal results are: Initial Temperature: 1e10 , Cooling Rate:0.998, Ending Temperature:0.001, Number of Cities:As per TSP Library.

The Genetic Algorithm has been run for the above-mentioned Benchmarks for 10 times and the results obtained are being categorized Optimum Results which is already known from the TSPLIB website, Best solution has been taken from the 10 test runs and mean has been taken as average results from all the 10 test runs. The parameter set for the algorithms were as follows: Population: As per TSP Library, Number of Iterations:1e4, Crossover operator: Two points crossover, Mutation operator: One mutation at a random, Crossover probability:0.75, Mutation probability:0.01.

The ANT Colony Optimization Algorithm has been run for the above-mentioned Benchmarks for 10 times and the results obtained are being categorized Optimum Results which is already known from the TSPLIB website, Best solution has been taken from the 10 test runs and mean has been taken as average results from all the 10 test runs. The parameter set for the algorithms were as follows: No. of Cities:As per TSP Library, Pheromone Rate (α):1, Heuristic Rate(β):3, Evaporate Rate (ρ):0.5, Iterations:1000.

While the deviation is best taken as the result through the formula

Deviation (%)= (Optimum-Best)/Optimum.

All the three meta heuristic has been run on the benchmark of the 5 TSPLIB varying from 150 to 1655 cities for the ten rounds. Best Result and Average Results has been noted while the Deviation in the percentage is also given with respect to the best-known solution from the TSP LIBs.

The above deviation results conclude that the nature inspired algorithms ANT Colony algorithm has outperformed the Evolution based and Annealing algorithms. But it must also be noted that the results vary from instance to instance and it is not always necessary that one algorithm will perform best for every situation. There are certain hybrid algorithms that has been presented recently which is performing more efficiently for many of the cases. While the tuning of the parameters is also one of the key factor in influencing the results.

Table 1 Comparative Results Table

No.	Instances	SAA Deviation	GA Deviation	ACO Deviation
1	D1655	5.62	5.01	0.78
2	KroA150	1.41	1.05	0.05
3	Lin318	2.32	1.77	0.26
4	Rat575	2.38	2.64	0.63
5	Rat783	3.1	3.63	0.69

References

[1] F. Alizadeh, R. M. Karp, L. A. Newberg and D. K. Weisser, "“Physical mapping of chromosomes: A combinatorial problem in molecular biology," in Proc. 4th ACM-SIAM Symp. Discrete Algorithms (SODA) pp. 52–76., 1993.

[2] S. Kirkpatrick, C. D. G. Jr and M. P. Vecchi, "Optimization by Simulate Annealing," 1983.

[3] D. Whitely, T. Starkweather and F. D’Ann, "“Scheduling problems and traveling salesman: The genetic edge recombination operator," in Proc.3rd Int. Conf. Genetic Algorithms pp 133-140,, 1989.

[4] K. Ilavarasi and K. S. Joseph, "International Conference on Information Communication and Embedded Systems (ICICES2014)," in Variants of travelling salesman problem: A survey, 2014.

[5] D. L. B. R. E. C. V. &. C. W. Applegate, The Traveling Salesman Problem; AComputational Study, Princeton, New Jersey.: Princeton University Press,, 2006.

[6] C. H., C. E. Leiserson, R. L. Rivest and C. Stein, Introduction to Algorithms, Second Edition, Massachusetts: MIT Press Cambridge, 2001.

[7] The Traveling Salesman Problem: A case study in local by David S. Johnson and Lyle A. McGeoch, 1995.

[8] E. Ballas and P. Toth, Branch and Bound Algorithms for Traveling Saleman Problems, PITTSBURGH, Pennsylvania: Carnegie-Mellon University, 1983.

[9] M. M. Abid and I. Muhammad, "Heuristic Approaches to Solve Traveling Salesman," TELKOMNIKA Indonesian Journal of Electrical Engineering, vol. 15, no. 2, p. 390 ~ 396, August 2015.

[10] D. M, M. V and C. A, "Ant system: optimization by a colony of cooperating agents.," IEEE Trans Syst Man Cybern, vol. 26, pp. 29-41, 1996.

[11] D. Applegate, R. Bixby, V.Chv´atal and W. Cook, The Traveling Salesman Problem, A Computational Study, Princeton and Oxford: Princeton University Press, 2006.

[12] Applegate, D. L., R. E. Bixby, V. Chvátal and W. J. Cook, "The Traveling Salesman Problem: A Computational Study," in Princeton Series in Applied Mathematics, Princeton, NJ, USA, Princeton University Press, 2007, pp. Chapter 1–5,12–17.
[13] Cook and William, ""History of the TSP." The Traveling Salesman Problem.," Georgia Tech, October 2009.
[14] G. Dantzig, R. Fulkerson and S. Johnson, "Solution of a large traveling salesman problem. P-510," RAND Corporation, Santa Monica, California, USA, 1954.
[15] G. Reinelt, "Tsplib - a traveling salesman problem library.," Journal On Computing, vol. 3, no. 4, pp. 376-384, 1991.
[16] S. Kirkpatrick, C. Gelatt and M. Vecchi, "Optimization by simulated annealing," Science, vol. 220, no. 4598, pp. 671-680, 1983.
[17] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller and E. Teller, "Equation of state calculations by fast computing machines," Journal of Chemical Physics, vol. 21, no. 6, p. 1087–1092, 1953.
[18] R. W. Eglese., "Simulated annealing: A tool for operational research.," European Journal of Operational Research, vol. 46, no. 3, p. 271–281, 1990.
[19] N. M. Sureja and B. V. Chawda, "Random Travelling Salesman Problem using SA," International Journal of Emerging Technology and Advanced Engineering, vol. 2, no. 3, 2012.
[20] M. Dorigo, V. Maniezzo and A. Colorni, "The ant system: Ant autocatalytic optimizing process," Technical Report TR91-016, Politenico di Milano, 1991.
[21] M. Dorigo, G. D. Caro and L. M. Gambardella, "Ant algorithms for discrete optimization," Artificial Life, vol. 5, no. 2, pp. 137–172,, 1999.
[22] Z. R. and H. P. N. A. Y. D. Lee, "Applying Ant Colony Optimization Algorithm to Dynamic Job Shop Scheduling Problems," Int. J. Manufacturing Research, vol. 3, no. 3, pp. 301-320, 2008.
[23] S. Wang and A. Zhao, "An Improved Hybrid Genetic Algorithm for Traveling Salesman problem," IEEE Proc. Int. Conf. Computational Intelligence and Software Engineering, 2009.
[24] Y. W., Y. H. and K. G., "Parallel Search Strategies for TSP’s using a Greedy Genetic Algorithm," IEEE Proc. Int. Conf. Natural Computation, 2007.
[25] Z. W., H. D. and X. Z., "An Improved Greedy Genetic Algorithm for Solving TSP," IEEE Proc. Int. Conf. Natural Computation, 2009.
[26] Z. Raza and D. P. Vidyarthi, "A computional grid scheduling model to minimize turnaround using modified GA," Int. J. of Artificial Intelligence, vol. 3, no. A9, pp. 86-106, 2009.
[27] V. K. Banga, Y. Singh and R. Kumar, "Simulation of Robotic Arm using Genetic Algorithm & AHP," J. Engineering and Technology, vol. 25, pp. 95-101, 2007.
[28] A. Fraser., " Simulation of genetic systems by automatic digital computer Part 1: Introduction," Australian Journal of Biological Science, vol. 10, pp. 484-491, 1957.
[29] A. Fraser., "Simulation of genetic systems by automatic digital computers. Part II. Effects of linkage on rates of advanced under-selection," Australian Journal of Biological Science, vol. 10, pp. 492-499, 1957.
[30] J. H. Holland., "Adaptation in Natural and Artificial Systems," MIT Press, 1957.
[31] Q.-B. Z. and C. Shuyan, "A new ant evolution algorithm to resolve TSP problem," in IEEE Proc. Int. Conf. Machine Learning and Applications, 2007.
[32] W. Yingzi, H. Yulan and G. Kanfeng, "Parallel Search Strategies for TSP’s using a Greedy Genetic Algorithm," in IEEE Proc. Int. Conf. Natural Computation, 2007.
[33] T. G. Battiti R., "The Reactive Tabu Search," ORSA Journal on Computing, vol. 6, no. 2, pp. 126-140, 1994.