Ianna Journal of Interdisciplinary Studies
ISSN:2735-9883 \ E-ISSN:2735-9891

VOL -07 NO 1 2025

ANALYZING GA AND ACO TO SOLVE THE PROBLEM OF TRAVELING SALESMEN

Navdeep Randhawa [1] Manju Bala Goel[2],

[1]Assistant Professor, Department of Electronics & Communication Engineering, Swami Vivekanand Institute of Engineering & Technology, Banur, Punjab-140601

[2]Assistant Professor, Department of Electronics & Communication Engineering, Swami Vivekanand Institute of Engineering & Technology, Banur, Punjab-140601

Abstract:

Despite the widespread and quick advancements in technology today, several complex real-world NP problems continue to baffle researchers. The Travelling Salesman Problem (TSP), Knapsack Problem, Graph Colouring, Vehicle Routing, and other similarly complex puzzles have attracted the interest of numerous academics over the past few decades. TSP has shown to be a useful testing ground for comparing a variety of both old and new algorithms. The TSP issue appears to be fairly straightforward; however it is actually one of the more challenging traditional optimisation problems. It has been demonstrated that trying to solve TSP using standard methods may take more years than we have lived on the planet. Consequently, using heuristic techniques is the only practical choice left.

Keywords: Travelling Salesman Problem, Ant Colony Optimization, Genetic Algorithm

I. INTRODUCTION

For many years, nature has served as the primary source of inspiration for tackling challenging and difficult problems. Researchers are increasingly using nature-inspired heuristic algorithms to tackle NP-hard issues including the Travelling Salesman Problem (TSP), the Knapsack Problem, and vehicle routing. The Travelling Salesman issue is an example of a traditional optimisation issue that cannot be solved using standard methods, especially as the number of cities rises. It will take years to discover the best answer if one uses the standard way to solve TSP and takes into account all feasible tours. Heuristic algorithms are therefore a suitable answer to this issue. In this study, the commonly utilised heuristics Ant Colony Optimisation and Genetic Algorithm to solve TSP are taken into consideration. ACO and GA are contrasted for TSP situations in this study. TSP is shown as:

The TSP is the issue of designing a tour that visits each city exactly once while minimising the overall distance travelled, given a set of cities and known distances between each pair of cities.

The travelling salesman issue [1] is the most popular and extensively researched optimisation problem. Almost all novel approaches to optimisation problems are tested on TSP initially. TSP can be represented as an undirected weighted graph, where cities serve as the graph's vertices, paths serve as its edges, and the length of an edge is equal to the path's distance. It is a minimization problem that begins and ends at a given vertex, visiting each vertex precisely once in between. Despite the fact that it appears straightforward, it is one of the classic optimisation problems that cannot be resolved using a traditional mathematical strategy. Researchers have suggested a number of meta-heuristic ways to handle these kinds of problems, including Ant Colony Optimisation (ACO)[3], Particle Swarm Optimisation (PSO), Simulated Annealing (SA), and Genetic Algorithm (GA)[18]. Here, we'll talk about the ACO and GA methods to TSP. The genetic algorithm is represented as:

One of the first and most effective optimisation methods based on evolution is the genetic algorithm [18]. Search algorithms based on the principles of biological evolution's natural selection process are known as genetic algorithms (GA). The most fundamental idea is that the strong adapt and thrive while the weak perish. In other words, the "Survival of the Fittest" theory and evolution serve as the foundation for optimisation. In order to limit their search to the most promising regions of the state space, GAs might first generate a population of workable solutions and then recombine them. Each workable solution is represented by a chromosome (string), sometimes known as a genotype, and each chromosome is assigned a level of fitness through an assessment or objective function called the fitness function. A chromosome's capacity for survival and procreation depends on its fitness. The number of chromosomes in the population is limited. To evolve a population from one generation to the next, GAs employ probabilistic rules.

Advantages of GA are as follows:

1. It always gives better solutions with time.
2. It supports multi-objective optimization.
3. It is more useful and efficient when search space is large.
4. The GA is well suited to and has been extensively applied to solve complex design optimization problems because it can handle both discrete and continuous variables.

Disadvantages of GA are as follows:

1. When fitness function is not properly defined, GA may converge towards local optima.
2. Operation on dynamic sets is difficult.
3. GA is not appropriate choice for constraint base optimization problem.

Ant colony optimization is depicted as:

The ACO [9] is the first and most well-known method, having been first suggested by Marco Dorigo in his PhD thesis on "Optimisation, Learning, and Natural Algorithms" from 1992. The ACO is modelled after how actual ants find food and navigate around their environment. It is an approach to solving challenging combinatorial optimisation issues that is population-based. The Ant Colony Optimisation algorithm is based on how ants forage; they use a specific chemical called a 'pheromone' to communicate between colonies in order to determine the best route between the colony and a food source in the surrounding environment. The term "stigmergy," which refers to this mechanism, describes indirect communication among the self-organizing agents or acts. Real ants will initially wander aimlessly from their nest to food when it is found. They leave behind a unique chemical 'pheromone' on their route from colony to food. While returning, they also leave some pheromone behind. Because there is more pheromone on the shortest path, ants who follow it return earlier. Because it is the shortest path, this path eventually sees more traffic. Pheromones dissipate at a specific pace. Therefore, longer trails that are not traversed by ants eventually disappear. In order to find the shortest route from their colonies to their food, ants use the history of pheromone trails. The ACO algorithm is used to mimic genuine ant behaviour. Constructing an ant solution, pheromone update, which includes pheromone deposit and daemon action are important parameters in ACO.

The advantages of ACO are as follows:

1. It works on distributed computing.

2. It is robust and also easy to accommodate with other algorithms.

3. ACO algorithms have advantage over Genetic Algorithms approaches of similar problem (such as TSP) when the graph may changes dynamically, the ant colony algorithms can be run continuously and adapt to changes in real time.

Disadvantages of ACO are as follows:

Though ant colony algorithms can solve some Optimization problems successfully, we cannot prove its convergence. It is failed in the local optimal solution because the ACO updates the pheromone according to the current best path.

II. LITERATURE SURVEY

Dorigo and co. [1] we examined how ant colonies operate and proposed the outline of a fresh computational paradigm, which we refer to as the ant system (AS). As a workable fresh method for stochastic combinatorial optimisation, they suggest it. The employment of a beneficial greedy heuristic, distributed computation and positive feedback are the major features of this concept. The greedy heuristic aids in finding workable answers in the early stages of the search process, distributed computation prevents premature convergence, and positive feedback accounts for the quick identification of good solutions. They describe simulation findings after applying the suggested methodology to the well-known travelling salesman problem (TSP).

They compare the model to tabu search and simulated annealing using TSP, and they talk about parameter selection and the early setups of the model. They use the ant system (AS) to solve several optimisation problems, including the asymmetric travelling salesman, the quadratic assignment, and the job-shop scheduling, to illustrate the resilience of the method. Finally, they go over the key features of the AS, including probabilistic transitions, distributed communication, and global data structure change.

Ant colony system (ACS), a distributed technique used to solve the travelling salesman problem (TSP), is introduced by Dorigo et al. in their paper from [6]. In the ACS, a group of cooperative agents known as ants work together to create effective TSP solutions. Ants construct solutions while employing an indirect form of communication that is facilitated by a pheromone they deposit on the edges of the TSP graph. They conduct experiments on the ACS to better understand how it functions. The findings demonstrate that the ACS performs better than other nature-inspired algorithms like simulated annealing and evolutionary computation. We conclude by contrasting ACS-3-opt, a variation of the ACS enhanced with a local search procedure, to some of the top performing algorithms for symmetric and asymmetric TSPs.

Blum C, et al. He discusses how ant colony optimisation methods are biologically inspired. He demonstrates how a discrete optimisation algorithm can be created using this biological inspiration. Then, He covers some of the current best-performing ant colony optimisation variations and provides a more basic overview of ant colony optimisation within the framework of discrete optimisation. Finally, he offered instances of an intriguing current research area after summarising some significant theoretical findings and demonstrating how ant colony optimisation works. the fusion of more traditional artificial intelligence and operations research methodologies.

By Dorigo et al. Numerous methods recently used to solve challenging discrete optimisation issues were modelled after the foraging behaviour of ant colonies. In this paper, we define the Ant Colony Optimisation (ACO) meta-heuristic, which we use to organise various methods into a common framework. Along with a brief summary of current applications, a few paradigmatic examples of this innovative meta-heuristic's applications are provided. They have provided a formal explanation of both the class of issues that the Ant Colony Optimisation meta-heuristic can be applied to as well as the meta-heuristic itself. The travelling salesman problem and routing in packet-switched networks were the next two paradigmatic uses of ACO algorithms that were discussed, and they closed with a brief summary of the present implementations. The study of the formal characteristics of an ant system simplified, the creation of Ant Net for Quality of Service applications, and the development of further applications to combinatorial optimisation issues are the three primary directions of ongoing work.

According to research by Nada M. A. Al Salami et al. [14], a hybrid approach is suggested to handle combinatorial optimisation problems by utilising Ant Colony and Genetic Programming algorithms. Ant Colony Optimisation algorithm's evolutionary process adapts genetic operations to speed up ant movement towards the solution state. By adding up the best subsolutions, the algorithm eventually reaches the ideal result.

May Aye Khine, Member, and others, as well as Zar Chi Su Su Hlaing [15] They stated that the heuristic algorithm known as ant colony optimisation (ACO) is one of the high performance computer techniques for the travelling salesman problem (TSP) and has been successfully applied to a number of combinatorial optimisation issues. One of the most well-known combinatorial optimisation (CO) issues, TSP has a diverse history of applications. To discover the best solution for TSP problems, the ACO algorithm takes too long to converge and gets stuck in local optima. They suggest two enhancements to the ant colony optimisation technique. The candidate set strategy is first modified for rapid convergence. The second is an entropy-based dynamic updating rule for heuristic parameters. On benchmark problems from the TSPLIB, algorithms are evaluated, and the evaluation results are reported. Our tests show that the proposed method performs better than the traditional ACO technique.

Members K.S. Tang, S. Kwong, and K.F. Man[19]They stated that the knowledge of this developing technology may be used with genetic algorithms (GA) to create the framework of an industrial engineer design tool. Additionally, an attempt has been made to justify "why" and "when" GA should be utilised as a tool for optimisation.

Sanghamitra Bandyopadhyay, Shubhra Sankar Ray, Sankar K. Pa, and others [20]They introduced that, a genetic algorithm application to the TSP (travelling salesman problem). A neighbourhood swapping operator and new knowledge-based multiple inversion operators are suggested. In comparison to some other existing methodologies, experimental results on various benchmark data sets have been proven to produce superior outcomes.

Keywords: knowledge based multiple inversion, order crossover, knowledge based neighbourhood swapping.

Gopal M. Pandey, Sharad N. Kumbharana, and others [18]Researchers are increasingly using nature-inspired heuristic algorithms to tackle challenging NP-hard issues including the Travelling Salesman Problem (TSP), Graph Colouring, and Vehicle Routing. It will take years to discover the best answer if one uses the standard way to solve TSP and takes into account all feasible tours. Heuristic algorithms are therefore a suitable answer to this issue. In this study, we investigate three commonly used, nature-inspired heuristic techniques to solve TSP: Simulated Annealing, Genetic Algorithm, and Ant Colony Optimisation. Additionally, we contrast the ACO, GA, and SA findings for typical TSPLIB situations.

III. GA METHOD OF SOLVING TSP

The various "encoding," "crossover," and "mutation" processes that we have seen so far can be combined to produce multiple genetic algorithms that can be used to solve the "travelling salesman problem." We do not have too many distinct genetic algorithms to investigate because it is obvious that some crossover methods can only be employed with a particular type of encoding. Additionally, only a limited number of ways have been tried, therefore we will only focus on them. Finally, we will remember that since these programmes have been tested on various issues, it will be challenging to compare them to one another.

GA uses three operations to maintain population that evolve from one generation to another.

- The first operation is “Selection” operation.
- The second operation is the “Crossover” operation.
- The third operation is “Mutation”.

STEPS FOR THE GENETIC ALGORITHM:

Step 1. DETERMINE AN INITIAL POPULATION

a) Random or

b) By some Heuristic

Step 2. REPEAT

A. Find out each person's level of fitness in the population. (Apply the goal function to every member of the population.) At this point, fitness scaling can be utilised. Fitness Scaling raises the fitness values of the lesser performers while lowering the fitness values of the top performers, encouraging rivalry amongst the strings. The truly awful strings will become less prevalent as the population ages. One illustration is linear scaling.

B. Determine which strings are "copied" or "selected" for the mating pool, as well as how often a string will be "selected" for the mating pool. More frequently than not, higher performers will be copied than lower performers. An illustration is the chance n of choosing a string with a fitness value of f, where ft is the total of the population's fitness values.

C. Crossover:

1. Mate each string randomly using some crossover technique.

2. For each mating, randomly select the crossover position(s).

(Note one mating of two Strings produces two strings. Thus the population size is preserved).

D. Mutation: Genes on chromosomes undergo random mutations. Although uncommon, mutation is vital. Consider changing a gene with a probability of.005 as an example. The likelihood of a mutation on any one gene in a population with g total genes (g = string length * population size) is 0.005g, for instance. Most of the time, this step is unnecessary. Every area of the issue space can be reached thanks to mutation. A gene that has changed is randomly chosen and substituted with a different alphabetic symbol. UNTIL there are no more generations to produce.

IV. ACO METHOD OF SOLVING TSP

The travelling salesman dilemma is the issue that a salesperson faces when trying to determine the quickest route through a collection of customer cities, stopping in each one once before finally making his way back home. A full weighted graph G = (N, A) can be used to represent the TSP, where N is the set of n=|N| nodes (cities), and A is the set of arcs fully linking the nodes. The weight dij, which corresponds to the separation between the cities i and j, is given to each arc (i,j) of type A. Finding a Hamiltonian circuit of the graph with the shortest possible length is the TSP. A Hamiltonian circuit is a closed walk (a tour) that stops at each node of G precisely once. There are two types of TSPs: symmetric TSPs (dij = dji for every pair of nodes) where the distances between the cities are independent of the direction of traversing the arcs, and asymmetric TSPs (dij dji for at least one pair of nodes i, j). An ACO algorithm [5] can be conceptualised as the interaction of three processes: The steps to build ant solutions, update pheromones, and activate daemons are as follows:

Step 1. Construct Ants Solutions: This programme controls a colony of ants that simultaneously and asynchronously go to neighbouring states of the problem under consideration by passing through nodes on the problem's construction graph (GC). They move by employing a stochastic local decision process that draws on heuristic data and pheromone trails. Ants solve the optimisation challenge in this way by constructing small, incremental changes. The ant assesses the (partial) solution that will be utilised by the Update Pheromones operation to determine how much pheromone to deposit once the solution has been formed, or while the solution is being built.

Step 2. Pheromone updates: This is the procedure used to change pheromone trails. As ants leave pheromone behind on the parts or connections they use, the value of the trail might either rise or fall as pheromone evaporates. Practically speaking, the deposit of new pheromone improves the likelihood that those parts/connections that were either used by many ants or that were used by at least one ant and generated an excellent solution will be utilised again by subsequent ants. On the other hand, pheromone evaporation employs a valuable type of forgetting: it prevents an algorithm's too quick convergence towards a suboptimal location, favouring the exploration of new regions of the search space.

Step 3. Daemon tasks: This method is used to carry out centralised tasks that are impractical for a single ant to carry out. The activation of a local optimisation technique and the gathering of global data, which can be used to determine if it is desirable to deposit extra pheromone to bias the search process from a nonlocal perspective, are examples of daemon actions. In practise, this means that the daemon can watch each ant's path and choose one or a small group of them (perhaps those who constructed the best solutions during the algorithm iteration) to allow them to add more pheromone to the connections and components they used. The optional method Daemon activity describes centralised activities carried out by a daemon with global awareness.

V. EXPERIMENTAL RESULTS:

Two algorithms—ACO and GA—are taken into consideration in this study to address the travelling salesman problem. The Java code is run on the Windows 7 platform. Genetic algorithms are better because their worst cases are handled more quickly and on average. On the other hand, the ant colony optimisation technique takes longer because in an ant colony, the optimised path must be followed, and choosing the ideal path takes more time.

TABLE 1: PERFORMANCE OF GA FOR TSP

No. of CITIES	Genetic Algorithm		
	Best	Worst	Average
City10	153	163	156
City15	215	225	218
City30	184	194	187
City35	199	209	202

TABLE 2: PERFORMANCE OF ACO FOR TSP

No. of CITIES	ACO		
	Best	Worst	Average
City10	427	437	430
City15	237	247	240
City30	403	413	406
City35	377	387	380

TABLE 3: ITERATIONS RESULT OF GA FOR TSP

No. of Cities	Shortest Path	Longest Path
City10	34	36
City15	24	50
City30	33	36
City35	11	57

TABLE 4: ITERATION RESULT OF ACO FOR TSP

No. of Cities	Shortest Path	Longest Path
City10	32	91
City15	61	97
City30	76	81
City35	43	63

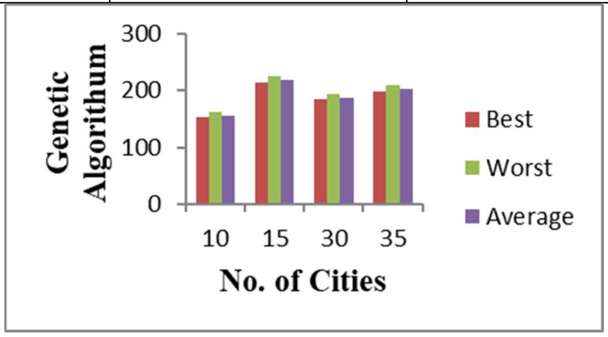

Fig. 1 shows graph of Genetic Algorithm for TSP

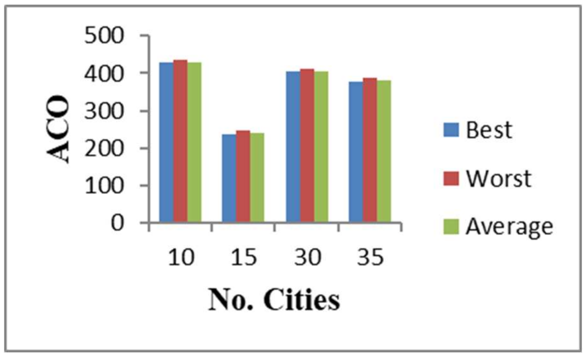

Fig. 2 Shows graph of ACO for TSP

VI. CONCLUSION & FUTURE WORK:

The two most popular optimisation algorithm approaches, ACO and GA, are compared in this research. Ants employ the ACO method to find food sources. They employ a process called pheromone trail deposition/evaporation. GA is an optimisation technique that draws its inspiration from the survival of the fittest hypothesis and the rule of biological reproduction, using crossover and mutation processes to identify the local best solution. The experimental findings related to TSP challenges are displayed in Section 4. It demonstrates that GA offers more superior options than ACO. Future studies could also take into account using the Particle Swarm Optimisation Algorithm to assess how well it performs in comparison to GA and ACO for TSP.

REFERENCES:

[1] Dorigo M, Maniezzo V, Colorni A. Ant System: Optimization by a colony of cooperating agents. IEEE Trans Syst Man Cybernet Part B 1996; 26(1):29–41.

[2] M. Dorigo. 1992. Optimization, Learning and Natural Algorithms (in Italian). Ph.D. Thesis, Dipartimento diElettronica, Politecnico di Milano, Italy.

[3] Blum C. Theoretical and practical aspects of Ant colony optimization. Dissertations in Artificial Intelligence Berlin: Akademische Verlagsgesellschaft Aka GmbH; 2004.

[4] Blum.C. Ant colony optimization : Introduction and recent trends, ALBCOM, LSI, Universitat Politècnica de Catalunya, Jordi Girona 1-3, Campus Nord, 08034 Barcelona, Spain, 2005.

[5] Dorigo M, Stützle T. Ant Colony optimization. Cambridge, MA: MIT Press; 2004.

[6] Dorigo, M., & Gambardella, L. M. (1997a). Ant colonies for the travelling salesman problem. BioSystems, 43(2), 73–81.

[7] Bullnheimer B, Hartl R, Strauss C. A new rank-based version of the Ant System: A computational study. Central European J Operations Res Econom 1999;7(1):25–38.

[8] Stützle T, Hoos HH. Improvements on Ant-System: IntroducingMAX–MIN Ant system. Future Generat Comput Syst 2000; 16(8):889–914.

[9] Dorigo, M. & Di Caro, G. (1999a). Ant colony ptimization: A new meta-heuristi In P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, & A. Zalzala (Eds.), Proceedings of the 1999 Congress on Evolutionary Computation (CEC'99) (pp. 1470-1477). Piscataway, NJ, IEEE Press.

[10] T. Stützle and H. H. Hoos, Improving the ant-system: A detailed report on the MAX-MIN ant system FG Intellektik, TU Darmstadt, Germany, Tech. Rep. AIDA 96- 12, 1996.

[11] M. Dorigo, G. Di Caro, and L.M. Gambardella, "Ant algorithm for discrete Optimization", Artificial Life, vol. 5, no. 2, pp. 137-172, 1999. Holland, "Adaptation In Natural and Artificial Systems".

[12] M. Dorigo and L. M. Gambardella. Ant Colony System: A cooperative learning Approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation, 1(1):53–66, 1997.

[13] M. Dorigo and G. Di Caro. The Ant Colony Optimization meta-heuristic. In D. Corne, M. Dorigo, and F. Glover, editors, New Ideas in Optimization, pages 11–32. McGraw Hill, London, UK, 1999.

[14] Nada M. A. Al SalamiUbiCC Journal, Volume 4, Number 3, August 2009.

[15] Zar Chi Su Su Hlaing and May Aye Khine, 2011 International Conference on Information Communication and Management IPCSIT vol.16 (2011) © (2011) IACSIT Press, Singapore.

[16] M. Dorigo and T. Stutzle. Ant Colony Optimization. MIT Press, Cambridge, MA, 2004.

[17] D. S. Johnson and L. A. McGeoch. The travelling salesman problem: A case study in local Optimization. In E. H. L Aarts and J. K. Lenstra, editors, Local Search in Combinatorial Optimization, pages 215–310. John Wiley & Sons, Chichester, UK, 1997.

[18] Sharad N. Kumbharana1, Prof. Gopal M. Pandey2International Journal of Societal Applications of Computer Science Vol 2 Issue 2 February 2013ISSN 2319 – 8443.

[19] K. F. Man, Member, IEEE, K. S. Tang, and S. Kwong, Member, IEEE, IEEE, VOL. 43, NO. 5, OCTOBER 1996.

[20] Shubhra Sankar Ray, Sanghamitra Bandyopadhyay, and Sankar K. Pal, "First International Conference, PReMI 2005, Lecture Notes in Computer Science, vol. 3776, pp. 617-622, 2005, Springer-Verlag, Berlin, 2005.