A Comparative Study of ACO, GA and SA for Solving Travelling Salesman Problem

Sharad N. Kumbharana[1], Prof. Gopal M. Pandey[2]

[1]*PG Scholar, IT Department, Shantilal Shah Engineering College, Sidsar, Bhavnagar, Gujarat – India*

[2]*Incharge Head, IT Department, Sir Bhavsinhji Polytechnic, Bhavnagar, Gujarat – India*

Abstract

Nature has been main source of inspiration for solving hard and complex problems for many years. Nature Inspired heuristic algorithms is getting more popular among the researcher for solving real world NP hard problems like Travelling Salesman Problem(TSP), Knapsack Problem, Graph Colouring, Vehicle Routing etc. One such classical optimization problem is Travelling Salesman Problem, which is NP hard problem that cannot be solved conventionally particularly when no. of cities increase. If one tries to solve TSP using conventional approach by considering all possible tours, it will take years to find optimal solution. So, Heuristic algorithm is the feasible solution to such problem. In this paper we consider three widely used nature inspired heuristic approaches Ant Colony Optimization, Genetic Algorithm and Simulated Annealing to solve TSP. We also compare results of ACO, GA and SA for standard TSPLIB instances.

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

I. INTRODUCTION

Travelling salesman problem is most studied optimization problem and it has been most favourable among the researchers. Almost every new approach for solving Optimization problems is first tested on TSP. Though it looks simple, it is one of the classical optimization problems which cannot be solve by conventional mathematical approach. To solve these type of problems researchers have proposed various meta heuristic approaches like Ant Colony Optimization (ACO), particle Swarm Optimization (PSO), Simulated Annealing (SA), Memetic Algorithm(MA), Genetic Algorithm (GA), Firefly Algorithm (FA). In this paper we'll discuss ACO, GA and SA approaches for solving TSP in next subsection.

II. ANT COLONY OPTIMIZATION

The ACO is the oldest and widely used approach proposed by Marco Dorigo in 1992 in his PhD thesis "Optimization, learning, and Natural Algorithms" [1]. The ACO is inspired by the food search behaviour of real ants and their ability in finding the optimum paths. It is a population-based general search technique for the solution of difficult combinatorial optimization problems

A. Inspiration

The Ant Colony Optimization algorithm is inspired by the foraging behaviour of ants, they use special chemical 'pheromone' to communication within colonies to find optimum path between the colony and a food source in an environment. This mechanism is called 'stigmergy' means indirect communication among the self-organizing agents or actions.

B. Metaphor

When food is located, real Ants initially roams randomly from their colony to food. They deposit special chemical 'pheromone' on their path from colony to food. They also deposit some amount of pheromone while returning. Thus the ants following shortest pathreturn earlier and amount of pheromone on that path is more. So after certain time this path has the more traffic because this is the shortest path.The Pheromones is evaporated at certain rate. So, longer paths which are not used by ants are eliminated after sometime. Thus ants using the history in terms of pheromone trail to search shortest path from colonies to their food. ACO algorithm is employed to imitate the behaviour of real ants and is as follows:[1][4]

C. Pseudo Code of ACO

```
begin procedure ACO
    generate pheromone trails and other parameters
    while(termination criteria not meet)
    {
        construct solutions
```

update pheromone Trails
}
post-process results and output
end *ACO procedure*

D. Working of ACO

Key parameters in ACO are distances between two ants,pheromone update which includes pheromone deposit, and pheromone evaporation.

Initially ants start their search roams randomly. An ant selects the next node to be visited by probabilistic equation. When ant k is on node I, the probability of going to node j is given by equation as follows:[2]

$$p_{ij}^{k} = \frac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{l \in N_i^k}(\tau_{il})^{\alpha}(\eta_{il})^{\beta}} \quad if \; j \in N_i^k$$

Where, N_i^k is the adjacent node of k which still not visited by ant i. α is the local pheromone coefficient that controls the amount of contribution pheromone plays in a components probability of selection and is commonly set to 0.1. β is the heuristic coefficient which controls the amount of contribution problem-specific heuristic information plays in a components probability of selection and is commonly between 2 and 5, such as 2.5.η = 1/d which is the inverse od distance between I and j.The total number of ants (m) is commonly set low, such as 10. The arcs which is used by the most ants and which is the shortest, receives more pheromone and will be used by the ants in future.

Another main function is pheromone update which consists of pheromone deposit and pheromone evaporation. Pheromone values are updated each time an ant travels from one node to another. A first Pheromone value on each arc is decreased by constant factor which is known as pheromone evaporation. Then some amount of pheromone is added to each node which is being traversed by each ant, is known as pheromone deposit. Pheromone evaporation is given by equation follows:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij}$$

Where, ρ is the evaporation rate.

Each ant drops some amount of pheromone on each node which is known as pheromone deposit and given by equation follows:

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$$

Where,

m is the number of ants,

$\Delta\tau_{ij}^{k}$ is the amount of pheromone drop on k node. It is calculated as

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{1}{C^k} & if \; arc(ic,jb) \; belongs^k \; to \; T \\ 0, & otherwise \end{cases}$$

Where C^k is the length of the tour by the K^{th} ant.

E. Advantages of ACO

1. It has advantage of distributed computing.
2. It is robust and also easy to accommodate with other algorithms. [5]
3. ACO algorithms have advantage over simulated annealing and Genetic Algorithms approaches of similar problems (such as TSP) when the graph may change dynamically, the ant colony algorithms can be run continuously and adapt to changes in real time. [5]

F. Limitations of ACO

1. Though ant colony algorithms can solve some optimization problems successfully, we cannot prove its convergence. [9]
2. It is prone to falling in the local optimal solution. because the ACO updates the pheromone according to the current best path.[7]

III. GENETIC ALGORITHMS

A Genetic Algorithm is one of the oldest and most successful optimization technique based on natural of Evolution. It was originally proposed by John Holland ain the 1960s at the University of Michigan, to study the process of evolution and adaption occurring in nature. [8]

A. Inspiration

Genetic Algorithm is inspired by Charles Darwin's theory of evolution and natural selection. This uses survival of the fittest approach for selecting the best (fittest) solution from the available solutions.

B. Metaphor

This approach uses the population of Chromosomes as the starting point then each chromosome is tested against fitness using an appropriate fitness function. Then the best chromosomes are selected and they undergo process of crossover and mutation to create new set of chromosome.

C. Pseudo Code of GA

begin *procedure GA*
generate populations and fitness function
evaluate population
while*(termination criteria not meet)*
{

```
    while (best solution not meet)
    {
        crossover
        mutation
        evaluate
    }
  }
  post-process results and output
end GA procedure
```

D. Working of GA

GA uses three operations to maintain population that evolve from one generation to another. The first operation is "Selection" operation which is inspired by the principle of 'Survival of the Fittest'. The search begins from a randomly generated population that evolve over successive generations (iterations).A fitness function is used to evaluate the performance of the solutions. Each time two solutions are chosen as parent solutions by selection process based on fitness function. The second operation is the "Crossover" operation, which is inspired by mating in biological populations. The crossover operator inherits features of good surviving designs from the parent population into the future population, which will have better fitness value on average. The third operation is "Mutation" which causes diversity in population characteristics. It causes local modifications to the new generation randomly. The new generation is identical to the parent except one or more changes made by mutation process. Repeat selection, crossover and mutation operations to produce more new solutions until the population size of the new generation is the same as that of the old one. The iteration then starts from the new population. Since better solutions have a larger probability to be selected for crossover and the new solutions produced carry the features of their parents, it is hoped that the new generation will be better than the old one. The procedure continues until the number of generations is reached to n or the solution quality cannot be easily improved.

E. Advantages of GA

1. It always gives solution and solution gets better with time. [5]
2. It supports multi-objective optimization. [5]
3. It is more useful and efficient when search space is large, complex and poorly known or no mathematical analysis is available. [11]
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, and nonlinear objective functions without requiring gradient information.[12]

F. Limitations of GA

1. When fitness function is not properly defined, GA may converge towards local optima.[11]
2. Operation on dynamic sets is difficult.[11]
3. GA is not appropriate choice for constraint based optimization problem.[11]

IV. SIMULATED ANNEALING ALGORITHM

Simulated Annealing (SA) is the oldest probabilistic meta-heuristic algorithm and one of the first algorithms having ability to avoid being trapped in local minima. It was first presented by Kirkpatrick, Gelatt and Vecchi in 1983. [10]

A. Inspiration

Simulated Annealing is inspired by the process of annealing in metallurgy. In this process a material is heated and slowly cooled into solid crystal state with minimum energy and larger crystal size to reduce defects in metallic structures. The heat increases the energy of the atoms allowing them to move freely, and the slow cooling schedule allows a new low-energy levelto be discovered. This annealing process requires the careful control of temperature and cooling rate. [13]

B. Metaphor

Each set of a solution represents a different internal energy of the system. Heating the system, results in a relaxation of the acceptance criteria of the samples taken from the search space. As the system is cooled, the acceptance criteria of samples are narrowed to focus on improving movements. Once the system has cooled, the configuration will represent a sample at or close to a global optimum. [13]

C. pseudo code for SA

```
begin procedure SA
  generate initial solutions
  set temperature, and cooling rate
  while(termination criteria not meet)
  {
      generate new solutions
      access new solutions
      if (accept new solution)
      {
          update storage
          adjust temperature
      }
  }
  post-process results and output
end GA procedure
```

D.Working of SA

The SA algorithm allows the search to sometimes accept worst solutions with a probability (p) that would decrease with the temperature of the system (t). In this way, the probability of accepting a solution that resulted in a certain increase in the objective Function (Δf), at a certain temperature, is given by the following formula described in the original paper by Kirkpatrick [10]:

$$p(\Delta f, t) = \begin{cases} e^{-\frac{\Delta f}{t}}, & \Delta f \leq 0 \\ 1, & \Delta f > 0 \end{cases}$$

While implementing the SA algorithm, the initial temperature and cooling rate should be set such that the slowerthe temperature is decreased, the greater the chance an optimal solution is found. In fact, Emile H. Aarts showed that running the simulation an infinite number of times are needed to be sure the optimal solution to the problem has been found. [18] Most times a reasonably good cooling schedule can be achieved by using an initial temperature (T_0), a constant temperature decrement (α) and a fixed numberof iterations at each temperature.

E. Advantages of SA [16]

1. It statistically guarantees finding an optimal solution.
2. It is relatively easy to code, even for complex problems
3. SA can deal with nonlinear models, unordered data with many constraints.
4. Its main advantages over other local search methods are its flexibility and its ability to approach global optimality.
5. It is versatile because it does not depend on any restrictive properties of the model.

F. Limitation of SA [17]

1. Continuous annealing with a $\frac{1}{\log k}$ schedule is very time consuming, especially if the cost function needs more computation.
2. SA is not that much useful when the energy landscape is smooth, or there are few local minima,
3. SA is a meta-heuristic approach, so it needs a lot of choices to turn it into an actual algorithm.
4. There is a trade-off between the quality of the solutions and the time needed to compute them.
5. More customization work needed for varieties of constraints and have to fine-tune the parameters of the algorithm.
6. The precision of the numbers used in implementation have a major effect on the quality of the result.

V. EXPERIMENTS AND RESULTS

Here we have considered ACO, GA and SA to solvestandard TSP instances downloaded from TSPLIB [19]. The code is in Matlab and executed on windows 7 platform with Intelcore i5 processor and 3 GB RAM. Table 1 lists TSP file name, no. of cities and lengths of the optimal tour for ACO, GA and SA. TSP instance provides some cities with their coordinates.The Results given here are best, worst and average of 15 runs for ACO, GA and SA.

TSP File	No. of Cities	ACO			GA			SA		
		Best	Worst	Average	Best	Worst	Average	Best	Worst	Average
city10	10	60.50	63.81	61.84	60.50	60.50	60.50	64.19	86.08	76.38
city29	29	9999.20	11151.22	10440.04	9074.15	9197.05	9106..3	19180.68	22332.12	20915.97
city51	51	529.15	578.12	558.04	441.03	456.27	448.56	1352.51	1456.92	1314.16
ulysses16	16	74.00	82.64	78.08	73.99	74.00	74.00	89.77	118.69	106.08
Oliver30	30	452.96	510.01	479.57	423.74	423.74	423.74	424.99	480.14	437.66
att48	48	39930.58	42895.88	41374.88	33818.16	35258.33	36649.40	34980.15	41864.32	35745.58
eli51	51	447.59	478.48	471.27	445.80	464.43	458.21	1275.73	1452.27	1392.72

Table 1: Comparison between ACO, GA and SA for Solving Standard TSP Instances

VI. Conclusion and Future Scope

This paper presents a comparative view of most widely used optimization algorithm techniques namely ACO, GA and SA Optimization. In sections II, III and IV we have outlined these three meta-heuristic approaches as they are described in the literature. ACO is the process used by ants to forage food source. They use pheromone trail deposition/evaporation technique to map their way. GA is an optimization technique, inspired by the law of biological reproduction and survival of fittest theory, where mutation and crossover operations used tofind by the local optimal solution. The SA is inspired by annealing process in metallurgy which is a technique of controlled cooling of materials to reduce defects.

Section 5 shows the experimental results obtained on standard TSP problems. It shows that ACO and GA provide almost same results when no. of cities is less. GA is provides more better solutions compared to ACO and SA.

All these three techniques have immense potential and scope of application ranging from engineering to software engineering, and real world optimization problems. These techniques need to be further explored to find their suitability to certain applications. Also, there is a need to combine two or more techniques so that they complement each other and nullify their respective limitations

Acknowledgement

We thank all who have supported us for this research.

References

1. M. Dorigo, L. Gambardella, *"Ant colonies for the Traveling salesman problem."* Biosystems 43, pp. 73-81, 1997
2. M. Dorigo, T. Stutzle, *"Ant Colony optimization"*, A Bradford book, MIT Press Cambridge, MassachusettsLondon, England, 2004 .
3. M. Dorigo, V. Maniezzo, A. Colomi, "The Ant System: Optimization by a colony of cooperating agents", IEEE Transactions on Systems, Man and Cybematics-Part B, Vol. 26, No.1, 1996.
4. Dorigo M, Gambardella L M, "*Ant colony system: a cooperative learning approach to the traveling salesman problem",* IEEE Trans. On Evolutionary Computation, vol-1(1), pp.53-66, 1997.
5. Shwetasinghal, shivangigoyal, shubhragoyal and divyabhatt, "*A comparative study of a class of Nature Inspired Algorithms"*, Proceedings of the 5th national conference :INDIACom, March 10-11, 2011
6. N. Sureja, B. Chawda, *"Random Travelling Salesman problem using Genetic Algorithms*," IFRSA's International Journal Of Computing,Vol. 2, issue 2, April 2012
7. Peiyi Zhao, Peixin Zhao, Xin Zhang, "*A New Ant Colony Optimization for the Knapsack Problem"*, 2007
8. Melanie Mitchell.,"*An Introduction to Genetic Algorithms"*. MIT Press, Cambridge,MA, 1998.
9. X. Yuneig, G. Junen, G.Bo, "*An Ant Colony Optimization Algorithm based on the Nearest Neighbour Node Choosing Rules and the Crossover Operator"*, International Conference on Computer Science and Software Engineering, 2008
10. S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi. "*Optimization bysimulated annealing"*. Science, Number 4598, vol. 220, 4598, pp. 671-680, 13 May 1983.
11. Binitha, S.S. Siva sathya, "*A survey of bio inspired optimization algorithm",* IJSCE, Vol-2, issue-2, May 2012
12. Rania Hassan,BabakCohanim, Olivier de Weck, "*A comparison of PSO and GA"*, American Institute of Aeronautics and Astronautics,2004.
13. Jason Brownlee, "*Clever Algorithms -Nature Inspired Programming Recipes"*, ISBN: 978-1-4467-8506-5, First Edition. LuLu., January 2011
14. Xin-She-Yang, "*Introduction to Mathematical optimization"*, ISBN 978-1-904602-82-8, Cambridge International Science Publishing, 2008
15. N. Sureja, B. Chawda, "*Random Travelling Salesman Problem usingSA,*" International Journal of Emerging Technology and AdvancedEngineering, Volume 2, Issue 4, April 2012
16. http://cs.adelaide.edu.au/~paulc/teaching/montecarlo/node139.html
17. http://cs.adelaide.edu.au/~paulc/teaching/montecarlo/node140.html
18. Emile H. Aarts and Jan Korst. "*Simulated annealing and Boltzmann machines: a stochastic approach to combinatorial optimization and neural computing"*. N.Y. John Wiley and Sons, New York, NY, USA, 1989.
19. TSPLIB95: Ruprecht - Karls - UniversitatHeildelberg (2011),http://www.iwr.uiniheidelberg.de/groups/comopt/software/TSPLIB95/