Implementing Intelligent Behavior of ACO for Calculating Optimized Cost

Geetu[1] Dr. Anand Sharma[2]

[1]Research Scholar (Ph.D. Comp. Applications), Guru Kashi University, Talwandi Sabo, Punjab, India.

[2]Assistant Professor, UCCA, Guru Kashi University, Talwandi Sabo, Punjab, India.

Abstract – The research paper is primarily focused on discussing the optimization techniques relevant to load balancing in cloud computing. Load balancing in cloud computing is the process in which workloads and computing resources are distributed across more than one server. Load balancing helps in maintaining system firmness, performance and protection against system failures. Load balancing allows enterprises to manage the application or workload demands by allocating resources among multiple computers, networks or servers. Cloud load balancing involves hosting the distribution of workload traffic and demands that reside over the Internet. By spreading the work evenly, load balancing improves application responsiveness. It also increases the availability of applications and websites for users. The research paper elaborated on the importance of load balancing in the cloud environment and detailed about the prominent optimization techniques of Swarm Intelligence and Ant-Colony Optimization. The research paper also illustrates the implementation of Ant-Colony Optimization through an example using 200 iterations to calculate the best cost.

Keywords – Ant-Colony Optimization, Best Cost, load balancing, Swarm Intelligence, Particle Swarm Intelligence.

I. INTRODUCTION – CLOUD LOAD BALANCING

Load balancing is a method for distributing tasks onto multiple computers. The workload is segregated among two or more servers, hard drives, network interfaces or other computing resources, enabling better resource utilization and system response time. A load balancing technique makes sure that each system in the network has the same amount of work at any instant of time. This means neither any of them is excessively over-loaded, nor under-utilized [1]. For instance, distributing incoming HTTP requests (tasks) for a web application onto multiple web servers. There are a few different ways to implement load balancing. The primary purpose of load balancing is to distribute the workload of an application onto multiple computers, so the application can process a higher workload. The load balancer distributes data depending upon how busy each server or node is. In the absence of a load balancer, the client must wait while his process gets processed, which might be too tiring and demotivating for him. Load balancing is a way to scale an application [2]. A secondary goal of load balancing is often (but not always) to provide redundancy in your application. That is, if one server in a cluster of servers fails, the load balancer can temporarily remove that server from the cluster, and divide the load onto the functioning servers [3]. Having multiple servers help each other in this way is typically called "redundancy". When an error happens and the tasks are moved from the failing server to a functioning server, this is typically called "failover". A set of servers running the same application in cooperation is typically referred to as a "cluster" of servers [4]. The purpose of a cluster is typically both of the above two mentioned goals: To distribute the load onto different servers and to provide redundancy/failover for each other. The advantages of using cloud load balancing as compared to conventional counterparts are mentioned as under [5].

- Applications having higher accomplishments
  Cloud load balancing techniques are less expensive and simple to implement. Enterprises can make their client applications work faster and deliver better performances, that too at potentially lower costs.
- Allows an increase in scalability
  Cloud balancing takes the help of the cloud's scalability and agility to maintain website traffic. By using efficient load balancers, you can easily match up the increased user traffic and distribute it among various servers or network devices. It is especially important for e-commerce websites, which deals with thousands

of website visitors every second. During sales or other promotional offers, they need such effective load balancers to distribute workloads.

- Enable handling of traffic spikes
  A normally running University site can completely go down during any result declaration. This is because too many requests can arrive at the same time. If they are using cloud load balancers, they do not need to worry about such traffic surges. No matter how large the request is, it can be widely distributed among different servers for generating maximum results in less response time.
- Continuing business with flexibility
  The basic objective of using a load balancer is to save or protect a website from sudden outages. When the workload is distributed among various servers or network units, even if one node fails the burden can be shifted to another active node.

II. SWARM INTELLIGENCE AND ANT-COLONY OPTIMIZATION

Swarm Intelligence is the seemingly intelligent behavior that emerges from the collective behavior of a large number of autonomous agents. In biology, this term is most widely used with reference to the colony-level behaviors seen in social insects [6]. For instance, whereas individual fire ants struggle and relatively rapidly drown after falling into a pool of water, groups of fire ants link together to form rafts that can float on the water surface for days. Swarm-intelligence principles inspired by the collective insect societies are used for developing computer algorithms and motion control principles for robotics [7]. The basic idea is that a swarm of individuals can coordinate and behave as a single entity that performs better than the individuals. Using cooperative behavior, individuals help each other and solve problems that cannot be handled by single individuals [8].

Particle Swarm Intelligence is another nature-inspired method, mimics the social behavior of bird flocking or fish schooling. It falls into the metaheuristics and swarm intelligence methods class. It is also a population-based stochastic optimization technique, introduced by Kennedy and Eberhart in 1995. Particle Swarm Optimization is characterized into the domain of Artificial Intelligence. The term Artificial Intelligence refers to the theory of simulating human behavior through computation [9]. It involves designing such computer systems that are able to execute tasks that require human intelligence. There are three approaches to artificial intelligence: Statistical Methods, Symbolic Artificial Intelligence, and Computational Intelligence. Computational Intelligence is implemented using either of the three methods: Artificial Neural Network, Fuzzy Logic, and Evolutionary Computation. Under Evolutionary Computation, are the Swarm Intelligence Techniques which include Particle Swarm Optimization [10]. Fig. 1 shows the detailed classification of different approaches to Artificial Intelligence.

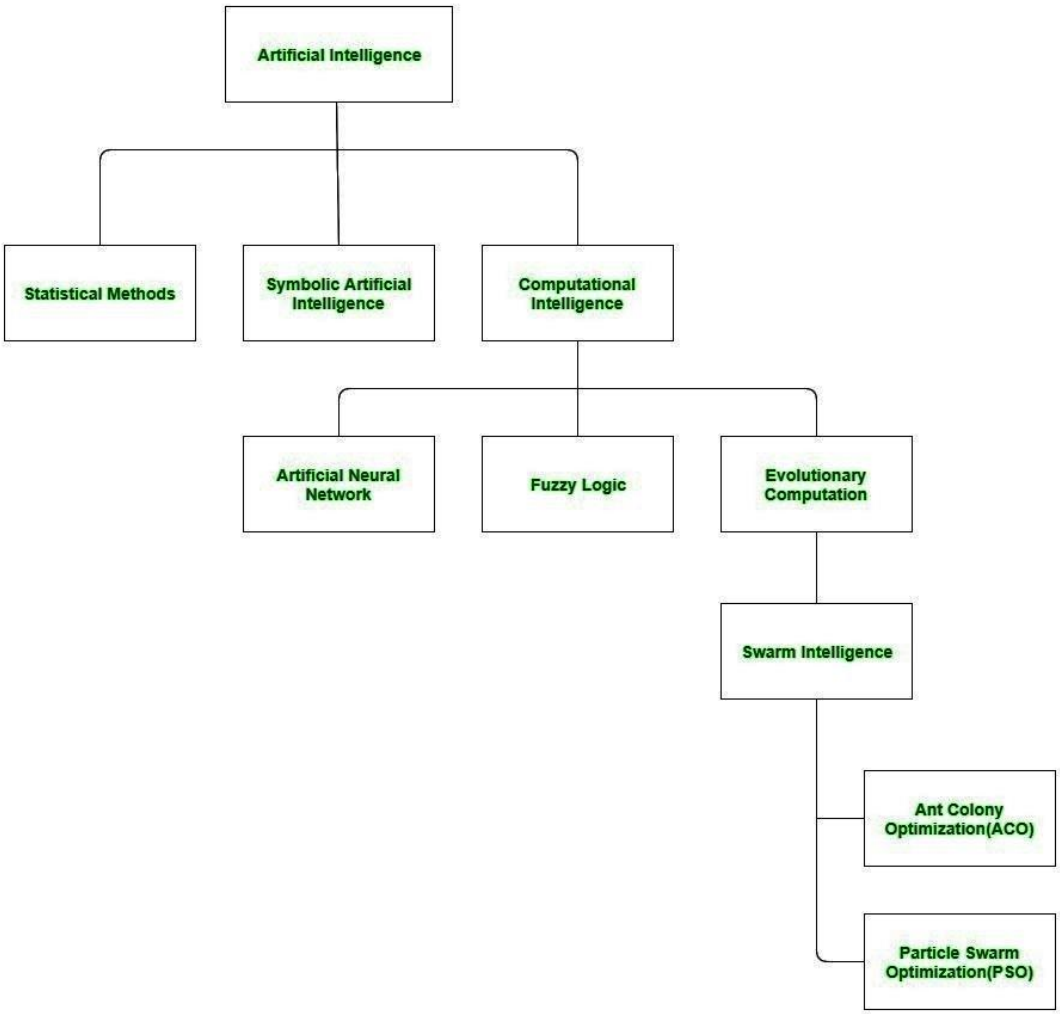

Fig. 1. The figure shows the classification of different approaches to Artificial Intelligence

In computational science, particle swarm optimization (PSO) is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. Swarm intelligence (as the name suggests) comes from mimicking nature. Swarms of social insects, such as ants and bees, operate using a collective intelligence that is greater than any individual member of the swarm. Swarms are therefore highly effective problem-solving groups that can easily deal with the loss of individual members while still completing the task at hand -- a capability that is very desirable for a huge number of applications [11]. Today this concept is being applied in concert with machine learning and distributed computing systems. A result is a group of connected machines that can communicate, coordinate, learn and adapt to reach a specific goal [12].

Ant colony optimization (ACO) is a population-based metaheuristic that can be used to find estimated solutions to challenging optimization problems. In ACO, a set of software agents called artificial ants hunt for good solutions to a given optimization problem. To apply ACO, the optimization problem is transformed into the problem of finding the best path on a weighted graph. The artificial ants incrementally build solutions by moving on the graph. The solution construction process is stochastic and is influenced by a pheromone model, that is, a set of parameters associated with graph components (either nodes or edges) whose values are modified at runtime by the ants [13]. Each ant starts from a randomly selected city (vertex of the construction graph). Then, at each construction step, it moves along the edges of the graph. Each ant keeps a memory of its path, and in subsequent steps, it chooses among the edges that do not lead to vertices that it has already visited. An ant has constructed a solution once it has visited all the vertices of the graph. At each construction step, an ant probabilistically chooses the edge to follow among those that lead to yet unvisited vertices. The probabilistic rule is biased by pheromone values and heuristic information: the higher the pheromone and the heuristic value associated with an edge, the higher the probability an ant will choose that particular edge [14]. Once all the ants have completed their tour, the pheromone on the edges is

updated. Each of the pheromone values is initially decreased by a certain percentage. Each edge then receives an amount of additional pheromone proportional to the quality of the solutions to which it belongs (there is one solution per ant). This procedure is repeatedly applied until a termination criterion is satisfied [15].

III. IMPLEMENTATION OF ANT COLONY OPTIMIZATION (ACO)

The research work shown in this section elaborates on the working of Ant Colony Optimization. Different parameters considered in calculating the best cost are mentioned as under.

Iterations	-	Refers to the number of iterations to be executed
colony_size	-	Refers to the number of ants in the colony.
evap_coeff	-	Refers to the evaporation coefficient that will be used in updating the pheromone (must be between 0 and 1).
alpha	-	This refers to the bias of using pheromone deposits in the decision process.
beta	-	Sets the bias for deciding if a shorter path is better.
tau	-	0.0001 * ones(52); Creates a 52 x 52 tau matrix full of 1.
el	-	Set the parameter for eliminating common costs
Q	-	Refers to the positive constant used for updating tau to control the level of exploration undertook by the ants

The dataset of CSV file titled “talwandi.csv” is mentioned in the table below.

Fig. 2 shows the readings of the dataset titled “talwandi.csv”

Node	Coordinate	Section
1	565	575
2	25	185
3	345	750
4	945	685
5	845	655
6	880	660
7	25	230
8	525	1000
9	580	1175
10	650	1130
11	1605	620
12	1220	580
13	1465	200
14	1530	5
15	845	680
16	725	370
17	145	665
18	415	635
19	510	875
20	560	365
21	300	465
22	520	585
23	480	415
24	835	625
25	975	580
26	1215	245
27	1320	315
28	1250	400
29	660	180
30	410	250
31	420	555
32	575	665
33	1150	1160
34	700	580
35	685	595
36	685	610
37	770	610
38	795	645
39	720	635
40	760	650
41	475	960
42	95	260
43	875	920
44	700	500
45	555	815
46	830	485
47	1170	65
48	830	610
49	605	625
50	595	360
51	1340	725
52	1740	245

Fig. 2. The figure depicts the dataset used in the implementation of ACO

Case 1:

The values assigned to the parameters, in this case, are mentioned as under.

Iterations - 200

colony_size - 40

evap_coeff - 0.1

alpha - 15

beta - 20

tau - 0.0001 * ones(52);

el - 0.97

Q - 0.2

Fig. 3 shows the plotted nodes of the dataset as per the readings shown in Fig. 2.

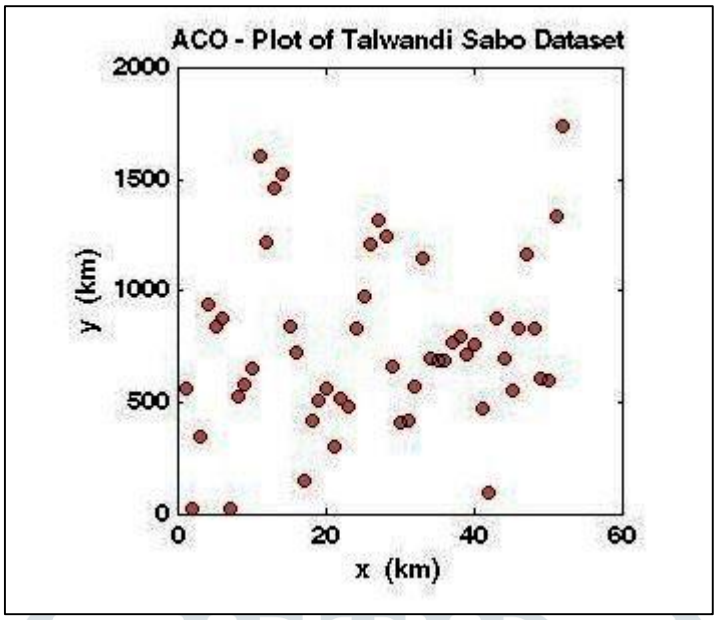

Fig. 3. The figure shows the plotted nodes as per dataset mentioned in Fig. 2

Fig. 4 shows the finally plotted graph depicting the path followed by the ants in carrying 200 iterations with the calculated best cost of 3668.6363.

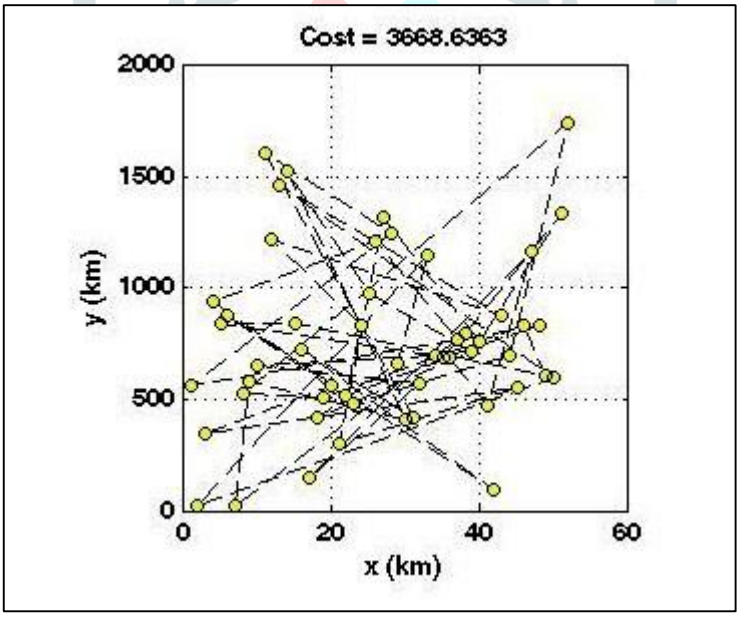

Fig. 4. The figure shows the finally plotted graph depicting the path followed by the ants in carrying 200 iterations

Fig. 5 shows the plotted graph between a number of iterations (X-axis) and an average of tour distance (Y-axis) as per readings of case 1.

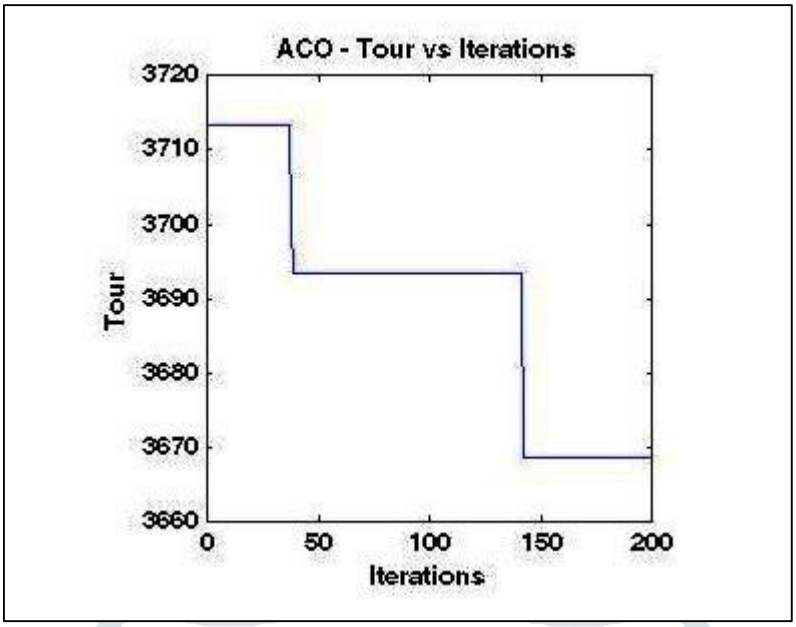

Fig. 5. The figure shows the plotted graph between the number of iterations (X-axis) and an average of tour distance (Y-axis)

Fig. 6 shows the plotted graph depicting the behavior of the ant during 200 iterations and finishing off with the best cost of 3668.6363.

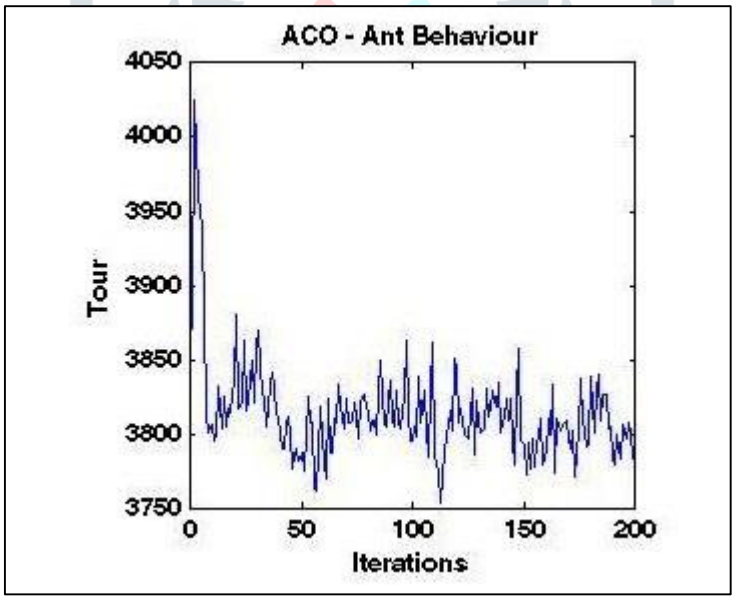

Fig. 6. The figure shows the plotted graph depicting the behavior of the ant during 200 iterations

The above implementation of Ant-Colony Optimization conducted on parameters of Case 1 has shown the calculated best cost of 3668.6363. Similarly, implementation of different cases can be performed using a different number of iterations.

IV. CONCLUSION

The study concludes that time may be needed for organizations to make changes to improve quality and lower costs. ACOs show promise in reducing spending and improving quality and more information about the high-performing ACOs is needed to inform the future direction of the program. The research paper elaborated on the importance of load balancing in the cloud environment and detailed about the prominent optimization techniques of Swarm Intelligence and Ant-Colony Optimization. The research paper also illustrates the implementation of Ant-Colony Optimization through an example using 200 iterations to calculate the best cost. The study concludes that time may

be needed for organizations to make changes to improve quality and lower costs. ACOs show promise in reducing spending and improving quality and more information about the high-performing ACOs is needed to inform the future direction of cloud computing.

REFERENCES

[1]. Liu, C.; Yang, W. A Multi-Objective Task Scheduling Based on Genetic and Particle Swarm Optimization Algorithm for Cloud Computing.Comput. Tech. Dev.2017,27, pp. 56–59.

[2]. Babukarthik, R.G.; Raju, R.; Dhavachelvan, P. Hybrid Algorithm for Job Scheduling: Combining the Benefits of ACO and Cuckoo Search. Adv. Intell. Syst. Comput.2013,177, pp. 479–490.

[3]. Tong, Z.; Chen, H.; Chen, M.; Mei, J.; Liu, H. A hybrid biogeography-based optimization algorithm for task scheduling in cloud computing. J. Comput. Eng. Sci.2018,40, pp. 765–772.

[4]. Zhao, M.; Li, S. Load Balancing of Task Scheduling Based on Ant Colony Optimization in Cloud Computing Environment.Electr. Design Eng.2016, 24, pp. 30–33.

[5]. Ren, J.; Huang, Y.; Zhong, X. Cloud task scheduling improved algorithm based on the genetic algorithm. J. Jiangxi Univ. Sci. Tech.2018,39, pp. 90–94.

[6]. Chandrasekaran, K., & Divakarla, U. (2013). Load balancing of virtual machine resources in cloud using genetic algorithm. National Institute of Technology Karnataka, Surath Nal, pp. 156-168.

[7]. Shetty, S. M., & Shetty, S. (2019). Analysis of load balancing in cloud data centers. Journal of Ambient Intelligence and Humanized Computing, pp. 1-9.

[8]. Gagandeep Jagdev et al., "Analyzing Commercial Aspects and Security Concerns Involved in Energy Efficient Cloud Computing" in International Journal of Scientific and Technical Advancements (IJSTA), ISSN-2454-1532, 2016.

[9]. Yu-Hsin Chen, G. (2013). A New Data Structure of Solution Representation in Hybrid Ant Colony Optimization for Large Dynamic Facility Layout Problems. International Journal of Production Economics, 142(2), pp. 362–371.

[10]. Rao, IAnitha and Hegde, K., Rao, A., Hegde, K., Rao, IAnitha and Hegde, K., Rao, A., & Hegde, S. K. (2015). Literature Survey On Travelling Salesman Problem Using Genetic Algorithms. International Journal of Advanced Research in Education Technology (IJARET), 2(1), pp. 4 - 10.

[11]. Salama, K. M., & Freitas, A. A. (2013). Learning Bayesian Network Classifiers Using Ant Colony Optimization. Swarm Intelligence, 7(2–3), pp. 229–254.

[12]. Liu, Y., Chen, W.-N., Hu, X., & Zhang, J. (2015). An Ant Colony Optimizing Algorithm Based on Scheduling Preference for Maximizing Working Time of WSN. Proceedings of 2015 on Genetic and Evolutionary Computation Conference -GECCO '15, pp. 41–48.

[13]. Moon, Y. J., Yu, H. C., Gil, J. M., & Lim, J. B. (2017). A Slave Ants Based Ant Colony Optimization Algorithm for Task Scheduling in Cloud Computing Environments. Human-Centric Computing and Information Sciences, 7(1), pp. 28 - 35.

[14]. Stũtzle, T., & Dorigo, M. (2002). A Short Convergence Proof for a Class of Ant Colony Optimization Algorithms. IEEE Transactions on Evolutionary Computation, 6(4), pp. 358–365.

[15]. Abdelhafidh, M. (2018). Linear WSN Lifetime Maximization for Pipeline Monitoring using Hybrid K-means ACO Clustering Algorithm, pp. 178–180.