International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:9, No:6, 2015

Performance Comparison of Prim’s and Ant Colony Optimization Algorithm to Select Shortest Path in Case of Link Failure

Rimmy Yadav, Avtar Singh

***Abstract*—**Ant Colony Optimization (ACO) is a promising modern approach to the unused combinatorial optimization. Here ACO is applied to finding the shortest during communication link failure. In this paper, the performances of the prim’s and ACO algorithm are made. By comparing the time complexity and program execution time as set of parameters, we demonstrate the pleasant performance of ACO in finding excellent solution to finding shortest path during communication link failure.

***Keywords*—**Ant colony optimization, link failure, prim’s algorithm.

I. Introduction

In decentralized computing architecture the overall activity is governed by the centralized node called as a master node [3]. Three major reasons limit the efficiency of the centralized architecture, and these are (1) Master node failure, (2) Highly dynamic environment where many resources join, leave, and can change characteristics at any time. (3) Approaches cannot scale well to the geographical distributed system across the Internet [3]. However, in the centralized and distributed computing environment, the fault tolerance is the major issue. Fault tolerance is the ability of the system to work well even in the presence of failure [4]. Faults occur due to the node or server crashes, Communication link failure, extremely dynamic workload, Great Communication delay, massively heterogeneous resources, etc. [3].

In this paper, the link failure problem is covered and ACO algorithm is applied to find the shortest path when link failure occurred in the distributed environment. After applying the ACO to the link failure problem, the performance comparison between the ACO and Prim’s algorithm shows that the ACO algorithm helps to select the shortest path in very minimum time. The organization of this paper is as follows: Section II describes the literature review; Section III describes prim’s and ACO algorithms and their implementations for shortest path selection after link failure; Section IV shows result and discussion; Section V summarizes the brief conclusion of the paper.

Rimmy Yadavis with the Lovely Professional University Jalandhar-Punjab-India (phone: 09501957500; e-mail: rimmy_yadav@ ymail.com).

Avtar Singh Assistant Professor Department of Computer Science G.G.S.Khalsa College Sarhali (Tarn Taran), Punjab-India(e-mail: avtarsidhu22000@gmail.com).

II. Related Work

Bheevgade [1] developed watchdog timer and sanchita algorithm to deal with the faults in the Grid system. Sapre [3] developed built in user transparent error detection mechanism to cover hardware failure. A priority based probe algorithm developed to deal with the faulty processors. Al-Jaroodi [2] developed a delay-tolerant, fault-tolerant algorithm to deal with the fault tolerance in the distributed cloud services. Malladi [4] developed an efficient coordinated check pointing protocol with pessimistic message logging. Zheng and Lyu [5] developed a model in which network nodes are assumed to be process the ability to test certain another network amenity for the presence of failure is employed.

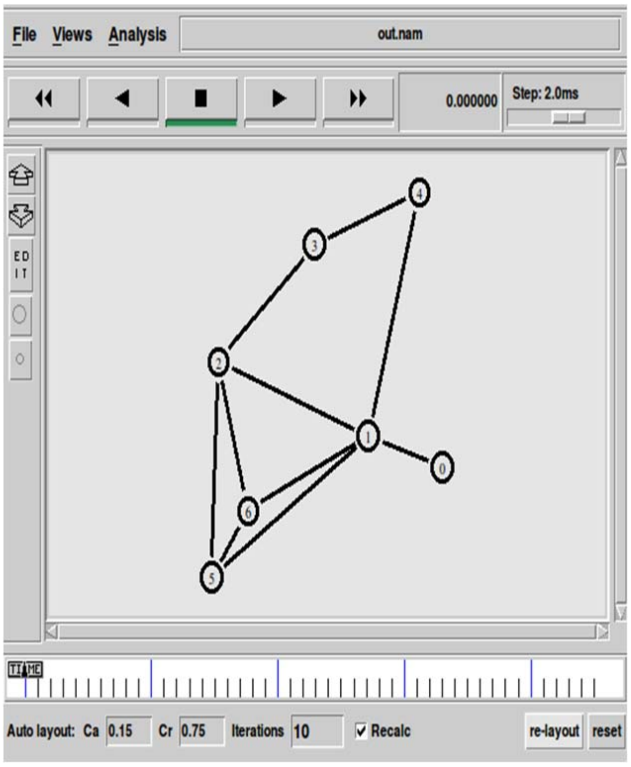
III. Fault Tolerance Using Prim’s and ACO Algorithm

Prim’s algorithm is a greedy approach which finds the shortest path from source node to the destination node from the undirected graph problem. The purpose of the prim’s algorithm is to find a minimum spanning tree for a connected weighted undirected graph. This algorithm finds subsets of the edges forms a tree that includes every vertex or node, where the total weights of all the edges in graph problem in minimized. Network Simulator, i.e. NS2 is used to implement both the algorithms. Consider the following diagram which depicts the nodes connected with arcs.

In Fig. 1, the circle represents the nodes, and these nodes are connected with the edges forming an undirected weighted graph. The set of nodes are: {0, 1, 2, 3, 4, 5 and 6}. Fig. 2 shows the communication between the nodes 0→1, 1→2, 2→3.

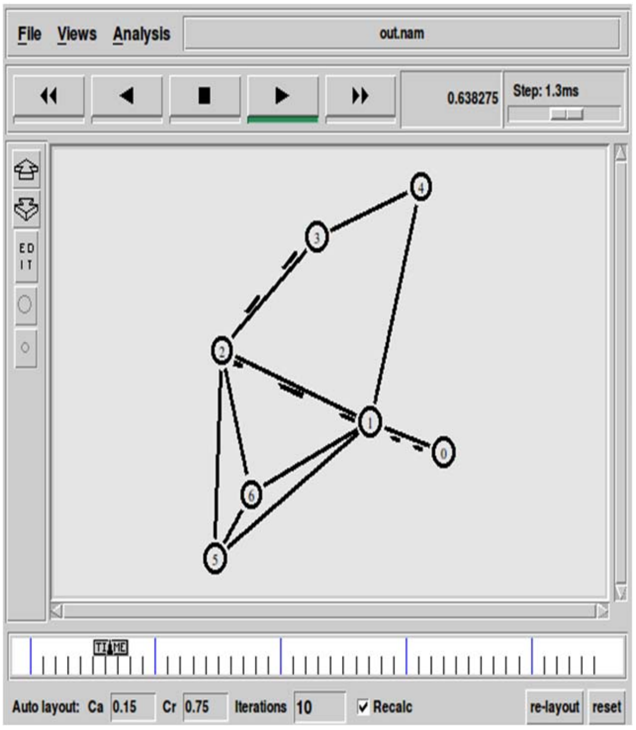
TABLE I
ASSIGN WEIGHT VALUES TO EDGES

Edge	Key Weight Value
0→1	2
1→2	Link Failed
1→4	2
1→5	5
1→6	3
2→3	Link Failed
2→5	4
3→4	3
5→6	4



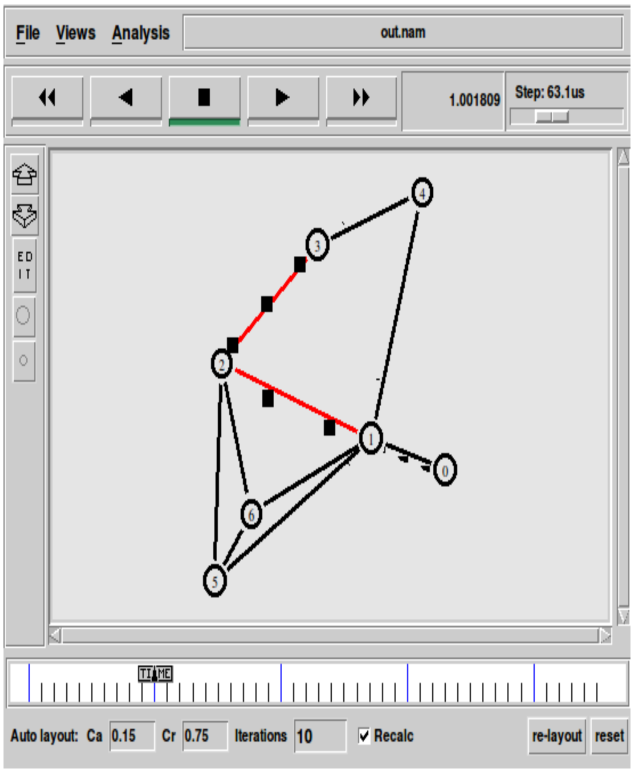
Auto layout: Ca 0.15 Cr 0.75 Iterations 10 ✓ Recalc

Fig. 1 Connection of Nodes



Auto layout: Ca 0.15 Cr 0.75 Iterations 10 ✓ Recalc

Fig. 2 Flow of Packet Data



Auto layout: Ca 0.15 Cr 0.75 Iterations 10 ✓ Recalc

Fig. 3 Link failure

The communication link breakdown due to congestion, after this path has to be detriment to reach the destination node that is node 3. Now smallest path originating from the node 0 to node 3 had to be obtaining with minimum spanning tree. Prim's algorithm is applied to obtain the shortest path from node 0 to node 3. The key weight values on the edges for Fig. 3 are given as:

A. Prim's Algorithm

Input: A non-empty weighted graph with vertices V and edge's E, i.e. G (V, E).MST set: V n = [x], where v is starting vertex from V, E n =[ ].Repeat until V n=V:

- Choose path or edge, [u, v] with minimum weight assigned to it.
- Add v to V n and their edge i.e. [u, v] =En.

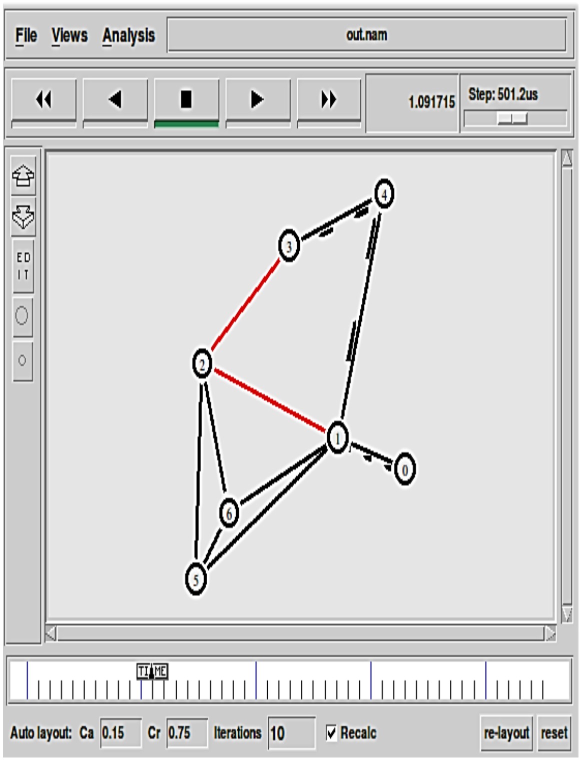
Output: V n and En described a minimum spanning tree.

1. Prim's Algorithm (In Case of Link Failure)

The Prim's Algorithm makes nature choices of the cut an every iteration it grows a single tree and adds a light edge in each iteration consider the following:

1) As a source node 0 is selected first.
2) From the node 0 the node 1 selected because it has minimum weight value, and this node is saved in MST set.
3) From the remaining set of edges, i.e. 1→2, 1→5, 1→3. The path from 1→4 is selected because of minimum weight key value. We cannot go for path 1→2because the link is failed.
4) Again this path is added to the MST set, and the total key value in the MST set is updated.

Lastly, the path from 4→3 are selected based on weight value. The total weight value is 7, and it has a minimum spanning tree. Fig. 4 shows the minimal path selection using Prim's algorithm.



Auto layout: Ca 0.15 Cr 0.75 Iterations 10 ☑ Recalc re-layout reset

Fig. 4 Path Selected by Prim's Algorithm

TABLE II
CALCULATED VISIBILITY VALUE

Edge	Visibility value	Probability value
0→1	0.5	1
1→2	----	----
1→4	0.5	1
4→3	0.33	0.33
0→1	0.5	1
1→2	----	----

TABLE III
VISIBILITY AND PROBABILITY VALUES OF WEIGHT ASSIGNED TO EDGE

Edge	Key Weight Value	Visibility Value
0→1	2	0.5
1→2	Link Failed	----
1→4	2	0.5
1→5	5	0.2
1→6	3	0.33
2→3	Link Failed	----
2→5	4	0.25
3→4	3	0.33
5→6	4	0.25

B. Ant Colony Optimization (ACO) Algorithm

Ant colony belongs to the class of Meta's heuristics [6]-[9], which are approaching algorithms used to obtain good solutions, to complex problems in a reasonable amount of time. ACO is one of the most-recent techniques for the close optimization.

The moving source of ACO algorithm is the real ant colonies. Natural ants are capable of finding the shortest path from their nest to the food source without using the visual signs, because ants are blind and then intellect the path with the help of their antennas and pheromone trail leave backward behind the ants [7]. So ants select the optimal prime path using the Meta heuristic search technique to reach to the food source or the destination and go to the rear using an indirect communication through the pheromone [8]. In the field of information technology, an ant is used as an artificial ant to find the finest optimal path form to set of paths [6]-[8].

1. Basic Procedure Followed by the ACO Algorithm

- Procedure ACO Meta heuristic
- Schedule Activities
- ConstructAntsSolutions
- UpdatePheromones
- DaemonActions // Optional
- End-Schedule Activities
- End-Procedure

2. ACO Algorithm (Pseudo Code) for the Shortest Path (In Case of Link Failure)

```
Gives the initial position to each ant on a random path
For each iteration do;
Do while each ant has not covered its path to reach the node in the graph // (node 0 to Node 3).
For each ant do;
Move ant to the next node by using probability equation (1); // (0→1)
End;
If Link is failed between nodes, i.e. (1→2)&& (2→3) then;
For ant select another shortest path using the visibility and probability value using equation (1) & (2) // (Node 4 is selected)
End;
For each ant with a complete path to reach to its destination (node 3) do;
Update the pheromone value;
End;
End;
```

The path 1→2 and 2→3 are a breakdown, so ants will have to decide the next path or route from the remaining paths. The ants will select the path from 0→1, 1→4 and 4→3. This path is selected by using probabilistic and visibility equations (1) and (2) and given as below:

To find the next neighbor nodes the probability function is the use, which is given as [6]:

$$P^{k}xy(t) = [txy(t)a.[nxy(t)]\beta \ /?[txy(t)]a.[nxy(t)]\beta \quad (1)$$
$$where \ \cdot a \geq 0 and \ \beta \geq 1$$

Visibility function which is used to detect the pheromone concentration value on the route / path is calculated by using:

$$\text{Visibility} \quad (n) = 1/d \qquad (2)$$

The visibility and probability values for the each edge's weight are calculated below: Table III shows the visibility and probability value of the shortest path followed by the ACO algorithm. Fig. 5 shows that the shortest path is selected by using ACO algorithm.

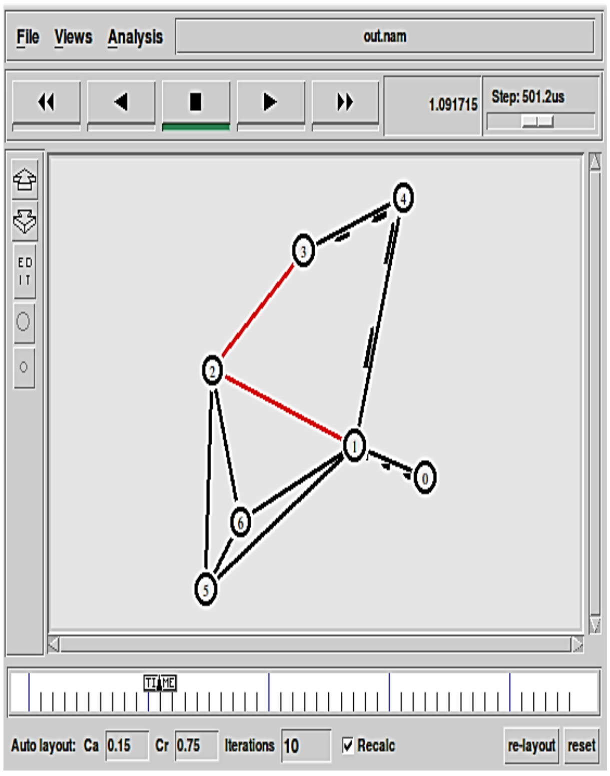


Fig. 5 Shortest path followed by ACO Algorithm

TABLE IV
OBTAINED RUNNING TIME VALUES AND TIME COMPLEXITY

Parameters	Algorithms		
	Prim's Algorithm	ACO Algorithm	
		Ants=1	Ants>1
Running Time (In Seconds)	2.003	1.001	1.001
Time Complexity	O(V^2)	O(V)	O(V)

As befits a function that measurement the running time of a program we shall assume that:

1. The argument V is restricted to be a positive integer number.
2. The value O(V) is positive for all arguments V.

Let f(V) be some function defined on the positive integer V. We say that "O(V) is O(f(V))".

IV. Results and Discussion

In this paper prim's and ACO algorithm selects the same path after the link failure problem. Based on time taken by the both algorithms, ACO is considered to be the best optimal technique to find the shortest path. Table IV shows the performance comparison between the prim's and ACO algorithm. The execution time taken by the both the algorithm is different. The time complexity is described in big-oh notation. The time complexity taken by the prim's algorithm is O (|V2|), and time complexity of ACO algorithm is O (|V|).

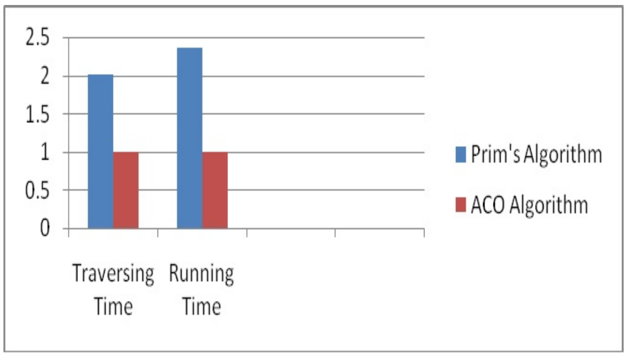


Fig. 6 Time variation graph of prim's and ACO algorithm

Fig. 6 shows the time variation chart of prim's and ACO algorithms. The ACO algorithm shows the linear line. ACO algorithm used single ant or multiple ants then the computation time remained constant.

V. Conclusion

In this paper, we presented an architectural framework in which shortest path assigned to the node provided by the ACO algorithm. Our work is concerned with the reliability, availability, safety in distributed cloud environment. The original contribution addresses the ACO algorithm to provide the shortest path in case of communication link failure in cloud environment and performance of the ACO algorithm compared with the prim's algorithm. From the performance comparison point of view, ACO algorithm takes Big-O, i.e. O(V) time complexity to complete the execution.

VI. Future Work

The future work considers the development of mechanism for fault avoidance.

References

[1] Bheevgade Meenakshi and Patrikar Rajendra (2009), *Implementation of Fault Tolerance Techniques for Grid Systems, Advanced Technologies*, Kankesu Jayanthakumaran (Ed.), ISBN: 978-953-307-009-4, In Tech, Available from: http://www.intechopen.com/books/advanced-technologies/implementation-of-fault-tolerancetechniques-for-grid-systems, PP. 531-546.

[2] Jaroodi-Al, Mohamed, Nuaimi, *An Efficient Fault-Tolerant Algorithm for Distributed Cloud Services* IEEE Second Symposium on Network Cloud Computing and Applications, 2012, pp.1-8.

[3] Sapre Bhushan, Garje Anup Fault Tolerant Environment Using Hardware Failure detection, Roll Forward Recovery Approach and Micro-rebooting For Distributed Systems. International Journal of Engineering Research and Applications (IJERA), ISSN: 2248-9622, Vol. 1, ISSUE 3, PP. 1065-1071.

[4] Rao Subba, Malladi Design, *Design, Analysis and Performance Evaluation of a New Algorithm for Developing a Fault Tolerant Distributed System*, in Technical Report CS-SWlab-2006-05/03.Proceedings of the 12th International Conference on Parallel and Distributed Systems (ICPADS'06) IEEE, 2006.

[5] Pei-yu Li, yu-Pei and McMillin Bruce, *Fault-tolerant Distributed Deadlock Detection/Resolution*, IEEE transactions on parallel and distributed systems,1993, pp.224-230.

[6] Raj Pratibha and sood Monica, *Ant Colony Optimization is the limited case of Prim's Algorithm*, Pratibha Raj et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, 2012, Vol. 3 (3), 4202-4204.

[7] Sim Mong and Sun Hong, 2003,*Ant Colony Optimization for Routing and Load-Balancing: Survey and New Directions*, IEEE Transactions on Systems, Man, and Cybernetics Part A: SYSTEMS and Humans, vol. 33.

[8] Dorigomarco and stutzle Thomas, Ant Colony *Optimization.*

[9] Muhammet Unal, Ayca Ak, Vedat Topuz, Hason Erdal, "Optimization of PID controller using Ants Colony and Genetic Algorithm" Publisher Springer Vertag Berlin Heidelberg in 2013.