Performance Analysis of Environment Aware Algorithms for Fault Tolerant Networks

Karunakara Rai B[1], Shantharama C[2]
[1]*Assoc. Professor, Dept. of ISE, Nitte Meenakshi Institute of Technology Bangalore - 560064, India*
[2]*Professor, Dept. of ECE, A.J. Institute of Engineering and Technology Mangalore - 575006, India*
(E-mail: karunakara.rai@nmit.ac.in[1], csraicec@gmail.com[2])

Abstract— In this paper, performance analysis of two environment aware algorithms is carried out for network topology. As Depth First Search BeeHive (DFSBeeHive) algorithm and genetic algorithm are prominent in finding out the optimal routes in large networks, they are being compared and evaluated for their performance. Researchers in the recent past decade have worked on various approaches to address different issues in fault tolerant communication networks using the aspects of evolutionary algorithms. In this paper the analysis is carried out with a presumption that 50 percent of bees will act as employed bees for beehive algorithm for ensuring early convergence of determining the optimal route. The analysis draws the inference that the DFSBeeHive algorithm shows better performance for a fault tolerant network. Here bandwidth is the QoS parameter used for the evaluation. Hop count is taken as one of the criterions for selecting one of the multiple optimal paths derived. The fitness function for selecting the link is based on the threshold value fixed on the basis of observation. The simulation result shows that DFSBeeHive algorithm has outperformed genetic algorithm by 35% for the determination of optimal path for given environment.

Keywords— *DFSbeeHive; Genetic Algorithm; Optimal Path; Real Time Fault Tolerant; Environment Aware*

I. INTRODUCTION

In the past decade many environment aware routing algorithms have been explored by researchers for determining optimal path in a given network topology. Genetic algorithm (GA) and artificial bee colony algorithm have played major role in giving optimal solutions to some complex problems like faults in network routing, robotics applications, game playing, bench marking optimization and many more. GA came to the forefront in the early 1970s after the remarkable work of John Holland [1]. Present demand for solutions to real time machine issues has contributed enough sustained work in GA theory. GA has given optimal solutions to business applications like information systems and production operations, numerical optimization and machine learning. There are a number of advantages of GAs. Some of them are as below:

- Parallelism is the basic feature of GA. GA starts with multiple solutions and converges to single solution whereas other conventional algorithms start with a single solution and reaches multiple solutions.
- GA is easy to implement as it starts with arbitrary known combinations and follows fixed steps in iterations. Hence it gives a wide range of selection sphere for consideration of input combinations. In each stage the justice is given in the process of subsequent selection of intermediate parameters depending on the fitness function.
- When the solutions are spread across wide range of inputs GA follows well defined pattern of evaluation without taking too many parameters at a time which reduces the complexity of search for optimal solution.
- GA is an attempt to converge to an optimal solution, but it need not be the best.
- As the number of iterations depends on the population, the time for execution of GA for a given problem is deterministic.
- The basic operations involved in implementation of GA are simple, and atomic in nature. Hence the complexity is tremendously reduced.

The drawbacks of GA in its working principles are discussed below:

- Handling the local maxima is a challenge. The challenge of obtaining the global maxima through brute force methods need to be addressed at an early stage.
- The convergence time for achieving a satisfactory solution is not appealing.
- Standardizing the stages of GA like chromosome selection, crossover and mutation is a big challenge.
- Fitness function for a given problem considers limited number of parameters.

The DFSBeeHive (Depth First Search BeeHive) approach is similar to Artificial Bee Colony (ABC) algorithm, which was introduced by Karaboga in 2005 [2]. These algorithms are used for optimizing numerical problems. The DFSBeeHive algorithm as worked out by Wedde and et al. [3] incorporates the foraging behaviour of honey bees. There are three categories of bees namely employed bees, onlooker bees and the scout bees. The food sources that we refer in the model are close to the hive. In the entire process of procuring the food by the bees depends on the threshold amount of nectar close to the hive. In the entire process of procuring the food by the bees depends on the threshold amount of source. Adaption of DFSBeeHive algorithm starts with assigning the employed bees to the initial known food sources. Onlooker bees move in the direction of the food depending on the foraging dances of employed bees. The scouts are assigned to the new food sources. The different applications of DFSBeeHive algorithm are Biological simulation, Continuous Optimization, Travelling Salesman Problem(TSP), Ride-Matching Problem, Dynamic Allocation of Internet Service, Telecommunication Network Routing, Job shop scheduling, Max-W-Sat Problem, Neural Network, Dynamic resources and Max-Routing and Wavelength Assignment [4-8]. The advantages of DFSBeeHive are simple in understanding, easy to implement and flexible to modify. The disadvantages of DFSBeeHive includes burden of calculation of fitness at each stage, huge number of function evaluations and computational cost [9].

In paper [10], a formal framework is being obtained for evaluating a large topology. The flexibility for changes to address faults in the network is implemented in the paper. Self-organizing is one of the most prominent features of environment aware network algorithms. This feature is strongly supported here, through behavioural study and local interactions. But the heterogeneous nature of the network with varying bandwidth and link delays are not addressed, for which an attempt has been done in this paper keeping bandwidth as the evaluating parameter. The authors in [11] have contributed tremendously for telecommunication network routing using the concepts of Artificial Bee Colony. Foraging zones are the working area of the bee agents. The routing tables are updated depending on the local information obtained. This approach is fault tolerant, as the actions are taken on the prevailing local environment. This paper provides a great scope for enhancing the volume of considerations among the local parameters that can be exploited for the future work. The author in [12] discusses the advantage of DFSBeeHive routing algorithm for a fault tolerant network which addresses the traffic issues. The traffic splitting approach is implemented in this paper. The performance of DFSBeeHive routing algorithm can be compared with other competitive algorithms like GA, Ant colony optimization algorithm and many more. This aspect is considered in our paper by taking DFSBeeHive and GA algorithms. In [13], an attempt is made to evaluate the performance of a distributed sensor network using GA. Fault tolerant networks vary their performance for different routing algorithms [14].

The rest of this paper is organized as follows. In section 2, we introduced the network aware routing algorithms and discussed thee fitness function used to carry out the simulation. Analysis of simulation result is dealt in section 3. Finally conclusion is dealt in section 4.

II. NETWORK AWARE ROUTING ALGORITHM

This paper implements the GA and DFSBeeHive algorithms for comparing the performance of a fault tolerant network [15] at different environments, as both the algorithms are network aware in nature.

Genetic algorithm

GA is a population based algorithm. Initially in the first stage a selection criterion is followed for choosing the chromosomes from the population. The prominent chromosome selection techniques are elitism and Roulette wheel. Roulette wheel uses a random approach in deciding the parents for further processing. But in this method of selection dominant chromosomes get the better share. Figure 1. represents a Roulette wheel which spins continuously before it stops to select a slice corresponding to the chromosome. In the second stage crossover of parent chromosomes is carried out to generate offsprings. In each stage of crossover the fitness of each generated chromosome is calculated and the probability of considering the same for the next stage is determined. Crossovers can be of different types like single-point crossover, multi point crossover and many more. Mutation is the third stage in the algorithm, wherein some of the genes are flipped randomly. GA provides a platform for finding the near optimal solutions out of huge number of available inputs, though the solution may not be the best.

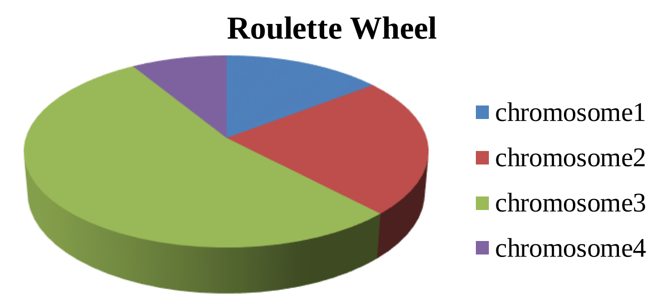

Fig.1: Roulette Wheel mechanism

Chromosome crossover

Through crossover the offspring inherits the parents. In single point crossover the string from beginning of first parent chromosome to the crossover point and the rest is copied from the second parent chromosome after the crossover point. In multi point crossover, two or more crossover points are considered.

For example:
Single point Crossover:

Parent 1: A C D | K M N Z
Parent 2: A B E | J L R Z

Offspring 1: A C D J L R Z
Offspring 2: A B E K M N Z

Multipoint Crossover:
Parent 1: A C | D K M | N Z
Parent 2: A B | E J L | R Z

Offspring 1: A C E J L N Z
Offspring 2: A B D K M R Z

In the above example ACDKMNZ and ABEJLRZ are the two parent chromosomes which represent the two routes from the source A to the destination Z. The parent chromosomes are made to crossover after the third gene, to obtain ACDJLRZ and ABEKMNZ offsprings. Similarly two point crossover is done at the positions second and fifth genes to obtain ACEJLNZ and ABDKMRZ.

Mutation

The Mutation is carried out to introduce diversity in the chromosomes by randomly replacing a gene with a different gene. It helps in minimizing the similarity in the solutions.
Consider the chromosome,

A C D K M N Z

The mutation can be done for the third gene D by replacing it by F to obtain ACDFMNZ. Always the percentage of mutation is kept very small compared to crossovers to see that the GA converges to the early solution. The figure 2, Depicts the working principle of GA. The input to the GA is a set of populated possible routes from source to destination in a fault tolerant network topology. The crossover and mutation are carried out at each iteration to produce the offsprings to replace the parent chromosome on fitness condition. The iterations are repeated for obtaining the near optimal solution.

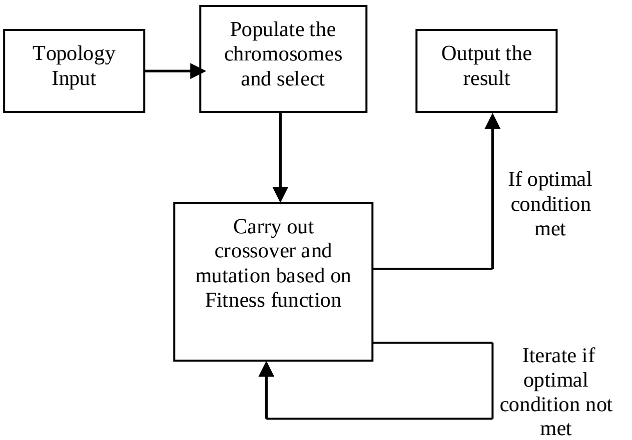

Fig.2: Genetic algorithm

The following are the steps involved while executing the GA:

Generate random population of suitable size

1. Determine the fitness of each chromosome.
2. Repeat the following steps until near optimal solution is obtained
   a. Select the chromosomes for crossover
   b. Carry out crossover operation and mutation to obtain the offsprings.
   c. Calculate the fitness of each offspring and determine the probability of replacing the parent from offspring.
   d. Include the offsprings in the new population.
3. Go to 3.

DFSBeeHive algorithm

DFSBeeHive algorithm which has inherited the features of ABC, is one of the powerful algorithms for fault tolerant network routing.

Figure 3, depicts the working principle of DFSBeeHive algorithm. The topology is divided into multiple foraging areas. The initial foraging area is the input to the algorithm with source and destination being mentioned. Initially a suitable number of bees are assigned to the food sources. The onlooker bee follows the employed bees. More sources of food are located and evaluated for the threshold nectar amount using the fitness function. The searching continues till the destination is found in the present foraging area. If destination is not found the searching process enters to the new foraging area. These steps are repeated until the destination is found.

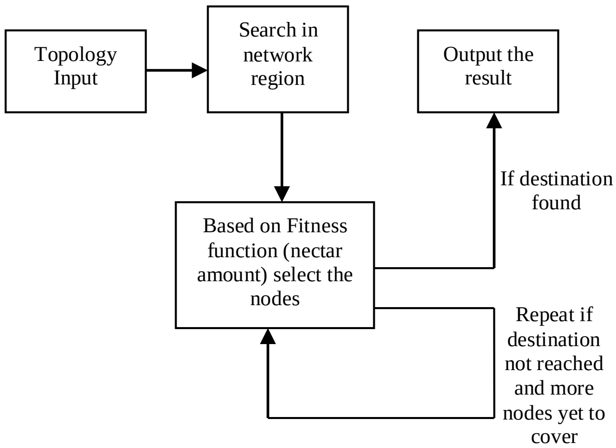

Fig.3: DFSBeeHive

The following are the steps in DFSBeeHive algorithm:
Consider a foraging area
REPEAT
Assign the employed bees on to the food sources and determine the nectar amount. Calculate the fitness value and determine the probability of the preference of the nectar amount.
Assign the scouts into the search area for discovering new food sources.

Exploit the complete foraging area using the scout bees.

Recognize the best food source obtained so far.

UNTIL (Condition satisfied)

Fitness function

The fitness function is calculated based on link bandwidth. In GA the probability of the offsprings are determined depending on the fitness function in each iteration. The fitness of offsprings are compared with the fitness of chromosome in the population of respective generations. In DFSBeeHive algorithm, fitness of the link corresponding to the newly invented node is calculated and evaluated for consideration depending on the threshold value. The fitness function is shown in equation (1):

$$f(t) = \frac{Bw(t)}{\sum_{i=0}^{n} Bw(t)} \quad (1)$$

The probability function for a link to consider is as shown in equation (2):

$$p(t) = 1 - \sum_{i=0}^{t-1} p(i) \quad (2)$$

III. SIMULATION RESULTS

Simulation is carried out using C code for both GA and DFSBeeHive algorithm for different topologies. One of the topologies considered for the simulation is depicted in Figure 4. DFSBeeHive and GA were considered to determine the near optimal path. The required assumptions for GA are as shown in Table 1.

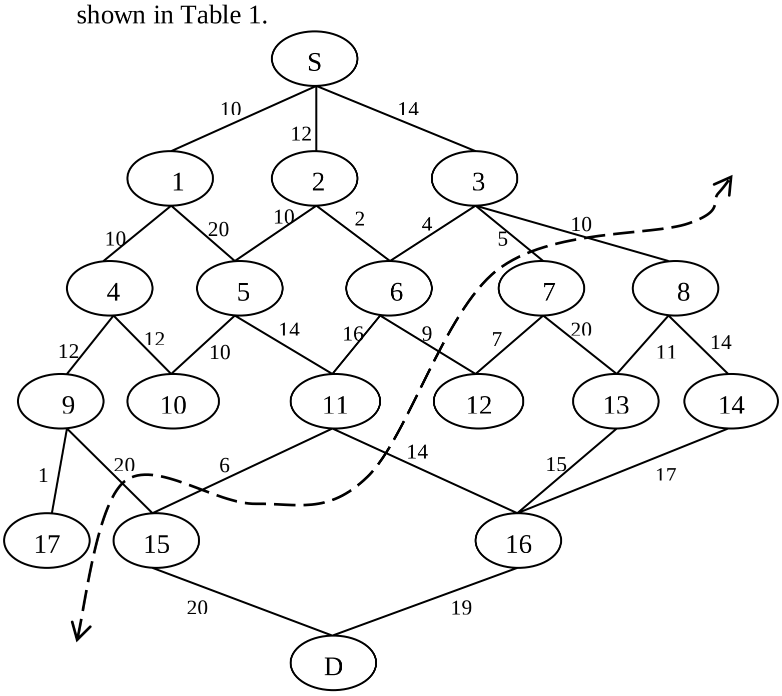

Fig.4: Topology

TABLE 1. GENETIC PROPERTY SETTINGS

GA properties	Values
Crossover probability	99%
Mutation probability	1%
Population size	17 nodes
Number of generation	4

It was observed that GA takes more memory for a given network compared to DFSBeeHive algorithm. It's because GA records more number of initial known chromosomes before selection process. Execution of fixed steps for fitness calculation and determination of offsprings to replace the parents leads to uniform pattern of tracing as opposed to DFSBeeHive.

Table 2, provides the performance recorded for the above mentioned algorithms with different destinations considered for different simulations (Runs) keeping the sources same. It was observed that DFSBeeHive performs better with respect to the fitness of the route compared to GA. As far as number of iterations considered genetic takes more time to converge to the near optimal result.

TABLE 2. FITNESS VALUE AND ROUTE LENGTH

Runs	DFSBeeHive		Genetic Algorithm	
	Fitness Value	No. of iterations	Fitness Value	No. of iterations
Run1	0.78	6	0.56	13
Run2	0.74	6	0.48	11
Run3	0.66	8	0.40	10
Run4	0.72	7	0.46	11

The hop count of the routes obtained in DFSBeeHive is comparatively more than that of GA as shown in Figure 5. The graph also depicts that GA attempts to converge to a near optimal route, where as DFSBeeHive performs relatively better.

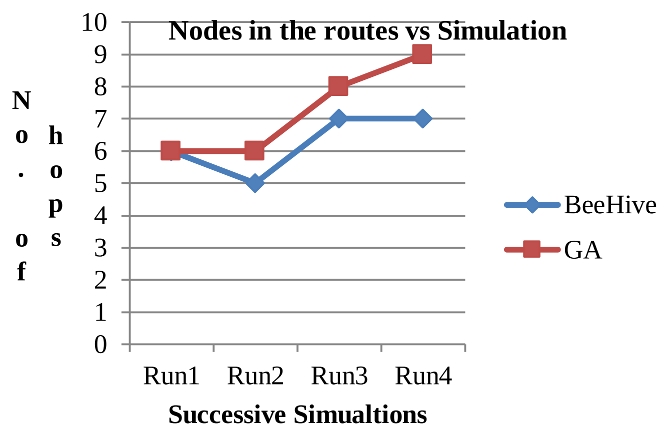

Fig.5: Nodes selected by the algorithms

But delays in reaching the optimal path are inevitable because of huge number of iterations involved in very nature of genetic approach. Throughput is more in GA compared to the uncertainty involved in DFSBeeHive when fault occurs. Overhead of finding the alternate path in DFSBeeHive algorithm adds complexity in finding the optimal path.

IV. CONCLUSION

The paper presented here determines the outperformance of DFSBeeHive algorithm on GA in environment aware fault tolerant network. Both the algorithms were tested on a particular predefined topology with bandwidth as QOS metrics. The work can be enhanced for larger networks considering addition QOS metrics. Future work will include the FPGA implementation for the algorithm at network layer for saving paramount time in carrying the huge number of iterations for fault tolerant network systems.

V. REFERENCES

[1]. Holland, John H. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT press, 1975.

[2]. Karaboga, Dervis, and Bahriye Basturk. "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm." Journal of global optimization 39.3 (2007): 459-471.

[3]. Adil Baykasolu, Lale Ozbakır and Pınar Tapkan, Artificial Bee Colony Algorithm and Its Application to Generalized Assignment Problem, Book Chapter Swarm Intelligence: Focus on Ant and Particle Swarm Optimization, pp no. 113-144, Itech Education and Publishing

[4]. Biradar, Arun, and Ravindra C. Thool. "Reliable genetic algorithm based intelligent routing for MANET." Computer Applications and Information Systems (WCCAIS), 2014 World Congress on. IEEE, 2014.

[5]. Grefenstette, John, et al. "Genetic algorithms for the traveling salesman problem." Proceedings of the first International Conference on Genetic Algorithms and their Applications. Lawrence Erlbaum, New Jersey (160-168), 1985.

[6]. Eiben, Ágoston E., Jan K. Van Der Hauw, and Jano I. van Hemert. "Graph coloring with adaptive evolutionary algorithms." Journal of Heuristics 4.1 (1998): 25-46.

[7]. Saeys, Yvan, Iñaki Inza, and Pedro Larrañaga. "A review of feature selection techniques in bioinformatics." bioinformatics 23.19 (2007): 2507-2517.

[8]. Tang, Kit-Sang, et al. "Genetic algorithms and their applications." IEEE signal processing magazine 13.6 (1996): 22-37.

[9]. Ramasubramanian, Venugopalan, and Emin Gün Sirer. "Beehive: (1) Lookup Performance for Power-Law Query Distributions in Peer-to-Peer Overlays." Nsdi. Vol. 4. 2004.

[10]. Muhammad Shahzad, Saira Zahid, and Muddassar Farooq, "A Scalable Formal Framework for Analyzing the Behavior of Nature-Inspired Routing Protocols", Proceedings of the 10th International Conference on Parallel Problem Solving from Nature pp. 1130–1139, 2008

[11]. Horst F. Wedde, Muddassar Farooq, and Yue Zhang, "BeeHive: An Efficient Fault-Tolerant Routing Algorithm Inspired by Honey Bee Behavior", International Workshop on Ant Colony Optimization and Swarm Intelligence pp. 83–94, 2004.

[12]. Kiran K, P Deepa Shenoy, Venugopal K R, Lalit M Patnaik, "Fault Tolerant BeeHive Routing in Mobile Ad-hoc Multi-radio Network", 2014 IEEE Region 10 Symposium, pp no. 116-120

[13]. Prof. R. K. Krishna, Anagha Nanoti, " A Review on Genetic Algorithm for Fault Tolerance in Distributed Sensor Networks", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 5, Issue 12, December 2016.

[14]. Gupta, Gaurav, and Mohamed Younis. "Fault-tolerant clustering of wireless sensor networks." Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE. Vol. 3. IEEE, 2003.

[15]. Hamada, Takeo, Peter J. Czezowski, and Ching-Fong Su. "Fault tolerant network routing." U.S. Patent No. 7,796,503. 14 Sep. 2010.

Karunakara Rai B holds M.Tech degree in VLSI and Embedded systems from VTU and MS in Information Science from MAHE Manipal. He is currently pursuing his Ph.D. program in Fault Tolerant systems. Presently he is working as Associate Professor in the Department of ISE at NMIT, Bangalore, India.

Shantharama C has obtained his Ph.D. from V.T.U Belgaum with specialization of power electronics and control system. He is member of Indian society for technical education New Delhi and Indian society for lighting engineers New Delhi, He has got teaching experience of more than 18 years. He has published /presented more than 40 papers in international/national journals /conferences in the field of engineering. Presently he is the Principal of AJ Institute of Engineering and Technology Mangalore, India.