Fault Tolerant ACO using Checkpoint in Grid Computing

Tanya Prashar
Department of M.Tech (C.S.E)
DAV Institute of Engineering & Technology, Jalandhar

Nancy
Asst Prof. B.Tech (C.S.E)
Beant College of Engg &Technology, Gurdaspur

Dinesh Kumar
Hod. M.Tech (C.S.E)
DAV Institute of Engineering & Technology, Jalandhar

ABSTRACT

This paper proposed an algorithm for fault tolerant distributed computation in a grid by the means of meta-heuristic Ant Colony Optimization (ACO) technique and Check-Pointing. Load Balancing is the process of distributing the workload among the nodes in a grid. The load can be CPU cycles, memory capacity or network load. Due to the emerging computing methodology of grid computing over the heterogeneous environment there is a need of hybrid fault tolerant load balancing technique which takes into account the grid scenario including computer heterogeneity, network bandwidth and communication delay.

General Terms

Algorithms Used: ACO Algorithm for Load Balancing, Check-Point Setter Algorithm and Fault Index Update Algorithm as fault tolerance mechanisms.

Keywords

Keywords: Computational Grid, Ant Colony Optimization, Check-Pointing, Fault Tolerance.

1. INTRODUCTION

The Grid is defined as a collection of computer resources in distributed system with optimal workloads. Grid Computing involves computation in a distributed manner.Grid computing is an evolutionary parallel and distributed computing practice used for managing various kinds of large scale computation problems in network [1].To facilitate and achieve the potentials of computational grids, an effective scheduling system is used to evenly distribute the workload among all the resources to minimize the job execution time.

Table 1. Grid vs. Distributed Computing [8]

Grid Computing Vs Distributed Computing	Grid	Distributed
Definition	It is defined as the efficient utilization and management of pool of heterogeneous systems with non-iterative workloads.	It is defined as managing hundreds or thousands of computer systems which have limited memory and processing power. Traditional Distributed Computing is defined as the subset of grid computing.

Load balancing and resource management are the key areas of research for a Grid Computing environment which is used to equally distribute the load on each computing node, maximizing the resource utilization rate and minimizing the task execution time [6]. ACO is the emerging methodology in grid computing for managing load balancing, job scheduling. Hence, efficient use of this algorithm will lead to better utilization of resources available thus improve the throughput and the overall performance [2]. Resource failure nad fault tolerance are the significant issues that needs to be handled in grid, thus fault tolerance related features must be used in grid task planning to improve the performance of the grid system [5]. This paper consider the fault-tolerant scheduling using ACO algorithm and Check Pointing.

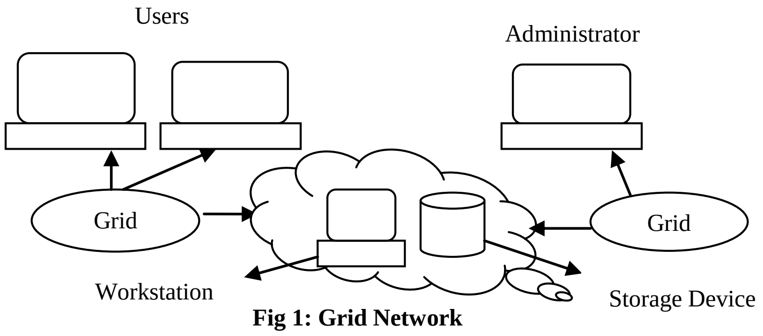

Fig 1: Grid Network

2. LOAD BALANCING

Load Balancing is a great challenge for dynamic and heterogeneous, complex grid and cloud computing environment [11]. The main objective of load balancing is to distribute the workload equally among the nodes to optimize the service time of the resources and the response time of the application. The main causes of load balancing are job migration, heterogeneity of resources and dynamic nature of resources performance. Load balancing is required so that maximum utilization of resource could be possible to improve the overall performance of the system. There are two types of load balancing namely:

- Static load balancing: In static load balancing, the work is distributed prior to the execution of the algorithm.

Scheduling takes place on basis of the condition of the system and no more scheduling will take place until the work is done.
Advantage: It is easy to design and implement.

- Dynamic load balancing: A dynamic load balancing distributes the work among the processors during the execution of the algorithm. The dynamic load balancing algorithms are more complicated.
  Advantage: It quickly manages the workload fluctuation.

The major goals of Load Balancing are:
- To improve the performance of the application.
- To manage and maintain the systems in a network [10].

2.1 Policies for Dynamic Load Balancing Algorithms

- Information policy: It states what workload information to be collected, when it is to be collected and from where it is to be collected.
- Triggering policy: It specifies the appropriate time period to start a load balancing process.
- Selection policy: It select the tasks that should be migrated from overloaded resources known as the sender to most idle resources called receiver nodes.
- Location policy: It specifies the results of the resource type policy to find a suitable partner for a server or receiver.
- Resource policy: It differentiates a resource as server or receiver of tasks according to its avalibility status [10] .

3. PREVIOUS ALGORITHM FOR LOAD BALANCING

3.1 Ant Colony Optimization

Ant Colony Optimization (ACO) is a population based metaheuristic approach for solving optimization problems. ACO provide good solutions to the optimization problems. The main task of ants in the algorithm is to redistribute work among the nodes. The ants traverse the grid network and leaves the pheromones on the path. On reaching the destination the ants update the pheromones tables [14]. The main source of ACO is the pheromone trail laying and following behaviour of real ants that use pheromones as a communication means .The ants move from node to node thus exploring the information provided by the pheromones values and in this way incrementally building the resultant solution [7].

3.2 Survey of Previous ACO Algorithm

Generation of Ants : The ants are generated, by the machine object. These Ants are treated as the mobile agents in the network. The agents would fetch the information about the resources. The TTL factor (Time to Live) is the evaporation parameter of the ants Hence the trail is evaporated as the TTL (Time To Live) reaches to the zero value [16].

Resource Allocation and Selection: Based on the information collected from the ants in the network, there will be the resource allocation for the job to complete. The mobile agents would send the information to the source machine thus this information is used to select the resource. The selection is based the factors like the bandwidth utilization and resource location[16].

Forward and Backward Direction:The routing in a network can be done in both forward and backward direction to discover overloaded and underloaded nodes in a grid.The backward ant take the same path as that of its corresponding forward ant but in opposite direction.

Foraging Pheromones: The ants continuously move in the forward direction in the grid network encountering overloaded node or under loaded node laying down the foraging pheromones [2].

Trailing Pheromones: If an ant encounters an overloaded node in its movement when it has previously encountered an under loaded node then it will go backward to the under loaded node in the grid to check if the node is still under loaded or not and if it finds it still under loaded then it will redistribute the workload to the under loaded node. The vice-versa is also applicable [2].

Pseudo code for ant colony optimization (ACO) :
- Initialize parameters
- Initialize pheromone trails
- Create Ants
- While stopping criteria is not reached do
- Let all ants construct their solutions to make the resultant solution
- Update pheromone trails
- Allow the Processes to execute
- End while [14].

Pheromones Tables: The routing tables in the network nodes are replaced by the tables of probabilities which are called pheromones tables and the pheromones strengths are exhibited by these probabilities [10].

Pheromones Updating: There will be two types of pheromones namely foraging and trailing pheromones associated with forward and backward movement of biological ants. The initial pheromone value of each resource was based on its status where job was issued to the resource with the maximum pheromone value. The pheromones evaporate over time. The strength of pheromone of each resource was updated after completion of the job.

The below flowchart is the basic ACO algorithm used for balancing the load in a computing network.

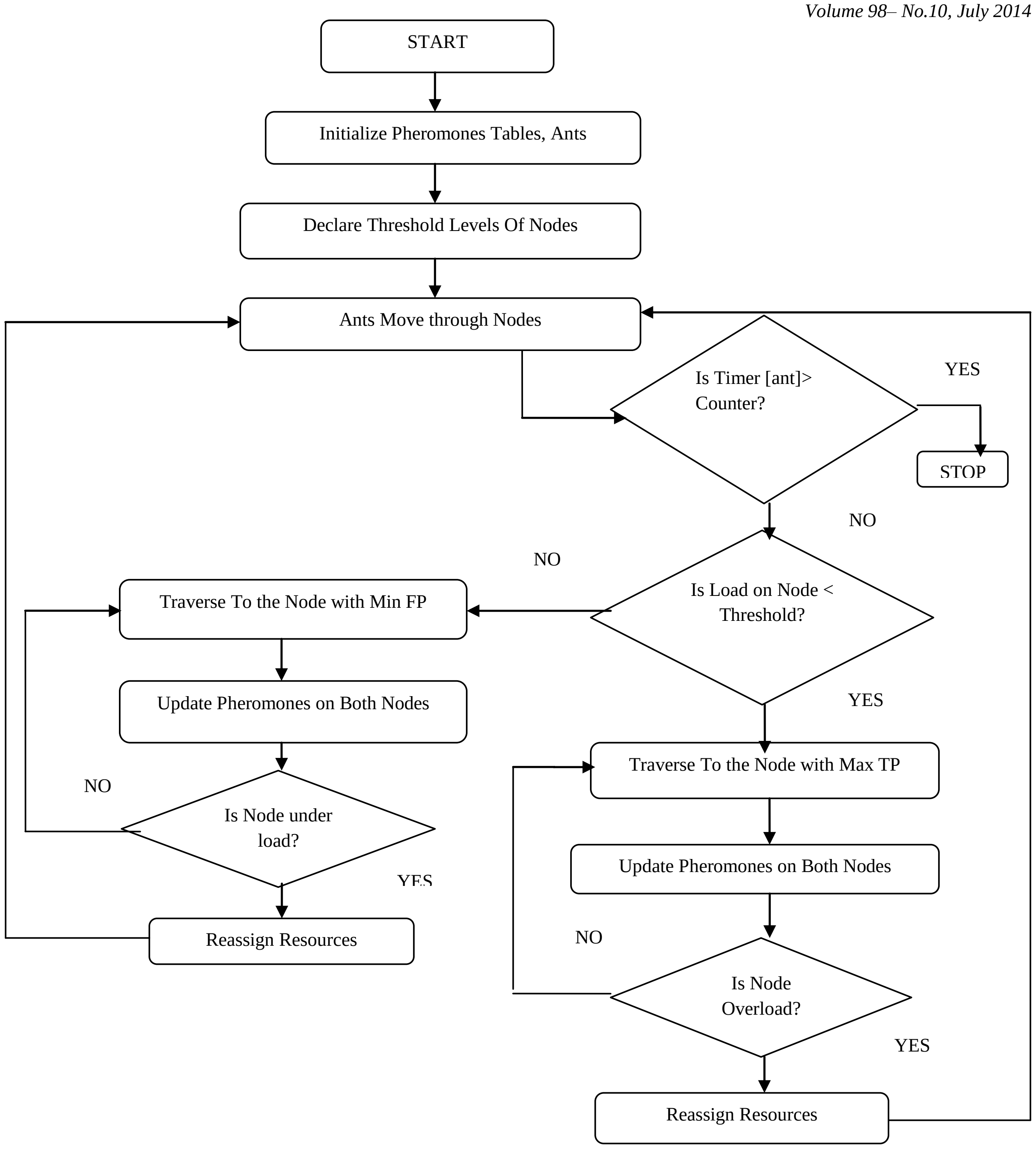
START
Initialize Pheromones Tables, Ants
Declare Threshold Levels Of Nodes
Ants Move through Nodes
Is Timer [ant]> Counter?
YES
STOP
NO
Is Load on Node < Threshold?
NO
Traverse To the Node with Min FP
Update Pheromones on Both Nodes
NO
Is Node under load?
YES
Reassign Resources
YES
Traverse To the Node with Max TP
Update Pheromones on Both Nodes
NO
Is Node Overload?
YES
Reassign Resources

Fig 2: Flowchart

4. PROPOSED ALGORITHM

The proposed algorithm have introduced the new concept of the fault tolerance mechanism while load balancing in grid of network using Check-Pointing technique. The general Ant Colony Optimization approach for load balancing have been modified by using checkpoint and fault index of resources to meet the fault tolerance issue, to improve the throughput and make the effective utilization of available resources in a computational grid environment.

4.1 Fault Tolerance Job Scheduling using Check-Points

Fault tolerance: It is the ability to preserve the delivery of expected services despite the presence of fault-caused errors within the system itself. It aims at the avoidance of failures in the presence of faults. A fault tolerant service detects errors and recovers from them without participation of any external agents, such as humans. Resource failures can have adverse effects on the applications. Fault tolerance can be achieved by scheduling replicated jobs on various processors [9].

Faults may be classified based on several factors:

- Network Faults: Faults like packet loss and packet duplication or packet corruption.
- Physical Faults: Faults that occur in Memory, Processor or Storage Media.
- Media Faults: Disk Head Crashes
- Resources Faults: Unavailability of resources, Lack of resources.
- Communication Faults: Protocol incompatibility.
- Service Time Faults: Expiry of service time of resource.
- System Performance Degradation [4].

This paper addresses the problem of fault tolerance in terms of resource or process failure.
Check-Pointing Approach: Check-Pointing approach balances the load of processors in a distributed system; processes are moved from heavily loaded processors to lightly loaded ones. Check-Pointing process periodically provides the information necessary to move it from one processor to another [3]. Check-Pointing can be initiated from within grid systems or within applications. There are three different categories of check-Pointing implementations:

- Kernel-Level
- User-Level
- Application–Level [12]

to the Checkpoint algorithm which includes the following components:

Checkpoint Manager:
It receives the scheduled job from the scheduler implemented by the ACO, sets the checkpoint based on the failure rate of the resource on which it is scheduled and submits the job with checkpoints to the resource. Then the checkpoint manager receives job failure or job completion message from the grid resource and responds to that accordingly [13]. Checkpoint manager implements checkpoint setter algorithm to set job checkpoints.

Checkpoint Server:
For each checkpoint the status of job is returned to the checkpoint server [1]. Checkpoint server save the job status and return the job status and last checkpoint whenever required i.e. during job/resource failure.

Fault Index Manager:
Fault index manager updates the fault index of a grid resource using fault index update algorithm depending upon the failure rate of the resource. The fault index of a grid resource is incremented every time the resource does not complete the assigned job within the deadline and also on resource failure. The fault index of a resource is decremented whenever the resource completes the assigned job within the deadline [1].

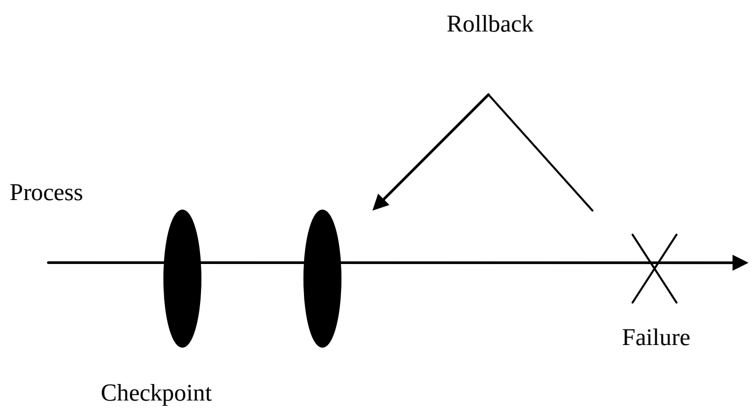

Fig 3: Rollback-Recovery [15]

These implementations differ in the level of transparency and efficiency and procedure used to initiate checkpoint and restart. Out of these three the most appropriate approach is application-level Check-Pointing since the developer has the detailed knowledge about the application.
The below flowchart depicts the hybridization of Ant Colony Optimization for load balancing and Checkpoint for handling the faults issues of resources to make timely completion of jobs assigned to the available resources. Initializing from the ACO where ants will move continuously through the nodes in the grid in search of food laying down the pheromones and updating the pheromones tables .ACO Algorithm will initially declare the threshold levels of each node in the grid and will check the timer of ants and the load on the node, if the load on the node is below the threshold value [2], the control will flow

Once the job is completed update the pheromones on both the previous as well as on the current node. Again check if the node is under load ,if yes then reassign the resources and reschedule the job and if no then traverse to the node with maximum trailing pheromones ,update the pheromones on both nodes and repeat traversing until the node is overload, otherwise reassign the resources again.The proposed flowchart is the hybridization of Ant Colony Optimization
Technique and the checkpointing mechanism to reduce the overhead of fault tolerance issue so as to improve the performance in a grid network and maximize the response time and utilization of resources.

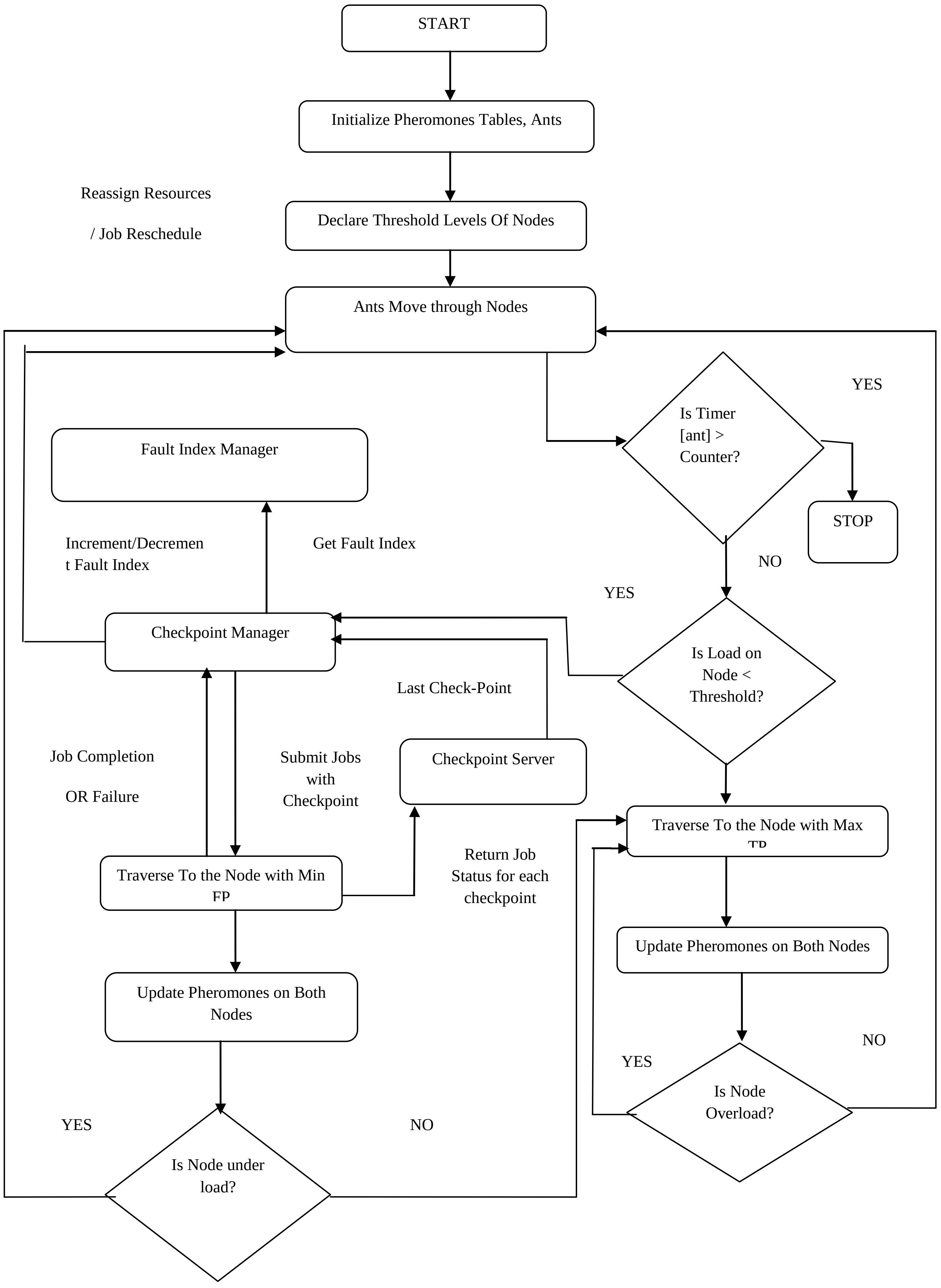
START
Initialize Pheromones Tables, Ants
Reassign Resources
/ Job Reschedule
Declare Threshold Levels Of Nodes
Ants Move through Nodes
YES
Is Timer [ant] > Counter?
Fault Index Manager
STOP
Increment/Decrement Fault Index
Get Fault Index
NO
YES
Checkpoint Manager
Is Load on Node < Threshold?
Last Check-Point
Job Completion
OR Failure
Submit Jobs with Checkpoint
Checkpoint Server
Traverse To the Node with Max TP
Return Job Status for each checkpoint
Traverse To the Node with Min FP
Update Pheromones on Both Nodes
Update Pheromones on Both Nodes
NO
YES
Is Node Overload?
YES
NO
Is Node under load?

Fig 4: Flowchart for fault tolerant ant colony optimization using checkpoint

5. CONCLUSION

The intended approach is computationally quick and reliable in which the ants continuously update a single result set along managing the fault of resources using check-Points by which job can be restarted from the last saved checkpoint instead from the beginning. In this way, the solution set can be continuously improved .The other advantage of the approach lies in the fact that the task of each ant depends on the type of first node that was encountered whether it was overloaded or under loaded and based upon the checkpoint/restart the job of the ants can be restarted from the last checkpoint rather than executing from the beginning in case resource or job failure.It also improves the performance by achieving the better results in terms of throughout, response time.Thus this approach is useful for effective utilization of resources and improving the service time, increases the job throughput and thus make the grid trustworthy.

6. FUTURE ENHANCEMENT

The proposed algorithm can also be implemented for other evaluation models in practice like Genetic algorithm (GA), Particle Swarm Optimization (PSO) technique and many others to apply the fault tolerance mechanisms like Check Pointing which can be used to minimize the execution time. Check-Pointing can be applied in the middleware Scenario and in various technologies and platform like C++ and Linux Operating system thus enhancing the overall network performance, throughput and scalability.Design and testing of fault tolerant load balancing in multi middleware environment.The proposed work can be extended by the migration of checkpointed jobs from the last checkpoint.The proposed work in future can be explored by embedding the fault tolerant load balancing techniques into the real world cloud environment using cloud analysis tool. Also it is planned to improve the checkpoint replication service by optimizing the recovery of checkpoint replica thus reducing the cost of fault tolerant load balancing. Hence the proposed algorithm for load balancing in cloud and grid Computing plays a very important role in future. There is a vast cope of improvement in this area.

7. ACKNOWLEDGMENT

It is with immense pleasure to acknowledge people who have helped tremendously during the research work and provided their guidance, motivation, and continuous support throughout the work.

8. REFERENCES

[1] Malarvizhi Nandagopal, V. Rhymend Uthariaraj. 2010 Fault tolerant Scheduling Strategy for Computational grid Environment. International Journal of Engineering Science and Technology.

[2] Kumar Nishant, Pratik Sharma, Vishal Krishna, Chhavi Gupta and Kunwar Pratap Singh, Nitin and Ravi Rastogi. 2012 Load Balancing of Nodes in Cloud Using Ant Colony Optimization. 14th International Conference on Modelling and Simulation.

[3] Pankaj Kumar, Paritosh Kumar and Gajraj Singh. 2013 Comparative Study of Job Migration Algorithms for Autonomic Grid Management. International Journal of Information & Computation Technology.

[4] S. Gokuldev, Shahana Moideen. May 2013 Global Load Balancing and Fault Tolerant Scheduling in Computational Grid. International Journal of Engineering and Innovative Technology (IJEIT).

[5] Arash Ghorbannia Delavar, Ali Reza Khalili Boroujeni, Javad Bayrampoor. August 2012 A Balanced Scheduling Algorithm with Fault Tolerance and Task Migration based on Primary Static Mapping (PSM) in Grid. International Journal of Computer Applications (0975 – 8887).

[6] Sowmya Suryadevera, Jaishri Chourasia, Sonam Rathore. Abdul Jhummarwala. 2012 Load Balancing in Computational Grids Using Ant Colony Optimization Algorithm. International Journal of Computer & Communication Technology (IJCCT).

[7] Shagufta Khan, Niresh Sharma. February 2014 Effective Scheduling Algorithm for Load Balancing using Ant Colony Optimization in Cloud Computing. International Journal of Advanced Research in Computer Science and Software Engineering.

[8]http://www.jatit.org/distributed-computing/gird-vs-distributed.html

[9] Jasma Balasangameshwara, Nedunchezhian Raju. 2012 A hybrid policy for fault tolerant load balancing in grid computing environments, Journal of Network and Computer Applications 35 (2012) 412–422.

[10] Ratan Mishra and Anant Jaiswal, April 2012, Ant colony Optimization: A Solution of Load balancing in Cloud, International Journal of Web & Semantic Technology (IJWesT) Vol.3, No.2.

[11] Kuppani Sathish , A Rama Mohan Reddy, October 2008 ENHANCED ANT ALGORITHM BASED LOAD BALANCED TASK SCHEDULING IN GRID COMPUTING, IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.10, 219

[12] Gokuldev S, Valarmathi M April 2013**,** Fault Tolerant System for Computational and Service Grid International Journal of Engineering and Innovative Technology (IJEIT) Volume 2, Issue 10, 236

[13] S.Baghavathi Priya, Dr.T.Ravichandran , July 2011 Fault Tolerance and Recovery for Grid Application Reliability using Check Pointing Mechanism, International Journal of Computer Applications (0975 – 8887) Volume 26– No.5, 32

[14] N. Sasikala and Dr. D. Ramesh, April 2014 Effective Load Balancing for Cloud Computing using Hybrid AB Algorithm, International Journal of Innovative Research in Computer and Communication Engineering,Vol.2 ,Issue 4.

[15] Ndeye Massata NDIAYE, Pierre SENS, and Ousmane THIARE, Performance comparison of hierarchical checkpoint protocols grid computing International Journal of Artificial Intelligence and Interactive Multimedia, Vol. 1, Nº 5.

[16] S.Nirmala Devi and A. Pethalakshmi, April 2012 Application of ACO for Resource Discovery in Grid Computing Environment,International Journal of Computer Applications (0975 – 8887) Volume 43– No.2.