IARJSET
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
ISO 3297:2007 Certified
Vol. 5, Issue 5, May 2018

Resource Scheduling based on Load Balancing with Fault Tolerance in Cloud Computing

Arshpreet Kaur[1], Gurpreet Kaur[2]

M. Tech. Student, Department of Computer Science, Eternal University, Himachal Pradesh, India[1]

Assistant Professor, Department of Computer Science, Eternal University, Himachal Pradesh, India[2]

Abstract: Cloud Computing is an emerging technology in today's world. The way tasks are distributed and coordinated in cloud computing is a challenging issue utilize the resource in an optimized way, avoid overload with resource failure. In this paper, we present a novel approach to load balancing through ant colony optimization (ACO) and to control resource failure. The strategies are used forward-backward ant mechanism, max-min rule, and checkpoint-based rollback recovery. To accelerate the searching process, moving probability of two ants. The proposed strategy provides the dynamic load balancing for cloud computing with less searching time whereas it will also have high network performance and fault tolerance related to the task.

Keywords: Load balancing, Cloud computing, Ant colony optimization, Fault Tolerance.

I. INTRODUCTION

Cloud computing is the latest computing and economic driven model for large-scale distributed computing for a variety of applications, cloud storage, and cloud security services have appeared. Through virtualization, Cloud computing provider its power and all kinds of resources to the custom [1]. The cloud computing system has a large scale of resources, heterogeneous resources, wide user base, different types of application tasks, QoS (Quality of Service) target constraints are different, and cloud computing systems have to deal with a large number of user tasks and massive data [2].

A. Load balancing

Heavy Load on the cloud is one of these bigger obstruction and load balancing becomes the major issue for the cloud computing. For efficient load balancing, parameters such as throughput, fault tolerance, response time, performance, scalability, and resource utilization, have to be evaluated. By having a complete analysis of these parameters, the load balancing techniques ensure better resource distribution for the user demands. The process of reallocating the total load to the individual nodes of the collective system to improve both resource utilization and job response time is called Load Balancing. It also tries to avoid a situation where some of the nodes are heavily loaded while other nodes are idle or doing very little work. Load balancing ensures that all nodes in the system approximately equal amount of work at any instance of time. The objective of load balance is to achieve optimal resource utilization, maximize throughput, minimum response time, and avoid overload [4]. Load Balancing techniques are categorized as in figure 1[2]. In cloud computing environment, data centers are intensively clusters with high-performance computers and network devices. Since the requirements of users are various and dynamically changing, and the resources and services are typically heterogeneous, it is common that the workload in the clusters is usually unbalanced. Specifically, some physical machines are over-loaded and the efficiency is affected, while others are often idle and thus the resources are wasted [3]. Therefore, how to ensure the load balance with fault tolerance is a significant problem is cloud environment.

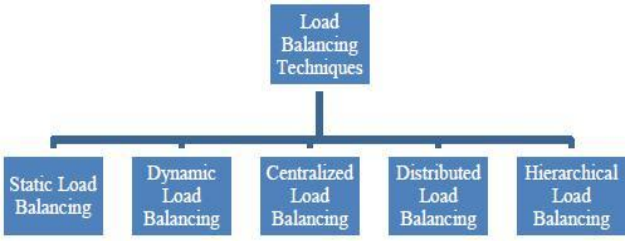

Fig.1. Different Techniques of Load Balancing

In Static approach, of load balancing, all the nodes are in the static state. Each and every node communicate among them using messages and report their status. This status of every node is reported at beginning of communication. The current state of the node is not taken into consideration. Based on prior knowledge of the system status in advance. However, in Dynamic approach, all nodes in a cloud environment will be communicating each other at a regular time

IARJSET
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
ISO 3297:2007 Certified
Vol. 5, Issue 5, May 2018

interval. Each node sends the status report of a particular node to the neighbor node.The neighbor node maintains a table knowing the current status of each node and their loads. By knowing the workload and status of every node in the cloud we can easily balance the load among all nodes [5]. In the centralized approach, a single node or server maintains the details of the nodes and the network. It updates this information periodically and centrally responsible for all load balancing activities. This approach is suitable for small networks but may suffer from bottleneck and single point failure. Distributed load balancing requires each processor in the network to keep a copy of the global state of the nodes and network in their own database. All nodes are responsible for load balancing. However, distributed load balancing can be sender initiated, receiver-initiated or symmetrically initiated by both sender and receiver. In hierarchical approach, multiple nodes are arranged in the tree structure such that the parent node broadcast the node/network performance information to their child nodes [2]. Apart from its technique there exist Load Balancing algorithms where loads in nodes are shared with the help of communication between neighbor nodes about the status of nodes to master or head node. Those algorithms are known as Swarm Based Algorithm.

B. Swarm intelligence

Swarm intelligence is the study of computational systems inspired by the 'collective intelligence'. Collective Intelligence emerges through the cooperation of large numbers of homogeneous agents in the environment. Examples include schools of fish, flocks of birds, and colonies of ants. Such intelligence is decentralized, self-organizing and distributed throughout an environment. In nature, such systems are commonly used to solve problems such as effective foraging for food, prey evading, or colony relocation. The information is typically stored throughout the participating homogeneous agents or is stored or communicated in the environment itself such as through the use of pheromones in ants, dancing in bees, and proximity in fish and birds [6].

Types of Swarm Based Algorithm for Load Balancing:

1) Genetic Algorithm
2) Particle Swarm Optimization
3) Ant Colony Optimization
4) Artificial Bee Colony

Ant Colony Optimization Algorithm is a probabilistic technique used to solve computational problems. The ants will be searching the food and find the shortest path to get the food material as needed. As wise the ACO provide the optimum solution to get the shortest path in search of food [5].

II. RELATED WORK

Load balancing plays an important role and its complexity within Cloud system generating substantial interest in the research community to optimize load balance by analyzing the task scheduling in cloud computing.

Hongwei Chen et. al. design the average random, Gaussian random and manual for setting the parameters to test the performance of the ant colony algorithm. In their paper, they analyzed the merits of ACO by comparing to FCFS, Greedy in the case of the same parameters and assumed that task is independent [1]. Later Arabi E. keshk et. al. proposed MACOLB algorithm to improve the cloud tasks scheduling for load balancing. MACOLB is used to find the optimal resource allocation for batch tasks in the dynamic cloud system and minimize the makespan of tasks on the entire system. The proposed algorithm uses the same self-adapting criteria for the MACO control parameters but has an added load balancing factor. MACOLB, MACO and ACO algorithms in applications with the number of tasks varying from 100 to 1000 evaluated using Cloudsim toolkit. Simulation results demonstrate that MACOLB algorithm outperforms MACO and ACO algorithms. MACOLB algorithm can be extended with improvements to handle precedence between tasks and costs of resources [7].

Ren Gao and Juebo Wu proposed a novel approach for dynamic load balancing based on the improved ant colony Optimization by two strategies were applied that were forward-backward ant mechanism and max-min rule and redefined by pheromone initialization and pheromone update. By means of such improvements, the speed for searching candidate nodes in load balancing operations can be greatly accelerated. Two kinds of moving rules for the forward ant were given to update the pheromone so as to speed up the convergence and the detailed dynamic load balancing algorithm was also described. The results showed that the proposed approach is feasible and effective on load balancing in cloud computing and also has better performance than a random algorithm and LBVS algorithm [8]. In 2015 Yang Xianfeng et. al. modified ACO algorithm for load balancing of virtual machines in the cloud environment. Specifically, we introduce GA and SA ideas for performance improvement [3]. Moreover, NIE Qingbin et. al. in 2016 presented advanced ant algorithm IACO based on time, cost and load balance in order to optimize task allocation in cloud computing. Test results show that IACO is superior to IABC and ACO in reducing processing time and cost and optimizing resource utilization rate in the cloud system [9]. In 2016 Jigna Acharya et. al. use swarm based load

IARJSET
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
ISO 3297:2007 Certified
Vol. 5, Issue 5, May 2018

balancing algorithm to achieve their objective which was to minimize the makespan time. They used Particle Swarm Optimization algorithm with Load balancing that covers the major goal of them. That proposed algorithm compared with another FCFS algorithm.PSO algorithm gives the better result as compared to FCFS [6].

Further Er. Mandeep Kaur and Er. Manoj Agnihotri proposed hybridization of the Genetic algorithm- Ant colony optimization (GAACO) for better performance of GA and ACO adopt multi-objective function. In this different evaluation, parameters were considered those are Execution time, Makespan time and Response time. The hybrid Genetic Algorithm- Ant colony optimization (GAACO) has shown the best global optimization solution [10]. Awatif Ragmani et. al. given the improved architecture for load balancing by offering to share the functions 01' the main controller representing the entry point to Cloud into two parts. Firstly, the main controller will keep the function of partitioning users' tasks through different regional Load Balancers. Secondly, the auxiliary controller will ensure the updating state of the system through various agents. The proposed architecture of the Cloud is organized on three levels and each level has a specific role and specific algorithm. The main controller applies an algorithm based on the geographic location of both user and data center in order to choose the closest load balancer. The regional load balancer applies a second policy based on ant colony optimization algorithm to allocate the appropriate nodes to users' tasks. Through this approach, we aim to optimize the whole response times of services in the Cloud [11].

In 2017 Hajara Idris et. al. describes the incorporation of a fault tolerance system, which is used to enhance the performance of the resource scheduling algorithm in Grid computing environment. This was achieved by introducing into the existing ACO a check pointing mechanism. By adding fault tolerant features within the scheduling approach, the overall performance of the Grid system is improved. The fault tolerance based resource scheduling strategy provides better results than the resource scheduling algorithm without fault tolerance in terms of various performance metrics, such as makespan, average turnaround time and the number of jobs completed [12]. M. Nagalakshmi et. al. aims to development of enhanced strategies through improved job and load balancing resource allocation techniques. Ant colony optimization algorithm and throttled algorithm dynamically allocates the resource to the job in a queue leading reduced cost in data transfer and virtual machine formation. The simulation result shows the reduction up to 50-60% in the cost and time [13]. Later on, Jyoti Rathore et. al. surveyed various load balancing techniques for cloud computing. The main purpose of load balancing is to satisfy the customer requirement by distributing load dynamically among the nodes and to make maximum resource utilization by reassigning the total load to the individual node. This ensures that every resource is distributed efficiently and evenly. So the performance of the system is increased [14]. Similarly, M. Padmavathi and Shaik Mahaboob Basha surveyed multiple ACO algorithms for load balancing in cloud computing environment. They discussed the factors should consider for Performance Evaluation of Load balancing Algorithm and different Nature inspired technologies and Ant Colony System (ACS) terminology [15].

Qiang Guo in his paper proposed the Multi-Objective- Improved Ant colony optimization (MO-ACO) algorithm, which considers the makespan, cost and load balancing. The algorithm establishes the constraint function of task completion time, cost, and load, improves the basic ant colony algorithm heuristic function and pheromone update rule, and adopts the pseudo-random transition probability rule in ant colony system [2].

Imen Chaouch et. al. applied 3 versions of a bio-inspired algorithm which are the Ant System (AS), the Ant Colony System (ACS) and a Modified Ant Colony Optimization algorithm (MACO) to solve the Distributed Job-shop Scheduling Problem with makespan minimization criterion that is a well-known NP-hard problem. First, we used the disjunctive graph to model the problem and then, an effective way to assign jobs to a factory is sketched based on the good workload balancing between the factories. The results show that the MACO outperforms the ACS and the classic AS [16]. Ashish Gupta and Ritu Garg multi-objective scheduling algorithm considering the optimization of makespan and load balancing has been proposed. The algorithm uses the ACO approach to obtain the local optimal solutions, finally, non-domination sorting is applied to obtain the Pareto set of solutions representing the trade-off between makespan time and the load balancing in the cloud. At last, the results specify that the proposed LB-ACO algorithm for task scheduling outperforms in comparison to NSGA-II [17].

In 2017 Young Ju Moon et. al. a novel ACO algorithm called SACO with slave ants for scheduling tasks in cloud computing environments. We adapt diversification and reinforcement strategies with slave ants to avoid long paths whose pheromones are wrongly accumulated by leading ants. It introduces minimal preprocessing overheads for slave ants and outperforms the existing ACO based cloud tasks scheduling strategies. Experimental results show that SACO solves the NP-hard problem in an efficient way while maximizing utilization of cloud servers [18]. Hongyan Cui and his colleague in 2017 present a cloud service task scheduling model TSS, which consists of three modules: a user module, a data center module, and a task scheduling module. In the user module, we assume that the tasks model is a Poisson process. In the task scheduling module, because of the deficiencies of GA and ACO, we have proposed a GA-CACO algorithm (Genetic Algorithm- Chaos ant colony optimization), which merges GA and CACO. And finally, we

IARJSET
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
ISO 3297:2007 Certified
Vol. 5, Issue 5, May 2018

compared GA-CACO, GA, and ACO for several different task sizes. The results indicate that GA-CACO algorithm is optimal for the optimization of the objective function and also has acceptable convergence speed [19].

Danlami Gabi and Abdul Samad Ismail presented a multi-objective task scheduling model based on execution time and execution cost objective is presented. An algorithm called Cloud Scalable Multi-Objective Cat Swarm Optimization based Simulated Annealing (CSMCSOSA) is proposed to solve the model. The obtained results have confirmed that our method has achieved better performance and returned better scalability value with an acceptable range of $0 < \psi < 1$. The proposed CSM-CSOSA has shown to improve its quality of solution at the latter stage of search procedure making it more efficient for cloud task scheduling [20].

R.K. Jena presented multi-objective CSA based optimization algorithm which can solve the task scheduling problem under the computing environment, whereof the number of data center and user job changes dynamically. The multi-objective CSA based algorithm is suitable for cloud computing environment because the algorithm is able to effectively utilize the system resources to reduce energy and makespan. The experimental results illustrated that the proposed methods (TSCSA) out-performed the maximum applications scheduling algorithm and random scheduling [21].

In 2018 Abdulrahman Abdulkarim et. al. proposed an improvement to the round robin load balancing algorithm by whereby a method was adopted to use average burst time of requests as the time quantum for the time-sharing principle on the round robin algorithm. It was expected that this work would result to more effective load balancing by achieving a minimal response time of user base tasks, minimal data center processing time, increase overall throughput and therefore results to better resource utilization on the cloud [22].

III. PROPOSED WORK

Master-slave architecture is a mature architecture with a single master server or job tracker and several slave servers, which has been widely used in cloud computing like Google's MapReduce and Hadoop. Figure 2 shows the typical scenario of the network topology of virtual resources in cloud computing, which is based on the master-slave architecture and the cloud platform discussed in this paper.

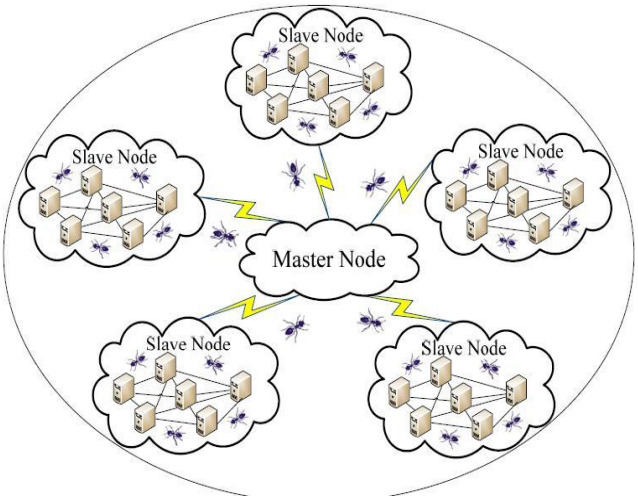

Fig 2: Network topology of virtual resources in cloud computing

In master-slave architecture, a job is first submitted to a master node by the user. Then the job is divided into several executable tasks in the master node and the generated tasks are distributed to different slave nodes. After that, the tasks are executed in the slave nodes separately with the guidance of the master node, and the results are returned to the master node. Finally, the distributed results are combined together in the master node and sent to the requesting user. Furthermore, the master node is responsible for monitoring the whole steps and re-executing the failed tasks.

During this process, the uneven distribution of tasks may cause that some slave nodes are in light load conditions while others are under heavy load conditions. In this case, load balancing operation ought to be carried out dynamically for the cloud platform in order to keep the platform stable and operating efficiently. In our approach, the main procedure of load balancing with ACO consists of two steps before load balancing execution as below.

1) Ant generation. Check the cloud platform periodically, and generate ants if and only if there are existing overload nodes or under load nodes; and

IARJSET
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
ISO 3297:2007 Certified
Vol. 5, Issue 5, May 2018

2) To find target nodes. According to searching rules, the ant is looking for the target nodes which meet the conditions of load balancing in its surrounding area. The target node is also called the candidate node for load balancing.

For fast convergence, we employ two diverse ants in search of the slave nodes. At the same time, we leverage max-min rules to trigger ant generation, so as to improve the efficiency of the algorithm.

A. Forward-Backward ant Mechanism

The ants are divided into two categories: forward ant and backward ant, which is the same mechanism described in but with the distinct definition.

i. The forward ant is responsible to find the candidate nodes for load balancing in cloud computing platform and it starts the searching activity from its generated node. The candidate nodes include overload nodes and under load nodes.

ii. The backward ant is in charge of updating information pheromones for the path as that of its corresponding forward ant but in the opposite direction. The backward ant is generated at each time when the forward ant identifies a candidate node.

To simulate the meeting process, we make use of a timer to record the life cycle of a backward ant after it is produced. Some storage unit is set for each node and they are used to save the information pheromone carried by backward ants, with one unit for one backward ant.

B. Max-Min Rules

We define two distinct rules, named max-min rules, to trigger the forward ant generation, with the purpose of reducing the time for searching candidate nodes as below.

Rule 1: Maximum value trigger rule. A forward ant is generated from a slave node when the load in this node is greater than a certain threshold. It indicates that the node has been running close to or beyond its maximum load, which needs to distribute the tasks to idle nodes so as to achieve optimal resource utilization.

Rule 2: Minimum value trigger rule. A forward ant is generated from a slave node when the load in this node is smaller than a certain threshold. It denotes that the node is running in the light load state, which can accept a range of new tasks, in order to share its resource to the overload nodes.

With respect to the strategy presented above, we further analyze the details of the moving probability with ant colony optimization in this section. To explore both load allocation efficiency and network performance, two critical issues must be addressed. First, pheromone initialization should be reasonable in the cloud computing environment, to satisfy the desired QoS. Second, the pheromone update has to meet the dynamic demand of the workload variability, with the aim of accelerating the convergence.

C. Pheromone Initialization

In cloud computing, the physical resources allocated to each virtual node are not the same and usually changing dynamically. Five physical resources are involved in pheromone initialization here, that is, CPU (number of cores and MIPS for each core), internal storage, external storage, I/O interface, and bandwidth. The CPU capability can be calculated by:

(1)

$$p_{CPU} = n \times p$$

To facilitate the calculation, we set an upper limit for each parameter. When a parameter exceeds its upper limit, the limit value is chosen instead of the actual value. The limits are given in Equation (2)

$$p_{CPU} \le p_{\max}, m_i \le m_{i\max}, m_e \le m_{e\max}, p_{i/o} \le p_{i/o\max}, p_b \le p_{b\max}$$

(2)

The capability definitions of physical resources of virtual machines are defined as below.

For CPU:

(3)

$$\tau_{CPU}(0) = \frac{p_{CPU}}{p_{\max}} \times 100\%$$

IARJSET
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
ISO 3297:2007 Certified
Vol. 5, Issue 5, May 2018

For internal storage

(4) $$\tau_{mi}(0) = \frac{m_i}{m_{i\max}} \times 100\%$$

For external storage

(5) $$\tau_{me}(0) = \frac{m_e}{m_{e\max}} \times 100\%$$

For I/O interface:

(6) $$\tau_{i/o}(0) = \frac{p_{i/o}}{p_{i/o\max}} \times 100\%$$

For bandwidth:

(7) $$\tau_b(0) = \frac{p_b}{p_{b\max}} \times 100\%$$

Therefore, we define the pheromone initialization for one slave node as:

(8) $$\tau_i = \psi_1\tau_{CPU}(0) + \psi_2\tau_{mi}(0) + \psi_3\tau_{me}(0) + \psi_4\tau_{i/o}(0) + \psi_5\tau_b(0)$$
$$\sum_{n=1}^{5} \psi_n = 1$$

where ψ*n* is a weight coefficient, which is used to adjust the influence of the physical resources in cloud computing**.**

D. Pheromone Update

The goal of pheromone update in our approach is to increase the pheromone values for slave nodes associated with good conditions and decrease those associated with bad ones. Three factors that impact the pheromone update are considered in our strategy, namely pheromone evaporation, update by task, and incentives for successful tasks

i) Pheromone Evaporation

Pheromone in the node is decreasing over time due to evaporation. We use the local update strategy to modify the pheromone on slave nodes where the pheromone is not zero. The pheromone update by evaporation is defined as:

(9) $$\tau_i(t+1) = (1-\rho) \times \tau_i(t), \ 0 < \rho < 1$$

where τ*i* (*t*) is the pheromone in slave node *i* at *t* moment and ρ is a coefficient of evaporation.

ii) Pheromone Evaporation by New Task

The capability of a slave node is changed when a new task is allocated to this node. After receiving new tasks, the capability of this slave node decreases because of the resources being consumed. In this case, the pheromone update is obtained by:

(10) $$\tau_i(t+1) = (1-\mu) \times \tau_i(t), \ 0 < \mu < 1$$

where μ is the factor to adjust the degree of resources consuming.

The capability of a slave node is also changing when a new task is performing as time goes by. The capability of this slave node increases owing to the releasing of resources. In this case, the pheromone is increased by:

$$\tau_i(t+1) = (1+\nu) \times \tau_i(t), \ 0 < \nu < 1$$

(11)

where ν is a factor to control how fast the resources are released.

iii) Pheromone Evaporation for Backward Ant

The pheromones of backward ants are stored in slave nodes, which are controlled by the relevant timers. Like normal pheromones of each node, the pheromones of backward ants are evaporated over time. The pheromone update by evaporation for backward ants is defined as:

$$\tau_i(t+1) = (1+\pi) \times \tau_i(t), \ 0 < \pi < 1$$

(12)

where π is a coefficient of evaporation for backward ant.

IARJSET
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
ISO 3297:2007 Certified
Vol. 5, Issue 5, May 2018

E. Checking Point

Checkpointing is a combination of two activities, first, it saves the running data and restores it after getting a suitable resource. Second, it captures the states and data of a running process including registers obtaining the address, variables, libraries, data structures, files containing data with a large size, etc. An application-level checkpointing tool is adopted, which is directly implemented within the application source code. That is, the application contains source code that saves and restores critical program variable to and from stable storage.

F. Rollback Recovery Analysis

In a conventional system, when a failure occurs, usually a job is rescheduled on another Grid resource and execution start from the beginning. A technique to avoid restarting the application from the beginning is the rollback recovery, which is based on the concept of checkpointing. The checkpointing mechanism periodically saves the application's execution state to stable storage. Hence, whenever a failure interrupts a volunteer computation, the job can be resumed from its last successful state, thereby improving the QoS requirement of the users.

When a resource failure occurs, the job is restarted exactly from the last saved checkpoint, thereby eliminating the need to start executing the job from scratch again. Hence, the execution time of the job is reduced. The submitted job starts its execution at time *JobST* and finishes its execution at time *JobET* in a normal environment, that is, the execution time of the job *ET* is given in

$$ET = (Job_{ET} - Job_{ST}) \quad (13)$$

And without considering checkpointing, if the failure occurs at time *JobFT* then the job restarts its execution from the beginning at time *JobRSTS* and finishes at time *JobET*, that is, the Total Execution Time (*TET*) of the job as given in equation

14 is the summation of the following: *JobFT-JobST; JobFT-JobRSTS; JobET-JobRSTS;*

$$TET = (Job_{FT} - Job_{ST}) + (Job_{RSTS} - Job_{FT}) + (Job_{ET} - Job_{RSTS})$$

(14)

If the job execution status is maintained in the form of a checkpoint file, and the last created checkpoint is at time *JobCP* which is before *JobF*, then the job can be restarted from the last checkpoint at time *JobRST*. Also, the total completion time as given in equation 15 will be the summation of *JobFT-JobST; JobFT-JobRST; JobET-JobRST*

$$TET = (Job_{FT} - Job_{ST}) + (Job_{FT} - Job_{RST}) + (Job_{ECT} - Job_{RST})$$

(15)

G. Checkpoint Interval

The efficiency of a checkpoint mechanism is strongly dependent on the length of the checkpointing interval. Checkpointing interval is the duration between two checkpoints. Each interval starts when a checkpoint is established and ends when next checkpoint is established. Frequent checkpointing leads to a large number of redundant checkpoints, which may enhance overhead like delay job processing by consuming computational and network resources. On the other hand, lazy checkpointing may lead to loss of significant computation because a substantial amount of work has to be redone in case of a resource failure.

Resource failure rate is used to represent the failure history of a resource. So, the resource failure rate (*FR*) is used to determine the checkpoint interval and the number of checkpoints.

Fault rates of the resources:

$$FR = \frac{N_f}{N_s + N_f} \quad (16)$$

Where *Nf* is the number of failed jobs assigned to the resources and *Ns* is the number of the jobs submitted to the resources.

The number of times to checkpoint a particular job when it is running:

$$CheckpointNumber = R_t * FR \quad (17)$$

IARJSET
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
ISO 3297:2007 Certified
Vol. 5, Issue 5, May 2018

where Rt = Response time.

Checkpoint interval time when a job should be checkpointed:

$$checkpointInterval = \frac{R_t}{R_t * FR} \quad (18)$$

VI. CONCLUSION

In this paper, two dynamic load balancing strategies were applied optimization with the forward-backward ant mechanism and max-min rules based on the improved ant colony and fault tolerance is introduced with another strategy that is checkpointing that comes with the feature of rollback and check interval. These strategies will speed up the searching candidate nodes in load balancing and convergence speed accreted with fault tolerance strategy overall performance will increase.

REFERENCES

[1] Hongwei Chen, Lei Xiong, Chunzhi Wang, "Cloud Task Scheduling Simulation via Improved Ant Colony Optimization Algorithm", *Journal of Convergence Information Technology (JCIT)* Volume8, Number7, April 2013.
[2] Qiang Guo, "Task scheduling based on ant colony optimization in the cloud environment", *5th International Conference on Computer-Aided Design, Manufacturing, Modeling and Simulation (CDMMS 2017),* 2017.
[3] Yang Xianfeng and Li HongTao," Load Balancing of Virtual Machines in Cloud Computing Environment Using Improved Ant Colony Algorithm", *International Journal of Grid Distribution Computing* Vol. 8, No.6, 2015.
[4] Amanpreet Kaur, Dr. Bikrampal Kaur, Dr. Dheerendra Singh" Optimization Techniques for Resource Provisioning and Load Balancing in Cloud Environment: A Review", *I.J. Information Engineering and Electronic Business*, 2017, 1, 28-35.
[5] M.Buvaneswari, M.P.Loganathan, Dr.S.Sangeetha," Cloud Challenges of Load Balancing and Security Issues using ICLoS Algorithm" *Second InternationalConferenceOnComputingandCommunications Technologies(ICCCT'17) IEEE.*
[6] Jigna Acharya, Manisha Mehta, Baljit Saini," Particle Swarm Optimization Based Load Balancing in Cloud Computing".
[7] Arabi E. keshk, Ashraf B. El-Sisi, Medhat A. Tawfeek," Cloud Task Scheduling for Load Balancing based on Intelligent Strategy", *I.J. Intelligent Systems and Applications,* 2014, 05, 25-36.
[8] Ren Gao and Juebo Wu" Dynamic Load Balancing Strategy for Cloud Computing with Ant Colony Optimization", *Future Internet* 2015.
[9] NIE Qingbin, LI Pinghua "An Improved Ant Colony Optimization Algorithm for Improving Cloud Resource Utilization", *2016 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, 2016.
[10] Er. Mandeep Kaur and Er. Manoj Agnihotri,"Performance Evaluation of Hybrid GAACCO for Task Scheduling in Cloud Computing" *2016 2nd International Conference on Contemporary Computing and Informatics (ic3i) IEEE, 2016.*
[11] Awatif Ragmani, Amina EI Omri, Noreddine Abghour, Khalid Moussaid, Mohammed Rida," A Performed Load Balancing Algorithm for Public Cloud Computing Using Ant Colony Optimization", *978-1-4673-8894-8/16/$31.00 ©2016 IEEE.*
[12] Hajara Idris1, Absalom E. Ezugwu, Sahalu B. Junaidu, Aderemi O. Adewumi," An improved ant colony optimization algorithm with fault tolerance for job scheduling in grid computing systems", *PLoS ONE,* 2017.
[13] M. Nagalakshmi, Dr. K. David," An Analysis of Load Balancing Algorithms in Cloud Computing" *International Journal of Advanced Research in Computer and Communication Engineering* Vol. 6, Issue 8, August 2017.
[14] Jyoti Rathore, Dr. Bright Keswani, Dr. Vijay Singh Rathore" Analysis of Various Load Balancing Techniques in Cloud Computing: A Review", *Suresh Gyan Vihar University Journal of Engineering & Technology* Vol. 3, Issue 2, pp. 48-52, 2017
[15] M. Padmavathi and Shaik Mahaboob Basha," A Conceptual Analysis on Cloud Computing ACO Load Balancing Techniques", *International Journal of Computer Science and Engineering (IJCSE)* Vol. 6, Issue 4, Jun - Jul 2017; 39-48.
[16] Imen Ben Mansoura, Ines Alayaa,"Indicator-Based Ant Colony Optimization for Multi-Objective Knapsack Problem", *19th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, 2015.
[17] Ashish Gupta and Ritu Garg, "Load Balancing Based Task Scheduling with
ACO in Cloud Computing", *International Conference on Computer and Applications (ICCA),* 2017.
[18] YoungJu Moon, HeonChang Yu, Joon-Min Gil and JongBeom Lim, "A slave ants based ant colony optimization algorithm for task scheduling in cloud computing environments", *Human-centric Computing and Information Sciences,* Springer, 2017.
[19] Hongyan Cui, Xiaofei Liu, Tao Yu, Honggang Zhang, Yajun Fang, and Zongguo Xia, "Cloud Service Scheduling Algorithm Research and Optimization", *Security and Communication Networks,* 2017, Article ID 2503153.
[20] Danlami Gabi, Anazida Zainal, Zalmiyah Zakaria, Abdul Samad Ismail," Scalability-aware Scheduling Optimization Algorithm for Multi-Objective Cloud Task Scheduling Problem" *978-1-5090-6255-3/17/$31.00 ©2017 IEEE.*
[21] R.K.Jena,"Energy Efficient Task Scheduling in Cloud Environment" *4th International Conference on Power and Energy Systems Engineering (CPESE)* 2017, 25-29.
[22] Abdulrahman Abdulkarim, Souley Boukari, Ishaq Muhammed Fatima Ahmed Abubakar," An Improved Round Robin Load Balancing Algorithm in Cloud Computing using Average Burst Time", *International Journal of Scientific & Engineering Research* Volume 9, Issue 3, March-2018.