RESEARCH ARTICLE OPEN ACCESS

Enhancing Load Balancing in Cloud Computing by ACO

1BETHALA SHIRISHA, 2THANDU NAGARAJU
1([1]Assistant Professor, MTech, Department of CSE, CMRIT, India)
2 ([2]Assistant Professor, MTech, Department of CSE, CMRIT, India)

Abstract:

Cloud computing is one of the growing technology. It provides users "pay as you go" services on demand. Now-a-days the number of users are increasing due to the attractive features it is providing. But it results in a fast growth of load on servers. Hence, load balancing has become important in the field of cloud computing. Load balancing will share the workload equally among all the nodes available in the network .So that no node is neither overloaded or underloaded and each node will do the work with the same amount of time[1]. It reduces the time and cost involved in the main computational models and helps to get better utilization of resources and the system will increase its performance. We are having many algorithms implemented by many researchers to solve the problem of load balancing. In this paper, we present a technique built on Ant Colony optimization to deal with the problem of load balancing in a cloud environment.

Keywords - Ant colony optimization, Cloud Computing, Load Balancing, Swarm intelligence;

1.Introduction

Cloud computing is one of the growing technology which offers online computing resources, Storage and it allows the users to use that resources and organize the applications with enhanced availability, Scalability and fault tolerance. The Cloud computing will allow the users to store the data on remote servers instead of their own devices. This information can be retrieved from anywhere with any device by using the internet, and that device should support cloud computing systems. Cloud computing system consists of back-end which is a collection of the computers and servers owned by a third party which stores the data and front-end, which is the client side. A central server which is a part of the back-end will follow some rules and it uses middleware to communicate between the connected computers. Cloud computing accumulates all the computing resources and manages them automatically. The Characteristics of cloud computing are automated backup and data maintenance, Automated upgrades, Web and mobile access from anywhere, elastic pay as you go i.e scale up or scale down and multitenant solution provided by vendor[2]. Cloud computing is everywhere with tools like Amazon Web Services substitutes are traditional enterprise data storage, Google Drives substitute with Microsoft Office, Dropbox storing all our data and files and banking websites substitutes are branch offices.

The cloud consists of Service layers and deployment models.

Cloud computing services are divided into three layers they are IaaS, PaaS and SaaS

1. Infrastructure as a Service (IaaS): Offering virtualised resources i.e storage, hardware and communication on demand is known as Iaas.

In an IaaS model a 3rd party provider hosts hardware, software servers, storage and other infrastructures components on behalf of its users. IaaS providers also host user applications and handle tasks including system maintenance backup and it enables on demand provisioning of servers running several choices of operating system and a customized software stack.

In this Iaas layer we requires Administrators for purchasing resources from the cloud.

Examples : Amazon EC2, Windows Azure, Rackspace, Google Compute Engine.

2. Platform as a Service (PaaS): In PaaS model the hardware, network and an operating system, are provided to the user.[3] This layer offers an

environment on which developers create and deploy applications and they do not necessarily need to know how many processor or how much memory that application is using.
In this layers we require developers to develop the application with the resources available.
Examples : AWS Elastic Beanstalk, Windows Azure, Heroku, Force.com, Google App Engine, Apache Stratos.

3. Software as a Service (SaaS): It is sometimes referred as On demand software has become a common delivery model for many business applications.
In this SaaS layer the User or clients will use the applications that are developed in the PaaS layer.
Example : Google Apps, Microsoft Office 365

Deployment Models :
The four deployment models present in cloud computing are:

1. Public cloud / Internet cloud: In the public cloud, the cloud is made available in a pay per use manner to general public. Any user can make use of the resources; it is unrestricted with the help of internet.

2. Private cloud / Enterprise cloud: The cloud here acts as internal data center of a business or other organization not made available to all users[4].

3. Hybrid Cloud / Mixed Cloud: This cloud is the combination of both private and public clouds. Some of the resources are provided and managed by public cloud and others as a private cloud.

Desired Features of a Cloud

1. **Self Service** : Cloud allow self service so that the users can request, customize, pay and use services without intervention of human operators.
2. **Per usage metering and Billing** : Services must be priced on a short time basis allowing users to release and will not pay for the resources as soon as they are not needed. Bandwidth is one of the example.
3. **Elasticity** : We can add the resources or delete the resources as per our requirement.
4. **Customization :** According to the user environment changes can also updated based on the application.

2. Load Balancing

Because of the attractive features of cloud the number of users are increasing, so the load on the servers and the quantity of processing done is rising drastically. So Load balancing is one of the serious problem in cloud computing. There are several nodes in the cloud, because of the random allocation of a request made by the client to any node, the nodes become randomly loaded[5]. So here some nodes are either over loaded or under loaded, so to avoid that problem the load balancer will equally divide the workload among all the nodes so it can aid in increasing the throughput , decreasing delays in communication, increasing resource utilization and decreasing execution time.

2.1 Objectives of load balancing:

Here we are pointing some of the objectives of load balancing algorithm

1. About system performance -- it has to give greater overall improvement at a least cost.
2. It should attain fault tolerance.
3. Load balancing have to maintain the stability of system.
4. It must be proficient of doing any changes or expansion in the distributed system configuration
5. It should take care of all jobs in the system equally.

2.2 Problems in Load Balancing

The problems in load balancing are described below:

1. Load balancing scheme will be better when it is scalable, general and stable and should add minimal overhead to the system. These requirements are interdependent.
2. The scheduling is one of the problem in load balancing. The main aim of load balancing is to avoid the inconsistency between the data locality and fairness of the requirements[6].
3. In the middle of execution, the processes may move between the nodes to make sure the equal workload in the system.
4. Task scheduling and Load balancing in distributed operating systems is a essential feature in system efficiency as the distributed system is not non-uniform and preemptive, that is, the processors may not be same.

5. How to accomplish a balance in load distribution amongst processors such that the computation can be done in the minimum possible time is one of the important problems to resolve.
6. Algorithms for load balancing have to be dependent on the hypothesis that the on hand information at each node is accurate to avoid processes from being continuously circulated the system without any progress.

2.3 Components of Load Balancing Algorithms:
A load balancing algorithm has five major components

1. **Transfer Policy**: When we are moving portion of data local node to a remote node is termed as Transfer strategy or Transfer policy.
2. **Selection Policy**: In this policy, it specifies the processors involved in the load exchange so that it can improve the overall throughput and response time.
3. **Location Policy**: The portion of the load balancing algorithm that is responsible for choosing a destination node for a task to transfer is stated as location policy or Location strategy.
4. **Information Policy**: The part of the dynamic load balancing algorithm that is in charge of gathering information about the nodes present in the system is started to as Information Policy or Information strategy[7].
5. **Load Estimation Policy**: In this policy, it determines the total workload of a node in a system.

3. Classification Of Load Balancing Algorithms:
Load balancing algorithms is divided based on present state of system and who started the task
Depending on which user initiates the process:

A. Sender-Initiated: Sender or client initiates the execution of load balancing algorithm on identifying the need for load balancing.
B. Receiver-Initiated: Receiver or server initiates the execution of load balancing algorithm on identifying the need for load balancing.
C. Symmetric: This type of algorithm is a blend of sender initiated type and receiver-initiated type algorithms.

Depending on current state of system Static algorithm: In the static algorithm, there is a uniform distribution of traffic among the servers[8]. This algorithm needs a prior understanding of system resources so that the judgment of shifting of the load does not depend on the current state of the scheme. The static algorithm is perfect in the system which has fewer inequalities in load.
Dynamic Algorithm: In the dynamic algorithm, for balancing the load the lightest server in the entire system or network is looked upon and preferred. For this real-time communication with the network is needed which can increase the traffic in the system. Here to make decisions for managing the load the current state of the system is used.

4. Related Work

Load Balancing is a vital part of Cloud Computing framework to accomplish maximum consumption of resources. Ant colony optimization (ACO) algorithm was projected by Marco Dorigo and his colleagues in 1992. It is motivated from real ants when finding for a food ant travels randomly, and in return trip, they deposited some chemical pheromone. By the quantity of this pheromone, other ants use the shortest path on which more pheromone value is deposited. The ants discover the shortest path from the nest to a food source by an indirect communication amongst the ants via pheromone trails.
Mayanka Katyal discussed various load balancing pattern such as static load balancing, distributed and non-distributed dynamic load balancing, centralized and hierarchical load balancing. Although static load balancing patterns offer simplest and effort free simulation and monitoring of the environment, they are not capable of handling heterogeneous nature of the cloud[9]. Considering dynamic load balancing algorithm, these are appropriate in a heterogeneous environment of cloud computing but are problematic to simulate.
Ekta Gupta Proposed technique based on the ACO where redistribution of overloaded nodes is done based on the threshold value. If the load on current node is less than the threshold ant will then search

for overloaded node among the neighboring nodes of the current node and move to the underloaded node by checking its Foraging Pheromone value. Here ants move only in one direction at a time.
Kun Li proposes a cloud task scheduling strategy by Load Balancing Ant Colony Optimization algorithm. The core aid of the work was to balance the whole system load while trying to minimize the make span of a given task set[10].

5. Proposed Work

Ant Colony concept was introduced by Macro Dorigo in his Ph.D. in 1992. One of the great features of ant is discovering the path between shell and food. Ants lay some chemical substances like Pheromone on the ground, when they find the path. All the ants can follow this while finding the path. Ants get attracted by this chemical pheromone, so that they follow the same path while searching for food and returning to shell. The ants consequently reach the food resources by following the pheromone trails. Pheromone strength varies based on different factors such as food source quality, food distance, etc. Based on pheromone strength ant's traverse from one place to another and they update the pheromone trail of that path at the same time. If more ants traverse through the same path, then that path becomes more feasible. If pheromone strength is maximum, then that path is the shortest path from best source of food to nest. Ants update a solution set individually by their movements. To get faster solution two different ants are used. Algorithm's efficiency can be improved by using max-min rules[11].
1. Forward movement -In the forward movement ants search for the food or they move forward to get food.
2. Backward movement - In the backward movement ants go to shell to store collected food.

5.1 Functioning of ACO Algorithm

Before the execution of load balancing our process of ACO follows two steps
1) Ant creation: cloud platform is checked regularly to know any node is overloaded or underloaded. If it happens then ants are created.
2) Finding target node: Ants based on searching rules, search for target nodes which satisfy load balancing conditions. Target nodes are also called as candidate nodes.

5.2 Max-Min Rules

We described two different rules to activate forward ant creation, which reduce time in searching target nodes. And these rules are named as max-min rules.
Rule 1: Maximum value trigger rule. From a slave node, in which load is greater than a certain threshold, forward ant is created. This implies that the node's load is closer or greater than maximum load and it needs to give out it tasks to other idle nodes.
Rule 2: Minimum value trigger rule. From a slave node, in which load is lessar than a certain threshold, forward ant is created. This implies that the node's load is very less and it needs to obtain some tasks from other overloaded nodes[12].
By above two rules optimal resource utilization can be achieved.

5.3 Process of Load Balancing

According to above strategies, primary steps of load balancing are as follows

1) Calculate moving probability of all neighbours and select the biggest one as its next destination;
2) After moving to a node decision is made whether that is candidate node or not. If yes, create a backward ant and initialize this backward ant. For forward ant, go to step 1;
3) The backward ant will go in the same path of its forward ant in opposite direction, until reaches to the starting point of its forward ant. All nodes pheromone information is updated as those are passed by backward ant. After reaching to starting point backward ant is removed.
4) Find the sum of resources of candidate nodes. If they meet the demand of load balancing stop the process. And,
5) Carry out the load balancing process. These steps are the same for max-min rules apart from the way to calculate the moving probability.

5.4 Dynamic Load Balancing

Two important issues must be considered while modelling with ACO to achieve load allocation efficiency and network performance[13].

First, in the environment of cloud computing pheromone initialization should be done properly in order to meet the required QOS. Second, pheromone update should meet the demand of the workload variation.

a. Pheromone Initialization

In cloud computing, allocation of physical resources changes dynamically to every virtual node. Because of this feature, in measuring a node's initial pheromone physical resources of virtual machines are used, as described in [14]. The physical resources concerned in pheromone initialization step are CPU, internal storage, external storage, input and output interface, and bandwidth.

By calculating physical resources capability, we can calculate CPU capability. We use Ψn for this. Ψn is a weight coefficient, that is used to adjust the influence of the physical resources in cloud computing.

5.5 Pheromone Update

For the movement of ants two types of pheromone updates are available. Ant updates a table in which there is an entry for every completed task. The type of ant movement is recognized by the type of pheromone updated. By this we can find the type of node for which ant is searching.

Two types of pheromones are given below:

1. **Foraging Pheromone**: The ants move in forward direction continuously when they come across overloaded or under loaded node. Foraging pheromone updating formula is
   FP(t+1) = (1 - βeva) FP(t) + ΔFP Where, βeva = Pheromone evaporation rate
   FP = Foraging pheromone of the edge before movement
   FP(t+1) = Foraging pheromone of the edge after movement
   ΔFP = Change in FP
2. **Trailing Pheromone:**

If an ant finds an overloaded node while moving, then it comes back to a node which was under loaded previously. If it finds that node again as under loaded, then it redistributes the work. The vice-versa is also possible.

Formula for updating value in trailing pheromone is,

TP(t+1) = (1 - βeva) TP(t) + ΔTP Where,
βeva = Pheromone evaporation rate
TP = Trailing pheromone of the edge before movement
TP(t+1) = Trailing pheromone of the edge after movement
ΔTP = Change in TP

Algorithm 1 The step by step procedure of the proposed load balancing

```
Begin
Step1: Slave nodes initialization
Step2: Read job node from the user for master node
Step3: Master node divides the job(jobn) into number of tasks
Step4: for i:=0 to n do  // n is the number of tasks alloted
        Now divide the tasks to slaves(task i);
Step 5:
  If the nodes or overloaded or underloaded Then
 {
        Produce forward_ant();
        Compute moving probability();
        Node←next node;
        If(Node==candidate)
                produce backward_ant();
                start timer::backward_ant(timer na);
               Update  pheromone::byforward_ant();
        If(na>0)//where na is timer
        Update pheromone :: by backward_ant();
                If(slave_task==successful)
                    Pheromone;
                        Else
                        Pheromone--;
 }
 Step 6: if( load_balancing==”satisfied”)
         Do::Load_balancing();
         Else if(new_tasks::”required”)
         Goto step2;
  End
```

6. **Conclusion And Future Work**

In this paper, we discussed cloud computing and load balancing. Apart from this, we also discussed the various goals, issues, components, classification, different techniques and metrics for load balancing. Load balancing is one of the major concern in cloud computing, and the main purpose of it is to satisfy

the requirements of users by distributing the load evenly among all servers in the cloud to maximize the utilization of resources, to increase throughput, to provide good response time and to reduce energy consumption. To optimize resource allocation and ensure the quality of service, this paper proposed a novel approach for dynamic load balancing based on the improved ant colony optimization. Two dynamic load balancing strategies were applied with the max-min rules and forward-backward ant mechanism. According to the characteristics of cloud computing, we redefined and improved the ant colony optimization by pheromone update and pheromone initialization. Using such improvements, the speed for searching candidate nodes in load balancing operations can be greatly accelerated. Simulations were illustrated by the improved approach in a cloud computing platform. In future work, we will study the triggering procedure for ant generation and the approach for pheromone update so as to considerably reduce the searching time for candidate nodes. Furthermore, we will investigate how to introduce other intelligent algorithms into our approach, with the purpose of improving system performance and efficiency.

REFERENCES

1. *D. Saranya et.al, "Load Balancing Algorithms in Cloud Computing: A Review," International Journal of Advanced Research in Computer Science and Software Engineering, vol. 5, Issue 7, July 2015.*
2. *T. Desai et.al, "A Survey of Various Load Balancing Techniques and Challenges in Cloud Computing," International Journal of Scientific & Technology Research, vol. 2, Issue 11, November 2013.*
3. *S. Sethi et.al, "Efficient Load Balancing in Cloud Computing using Fuzzy Logic," IOSR Journal of Engineering (IOSRJEN) ISSN: 2250-3021 vol. 2, pp. 65-71, July 2012.*
4. *Sharma S. et.al, "Performance Analysis of Load Balancing Algorithms," World Academy of Science, Engineering and Technology, 38, 2008.*
5. *S. Rajoriya et.al, "Load Balancing Techniques in Cloud Computing: An Overview," International Journal of Science and Research (IJSR), vol. 3, Issue 7, July 2014*
6. *Considering Load Balancing", Proceedings 21st European Conference on Modelling and Simulation (ECMS), 2007.*
7. *Nikravan M. et.al, "A Genetic Algorithm for Process Scheduling in Distributed Operating Systems*
8. *M. Katyal et.al, "A Comparative Study of Load Balancing Algorithms in Cloud Computing Environment", International Journal of Distributed and Cloud Computing Volume 1 Issue 2 December 2013*
9. *M. Amar et.al, "SLA Driven Load Balancing for Web Applications in Cloud Computing Environment",Information and Knowledge Management, 1(1), pp. 5-13, 2011.*
10. *D. Kashyap et.al, "A Survey Of Various Load BalancingAlgorithms In Cloud Computing", INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH VOLUME 3, ISSUE 11, NOVEMBER 2014.*
11. *Book: Ant colony optimization by Macro Dorigo and Thomas Stutzle.*
12. *Calheiros, R.N.; Ranjan, R.; Beloglazov, A.; de Rose, C.A.F.; Buyya, R. CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software 2011, 41, 23–50.*
13. *T.Deepa, & S Sharon Amulya Joshi. (2016). A Survey on Load Balancing Algorithms in Cloud. International Journal of Computer Engineering In Research Trends, 3(7), 371-374. Retrieved from http://ijcert.org/ems/ijcert_papers/3703.pdf*
14. *Kun Li et.al, "Cloud Task scheduling based on Load Balancing Ant Colony Optimization", 2011 Sixth Annual ChinaGrid Conference, 2011 IEEE.*