IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727,
PP 11-18
www.iosrjournals.org

Cloud Load Balancing Based on Ant Colony Optimization Algorithm

Namrata Goswami[1], Kunjal Garala[2], Prashant Maheta[3]
[1, 2, 3]*(B. H. Gardi College of Engineering and Technology, Rajkot, Gujarat, India)*

ABSTRACT: *Cloud Computing is growing rapidly and clients are demanding more services with better results. Therefore, load balancing as well as task scheduling in the Cloud has become a very interesting and important research area. In this paper, we present proposed approach for improving parameters in ant colonies function. The suggested definition of a new computational paradigm, is called Ant Colony Optimization (ACO) algorithm.*
Keywords—*Load Balancing, Ant colony Optimization, Pheromone, CloudSim*

I. INTRODUCTION

Cloud computing is experiencing a rapid development both in academia and industry; it is promoted by the business rather than academic which determines its focus on users' applications. This technology aims to offer distributed, virtualized, and elastic resources as utilities to end users. It has the potential to support full realization of 'computing as a utility' in the near future. With the support of virtualization technology, cloud platforms enable enterprises to lease computing power in the form of virtual machines to users. As, these users may use hundreds or thousands of virtual machines (VMs), it is difficult to manually assign tasks to computing resources in clouds. Therefore, required an efficient algorithm for load balancing in the cloud environment.

The load can be differentiated in different categories, CPU load, memory capacity, delay or network load, for example. Load balancing ensures that all the processor in the system or every node in the network does approximately the equal amount of work at any instant of time. Many methods to resolve this problem has been came into existence like Particle Swarm Optimization, hash method; genetic algorithms and several scheduling based algorithms. In this paper, we are proposing a method based on Ant Colony optimization to resolve the problem of load balancing in cloud environment.

I.

II. LITERATURE SURVEY

Ant Colony Optimization is a class of heuristic search algorithms that have been successfully applied to solving NP hard problems. This algorithm inspired from the behavior of real ants to find optimum solution. Ants have ability to find food, when it moving on the way, it laid pheromone on the ground which will be detected by other ants for following that path. When more ants choose same path it create denser pheromone and that attracts more ants. Each ant creates a pheromone table according to resource utilization. Under loaded and overloaded nodes will be founded when ants moves in forward direction. For updating pheromone table ant uses one strategy, when it find overloaded node move reverse to inform under loaded node. Each ant has separate pheromone table and its update consistency of path after finished trip. Consistency remain same while proceed from source node to destination node. This algorithm [12] is used to find optimal resource allocation for each task in the dynamic cloud system. The pheromone value will be low, if ant traverses with poor path; but if ants traverse with good path, pheromone value will be high. Best case scenario is that the under loaded node is found at beginning of the search.

III. CLOUDSIM

Simulation is a technique where a program models the behavior of the system (CPU, network and so on) by calculating the interaction between its different entities using mathematical formulas, or actually capturing and playing back observations from a production system [9]. CloudSim infrastructure modeling and simulation toolkits must support for real-time trading of services between customers and providers.

A. Cloud Simulator Characteristics

CloudSim allows simulation for modeling scenarios like IaaS (Infrastructure as a Service), PaaS (Platform as a Service), and SaaS (Software as a Service), because it offers basic components such as Hosts, Virtual Machines, and applications that model the three types of services. Cloudsim can be used to model data centers, host, service brokers, scheduling and allocation policies of a large scaled cloud platform. Hence, the researcher has used Cloudsim to model datacenters, hosts, VMs for experimenting in simulated cloud environment [7],[8],[9]. Cloudsim offers the following features:

- Support model of large scale Cloud computing.
- A self contained platform for modeling data centers, service brokers, scheduling, and allocations policies.
- Flexibility to switch between space-shared and timeshared allocation of processing cores to virtualized services.
- Cloud supports VM provisioning at two levels

1) At the host level

It is possible to specify how much of the overall processing power of each core will be assigned to each VM. Known as VM policy Allocation

2) At the VM level

The VM assigns a fixed amount of the available processing power to the individual application services (task units) that are hosted within its execution engine (Known as VM Scheduling).

B. CloudSim Work Style

The main parts of CloudSim that are related to our experiments in this paper and the relationship between them are shown in figure:

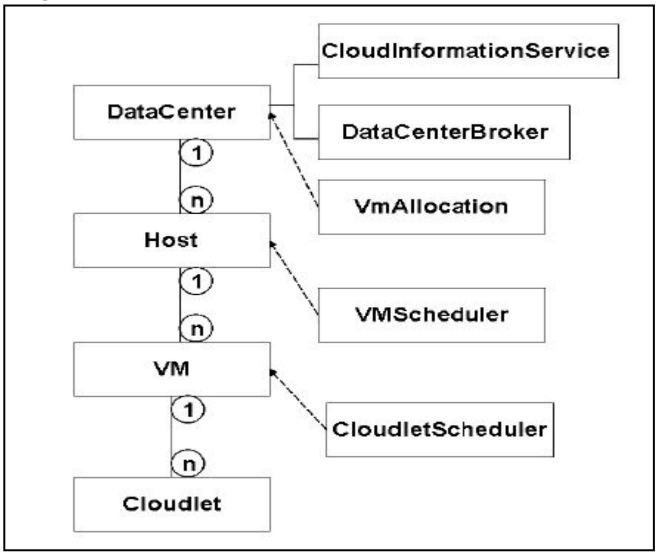


Fig.1. CloudSim Work Style

- CIS (Cloud Information Service)

It provides database level match-making services; it maps user requests to suitable cloud providers. CIS and Data-CenterBroker of CloudSim realized resource discovery and information interaction.

- DataCenterBroker

This class models a broker, which is responsible for mediating between users and service providers depending on users' QoS requirements.

- VmAllocation

The chief functionality of the VmAllocation is to select available host in a datacenter, which meets the memory, storage, and availability requirement for a VM deployment.

- VmScheduler

This is an abstract class implemented by a host component that models the policies (space- shared, time-shared) required for allocating processor cores to VMs. It is run on every host in datacenter.

- DataCenter

It models the core infrastructure-level services, which is offered by Cloud providers. It encapsulates a set of compute hosts that can either be homogeneous or heterogeneous.

- Host

It intermediate between datacenter and virtual machines (VMs). It also models a physical server.

- VM

It models a virtual machine which is run on Cloud host to deal with the cloudlet.

- Cloudlet

It models the cloud-based application services.

IV. The Basic Ant Colony Algorithm

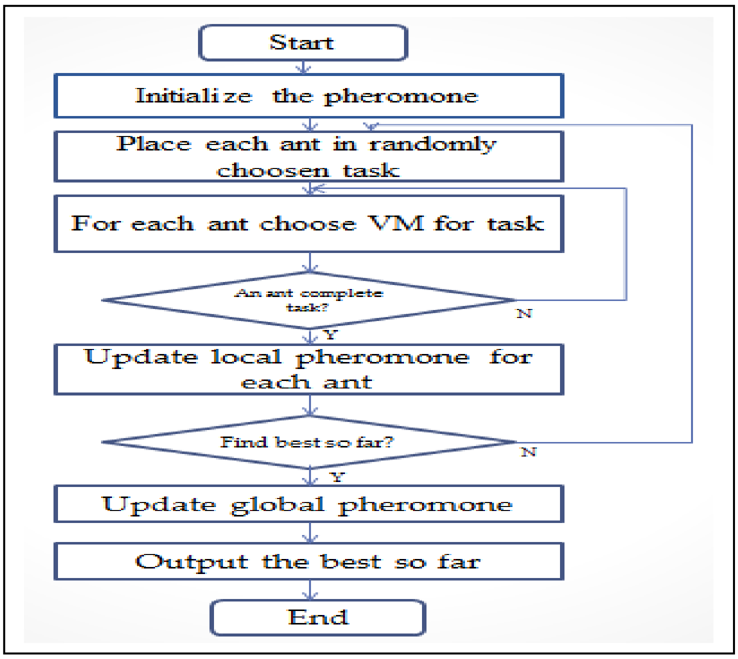
Dorigo M. introduced the ant algorithm based on the behavior of real ants in 1996, it is a new heuristic algorithm or the solution of combinatorial optimization problems. Here describes programming steps of the basic ACO(Ant Colony Optimization) as shown in Fig 2.

```
Procedure of ACO
Begin
    Initialize the pheromone
       While(stopping criterion not satisfied)do
            Position each ant in a starting VM
                  While(stopping when every ant has
                           build a solution)do
                         for each ant do
                            chose VM for next task by
                            pheromone trail intensity
                         end for
              end while
            Update the pheromone
```

Fig.2. Programming steps for basic ACO

V. The Proposed Algorithm

As shown in flowchart, utilize the characteristics of ant algorithms mentioned above to load task. Here in proposed work,carry out new load balancing depending on the result in the past load balancing that is very helpful in the cloud environment.



As shown in Fig.3 initialize the pheromone for load balancing and place each ant in a randomly chosen task. For each ant choose next task and each ant has to visit for more tasks if the result is yes then repeat the procedure for each ant otherwise return to the initial task. The next step is to update the local pheromone level using task completion . After that updating global pheromone using best so far among all local pheromone. If the result is yes then output best so far otherwise return and start again for each ant travel.

C. Programming Steps of the proposed ACO Algorithm

Assume that Tasks are mutually independent i.e. there is no precedence constraint between tasks and Tasks are not preemptive and they cannot be interrupted or moved to another processor during their execution. The experiment is implemented with 10 Datacenters with 50 VMs and 100-1000 tasks under the simulation platform. The length of the task is from 1000 MI (Million Instructions) to 20000 MI. The parameters setting of cloud simulator are shown in Table 1.

The ACO algorithm has been implemented and investigated for their relative strengths and weaknesses by experimentation. The parameters (α, β, p, t_{max}, m the number of ants and Q) considered here are those that affect directly or indirectly the computation of the algorithm. We tested several values for each parameter while all the others were held constatnt on 100 tasks. The ACO performance for different values of parameters (α, β,p, t_{max}, m the number of ants and Q) has been evaluated. Table 2 shows the selected best parameters of ACO. The programming steps of the proposed ACO algorithm based ACO as shown below:

Input*: List of Cloudlets and List of VMs*
Output*: the best solution for load allocation on VMs*
Steps*:*
*1.**Initialize** :*
***Set** Current_Best_so_far=null.*
***Set** an initial value τ(t)=0 for each path between tasks and VMs*
*2. **Place** the m ants on the starting VMs randomly.*
*3. **For** k :=1 to m do*
***Place** the starting VM of the k-th ant in VM*
***Do** ants_trip while all ants don't end their trips*
***Every** ant chooses the VM for the next task*
***End** Do*
*4. **For** k :=1 to m do*
***Update** the current_optimal_solution with the best_so_far solution.*
*5. **For** every edge task apply the local pheromone.*
*6. **Apply** global pheromone update*
***Goto** step 2*
Else
***Print** Best_so_far.*
***End** If*
Stop

Fig.4. Programming steps for ACO Algorithm

Entity Type	Parameters	Value
Task (cloudlet)	Length of task	1000-20000
	Total number of task	100-1000
Virtual Machine	Total number of VMs	50
	MIPS	500-2000
	VM memory(RAM)	256-2048
	Bandwidth	500-1000
	cloudlet Scheduler	Space_shared and Time_shared
	Number of PEs Requirement	1-4

Datacenter	Number of Datacenter	2-6
	Number of Host	2-6
	VmScheduler	Space_shared and Time_shared

Table.1. Parameters setting of cloudsim

Parameter	Value
α	.8
β	1
ρ	.3
Q	100
m	10
tmax	100

Table.2. Selected Parameters Of Aco

D. Experimental results

IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727,
PP 11-18
www.iosrjournals.org

```
Starting CloudSimExecution...
Initialising...
Starting CloudSim version 3.0
Datacenter_0 is starting...
Broker is starting...
Entities started.
0.0: Broker: Cloud Resource List received
with 1 resource(s)
0.0: Broker: Trying to Create VM #0 in
Datacenter_0
0.0: Broker: Trying to Create VM #1 in
Datacenter_0
0.1: Broker: VM #0 has been created in
Datacenter #2, Host #0
0.1: Broker: VM #1 has been created in
Datacenter #2, Host #0
0.1: Broker: Sending cloudlet 0 to VM #0
0.1: Broker: Sending cloudlet 1 to VM #1
1000.1: Broker: Cloudlet 0 received
1000.1: Broker: Cloudlet 1 received
1000.1: Broker: All Cloudlets executed.
Finishing...
1000.1: Broker: Destroying VM #0
1000.1: Broker: Destroying VM #1
Broker is shutting down...
Simulation: No more future events
CloudInformationService: Notify all
CloudSim entities for shutting down.
Datacenter_0 is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.
XXXXXXX PHEROMONE TESTING XXXXXXXXX
Testing started for 1.0 cloudlet

Ant returned with new best distance of:
9067.260876301518
Ant returned with new best distance of:
8950.528901114294
Ant returned with new best distance of:
8759.160378517805
Ant returned with new best distance of:
8746.710608320653
Ant returned with new best distance of:
8546.741493442989
Ant returned with new best distance of:
8225.860633471162
Ant returned with new best distance of:
8064.060024114328
Ant returned with new best distance of:
7950.114829212856
Ant returned with new best distance of:
7933.563248932396
Ant returned with new best distance of:
7814.093394253436
Ant returned with new best distance of:
7626.337606085747
Found best so far: 7626.337606085747
```

```
[7, 40, 18, 44, 31, 48, 0, 21, 30, 17, 2,
16, 20, 22, 19, 49, 6, 1, 41, 29, 28, 15,
45, 43, 33, 34, 35, 38, 39, 36, 37, 4,
23, 47, 14, 5, 3, 24, 11, 27, 26, 25, 46,
13, 12, 10, 32, 42, 9, 8]

Best Path found until this :
[path=1.0, best=7626.337606085747,
nestedPath=null]

Testing started for 2.0 cloudlet

Ant returned with new best distance of:
10739.598521473465
Ant returned with new best distance of:
10498.875418196698
Ant returned with new best distance of:
8834.783992536415
Ant returned with new best distance of:
8386.558407274773
Ant returned with new best distance of:
8293.559354742034
Ant returned with new best distance of:
8025.24477647533
Ant returned with new best distance of:
7894.615630270329
Ant returned with new best distance of:
7866.195053110021
Ant returned with new best distance of:
7847.5398722121445
Found best so far: 7847.5398722121445
[7, 40, 18, 44, 31, 48, 0, 21, 34, 33,
35, 38, 39, 36, 37, 23, 47, 4, 14, 5, 3,
24, 45, 15, 43, 49, 19, 22, 30, 17, 2,
16, 20, 41, 6, 1, 29, 28, 46, 25, 27, 26,
12, 13, 10, 11, 32, 42, 9, 8]

Best Path found until this :
[path=1.0, best=7626.337606085747,
nestedPath=null]

****************************************
****
Desplaying Best path found for after 2
Analization
****************************************
****
Best path found to transfer the load to
nearest node:

[path=1.0, best=7626.337606085747,
nestedPath=null]
[path=2.0, best=7847.5398722121445,
nestedPath=null]

Took Response Time: 10469 ms!

CloudSimExecution finished!
```

Using the parameters from the previous section, experiments performed for 100 jobs for experiments according to shown in Tables 1 and 2 respectively. The outputs generated by CloudSim as shown below. Here implementation done in CloudSim simulator.

VI. RESULT ANALYSIS

Here compared the proposed ACO with basic ACO for different algorithm parameters. As shown in figure 5 the response time is significantly better than the basic ACO algorithm. The results show that response time is reduced in proposed ACO compared
With Basic ACO and throughput is increased.

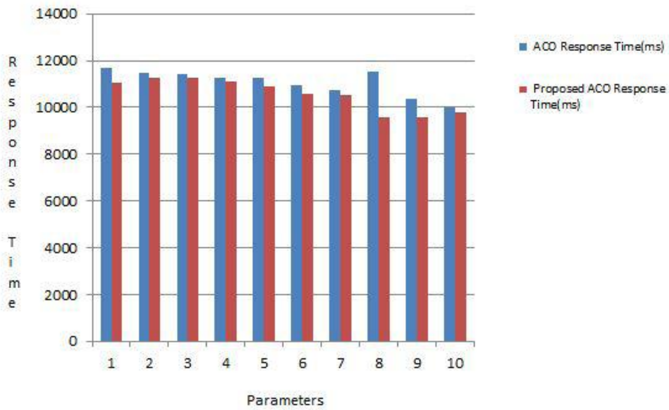


Fig.5. Comparison

VII. CONCLUSION

The proposed algorithm aims to improve performance parameters like response time and throughput in cloud environment. At present, the proposed algorithm implemented using the CloudSim simulator. The simulation result show that the response time is reduced and throughput increased compared with the Basic ACO(Ant Colony Optimization).The future work is to improve other load balancing performance parameters in proposed algorithm for more resource utilization and better performance in cloud environment.

REFERENCES

[1] Kunjal Garala, Namrata Goswami, Prashant D. Maheta,A Performance Analysis of Load Balancing Algorithms in Cloud Environment, International Conference on Computer Communication and Informatics (ICCCI -2015),2015.

[2] Raza Abbas Haidri,C. P. Katti, P. C. Saxena,A Load Balancing Strategy for Cloud Computing Environment, IEEE conference,2014, 636-641.

[3] A.KHIYAITA,M.ZBAKH,H. EL BAKKALI,Dafir EL KETTANI,Load Balancing Cloud Computing : State of Art,IEEE,2012, 106-109.

[4] Reena Thakur and Arun Yadav,Solving the Single Processor Scheduling Problem by Using Ant Colony Optimization Technique,IEEE International Conference on Communication Systems and Network Technologies,2012, 759-763.

[5] Kwang Mong Sim and Weng Hong Sun,Ant Colony Optimization for Routing and Load-Balancing: Survey and New Directions,IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, 33(5),2003.

[6] Ratan Mishra and Anant Jaiswal, Ant Colony Optimization: A Solution of Load balancing in cloud, International Journal of Web & Semantic Technology (IJWesT) ,3(2), 2012.

[7] Junjie Peng, Xuejun Zhang, Zhou Lei, Bofeng Zhang, Wu Zhang, Qing Li, Comparison of Several Cloud Computing Platforms, Second International Symposium on Information Science and Engineering, IEEE, 2009.

[8] BhathiyaWickremasinghe, Rodrigo N. Calheiros, and RajkumarBuyy, CloudAnalyst: A CloudSim-based Visual Modeller for Analysing Cloud Computing Environments and Applications, 24th International

Conference on Advanced Information Networking and Applications (AINA)IEEE Computer Society, 2010.

[9] BhathiyaWickremasinghe, RajkumarBuyya, CloudAnalyst: A CloudSim-based Tool for Modelling and Analysis of Large Scale Cloud Computing Environments, MEDC Project Report, 2009.

[10] Kun Li, Gaochao Xu, Guangyu Zhao, Yushuang Dong, Dan Wang, Cloud Task scheduling based on Load Balancing Ant Colony Optimization, IEEE Sixth Annual China Grid Conference,2011,3-9.

[11] Le Lu, Shanguo Huang, Wanyi Gu, A Dynamic Ant Colony Optimization for Load Balancing in MRN/MLN ,SPIE-OSA IEEE,2011.

[12] M. Dorigo, Ant colony optimization web page, http://iridia.ulb.ac.be/mdorigo/ACO/Aco.html

[13] Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, Cesar A. F. De Rose, and Rajkumar Buyya,CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Wiley Online Library,2010.

[14] M.Dorigo, and C.Blumb, Ant System: Optimization by colony of cooperaing agents, IEEE Trans. On SMC,1996, 29-41.