Australian Journal of Basic and Applied Sciences, 9(10) Special 2015, Pages: 356-363

ISSN:1991-8178

Australian Journal of Basic and Applied Sciences

Journal home page: www.ajbasweb.com

QoS Development of Cloud Platform Using T^{3}C Mechanism Based Dynamic Load Balancing and Target VM Selection Approach

[1]A. Selvakumar., M.Tech and [2]Dr. G. Gunasekaran., M.E., Ph.D

[1]*Research Scholar Manonmaniam Sundaranar University Tirunelveli, Tamilnadu, India*
[2]*PrincipalMeenakshi College of Engineering Chennai – 600 078, Tamilnadu, India*

ARTICLE INFO

Article history:
Received 28 January 2015
Accepted 25 February 2015
Available online 6 March 2015

Keywords:
Cloud Computing, Load balancing, CSP, T^{3}C, VM Selection, QoS.

ABSTRACT

As the cloud synonyms the environment has been deployed with number of services could by accessed through the internet technology independent of location. The loosely coupled nature of the cloud environment introduces many challenges for the service providers and the quality of service of cloud affected by various parameters like throughput, load, latency and more, where these factors are dependent on the load balancing mechanism enforced for the selection of service provider for each user request. This paper, discusses about the load balancing mechanism to improve the quality of service of the cloud environment. The paper proposes a T^{3}C (Throughput-Traffic-Time-Completeness) mechanism to perform load balancing. The method computes the all the factors for each of the service and based on that the target VM is selected to perform load balancing. The VM selection approach is performed based on the completeness measure of the VM which is computed using the previous log stored. The proposed method improves the performance of the cloud environment and increases the quality of service.

To Cite This Article: A. Selvakumar., M.Tech and Dr. G. Gunasekaran., M.E., Ph.D., QoS Development of Cloud Platform Using T3C Mechanism Based Dynamic Load Balancing and Target VM Selection Approach. ***Aust. J. Basic & Appl. Sci.,*** *9(10): 356-363, 2015*

INTRODUCTION

The development of information technology has paved the way for the growth of cloud environment. The cloud is the environment which is a collection of resources and the resources can be deployed and accessed by the external users through the internet. The cloud is a shared medium which can be accessed by different user upon registration. For example, the cost of purchasing a super computer is not possible for all the organization but needs to compute a batch job of huge size. In such situation, the organization could use the cloud.

In cloud environment the costly resources can be deployed and the external users can access the resources by paying or by getting authorization from the cloud environment. Whatever the cloud resource it can be accessed through the set of services provided. For example, there is a large set of processors deployed and it can be accessed only through the service provided to access the resource. The cloud environment provide different level of services namely IaaS (Infrastructure as a Service), PaaS (Platform as a Service), SaaS (Software as a Service). The services can be accessed through the internet technology and there are many service providers named CSP (Cloud Service Provider) who provide different services. The same set of services can be provided by different service providers and each service may be running at different locations.

The services provided by the service provider will be running on different virtual machines (VM) and each has capable of handling number of request. The capacity of each VM is limited and cannot handle more than the size of the number of request. In real world condition, the number of request approaches the cloud environment for the same task is unlimited and has to be regulated and scheduled and balanced in such a way that the cloud user must get the feeling that the cloud environment respond for his request. To provide such a mentality to the cloud user, the request has to be balanced between and shared between different virtual machines and different services dynamically. Even though there are number of types of cloud services available in the cloud environment, we focus on the SaaS to improve the performance of the cloud environment.

In general load balancing the requests approaches the cloud controller, could be balanced or shared between different virtual machine according to the traffic, or the capacity of the virtual machine. When the time orient application has entered into the

Corresponding Author: A. Selvakumar., M.Tech, Research Scholar Manonmaniam Sundaranar University Tirunelveli, Tamilnadu, India
E-mail: igaselva@yahoo.com

modern world, the users access different services through the web and they look for the timely result and the time should be shorter one and the user does not want to wait for long time. This requires more efficient load balancing inorder to improve the quality of service of the cloud environment.

The T^3C (Throughput, Traffic, Time and Completeness) based load balancing is the mechanism which considers the traffic, time complexity, throughput and completeness of each service provided by different service providers. The mechanism will be explained in detail in the section 3 of this paper.

Related Works:

There are many approaches has been discussed for the problem of load balancing in cloud environment and we discuss few among them here.

Honey Bees Inspired Optimization Method (Baris Yuce, et al., 2013), describe an optimization algorithm called the Bees Algorithm, inspired from the natural foraging behavior of honey bees, to find the optimal solution. The algorithm performs both an exploitative neighborhood search combined with random explorative search. In this paper, after an explanation of the natural foraging behavior of honey bees, the basic Bees Algorithm and its improved versions are described and are implemented in order to optimize several benchmark functions, and the results are compared with those obtained with different optimization algorithms.

A Genetic Algorithm (GA) based Load Balancing Strategy for Cloud Computing (Kousik Dasguptaa, et al., 2013), proposes a novel load balancing strategy using Genetic Algorithm (GA). The algorithm thrives to balance the load of the cloud infrastructure while trying minimizing the make span of a given tasks set. The GA load balancing strategy has been simulated using the Cloud Analyst simulator.

Load Balancing in Cloud Computing using Stochastic Hill Climbing-A Soft Computing Approach (Brototi Mondala, 2012), soft computing based load balancing approach has been proposed. A local optimization approach Stochastic Hill climbing is used for allocation of incoming jobs to the servers or virtual machines (VMs). Performance of the algorithm is analyzed both qualitatively and quantitatively using Cloud Analyst. A comparison is also made with Round Robin and First Come First Serve (FCFS) algorithms.

Cross-Breed Job Scheduling for Reducing the Server Load Using RBAC at Cloud (Kaur, N. Bansal, 2013), developing a technique named Cross Breed Job scheduling technique which would be a combination of FCFS, Priority and would be monitored by RBAC (Role based access control). RBAC is a system which checks that whether the user of the system has the access to particular content or not. If the user doesn't have the access to the content, he will be denied and the server's load would be minimized.

A Model for load balancing by Partitioning Public Cloud (Divya Thazhathethil et al., 2014), introduces a better load balancing model for the public cloud based on the cloud partitioning concept. A switch mechanism is introduced here to choose different strategies for different situations. The public cloud is divided into cloud partitions and different strategies are applied to balance the load on clouds. This paper introduces a system which has main controller, balancers and servers. The main controller selects the appropriate balancer for a particular job. The balancer further selects the server having minimum load. Hence, this system will help dynamically allocate jobs (data) to the least loaded server which will result in an efficiently balanced cloud system.

A Load Balancing Model Based on Cloud Partitioning for the Public Cloud (Gaochao Xu et al., 2013), introduces a better load balance model for the Job Seekers Web Portal based on the cloud partitioning concept in which the jobs are partitioned based on the arrival date and the Main Controller (Admin) balances the load. Enhanced Load Balancing Approach to avoid Deadlocks in Cloud (Rashmi, K.S and V. Suma and M. Vaidehi, 2012), a load balancing algorithm has been proposed to avoid deadlocks among the Virtual Machines (VMs) while processing the requests received from the users by VM migration. Further, this paper also provides the anticipated results with the implementation of the proposed algorithm. The deadlock avoidance enhances the number of jobs to be serviced by cloud service provider and thereby improving working performance and the business of the cloud service provider.

A Thrseshold – Based Dynamic Resource Allocation Scheme for Cloud Computing (Weiwei Lin, et al., 2011), study the resource allocation at the application level, instead of studying how to map the physical resources to virtual resources for better resource utilization in cloud computing environment. An threshold-based dynamic resource allocation scheme has been proposed for cloud computing that dynamically allocate the virtual resources (virtual machines) among the cloud computing applications based on their load changes (instead of allocating resources needed to meet peak demands) and can use the threshold method to optimize the decision of resource reallocation.

All the above discussed approaches produces poor load balancing and increases the latency and produces less throughput ratio.

Proposed Method:

The proposed Traffic, throughput, time and completeness based dynamic load balancing approach has different functional components namely, Preprocessing, T^3C Computation, Dynamic

Load balancing and Target Vm Selection. We discuss each of the functional component in detail in this section.

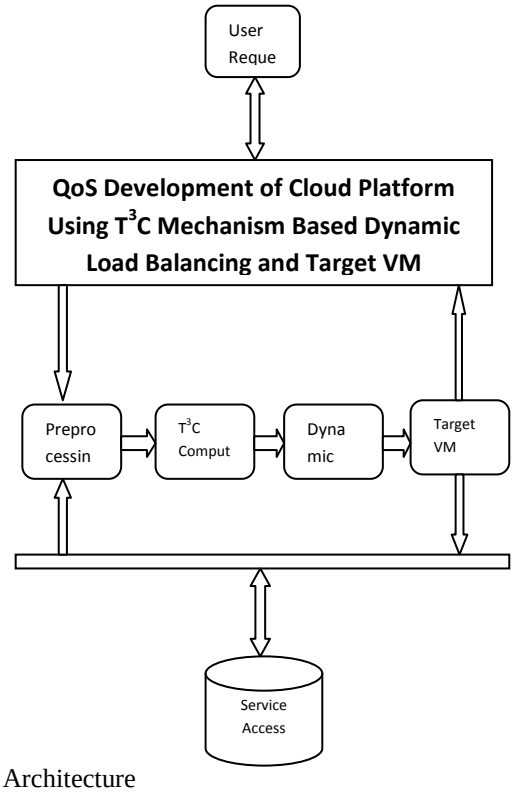


Fig. 1: Proposed system Architecture

The Figure 1, shows the architecture of the proposed system and shows the functional components of the proposed system, we discuss each of the functional component in detail in this section.

Preprocessing:

The method reads the service access trace and identifies the unique service and available virtual machine. The method identifies the set of all service providers who provides the service and for each service available, the method splits the logs and identifies the status of the service request. The method splits the logs into different time window and based on that the service access trace are split into number of groups. The identified and grouped logs are used to compute the remaining measures in the proposed approach.

```
Procedure:
Input: Service Access Trace SAT.
Output: Preprocessed Log Pl.
Start
        Identify the requested Service Rs = Request.ServiceID.
         Identify set of all similar services available.
          Service set Ss = ∑service(SAT)∑≈Rs
            For each time window Ti from Tw
                  For each service Si from Ss
                                    If SAT.ServiceID==Si then
                                    Add to Pl = ∑SAT€Pl)+SAT(i)
                                  End
              End
          End
Stop
```

The above discussed algorithm splits the service access trace of each service according to different time window and add them to the service trace or preprocessed log list. This will be used to compute the T^3C measure in the next stage of load balancing.

T^3C Computation:

The method computes the three measures namely throughput, traffic rate, time complexity and completeness of each of the service identified in the previous stage using the preprocessed log. The preprocessed log is taken into the account and at each time window for each unique service being identified, the method computes the above mentioned measures using the log available. The computed measure will be used to perform load balancing in the next stage and to perform VM selection.

Procedure:
Input: Preprocessed Log Pl.
Output: T^3C Set.
Start

For each time window T_i from Tw
For each service S_i from SSϵPl

Compute Access Frequency Af = $\sum_{k=1}^{Size(Pl)@Ti} \frac{\sum Pl(i).ServiceID==Si}{Total\ Number\ of\ trace\ at\ Ti}$

Compute Throughput ThRatio = $\frac{\sum Pl(i).ServiceID==Si \&\& Pl(i).ServiceStatus==Success}{AF} \times 100$

Compute Traffic Rate Trate = $\frac{AF}{Size(Pl)} \times 100$

Compute Time Complexity Tc = $\sum_{k=1}^{Size(Pl)@Ti} \frac{\sum Pl(i).ServiceID==Si \&\& \sum Pl(i).Time}{Total\ Number\ of\ trace\ at\ Ti}$

Compute Completeness ratio CRatio = $\frac{\sum Pl(i).ServiceID==Si \&\& Pl(i).ServiceStatus==Success}{Total\ Trace\ at\ Ti} \times 100$

Add to T^3C = $\sum T3C + \{Thratio, Trate, Tc, Cratio\}$
End
End
Stop

The above discussed algorithm computes the T^3C measure for each of the service available at each time window and add to the set. The computed measure will be used to perform load balancing.

Dynamic Load Balancing:

The load balancing is performed based on the traffic rate which represents the incoming rate of service request. Whenever a service request being received the method computes the first two stage process and with the result of T^3C measure, the method computes the possible load could be handled by the VM which executes the service. Once the load handling weight of each VM has been computed then current traffic rate for each of the VM is identified and the process of VM selection is performed to schedule the service request to a specific VM in the cloud.

Procedure:
Input: T^3C Set
Output: Scheduled VM.
Start
For each service Si from T^3C set
Compute load handling weight LHW.
LHW = $\sum_{i=1}^{size(Tw)} \sum \frac{Thratio(Ti)+Trate(Ti)+Tc(Ti)}{size(Tw)}$
Compute current time window traffic rate CTR.
Compute Current Time widnows Access Frequency CAF.

$$CAF = \sum_{k=1}^{Size(Pl)@Ti} \frac{\sum Pl(i).ServiceID==Si \&\& Pl(i).Time==CurrentTw}{Total\ Number\ of\ trace\ at\ Ti@CTW}$$

$$\text{Compute Current Traffic Rate } Trate = \frac{CAF}{Size(Pl)@CTW} \times 100$$

$$\text{Add to Service Weight List } Swl = \sum Swl(Si) + \{LHW, Trate\}$$

End

VM = VM-Selection(Swl)

Stop.

The above discussed algorithm performs load balancing in the cloud environment with the support of all the functioning modules described and choose a efficient and optimal virtual machine according to the T^3C measure and vm selection approach.

Virtual Machine Selection:

The virtual machine selection is performed using the load handling weight computed by the dynamic load balancing algorithm. From the computed load handling weight and the service weight list provided, the vm selection approach sort the available services according to the weight and computes cumulative completeness measure. Using the measures computed the services and the providing VM are ranked and sorted to choose most efficient one.

Procedure:

Input: Service Weight List Swl, T^3C Set

Output: VM

Start

For each service Si from Swl

Compute cumulative completeness measure CCM.

$$CCM = \sum_{i=1}^{size(T3set)} \sum \frac{CRatio(Si) \forall Ti}{size(Tw)}$$

End

Sort services according to LHW.

Swl = sort(Swl(LHW)).

From top N Service

Choose the service running Vm with Tratio < Th and CCM>CTH.

VM = Service.VM@Tratio<Th && Service.VM@CCM>CTH

Stop

The VM selection algorithm, computes the cumulative completeness measure for each of the service and the running virtual machine. Then it sorts the vm's according to the load handling weight and the cumulative completeness measure. If both are <> the mentioned thresholds then the VM is selected as the target to fulfill the service request.

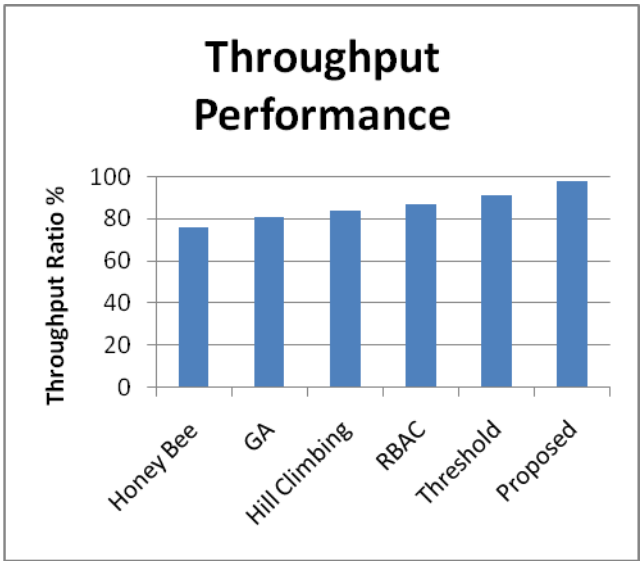
RESULTS AND DISCUSSION

The proposed T^3C Mechanism based dynamic load balancing and VM selection approach has been implemented and tested for its efficiency in load balancing. The proposed approach has produced efficient results in all the factors of quality of service of the cloud environment. The approach has been evaluated for its efficiency as per the parameters mentioned in the below table.

Table 1: Details of Simulation parameter

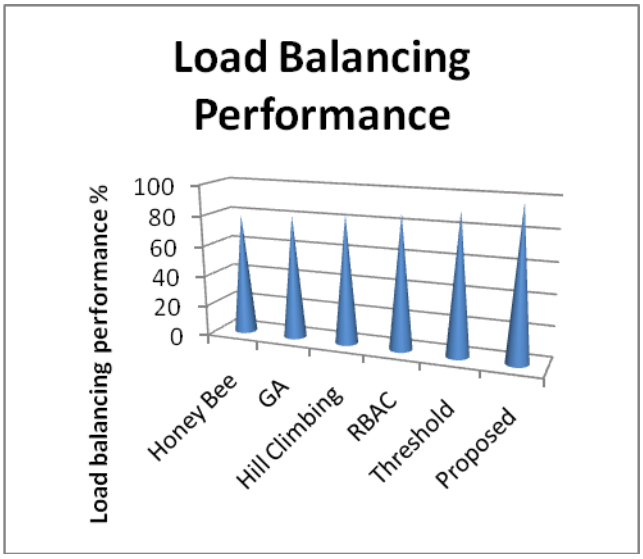
Parameter Name	Value
Tool Used	Cloud Sim
Number of Service Types	50
Average Services at each type	5
Number of users	1000

The Table 1, shows the details of simulation parameters being used to evaluate the efficiency of the proposed load balancing algorithm. The simulation environment is set with 50 different type of services and each has minimum of 5 similar services of the same type. The environment has been evaluated with the request from 1000 users.



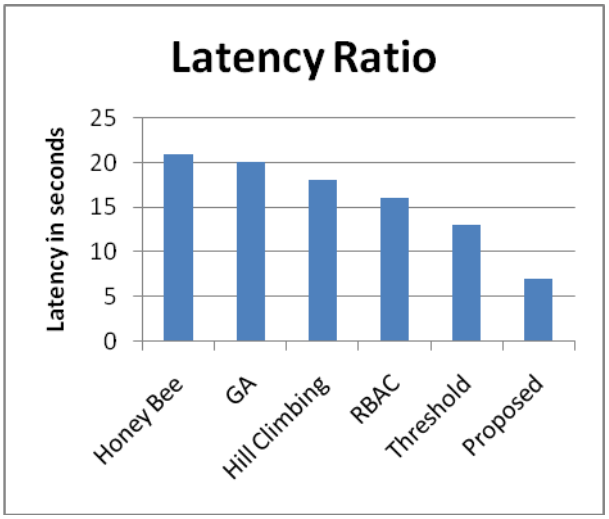
Graph 1: Comparison of throughput performance of different methods

The Graph 1, shows the results of comparative analysis performed on throughput performance with different methods and it shows clearly that the proposed approach has produced more efficient result than others.



Graph 2: Comparison of load balancing performance

The Graph 2, shows the comparative result of load balancing performance produced by different methods and it shows clearly that the proposed method produces more efficient load balancing than other methods.



Graph 3: Comparison of latency introduced by different methods.

The Graph 3, shows the comparative result on latency at load balancing introduced by different methods and it shows clearly that the proposed method produces less latency than other approaches

Conclusion:

We proposed a novel T^3C approach based dynamic load balancing and VM selection approach to improve the quality of service of the cloud environment. The method identifies the similar services available and then it splits the access trace of the services into number of time window. Then the method computes the Traffic, Throughput, Time complexity and Completeness measure for each of the service at each time window. Using the computed measure, the load balancing mechanism computes the load handling weight for each service. Then the Vm selection approach computes the cumulative completeness measure for each service and based on all the factors and measures, a top valued Vm is selected based on the completeness threshold and current traffic ratio. The proposed method produces more efficient load balancing and increases the throughput ratio. Also the method reduces the latency of load balancing which is hugely less than other methods.

REFERENCES

Baris Yuce, Michael S. Packianather, Ernesto Mastrocinque, 2013. Duc Truong Pham and Alfredo Lambiase "Honey Bees Inspired Optimization Method: The Bees Algorithm" insects Published:

Kousik Dasguptaa, Brototi Mandalb, Paramartha Duttac, Jyotsna Kumar Mondald, Santanu Dame, 2013. A Genetic Algorithm (GA) based Load Balancing Strategy for Cloud Computing" International Conference on Computational Intelligence: Modeling Techniques and Applications (CIMTA) vol 10.

Brototi Mondala,∗, Kousik Dasguptaa, Paramartha Duttab, 2012. "Load Balancing in Cloud Computing using Stochastic Hill Climbing-A Soft Computing Approach" Procedia Technology, 4: 783-789 2212-0173 © 2012 Published by Elsevier Ltd. doi: 10.1016/j.protcy.2012.05.128 C3IT-2012.

Kaur, N. Bansal, 2013. "Cross-Breed Job Scheduling for Reducing the Server Load Using RBAC at Cloud", International Journal of Advanced Research in Computer Science and Software Engineering", 3(5): ISSN: 2277 128X.

Divya Thazhathethil*, Nishat Katre, Jyoti Mane-Deshmukh, Mahesh Kshirsagar, 2014. A Model for load balancing by Partitioning the Public Cloud, International Journal of Advanced Research in Computer Science and Software Engineering, 4: 3.

Gaochao Xu, Junjie Pang, Xiaodong FuJaya, Bharathi Chintalapati,Srinivasa Rao T.Y.S. 2013." A Load Balancing Model Based on Cloud Partitioning for the Public Cloud" IEEE transactions on cloud computing year.

Akhil Goyal and Bharti. Article: A Study of Load Balancing in Cloud Computing using Soft Computing Techniques. International Journal of Computer Applications., 92(9): 33-39.

Ashish Kumar Singh1, Sandeep Sahu2, Mangal Nath Tiwari3, R.K. Katare4, 2014. Scheduling Algorithm with Load Balancing in Cloud Computing, International Journal of Scientific Engineering and Research (IJSER), 2: 1.

Saurabh, K., Garg,Chee Shin Yeo, Arun Anandasivam, 2010. Rajkumar Buyya "Environment – Conscious Scheduling of HPC application on distributed Cloud – oriented centers", Science Direct.

Rashmi, K.S and V. Suma and M. Vaidehi, 2012. "Enhanced Load Balancing Approach to avoid Deadlocks in Cloud" IJCA-ACCTHPCA.

Weiwei Lin, james Z. Wang, Chen Liang, Deyu Qi, 2011."A Thrseshold – Based Dynamic Resource Allocation Scheme for Cloud Computing" Science Direct.

Alexey Tumanov, James Cipar and Michae, A Kozuch, 2012. "Algebraic Scheduling of Mixed Workloads in Hetrogeneous Clouds", ACM.

Monir Abdullah, 2013. Mohamed Othman "Cost – Based Multi – QoS Job Scheduling using Divisible Load Theory in Cloud Computing", Science Direct.

Santosh, R. and T. Ravichandran, 2012. "Non-Preemptive onLine Scheduling of Real-Time Services with Task Migration for Cloud Computing", EJSR.

Soumya Ray and Ajanta De Sarkar, 2012. "Execution Analysis of Load Balancing Algorithm in Cloud Computing Environment in International Journal on Cloud Computing : Service and Architecture (IJCCSA)", 2.