A Dynamic Resource Allocation Framework Based On Workload Prediction Algorithm For Cloud Computing

J. Aswini, N. Malarvizhi, T. Kumanan

Abstract: ***The conventional load balancing algorithms feature severe limitations and drawbacks in cloud environments. In order to address these challenges, researchers have proposed prediction algorithms using genetic algorithms and genetic programming. These algorithms aim to simplify task scheduling in cloud platforms characterized by a large volume of users. The proposed scheme meets the requirements for inter-nodes load balancing. Simulations to compare the performance of the proposed scheme and the AGA demonstrated the effectiveness and validity of the proposed method in cloud computing.***

Index Terms: Cloud computing, Resource allocation, Workload prediction, CloudSim

I. INTRODUCTION

Over the past decade, distributed systems such as Big data , on-line social networking and Internet of things applications and cloud computing have been growing dramatically and is becoming pervasive. Cloud computing as well as social network based applications will become dominant in many aspects of life in the next few decades. The performance of such large scale systems is characterized by system capacity in terms of number of users/clients, flexibility, scalability, and effective cost of operation, etc. Among many technical challenges, resource allocation becomes one of the most important factors determining the viability of systems. Good resource allocation schemes help increase the availability and scalability of the systems as well as reducing operational costs significantly. With the number of users of social networking applications increasing quickly during last few years [1], the data generated has grown dramatically accordingly.

Manuscript published on 28 February 2019.
***** Correspondence Author (s)
J. Aswini*, Research Scholar, Department of Computer Science and Engineering,, Meenakshi Academy of Higher Education and Research, Chennai – 600 069, Tamil Nadu, India and Assistant Professor, Department of Information Technology, Jawahar Engineering College, Chennai – 600093, Tamil Nadu, India.
N. Malarvizhi , Professor & Head, Department of Computer Science and Engineering, School of Computing, Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Chennai-600062, Tamil Nadu, India.
T. Kumanan, Professor, Department of Computer Science and Engineering, Meenakshi Academy of Higher Education and Research, Chennai – 600 069, Tamil Nadu, India.

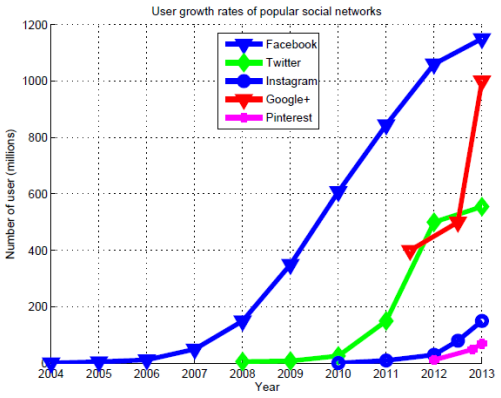

Figure 1: user growth rates of some popular social networks over the past few years

Figure 1 shows the user growth rates of some popular social networks over the past few years. Popular social networks have hundreds of millions users and continue to grow. There is no single server or even cluster (a group of servers) can handle such large amount of data. These social networking applications are served by data centers which consist of hundreds of thousands or even more than one million networked servers in total[2].

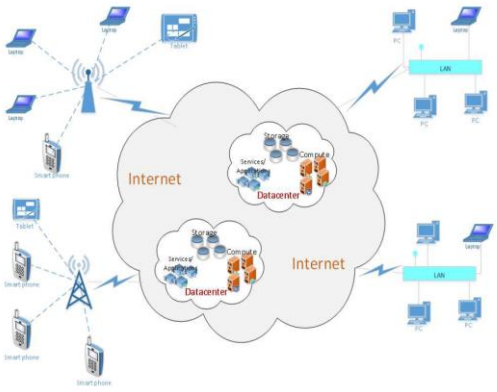

Figure 2: Example of relationships in cloud data centre

Figure 2 present a simple example of relationships between clients/users and cloud datacenters. In recent years, cloud service providers have shifted towards dynamic resource management to enable sharing of cloud computing resources between different users [3].

This intelligent resource balancing is known as workload balancing in a cloud service model[4]. Cloud service environments have adapted different provisioning strategies to improve their service level. Dynamic cloud computing technique enables resources to be assigned to different clients based on the current demand of each client turning the cloud to a limitless computational platform with limitless storage space which improves the performance of cloud services [5]. To achieve best resource allocation in dynamic hosting frameworks, cloud service providers should provision resources intelligently to all clients [6]. This intelligent resource balancing is known as workload balancing in a cloud service. Cloud service environments have adapted different provisioning strategies to improve their service level [7]. Today cloud computing enables companies to delivery different computing services such as storages, software, and databases to their clients over the Internet [8]. This resource sharing technique enables organizations to focus on their main objectives rather than on computer infrastructure and maintenance [9]. There are two resource management models, static and dynamic. Initially, cloud computing services were introduced as static computing services where a specific amount of resources was assigned to specific organizations however over the time with the rapid growth of computing needs for many organizations and business, dynamic cloud computing was introduced[10]. Dynamic cloud computing allowed cloud service providers to share and assign resources based on the demand for a specific workload. The dynamic resource management model enabled limitless computational platform with unlimited storage which improves the performance of cloud computing [11]. For instance, in a static computing, any outrages can generate downtime, wherein dynamic computing, if any outages occur the computing job can be automatically shifted to another location [12][13].

II. TECHNICAL BACKGROUND

Software As A Service is the top layer of cloud computing services where software applications mainly standard software is offered as a cloud service to the users. An outstanding example of a SaaS service is Google Docs. Google docs offer a free fully functional word processor, the spreadsheet application, and presentation creator software enabling users to collaborate with each other from different locations[14]. If users need to develop their own application on the cloud, they must use Platform As A Service (PaaS). This platform provides a cloud service environment in which developers can use appropriate APIs to make an application such as Facebook, which can be run and shared in anywhere in the world with any platforms without the risk of software pirating.

Infrastructure as a Service segment of cloud services provide developing tools with limitless storage and computing powers to developers and ordinary users. For example, Google drive and Apple iCloud offer cloud storage service for all people including ordinary users and developers. Allowing them to develop, run, and store different applications in cloud environments as shown in figure 3. For example, Amazon EC2 and Windows Azure are typical. Cloud deployment model can be categorized into 5 types:

1. Private clouds
2. Public clouds
3. Community
4. Hybrid
5. Hybrid with Cloud bursting application

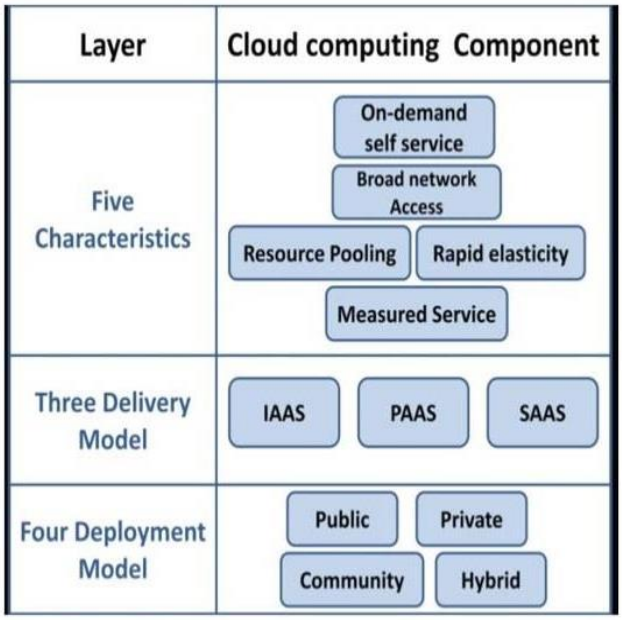

Figure 3: Cloud computing framework

A cloud-computing environment is called a private cloud when the provider and consumer are associated with each other, however, in public clouds, there are no associations between the provider and the customer as shown in figure 3. The customer rents machines from the provider either by the hour or by a different function of time. Hybrid computing is a mixture of public and private computing models and a community cloud is computing infrastructure shared between different organizations [15]. In public cloud computing, workload balancing is needed for both provider and consumer. In public cloud computing, providers must utilize their resources so that their consumers can have the assurance of receiving sufficient amount of resources. In public cloud computing, workload balancing is needed for both provider and consumer.

III. PROPOSED FRAMEWORK

The core contribution of this paper is a new predictive load balancing of running tasks, for the purpose of resource allocation. Predictive workload balancing enables cloud service providers to prepare their resource allocation for all different scenarios beforehand of any events. We will call the algorithm of allocating resources based on Cicada predictions C-Rule algorithm. Cloud resource provisioning and power consumption of data centers and its efficiency has been proven in previous research papers. C-Rule algorithm first predicts workloads during the early stage by a predictor called Cicada. Then, Cicada uses CloudSim framework to simulate the workload balancing by our rule-based algorithm.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

C-Rule Algorithm focuses on preventing over-loads in a first place rather than balancing current over-loads. In this new approach, a prediction can be achieved in a less than 20 milliseconds (LaCurts, 2014) and with a help of a Cloud simulator, an overload can be in a matter of seconds. If C-Rule algorithm detects any over-loads, CloudSim can find the most accurate resource allocation in a matter of seconds which is faster than all previous algorithms. Resource allocation with a CloudSim requires less computational power than using complex statistical and mathematical formulas for resource allocation. C-Rule algorithm can achieve the most efficient cloud resource allocation which includes number of host machines and the required number of virtual machines for each host machine with minimal resources. After finding the most system configuration for a specific workload, C-Rule algorithm will lower number of virtual machines and amount of physical memory for every given task, up until it finds the minimum resource requirement for a specific workload. C-Rule algorithm needs to receive efficient prediction data and if there are no historical prediction data then CloudSim will use the random algorithm for workload balancing until it receives a reliable workload data. Previous researches have proven that Random algorithm is the most efficient algorithm for peak time traffic and when Cicada cannot detect a reliable prediction, random algorithm can distribute the workload evenly between different VMs. Unlike other algorithms Random Algorithm connects cloudlets and servers randomly by assigning random numbers to each servers and can handle large number of requests and evenly distribute the workload to each node. In a load balancing dependent on the Random algorithm each client can be given list of available servers which can eliminate the need for a centralized broker. The main purpose of a predictive workload balancing with C-Rule is that, cloud service providers can install SFlow-enabled devices on their cloud network and gather workload data from a traffic link of their cloud network and use C-Rule to simulate the workload on a simulated network based on a specific workload and later increase the amount of workload to test the maximum handling of their workload as shown in figure 4. This method of the provisioning can also prevent any over-provisioning by finding the minimum amount of computing resources for a workload.

A. Workload Prediction

Cicada uses the data gathered from the SFlow-enabled devices to predict incoming workload as shown in figure 5.

The data collection process will comprise of the following three steps:

1. Firstly, the SFlow-enabled devices will transmit the samples to a centralized server.
2. Secondly, the centralized server will collect detailed information about the data sample including the IP address, time-stamp, and transferred bytes.
3. Thirdly, the aggregate dataset will be exported to Cicada for further estimation.

After completing the first three phases.

1. Cicada imports data from sFlow-enabled device and compares it to historical traffic data generating a workload prediction.
2. Cicada exports the prediction data to a file, which can be exported to Cloud Simulator framework.
3. C-Rule algorithm can compare the prediction data to historical predictions and if it finds any similar overload-scenario in the historical data then it can execute the previous resource allocation policy rather than a new load balancing scenario.

Both Cicada system and CloudSim framework can be installed on a same computer.

The following parameters must be transferred from Cicada to CloudSim in order to stablish a reliable simulation.

TaskCPUNum	// Number of the CPU of the task and workload /
cloudletLength	This variable contain the length of each cloudlet (the actual workload)
cloudletInputFileSize	This variable will import //input file size from the (task and workloads) section
cloudletOutputSize	//output file from the (task and workloads) section Length of Instruction from the (task and workloads) section

Figure 4: Input Parameters to Cloud Simulator.

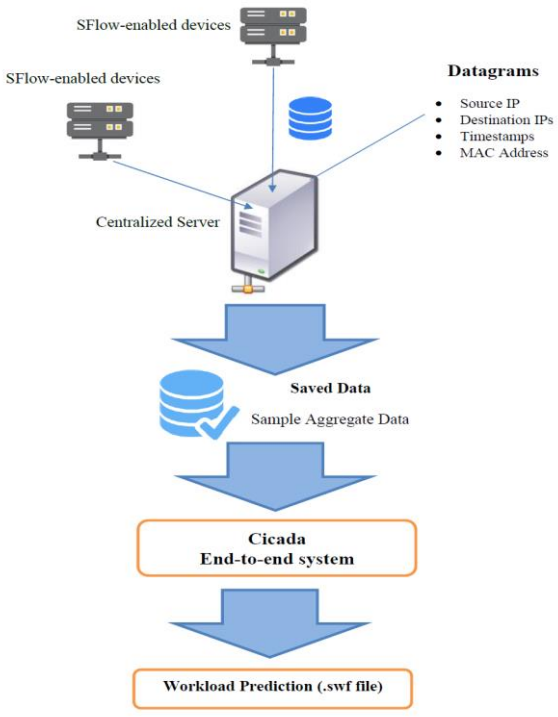

Figure 5: Architecture of workload prediction

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

All predictions are transmitted from Cicada to Cloud. CloudSim will run a simulation and detect any possible overloads.

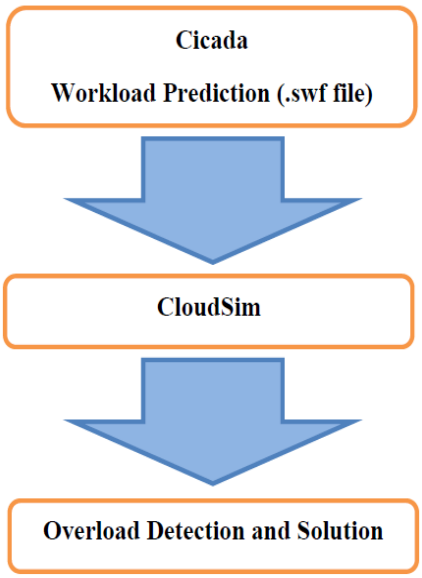

Figure 6: Architecture of overload detection and solution

If the simulation detects any overloads, it will simulate different scenarios by adding more host machines or virtual machines to achieve zero CPU waiting time as shown in figure 6. Cloud simulation can also generate list of CPU wait times for each cloudlet each time that CloudSim adds or removes different cloud resources. All waiting times can stored as set and compared by Paired t-test to the previous data set of CPY waiting times. In the following example the number of host machines has been increased from 1 hosts to 2 hosts. The sum of the waiting times have been decreased from 200020.38 to 40003.75. To make a statistical comparison, all waiting times will be added into a list and will be compared by Paired t-test.

In the following example the final values for t is 6.98 which indicates there has been a significant change.

Total # of Virtual Machines:		19	19		
Total Number of Host Machines		1 Hosts	2 Hosts		
CloudletID	STATUS VmID	WaitTime	CloudletID	STATUS VmID	WaitTime
3	Success 4	0.00	7	Success 8	0.00
1	Success 2	0.00	3	Success 4	0.00
0	Success 1	0.00	1	Success 2	0.00
2	Success 3	0.00	5	Success 6	0.00
7	Success 4	5000.06	11	Success 12	0.00
5	Success 2	5000.62	2	Success 3	0.00
4	Success 1	5000.75	8	Success 9	0.00
6	Success 3	5000.86	0	Success 1	0.00
9	Success 2	10000.88	9	Success 10	5000.24
11	Success 4	10000.77	6	Success 7	5000.13
10	Success 3	10000.99	4	Success 5	5000.24
8	Success 1	10000.99	10	Success 11	5000.94
13	Success 2	15000.91	19	Success 8	5000.94
14	Success 3	15001.69	15	Success 4	5000.49
15	Success 4	15001.55	17	Success 6	5000.94
12	Success 1	15001.95	13	Success 2	5000.84
17	Success 2	20001.13	18	Success 7	10000.46
18	Success 3	20001.91	14	Success 3	10001.53
19	Success 4	20002.43	12	Success 1	10001.31
16	Success 1	20002.88	16	Success 5	10001.53

Figure 7: Table of results from CloudSim. Adding a new host machine decreases the total wait-time.

C. C-Rule Workload balancing diagram

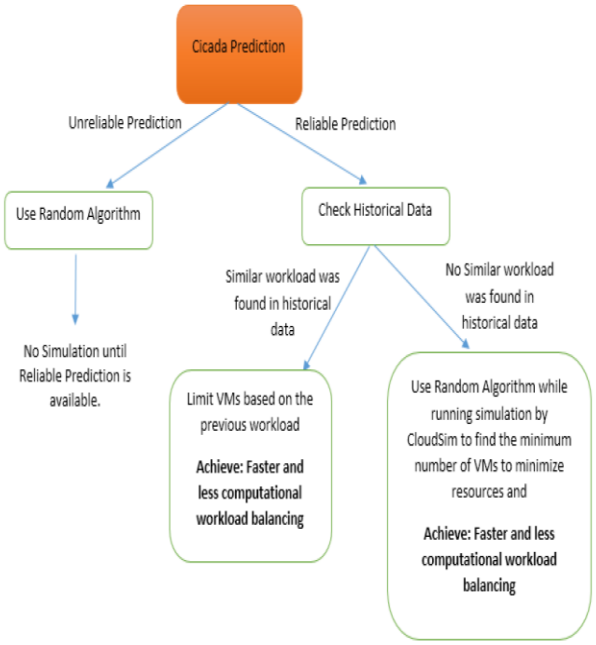

Figure 8: Architecture of the Workload Balancing Algorithm

Algorithm 1 : Proposed C-rule algorithm

1: Calculate global parameters: S, S, ΔS, F
2: Initialize T b
3: while T b > ε do
4. Select an user u
5. Select a new server s
6. Calculate Estimate F
7. if Estimate F < F then
8. Switch user u to new server s
9. else
10. Switch user u to new server s
11. with probability exp(F−EstimateF)
12. end if
13. if T b > T Decrease T b
14. end if
15. end while

Initially, Cicada generates a load prediction and if the prediction is unreliable, then it will use the random algorithm for load balancing until it receives a reliable prediction. Random Algorithm connects cloudlets and servers randomly by assigning random numbers to each server. Unlike Round Robin algorithm, Random algorithm can handle a large number of requests and evenly distribute the workload to each node as shown in figure 9. Similar to the Round Robin algorithm, another advantage of the Random algorithm is that it is sufficient for machines with similar Ram and CPU specs. The Random algorithm is the most efficient algorithm for peak time traffic and when Cicada cannot detect a reliable prediction, The Random algorithm can distribute the workload evenly between different VMs.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

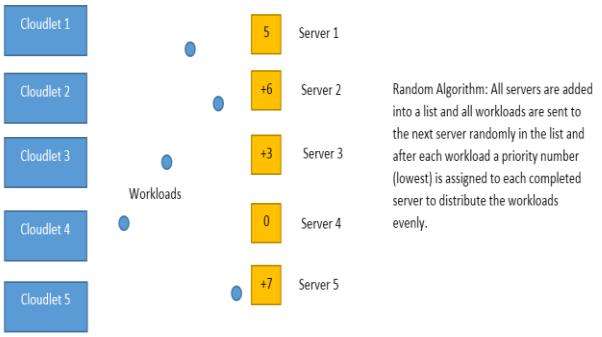

Figure 9: Random Algorithm

In the C-Rule algorithm, any unreliable prediction can be handled with the random algorithm and when a reliable workload arrives, the C-Rule algorithm can compare the incoming workload with historical workloads. If a similar historical workload is detected, the C-Rule can detect whether current resource management policy is suitable for that specific workload. If C-Rule algorithm does not find any historical data related to the incoming workload then it will begin simulating the workload

in the CloudSim simulator. Initially, C-Rule will find the optimal number for physical machines and virtual machines to achieve minimum CPU waiting time. The ideal CPU waiting time is always zero. Each time C-rule algorithm can also use the paired t-test to compare the new result to the previous one.

IV. RESULT AND EVALUATION

Previously introduced load-balancing algorithms, C-Rule Algorithm focuses on preventing over-loads in a first place rather than balancing current over-loads. C-Rule Algorithm can find a solution for any workloads in a fraction of a second. In most cases, Cicada can make a prediction in less than a 25 milliseconds and it needs a minimum of only 1 hour of historical data to make a prediction. In some cases, the speed of predictions is less than 5 milliseconds. The figure below demonstrates the speed of Cicada's prediction based on the size of the Dataset as shown in figure 10.

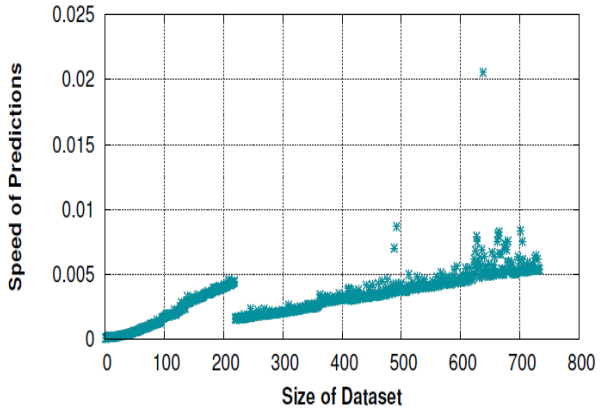

Figure 10: Speed of Predictions based on dataset

CloudSim can also simulate a workload of less than a second depending on the processing power of the centralized server. All previously introduced algorithms need complicated mathematical and statistical computation and demand a very high computational power, where the C-Rule algorithm can require a very small processing power. After achieving the minimum CPU waiting time. C-Rule algorithm will reduce the amount of resources in the simulations to find the minimum number of required resources for that workload to prevent any over-provisioning as shown in figure 11.

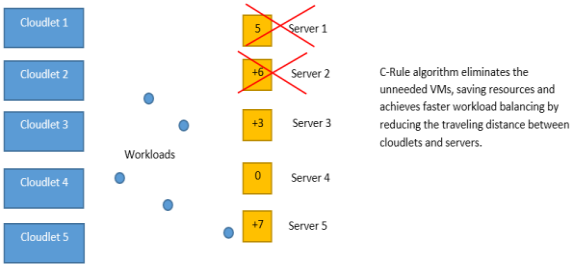

Figure 11: C-Rule eliminates excessive host machines to eliminate over-provisioning

The following figure is an example of resource reduction by a C-Rule algorithm as shown in figure 12.

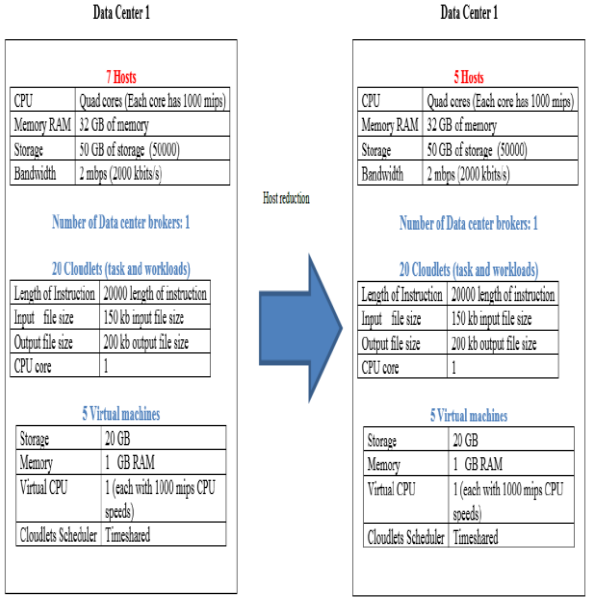

Figure 12: Resource Reduction

Total # of Virtual Machines: 20 Total Number of Host Machines: 7 Hosts / 5 Hosts						
CloudletID	STATUS	VmID	WaitTime	CloudletID	STATUS	VmID	WaitTime
3	Success	9	0.00	7	Success	8	0.00
1	Success	14	0.00	3	Success	4	0.00
0	Success	15	0.00	1	Success	2	0.00
2	Success	12	0.00	5	Success	6	0.00
7	Success	5	0.00	11	Success	12	0.00
5	Success	1	0.00	2	Success	3	0.00
4	Success	11	0.00	8	Success	9	0.00
6	Success	3	0.00	0	Success	1	0.00
9	Success	8	0.00	9	Success	10	0.00
11	Success	13	0.00	6	Success	7	0.00
10	Success	10	0.00	4	Success	5	0.00
8	Success	17	0.00	10	Success	11	0.00
13	Success	18	0.00	19	Success	8	0.00
14	Success	20	0.00	15	Success	4	0.00
15	Success	6	0.00	17	Success	6	0.00
12	Success	7	0.00	13	Success	2	0.00
17	Success	4	0.00	18	Success	7	0.00
18	Success	16	0.00	14	Success	3	0.00
19	Success	2	0.00	12	Success	1	0.00
16	Success	19	0.00	16	Success	5	0.00

Figure 13: Final result of resource reduction after achieving a zero processing wait-time.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

The above chart demonstrates a simulation result for host reduction in a successful load balancing as shown in figure 13. CPU waiting time is zero whether service provider uses 7 host machines or 5 hot machines.

V. CONCLUSION

The proposed approach enhances cloud services to achieve faster and more reliable workload balancing, allowing them to utilize their resources more efficiently by preventing any over-provisioning. Cloud service providers can use the workload prediction and if any similar workload exists in the historical data then C-Rule algorithm can simply use the result from the previous prediction. If no previous data exists in the database then C-Rule algorithm can use the prediction data and simulate a workload balancing and use that simulation data in future for a faster workload balancing. The C- Rule algorithm can balance a workload in a matter of seconds rather than several minutes.

REFERENCES

1. Al-Qudah, Z., Alzoubi, H. A., Allman, M., Rabinovich, M., & Liberatore, V. (2009). Efficient application placement in a dynamic hosting platform. In '09 Proceedings of the 18th ACM International Conference on World Wide Web, Madrid, Spain, pp. 281-290.
2. Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., & Buyya, R. (2011). CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience, 41(1), 23-50.
3. Calheiros, R. N., Ranjan, R., De Rose, C. A. F., & Buyya, R. (2009). CloudSim: A novel framework for modeling and simulation of cloud computing infrastructure and services. Technical Report GRIDS-TR-2009-1, Grid Computing and Distributed Systems Laboratory.
4. Chase, J. S., Anderson, D. C., Thakar, P. N., Vahdat, A. M., & Doyle, R. P. (2001,October). Managing energy and server resources in hosting centers. In ACM SIGOPS Operating Systems Review, 35(5), 103-116.
5. Devi, C., & Uthariaraj, R. (2016). Load balancing in cloud computing environment using improved weighted Round Robin Algorithm for non-preemptive dependent tasks. http://dx.doi.org/10.1155/2016/3896065.
6. Doyle, B., & Lopes, C. V. (2005). Survey of technologies for Web application development. ACM Journal, 2(3), 1-43.
7. Duggan, J., Cetintemel, U., Papaemmanouil, O. & Upfal, E. (2011). Performance prediction for concurrent database workloads. SIGMOD '11. June 12-16, 2011, Athens, Greece. 978 (1): 337-348.
8. Issawi, S. F., Halees, A. A., & Radi, M. (2015). An efficient adaptive load-balancing algorithm for cloud computing under bursty workloads. Engineering, Technology, & Applied Science Research, 5(3), 795-800.
9. Jena, S. R., & Ahmad, Z. (2013). Response time minimization of different load balancing algorithms in cloud computing environment. International Journal of Computer Applications, 69(17), 22-27.
10. LaCurts, K. L. (2014, June). Application workload prediction and placement in cloud computing systems (Unpublished doctoral dissertation). Massachusetts Institute of Technology, Cambridge Massachusetts.
11. Lee, R., & Jeng, B. (2011). Load-balancing tactics in cloud. In Proceedings of the International Conference on Cyber-Enabled Distributed Computing and Knowledge CyberC Discovery, pp. 447-454.
12. Mahmood, Z. (2011). Cloud computing: characteristics and deployment approaches. In the 11th IEEE International Conference on Computer and Information Technology, pp. 121-126.
13. Mathur, S., Larji, A. A., & Goyal, A. (2017). Static load balancing using SA Max-Min algorithm. International Journal for Research in Applied Science & Engineering Technology, 5(4), 1886-1893.
14. Nae, V., Prodan, R., & Fahringer, T. (2010, October). Cost-efficient hosting and load balancing of massively multiplayer online games. In the 11th IEEE/ACM International Conference on Grid Computing (GRID), Brussels, Belgium, pp. 9-16.
15. Nema, R., & Edwin, S. T. (2016). A new efficient virtual machine load balancing algorithm for a cloud computing environment. International Journal of Latest Research in Engineering and Technology, 2(2), 69-75.

AUTHORS PROFILE

Ms. J. Aswini, currently pursuing Ph.Din theDepartment of Computer Science and Engineering at Meenakshi Academy of Higher Education and Research (Deemed to be University), Chennai. She did her B.Sc Computer Science, Master of Computer Application and Master of Computer Science and Engineering in 2000 , 2003 and 2011 respectively from Madras University and Anna University, India. At present she is working as an Assistant Professor in Department of Information Technology at Jawahar Engineering College, Chennai, Tamil Nadu and She has eight years of teaching experience. She has published several papers in International Conferences and Journals. Her research interests include Cloud Computing and Internet of Things.

Dr. N. Malarvizhi, currently working as Professor & Head in the Department of Computer Science and Engineering at Vel Tech RangarajanDr.Sagunthala R&D Institute of Science and Technology, Chennai-62, Tamilnadu, India. She is having more than 15 years of teaching experience. She has written a book titled "Computer Architecture and Organization", Eswar Press, The Science and Technology Book Publisher, Chennai. She serves as a reviewer for many reputed journals. She has published numerous papers in International Conferences and Journals. Her area of interest includes Parallel and Distributed Computing, Grid Computing, Cloud Computing, Big Data Analytics, Internet of Things, Computer Architecture and Operating Systems. She is a life member of Computer Society of India (CSI), Indian Society for Technical Education (ISTE), IARCS and IAENG. She is a Senior Member of IEEE and IEEE Women in Engineering (WIE). She is a Member of Association for Computing Machinery (ACM).

Dr. T. Kumanan, currently working as Professor in Department of Computer Science and Engineering at Meenakshi Academic of Higher Educationand Research(Deemed to be University), Chennai. He received M.E and Ph.D Degrees in 2005 and 2014 from Anna University, Chennai. He has published 10 paper in International Level Conferences and Two papers in National Level Conferences. He has published 15 paper in Internal National Journal. His areas of interests are in Computer Networks, Mobile Computing, Cryptography and Network Security, Image Processing and High Speed Network. He is member of ISTE and CSI.