Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 159 (2019) 902–912

Procedia Computer Science

www.elsevier.com/locate/procedia

23rd International Conference on Knowledge-Based and Intelligent Information & Engineering Systems

A Predictive Workload Balancing Algorithm in Cloud Services

Mahdee Jodayree[a,*], Mahmoud Abaza[b], Qing Tan[b]

[a]*Department of Computing and Software, McMaster University, Canada*
[b]*School of Computing Information, Athabasca University, Canada*

Abstract

Performance of dynamic clouds depends on the efficiency of its load balancing and resource allocation. This paper is an exploratory study on the predictive approach for dynamic resource distribution of cloud services. Efficient cloud resource management can be achieved by simulating cloud services based on the predictions of incoming workloads, which can be more efficient than static allocation methods. This paper introduces a rule-based workload-balancing algorithm based on the predictions of an end-to-end system called Cicada. A simulation of cloud services can be achieved by a cloud service simulator called CloudSim and it will be used to achieve an algorithm with lower computational demand and a faster workload balancing. The final result will demonstrate the effectiveness of a predictive workload balancing approach that can achieve faster workload balancing with a lower computational power usage.

Keywords: predictive; workload; load balance; algorithm; dynamic cloud;

1. Introduction

Cloud computing services play a major role in today's computing. Leading information technology companies like Amazon's AWS, HP, Microsoft, and Google deploy large data centers with extensive hardware network for effective service delivery to cloud clients. Cloud service providers require proper resource management and provisioning to allow clients to access cloud services from the Internet [1]. In recent years, cloud service providers have shifted towards dynamic resource management to enable sharing of cloud computing resources between different users. 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. To achieve best resource allocation in dynamic hosting frameworks, cloud service provider should provision resources intelligently to all clients. This intelligent resource balancing is known as workload balancing in a cloud service models. Cloud service environments have adapted different provisioning strategies to improve their service level.

* Mahdee Jodayree. Tel.: +0-000-000-0000 ; fax: +0-000-000-0000 .
E-mail address: mahdijaf@yahoo.com

10.1016/j.procs.2019.09.250

The use of static resources for cloud services [2] has many drawbacks, for instance, static hosting in web-based platforms has unreliable service level, where a single outage can make the platforms unusable. It is also very costly and inefficient to assign a static amount of resources to a specific workload where static resources can remain unused for various periods of time. The dynamic hosting approach can enable vendors and providers to support efficient resource allocation and resource management mechanisms for their hosting platforms, however dynamic resource management for cloud services requires an efficient and a fast resource allocation algorithm.

The main contribution of this research is a proposed ruled-based algorithm called C-Rule Algorithm that would use a very efficient prediction tool [3] and simulation framework to prevent any unbalance in the system overload in a dynamic environment. This unbalance prevention will be achieved by simulating different resource allocation scenario of a predicted workload, in order to achieve an optimal resource provisioning for a specific workload with a low computation need.

2. SELECTING A PROPER PREDICTION AND SIMULATION METHOD

2.1. Selecting a Load predictor: Cicada Toolkit

There is a relationship between network traffic and the processing load of a cloud [3]. Cloud computing computers receive and forward packets via physical interfaces, typically Layer 2 technologies like the Ethernet. These technologies, or so-called network links, have their characteristics defined in terms of parameters such as bandwidth. Therefore, the amount of network traffic determines the required capacity of the network links due to the nexus between bandwidth and packet forwarding rate. The relationship between the network traffic and cloud processing workload in any region of a network is often expressed using Little's Law, which is derived from queuing systems theory [3]. The Little's Law states that the average number of items in a queue system is a product of the average rate at which the items arrive and the average time that an item spends in the system. The Little's Law expresses the ratio of the mean traffic demand to the mean number of users in a network segment [4].

Cicada toolkit and an extension called Choreo will be used in this paper for predicting incoming workload [3]. The following graph demonstrates the speed of predictions of Cicada based on the size of the Dataset

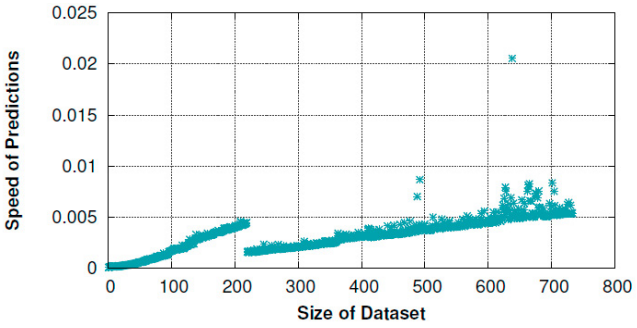

Figure 1: [5] Speed of Predictions of Cicada based on the Size of Dataset

2.2. Selecting a Proper Cloud Simulator Framework

The main step in analyzing a cloud provisioning is to simulate a Cloud computing model, where simulation enables provisioning of a Cloud computing model. To evaluate the performance of a workload model, the simulation software must be able to simulate application models, resources, and policies [5].

- CloudSim is one of the most popular and well know open-source cloud simulator. CloudSim can simulate large-scale data centers by virtualizing server hosts. CloudSim is capable of provisioning host resources to virtual machines. CloudSim can also model and simulate energy-aware computational resources and dynamic provisioning of simulation elements. In CloudSim simulation can be stopped or resumed at any time. CloudSim can simulate a cloud computing workload efficiently with a set of applications. CloudSim offers support for system modeling of Cloud systems but also it enables users to simulate system component behavior for resource provisioning such as simulation of virtual machines (VMs). CloudSim can support a single cloud as well as inter-

networked clouds, which consists of integrated clouds [5]. CloudSim allows researchers to investigate Cloud resource provisioning and power consumption of data centers.

- ICanCloud is another cloud simulation platform, which is capable of modeling and simulating many cloud computing systems. The main functionality of iCanCloud is to analyze and predict the trade-offs between performance and cost of different applications. ICanCloud is capable of simulating multiple applications in different hardware while considering information about cost. ICanCloud can model and simulate many different computing architectures with different cloud brokering policies such as customized VMs with different unicore and multi-core systems.

- GreenCloud [6] simulator is another cloud simulator which focuses on energy power consumption and cost of the physical components of a cloud computing network. With GreenCloud simulator, the workload of cloud computing scenarios and of all its infrastructural elements of a data center can be simulated in order to calculate the total cost of energy consumption.

- CloudSched is also another simulation platform which can model and simulate large Cloud computing environments such as VMs, data centers, and physical machines. CloudSched can also use different resource scheduling policies and algorithms to simulate a network infrastructure.

- An extensive study [7] on the most popular open-source cloud simulators such as ICanCloud, GreenCloud, CloudSched, and CloudSim has proven that the most efficient cloud simulator for computationally intensive tasks, data interchanges between data centers and internal network communications is **CloudSim**.

2.3. PREVIOUSLY INTERDUCED PREDICTIVE LOAD BALANCING ALGORITHMS

Many research has been conducted to explore the predictive load balancing for the cloud while introducing many different algorithms. One load balancing method introduced in [8] is to use Predictive Load Balancing Algorithm in both burst and non-burst periods to maintain service quality and minimize energy consumption of cloud network. [8] also, suggest the use of Right Scale Algorithm (RSA) for consolidating Virtual Machines (VM) into physical machines. Both algorithms use mathematical equations for load balancing and management of cloud resource, however, the research paper does not provide any simulation or real scenario to prove the efficiency of the algorithms. The prediction simply predicts the burst time and it caps the cloud resources in burst time for better management and the algorithm uses the QoS parameters to add or remove virtual machines in order to meet the QoS goal.

Another predictive load balancing research based on ensemble forecasting [9] uses a reactive overload detection method to predict any overloads. Reactive overload detection uses different CPU parameters such as static threshold (ST) value and when CPU utilization exceeds the static threshold value by 80% or 90% then it will detect it as an overload. This approach also uses various computation intensive statistical calculations to compare CPU utilization values to historical data. After detecting an overload, the research paper suggests the use of CloudSim for forecasting CPU utilization at a theoretical level. The proposed concept in this paper only detects an overload when it already has happened and it proposed prediction method is very computationally intensive. The CloudSim simulation proposed in this paper is only in a theoretical level and this paper does not provide any clear simulation results to prove its method and suggest further study in order to improve the load balancing results.

Another load prediction study for energy-aware scheduling [10], suggests training predictors for predicting a load without mentioning any accurate tool for prediction.

The literature review of predictive load balancing algorithms for cloud indicates that all previous literature has used statistical calculations for the prediction that predicts overloads that have already begun to happen. All the previous methods need high computational and centralized approaches that need to be configured and trained for overloads. For load balancing all previous literature have used mathematical equations which needs high computation power for load balancing and management of cloud resource without offering any simulation or real scenario to prove the efficiency of the introduced algorithms. This literature tends to introduce a new accurate and reliable and less computational approach for cloud load prediction which can accurately predict cloud load. This literature will also investigate all cloud simulation frameworks, in order to find the most accurate simulation platforms.

2.4. Workload Balancing Algorithms

The main goal of load balancing is to achieve the minimum process execution wait time with minimum amount of computational resources. In a perfect load balancing which has a zero execution wait time, all processes are handled simultaneously and there are no wait-times for processing information. Many algorithms were introduced to address workload prediction and workload balancing, the most popular algorithm for workload balancing are Round Robin, Random Algorithm and least loaded algorithm. The following literature below explains the most concept behind the popular workload balancing algorithms.

2.5. Round Robin and Random Algorithms

- Round-Robin (RR) is scheduling technique that achieves load balancing by assigning equal time quanta to cyclic tasks and processes [11]. In RR, the algorithm divides time quanta is into equal slices and assigns with the specific time interval. The time scheduling principle describes the scheduling of the time slides when using the algorithm such that all the nodes are assigned with a quantum and with an operation. All resources are treated as time slices. While RR provides an efficient mechanism for load balancing in terms of meeting peak user demands and providing high quality services, this approach presents significant challenges in bursty workloads [12].
- Bursty workload refers to uneven pattern of data transmission, a common problem in large systems such as web-based applications. The problem with bursty workload is that it can degrade system performance and lead to system unavailability. Burstiness is a major problem in the context of cloud computing given the increasing number of cloud users. Static algorithms such as RR have inherent limitations given that they depend on prior knowledge without considering current state of a node. This means that the algorithm can degrade system performance. The limitations of RR algorithms in environments characterized by bursty workloads indicate the need for enhanced algorithms.

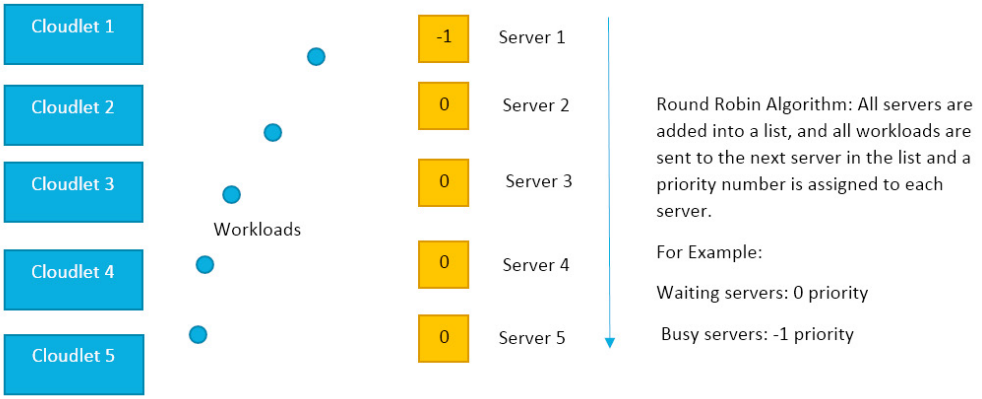

Figure 2: Round Robin Algorithm

- Random-Algorithm: Random Algorithm connects cloudlets and servers randomly by assigning random numbers to each server. Unlike the Round Robin algorithm, the Random algorithm can handle a large number of requests and evenly distribute the workload to each node. Similar to the Round Robin algorithm, another advantage of 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, random algorithm can distribute the workload evenly between different VMs.

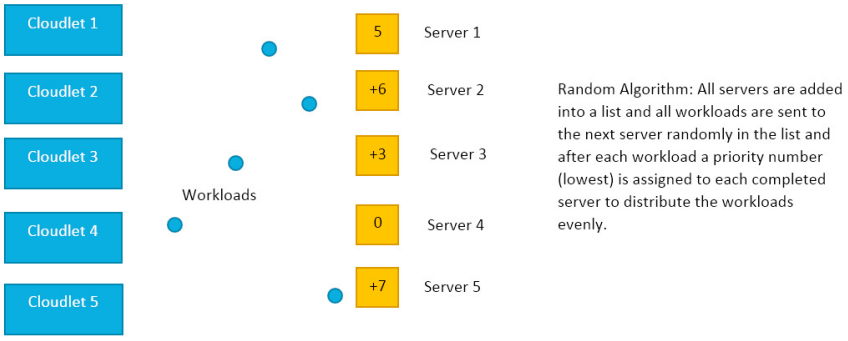

Figure 3: Random Algorithm

2.6. Introducing Predictive Workload Balancing Algorithm For Cloud Services

- Introducing C-Rule Algorithm

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.

The most reliable (in our estimation) load prediction tool called Cicada and a reliable cloud simulation framework called CloudSim, which allows researchers to investigate 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. C-Rule Algorithm focuses on preventing over-loads in a first place rather than balancing current overloads. In this new approach, a prediction can be achieved in less than 20 milliseconds [3] 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 a 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, the C-Rule algorithm will lower the 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 reliable workload data. Previous researches have proven that 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. Unlike other algorithms, Random Algorithm connects cloudlets and servers randomly by assigning random numbers to each server and can handle a 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 a 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. This method of the provisioning can also prevent any overprovisioning by finding the minimum amount of computing resources for a workload.

2.7. Workload Prediction Concept Introduced by Cicada and Choero

Cicada uses the data gathered from the SFlow-enabled devices to predict incoming workload.

The data collection process will comprise 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, timestamp, 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 a 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 the 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 the same computer. The following parameters must be transferred from Cicada to CloudSim in order to establish 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

Table 1: Table of Input parameters from Cicada to CloudSim Simulator.

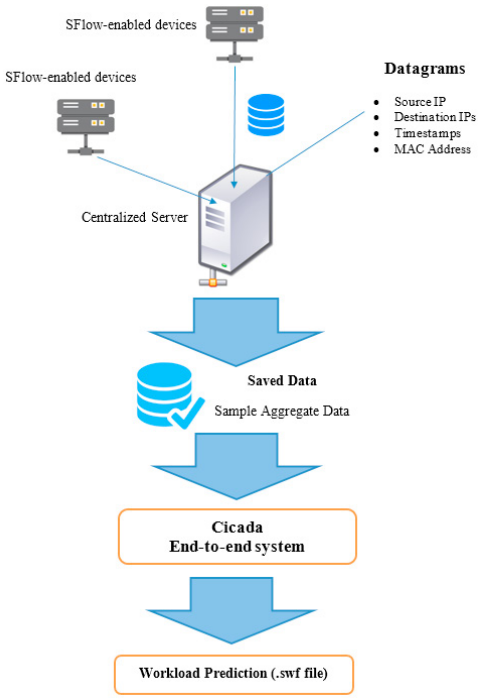

Figure 4: Cicada Data Gathering Diagram

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

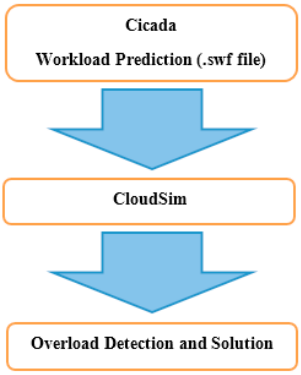

Figure 5: Data Importation from Cicada to CloudSim

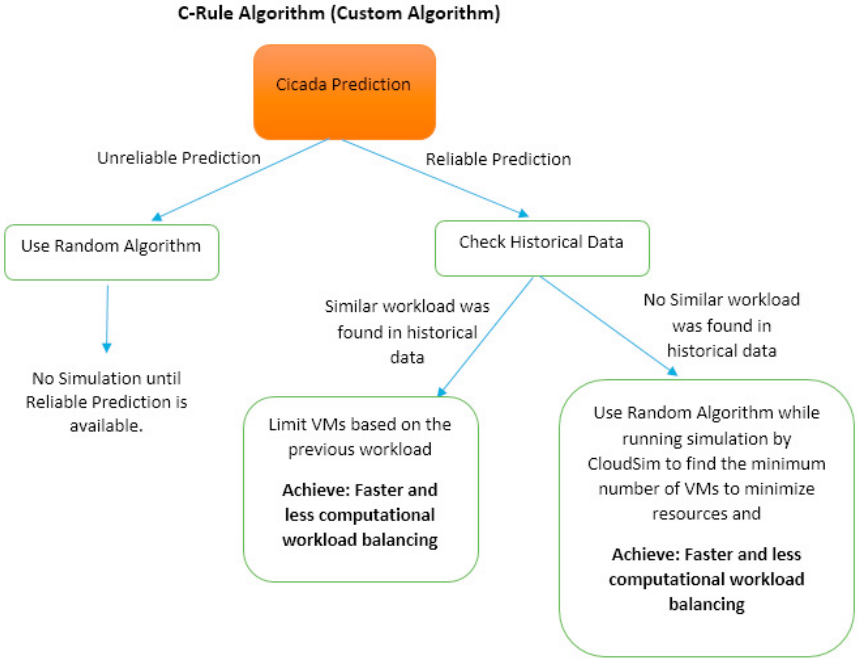

Figure 6: C-Rule Workload balancing diagram

The above diagram demonstrates the overall concept of C-Rule algorithm. 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. The literature in Chapter 2, indicates that Cicada cannot provide any prediction for any burstiness of a workload and the only algorithm that can efficiently handle load balancing of a burstiness workload is the Random algorithm. Random Algorithm connects cloudlets and servers randomly by assigning random numbers to each server. 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.

3. SAMPLE RESULTS

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. Cloud Simulation can also generate a list of CPU wait times for each cloudlet each time that CloudSim adds or removes different cloud resources. All waiting times can be stored as a 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 host to 2 hosts. The sum of the waiting times has 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.

3.1. Simulation load balancing results

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

t-Test: Paired Two Sample for Means

	1 Hosts	*2 Hosts*
Mean	10001.0185	4000.48
Variance	52642374.38	14740394
Observations	20	20
Pearson Correlation	0.944917784	
Hypothesized Mean Difference	0	
df	19	
t Stat	**6.989885375**	
P(T<=t) one-tail	5.85249E-07	
t Critical one-tail	1.729132812	
P(T<=t) two-tail	1.1705E-06	
t Critical two-tail	2.093024054	

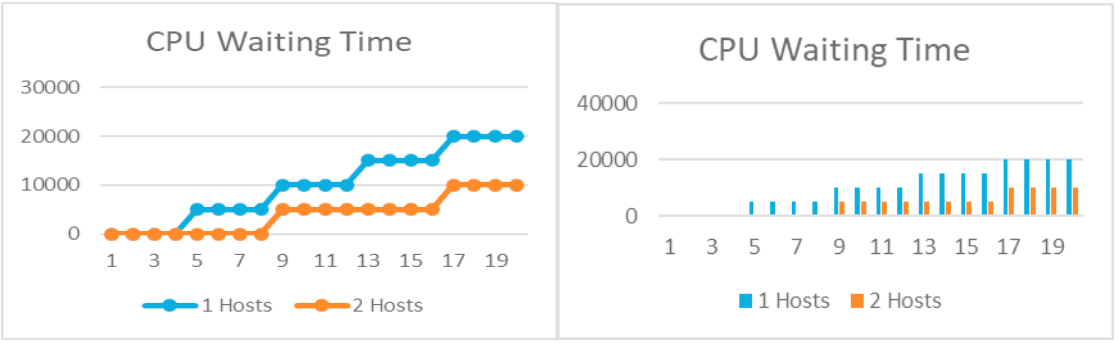

Figure 8: Chart demonstrating effect of adding a new host on total CPU waiting time.

3.2. Resource reduction results by a C-Rule algorithm

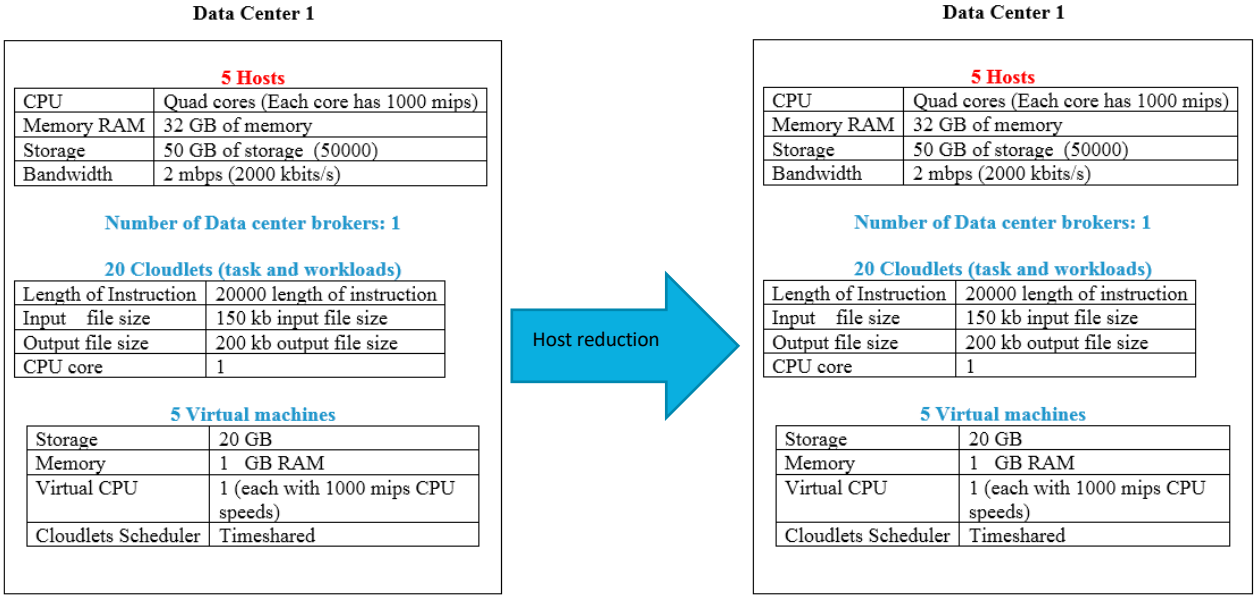

Figure 9: Resource Reduction by C-Rule algorithm

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

Total # of Virtual Machines: Total Number of Host Machines			20 7 Hosts	20 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

Table 2: Final result of resource reduction after achieving a zero processing wait-time.

The above result concludes that each host machine must run 20 VM machine in order to achieve zero waiting time with only 5 host machine rather than 7 host machines. The above simulation was completed in small fractions of a second.

Acknowledgements

The first author did most of the work when he was at Athabasca University. Moreover the first author acknowledges partial support of Discovery NSERC Grant of Canada.

Summary

The objectives of this work are three-fold: to investigate under what conditions to use Cicada for predicting workloads in dynamic Internet hosting platforms, to determine the conditions to use CloudSIM for reliable workload simulation, and to identify the challenges of rule-based algorithm for load balancing for Internet-based platforms compared to Cicada predictions. The methodology envisaged in this work entails three phases: workload prediction using Cicada, simulation using CloudSIM framework, and the development of a space-shared algorithm (C-algorithm) for dynamic workload balancing in cloud environments.

This new approach helped cloud services 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.

The final result proved that prediction workload balancing based on a successful simulation can achieve faster results and can require less computational power.

References

[1] S. Singh and T. Jangwal, "Cost breakdown of Public Cloud Computing and Private Cloud Computing and Security Issues," *International Journal of Computer Science & Information Technology (IJCSIT),* vol. 4 No 2, pp. 17-31, April 2012.

[2] Q. Z. Sheng, X. Qiao, A. V. Vasilakos, C. Szabo, S. Bourne and X. Xu, "Web services composition," *A decade's overview,* vol. 280, pp. 218-238, 2014.

[3] K. L. LaCurts, "Application workload prediction and placement in cloud computing systems," *Massachusetts Institute of Technology,* 2014.

[4] B. Błaszczyszyn, M. Jovanovicy and M. K. Karray, "How user throughput depends on the traffic demand in large cellular networks," *IEEE,* pp. 611 - 619, 2014.

[5] K. Hwang, "Cloud Computing for Machine Learning and Cognitive Applications: A machine Learning approach.," *MIT Press,* p. 624, 2017.

[6] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. D. Rose and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Wiley Online Library,* vol. 41, pp. 23-50, 2011.

[7] J. Zhihua, "GreenCloud for Simulating QoS-based NaaS in Cloud Computing," *Ninth International Conference on Computational Intelligence and Security,* pp. 766 - 770, 2013.

[8] W. Tian, M. Xu, A. Chen, G. Li, X. Wang and Y. Chen, "Open-source simulators for Cloud computing: Comparative study and challenging issues," *Elsevier,* Vols. 58, Part 2, pp. 239-254, 2015.

[9] U. .K.S and P. Chaturvedi, "Predictive Load Balancing Algorithm for Cloud Computing," *IEEE,* pp. 1-5, 2017.

[10] M. Sommer, K. Michael, T. Sven and J. Hähner, "Predictive Load Balancing in Cloud Computing Environments Based on Ensemble Forecasting," *IEEE International Conference on Autonomic Computing,* vol. 11 Ed 4, 2016.

[11] A. Dambreville, J. Tomasik, J. Cohen and F. Dufoulon, "Load Prediction for Energy-Aware Scheduling for Cloud Computing Platforms," *IEEE,* pp. 2604 - 2607, 2017.

[12] N. Pasha, A. Agarwal and R. Rastogi, "Round Robin approach for VM load balancing algorithm in cloud computing environment," *International Journal of Advanced Research in Computer Science and Software Engineering,* vol. 4, no. 5, pp. 34-39, 2014.

[13] S. F. Issawi, A. Al Halees and M. Radi, "An efficient adaptive load-balancing algorithm for cloud computing under bursty workloads.," *Engineering, Technology & Applied Science Research,* vol. 5, no. 3, pp. 795-800, 2015.