MCCVA: A NEW APPROACH USING SVM AND KMEANS FOR LOAD BALANCING ON CLOUD

Hieu Le Ngoc[1, 3], Trang Nguyen Thi Huyen[2], Xuan Phi Nguyen[3] and Cong Hung Tran[3]

[1]Department of Information Technology, Ho Chi Minh City Open University, Vietnam
[2]Sai Gon University, Ho Chi Minh City, Vietnam
[3]Post and Telecommunication Institute of Technology, Ho Chi Minh City, Vietnam

ABSTRACT

Nowadays, the demand of using resources, using services via the intranet system or on the Internet is rapidly growing. The respective problem coming is how to use these resources effectively in terms of time and quality. Therefore, the network QoS and its economy are people concerns, cloud computing was born in an inevitable trend. However, managing resources and scheduling tasks in virtualized data centres on the cloud are challenging tasks. Currently, there are a lot of Load Balancing algorithms applied in clouds and proposed by many authors, scholars, and experts. These existing methods are more about natural and heuristic, but the application of AI, or modern datamining technologies, in load balancing is not too popular due to the different characteristics of cloud. In this paper, we propose an algorithm to reduce the processing time (makespan) on cloud computing, helping the load balancing work more efficiency. Here, we use the SVM algorithm to classify the coming Requests, K - Mean to cluster the VMs in cloud, then the LB will allocate the requests into the VMs in the most reasonable way. In this way, request with the least processing time will be allocated to the VMs with the lowest usage. We name this new proposal as MCCVA - Makespan Classification & Clustering VM Algorithm. We have experimented and evaluated this algorithm in CloudSim, a cloud simulation environment, we obtained better results than some other well-known algorithms. With this MCCVA, we can see the big potential of AI and datamining in Load Balancing, we can further develop LB with AI to achieve better and better results of QoS.

KEYWORDS

Cloud computing, load balancing, processing time, makespan in cloud, MCCVA, SVM, Kmeans

1. INTRODUCTION

Cloud computing [1], [2], [3] is the trend of information technology development, it inherits previous networks and distributed computing concepts, integrate all resources to build up a perfect environment with better service. Cloud provides network, computation, storage or even platforms and many other services on demand with conveniently and quickly. It has been enabling a high demand of service, easiest way to release the resources. It helps to reduce communication between suppliers, providers, and end-users. The users just only pay for the resources used (pay-by-use). One of the most challenging problems in cloud, it is load balancing and improving QoS. To maximize the benefits of cloud computing, load balancing is one of the most important factors that we need attend to. Load balancing has a lot of parameters and many ways/properties that we can modify to have a better performance like throughput, response time, makespan (execution time), bandwidth etc. The load balancing (LB) [4] is aimed to solve the problem of unbalance in cloud, it helps to enhance the services providing for clients. The LB in cloud must faces a lot of difficulties to handle the requests and tasks from users. The main problem of cloud includes : dynamic nature of client request, the complexity of traffic flow of cloud, there is no exact mapper and generator to map a request to a resource rightly, the various of requests and distribution of tasks, etc. Besides these problems, LB also need to adapt the QoS and the green energy requirements.

In this paper, we focus on the execution time of load balancing. Therefore, we came up with the idea of applying some classification and clustering algorithms in AI, to allocate requests based on makespan (execution time of request) into the most reasonable resources, to improve and improve processing / execution time. With this approach, the cloud can have better performance, avoid the load imbalance.

The content of the paper is divided into the following 5 sections: Section I is about the introduction of the approach. Section II would present the basic theoretical basis and Section III shows the related works which have been publicized. Section IV will describe the proposed algorithm, MCCVA. Finally, Section V shows how we experiment and assess results, and then we come out with the conclusion, respectively.

2. LOAD BALANCING IN CLOUD COMPUTING

For better understanding, we should have taken an overview of some background information and its features, including cloud computing, load balancing on cloud, SVM method in AI, and K-Means clustering.

2.1. Cloud Computing

According to the document [5], cloud computing, also known as virtual server computing, is a computing model that we use computer technology and developments based on the Internet. The term "cloud" refers to the Internet (based on how it is arranged in a computer network diagram) and as a reference to the complexity of the infrastructure contained within it. In this computing model, all possibilities related to information technology are provided as "services", allowing users to access technology services from a certain provider "in cloud" without the knowledge and experience of technology, and no need to pay attention to the infrastructure that serves the technology. And the most advantage of cloud is about pay-per-use, this means that users only pay for the service which they have used, and they can increase or decrease the usage of the service on cloud. Service providers divide cloud computing into 3 major service models: IaaS - Infrastructure as a Service, PaaS - Platform as a Service, SaaS - Software as a Service.

2.2. Load Balancing on Cloud

Load balancing [6], [7] is a method of distributing the load on many computers or a cluster of computers to optimize resources, maximize throughput, reduce response time, and avoid server overload or deadlock. The purpose of load balancing is to improve the performance of the whole system basically; to minimize work timeout; to have a backup plan in case the system fails or partially fails; to maintain stability and to adapt the future changes in the system; to ensure small jobs which do not starve for a long time. Load balancing also ensures that one node is heavily loaded while the others are under light load, so a suitable load balancing technique for the system should be selected.

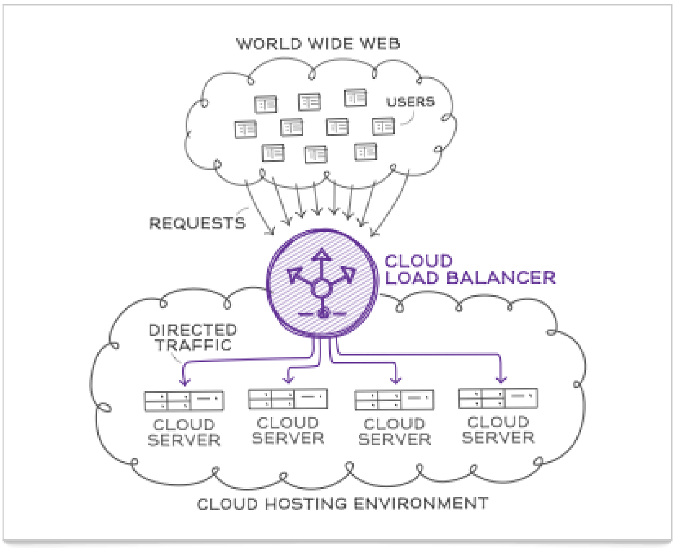

Figure 1. Cloud computing environment and load balancing [8]

Till 2020, there are too many LB Algorithms, according to the article [4], we can see there are more than 100 algorithms have been developed and released. But the existing methods still have a lot of weakness: algorithm complexity is high; low execution time; low resource utilization, High SLV and task rejection ratio, etc. With each method has each owns advanced features and disadvantages, meanwhile the cloud has many features need to be enhanced. There is no algorithm or approach that can fully solve all parameters of LB in cloud. According to this article [4], the authors have carried out the statistic of LB parameters as below:

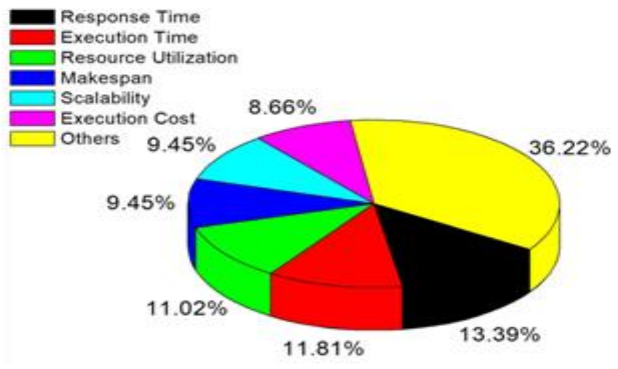

Figure 2. Percentage of LB metrics in Existing approaches [4]

2.3. Support Vector Machine Algorithm – SVM

SVM [9] is considered a powerful tool for non-linear classification problems developed by Vapnik and Chervonenkis in 1995. This method is based on the principle of SRM (Structural Risk Minimization), which is considered one of the most sophisticated non-parametric monitoring methods to date. SVM's diverse tool functions allow you to create transformable spaces to build layered planes. SVM uses tricks to map the original data set into more dimensional space. Once mapped to multi-dimensional space, SVM will consider and select the most appropriate super planar to classify that dataset. With the characteristics of SVM in classification, SVM becomes a very potential candidate for selecting / filtering / classifying the requests of cloud, so we can have a right allocation, respectively.

With the flexibility of SVM in a limited dataset and its performance, we can apply SVM to classify the request based on the history of requests. We can use the big enough dataset of previous requests to build the training set for the next coming request. With the advantage of SVM is fast performance in small set, we can classify more quickly to perform LB.

2.4 K-Means Algorithm

K-Means [10],[11] is an especially important algorithm and is commonly used in clustering techniques these days. The main idea of the K-Means algorithm is to find a way to group the objects (objects) into K clusters (K is the number of predefined clusters, K positive integers) such that the sum of squares of distances between objects. The centroid is the smallest. With K-means features, we can see the value of clustering for the hosts / VMs / Nodes in cloud environment. With the signification of clustering, we can reduce the wasteful work in allocating resources, and have the right choice for the right resource

3. Related Works

To conduct the study purpose of this paper, after understanding the background situation, we should review and see more about the status of Load Balancing using these kinds of AI methods.
In 2016, the article *"Minimum makespan task scheduling algorithm in cloud computing"* [12], N. Sasikaladevi of India's SASTRA University studied the minimum task scheduling algorithm in cloud computing. The authors pointed out that cloud computing provides a robust and on-demand business environment. They proposed a task scheduling mechanism with minimum processing time (makepan) named MMSF and the algorithm to make minimum task tasks named MMA with the main goal of minimizing the total number of makepan and Maximize the use of virtual machines. Experimental results show that MMA outperforms traditional task scheduling algorithms based on the makespan and total virtual machine usage. With this proposal, the MMSF is more about classical heuristic solution, it is not a modern AI method, but the results come better then the original LB.

In the year 2017, the paper *"Time Efficient Dynamic Threshold-based load balancing technique for cloud computing"* [13], the authors proposed minimizing system optimization, maximizing resource use, and reducing consumption of general energy. This approach has reduced waiting time compared to existing methods and makespan optimization of cloud resources. In the paper, the main goal is to minimize makepan - processing time on virtual machines. After the experiment of the paper, the proposed DLBA algorithm is a natural dynamic load balancing method, considering the current responses and its variables to decide the allocation of new requirements. However, the algorithm will be ineffective if the thresholds are incorrect or inappropriate. This paper shows us that if the thresholds are more correct, the Load Balancing will have better performance.

Also in the same year, there was an article *"An Efficient Load Balancing Scheme for Cloud Computing"* [14], from Atyaf Dhari and Khaldun I.Arif of India. They proposed the Load Balancing Decision Algorithm (LBDA) algorithm to manage. and load balancing between virtual machines in the data centre with reduced completion time (Makespan) and response time. The authors compared the proposed algorithm with the Round Robin, Max-Min and SJF algorithms in the same configuration environment and same input. The simulation results show that the proposed algorithm is superior to all cases compared to Max-Min, SJF and Round Robin algorithms by reducing the average value, average response time and total execution time in all VMs.

A new study was published in the same year, *"A Load Balancing Game Approach for VM Provision Cloud Computing Based on Ant Colony Optimization"* [15]. This study is from the author Tran Cong Hung and his colleagues, which is about load balancing based on game theory using for virtual machines on the cloud. The cloud here is based on the Ant Colony Optimization algorithm. In this study, these authors proposed a solution for virtual machines to ensure a balance of goals that users need to include service providers and their customers based on the role play theory. The key idea was to use the meta-heuristic algorithm Ant Colony Optimization (ACO) based on Nash equilibrium. The experimental results of this study showed a very bright sign of Load Balancing, it helped the cloud running more efficiency.

After one year, in 2018, the article *"A Modern Approach for Load Balancing Using Forest Optimization Algorithm"* [16] has been public. The authors presented a modern approach to load balancing by tree optimization algorithm. This research proposes a forest optimization algorithm to balance the load in a distributed computing structure. This depends on the behaviour of the trees in the forest and the use of seed dispersion strategies to streamline the makespan (the time spent on project implementation), help improve average response time and overall execution time. The authors performed a comparative analysis of evolution algorithms such as GA and PSO with Forest optimization algorithm (FOA) for load balancing. The techniques used by the trees and trees are used to find global and local optimal functions. The simulation results prove that the proposed algorithm gives better results than its partner load balancing algorithm.

Also in 2018, with the article *"Proposed Load Balancing Algorithm To Reduce Response Time And Processing Time On Cloud Computing"* [17], Tran Cong Hung and partners proposed Throttled Modified Algorithm algorithm to optimize response time on the Virtual machine in cloud computing and Makespan optimization of cloud data centre. This algorithm will help detect virtual servers are available with the table size "Available Index" changed more flexibly than the Throttled algorithm. This helps to increase processing performance for the system. The testing and experimental results are quite good comparing to others well-known algorithms like Throttle, Round Robin and MaxMin. But the testing data broad is not so wide, we can open more cases and more request to test.

In 2018, Geeta et had focused more on QoS of LB in cloud [18], the paper reviewed all existing algorithms of LB, otherwise proposed the important QoSs for LB in cloud. The cloud QoS has some characteristics: Self Service on demand, Location independence, Broad network access, Rapid elasticity, Measuring service. And the coming challenges: Security and Privacy, performance, Efficient Load Balancing, Resource Management and Scheduling, Require a constant and Fast Internet speed, Data center Energy Consumption, Scale and Quality of Service Management.

In 2018, Sambit et had showed a bigger picture of LB in cloud [19], described more detail and added a lot of specific features of LB in cloud. This study reviewed better and clearer, depicted very detail the model of LB, especially is the performance matrices that effects the load balancing. The authors had done a statistic review with a better view, besides once again stated out the important parameters of LB in cloud: Throughput (TP), Thrashing (TH), Reliability (R), Accuracy (A), Predictability (PR), Makespan (MS), Scalability (S), Fault Tolerance (FT), Associate Overhead (AO), Migration Time (MT), Response Time (RT), Associated Cost (AC), Energy Consumption (EC).

In 2019, Kothapuli Venkata Subba Reddy and partners [20] have proposed "A Dynamic Hierarchical Load Balancing Service Architecture for Cloud Data Centre Virtual Machine Migration". With this article, the authors focused on the bottlenecks of the architecture to improve the performance of LB. They enhanced the optimization settings of the components in the architecture of the cloud, proposed a dynamic and hierarchical hybrid data centre, a performance comparison matrix to evaluate the existing data centre architectures and hybrid architectures. This is a good optimization for LB and it can be a good model for enhancing the LB algorithms.

4. PROPOSAL WORK: MCCVA

With the idea of applying some classification and clustering algorithms in AI, to allocate requests into resources in the most reasonable way, we can help improve the processing time / task execution time, completion time, component and resource of the virtual machine, optimize the load balance in cloud computing environment. We would like to propose a new algorithm named MCCVA- Makespan Classification and Clustering VMs Algorithm.

4.1. Research Model

The research model aims to use SVM (Support Vector Machine) classification algorithm to classify requests based on previous processing time. In parallel, the algorithm uses clustering (here we use K-means) to cluster virtual machines (hosts), into groups with high, medium, and low performance levels. From there, the allocation of requests with high processing demand into the virtual machine has the lowest level of activity. With this approach, the proposed algorithm will improve the load processing time on cloud, and applications in cloud environment in real time. In this paper, we temporarily name the algorithm as MCCVA (Makespan Classification & Clustering VM Algorithm).

In this model, we use SVM (mainly based on processing time) to classify input requests, based on this algorithm, and know which virtual machine this request will allocate to which virtual machine group. To cluster virtual machines, here use K-Mean to divide them into 3 clusters (with k = 3). For machines in cluster 3 (the cluster with the freest resources), the priority will be to handle complex requests, the longest processing time.

This model is a natural simulation of the Max-Min algorithm and plans for subsequent requests to avoid load imbalance. But it is an enhancement and a better improvement of MaxMin with AI methods. According to this algorithm will reduce the communication load between the virtual machine and existing resources, thus reducing unnecessary bandwidth and throughput, increasing service requirements for users.

4.2. Makespan Classification & Clustering VM Algorithm – MCCVA

The proposal algorithm consists of three main modules: Module 1 - classifying requests using SVM algorithm; Module 2 - clustering virtual machines / hosts / resources; Module 3 - allocating services (distribute the right request to the right virtual machine respectively).

(1) Module of classifying requests using SVM algorithm

In this module, the SVM algorithm will rely on the properties of the request that calculate the processing time of that request, thereby classifying this request. Attributes include size, Response length, Max Length, ...

$$\textit{Processing Time Group} = \textit{MKNew} = \textit{SVM}\ (X_1, X_2, ..., X_n)$$

In which X_i is the attribute of the Request when sending to the cloud.

It can be divided into several groups (from 4 ~ 10 groups) or more based on the variability of the Request. The SVM will use the latest requests info (around 1 minute to 10 minutes) to collect the data for training. This training dataset are always updated and changing by the time.

(2) Module of clustering virtual machines / hosts / resources

In this module, we will use the clustering algorithm k-Means (with k = 3) to cluster virtual machines based on activity level, using virtual machine resources, including high, medium, and low clusters. This clustering of virtual machines is based on instantaneous parameters of virtual machines.

Cluster i = kMeans(cpu usage, ram, ...);

In which: i = 1 is the low group
i = 2 is the average group
i = 3 is the high group

In this proposal, we selected k =3, means we have 03 clusters. However, if the cloud is more complicated and bigger enough, we can extend k to 4 or 5 even to 100. The number of k is a further study of this paper. To achieve the more accurate outcome, we use the same way to select training dataset, we take latest VMs info as training dataset.

(3) Service allocation module (select virtual machine)

This module is responsible for allocating requests to virtual machines through the appropriate type of virtual machine requests and assemblies. If a request is sent, this request is classified by module 1, and the VMs that are considering including the unloaded VM are also clustered according to module 2. The algorithm will then calculate which request matches that. Which virtual machine is most likely passed by the return parameters of the SVM and K-Means functions above. If the request's processing time is calculated (calculated from module 1), this request will be processed on the VM with the furthest means (i.e., group 1, and the usage level is low) or the shortest. For small and large requests, we can use calculation methods such as deduction or variance to calculate allocations.

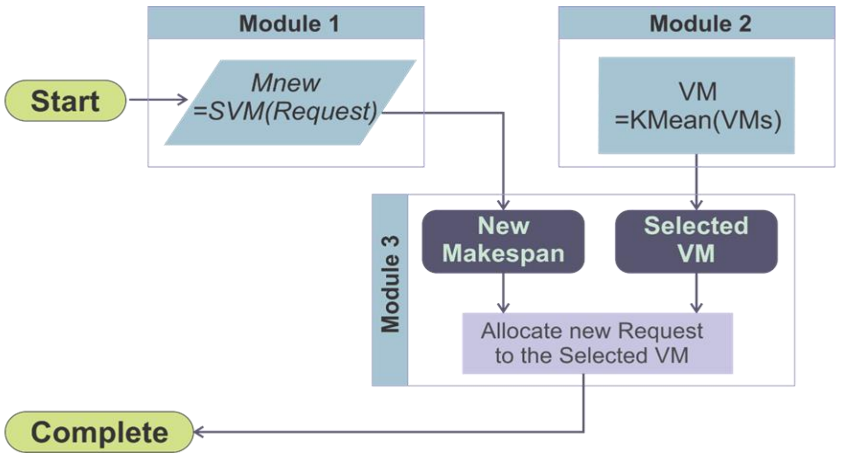

Figure 3. Process flow MCCVA algorithm

Pseudocode of the MCCVA algorithm

```
1. For each Request in CloudRequests
2.     isLocated = true;
3.     MakeSpan_Classnew = SVM (T1, T2 ... ..); // Module 1
4.     VM_Cluster = kMeans (situation); // situation: The status of the VM Module 2
5.     For each VM in VMList
6.         If isFitSituation (Request.MakeSpan_Class, VM.VM_Cluster)
7.             AllocateRequestToVM (VM, Request); // Module 3
8.             isLocated = true;
9.         End If
10.    End For
11.    If (! IsLocated)
12.        VM = VMList.getMinFromMean (); // Module 2
13.        AllocateRequestToVM (VM, Request);
14.    End If
15. End For
```

4.3. Experiment results

To simulate cloud environment, we use CloudSim library and programming in JAVA language. We developed this proposal in JAVA with APACHE NETBEAN IDE. The cloud simulation environment is from 5 to 15 virtual machines and creates a random request environment for these cloud services. Includes cloud provisioning service, CloudSim provisioning and user service for testing.

For the AI methods, we use Weka library to install SVM and K-Means algorithms on the simulation environment and test the results.

Parameters of simulation network model

Table 1. Configuration figures of data centre.

Datacentre Info	Host info (in the datacentre)
- Number of hosts in datacenter: 5 - No Storage (Using SAN for storage) - Architecture: x86 - OS: Linux - VMM: Xen - Time Zone: +7 GMT - Cost: 3.0 - Cost per Memory: 0.05 - Cost per Storage: 0.1 - Cost per Bandwidth: 0.1	With a host in Datacenter, the info is as below: - CPU with 4 cores, each core is 1000 (mips) - Ram: 16384 (MB) - Storage: 1000000 - Bandwidth: 10000

Table 1. Configuration of a VM

Size	Ram	Mips	Bandwidth	Pes no.	VMM
10000 MB	512 MB	250	1000	1	Xen

Table 2. Request information

Request Length (Byte)	File Size (Byte)	Output Size (KB)	PEs
3000 ~ 1700	5000 ~ 45000	450 ~ 750	1

Evaluation criteria

Experiment cloud simulation with the above parameters, run CloudSim's load balancing algorithm, and run the newly installed proposed algorithm. With the same input and same environment, we compare the output, especially the parameters processing time (Makespan). The lower the processing time of virtual machines, as well as the processing time of the cloud, the better the algorithm's effectiveness.

To have a better understand, we select the tradition algorithms and the currently using algorithms in many of current system, to compare with our proposal works. They are MaxMin algorithm, MinMin algorithm, Round-Robin algorithm, and the nature algorithm FCFS [4]. Maxmin is a static algorithm, it is based on task scheduling, it has low overhead, less makespan, high resource utilization. MinMin is also static, and it is based on task or resource scheduling, it has low makespan, low response time but hight resource. Round-Robin is a heuristic alogorithm, it is dynamic and based on task scheduling, it also has low makespan, low power consumption and low SLV. FCFS is as nature algorithm, it just handle the tasks as natural, what comes first will handle first.

Simulation and experimental results

The result of running simulation experiments on CloudSim with 5 virtual machines built to meet the requirements, the requests are initialized with random length and size, the number of requests is 20-50 and compared with the algorithms Round-Robin, MaxMin, MinMin and FCFS:

Table 3. Experiment result (in millisecond) with simulation of 50 Requests

Request No.	Round-Robin	MaxMin	MinMin	FCFS	MCCVA
20	**73.48**	112.81	166.77	127.95	123.98
25	**86.5**	129.1	186.93	148.7	192.54
30	**112.73**	145.79	217.82	164.41	140.4
35	**129.81**	170.22	244.26	189.48	156.35
40	**150.1**	198.91	282.34	230.53	176.63
45	**175.15**	212.54	300.53	256.17	195.01
50	231.83	229.8	334.34	316.47	**213.57**

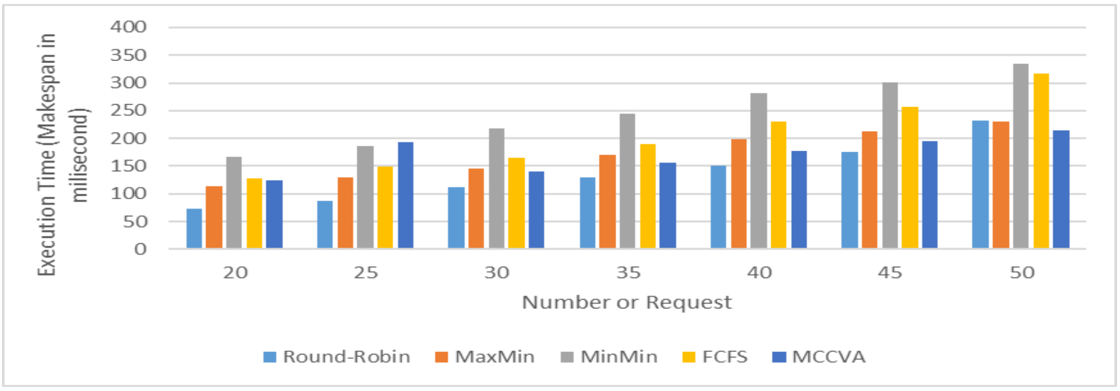

Figure 1. The comparing the execution time of 5 algorithms with 50 Requests

With the experimental results with 50 Requests, we find that the Round-Robin algorithm is dominant and fast processing, MaxMin algorithm is also quite stable. FCFS algorithm is not yet strong. However, the proposed algorithm MCCVA is also quite stable.

Results of running simulation experiments on CloudSim with 5 virtual machines built to meet the requirements, the requests are initialized with random length and size, the number of requests respectively is 100 to 1000:

Table 4. Experiment result (in millisecond) with simulation of 1000 Requests

Request No.	**Round-Robin**	**MaxMin**	**MinMin**	**FCFS**	**MCCVA**
100	**280.46**	529.01	564.25	915.94	359.47
200	**685.99**	995.76	923.2	1746.95	693.87
300	**936.88**	1402.47	1276.19	2450.61	972.44
400	**1175.86**	1806.76	1625.49	3148.05	1253.25
500	**1419.01**	2205.04	1978.6	3856.57	1531.72
600	**1659.67**	2605.43	2323.7	4565.09	1810.83
700	1898.65	3004.55	2673.20	5258.63	**1444.51**
800	2848.74	3400.15	3184.27	5967.59	**2367.79**
900	3097.55	3978.18	3521.57	8930.12	**2802.13**
1000	4114.58	4579.85	3899.78	12682.62	**3282.37**

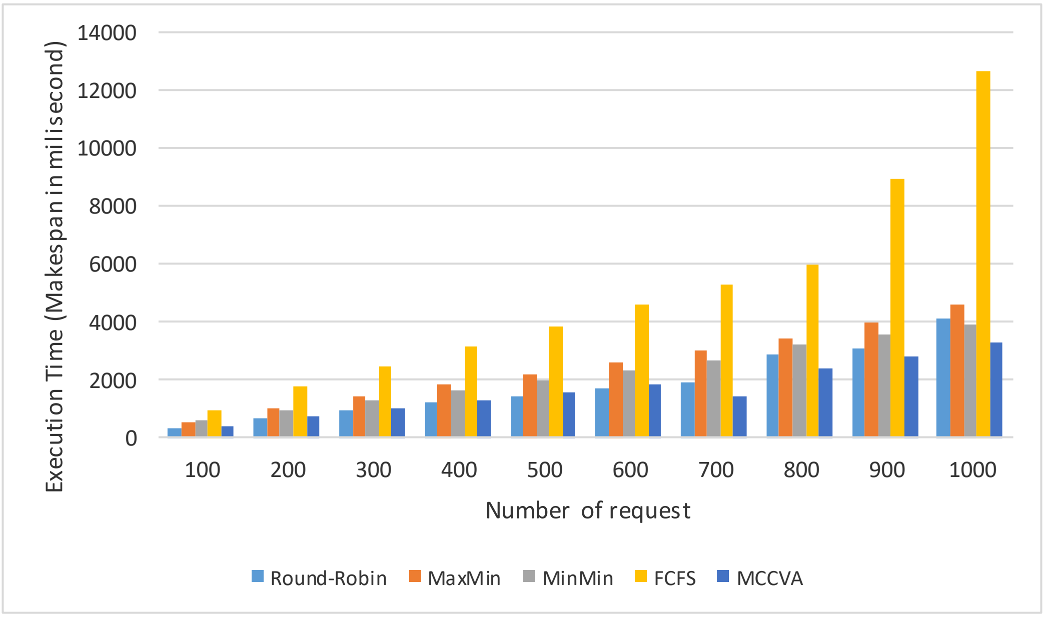

Figure 2. The comparing the execution time of 5 algorithms with 1000 Requests

From the 100th request onwards, MCCVA algorithm is superior to MaxMin, MinMin. However, the advantage over RoundRobin has not been found. But the larger the number of requests is, the more advantage MCCVA is and it gradually absolute dominance over the remaining algorithms. Clearly FCFS demonstrates the lack of intelligence and the nature of the algorithm.

Through the two charts comparing the processing time of the algorithms with the same input conditions, we can see the allocation of the proposed algorithm MCCVA is quite stable and reasonable. With proposal MCCVA, the processing time of virtual machines are not too different from processing time of other algorithms on cloud (in case of few and many requests).

5. Conclusion

A new algorithm for load balancing on cloud environment using classifying requests with processing time has been proposed and experimented in simulating models with small scale. Based on previous ideas and studies, the data mining application algorithm is the SVM algorithm to balance the load based on processing time. In particular, the more accurate the processing time is, the higher efficiency is. However, the more accurate calculation requires more memory and processor. But the user in the cloud environment has extremely diverse and rich requests, so the processing time also varies immensely on the cloud. The algorithm proposed in this paper approaches a generalized and promoted idea of classification according to regression and mathematics, using processing time, typically SVM algorithm. Therefore, the proposed algorithm has taken a fairly new approach in balancing the load in the cloud environment, while achieving some positive experimental experimental results, showing the good development direction of the algorithm and further works of enhancement.

The development direction of the proposed algorithm MCCVA can be more accurate measurement and calibration when combining SVM with machine learning, unsupervised or supervised learning by giving off peak or low cloud times. To develop better and deeper algorithms, it is necessary to experiment on computer simulation with a more powerful configuration and larger scale simulations. In addition, the implementation of this MCCVA on

CLOUD will help us to research more in-depth and specific, because the actual cloud environment will generate more problems related to processing time, thereby we can have a better modification for more reasonable and more effective.

With the try of SVM and Kmeans in LB, we can furthermore develop LB with many many other parameters of LB, QoS of cloud. This shows us a very Brightside that we can apply AI, datamining to LB to solve the challenges of LB in cloud.

ACKNOWLEDGEMENTS

Sincerely thank the Saigon University and Post & Telecommunication Institute of Technology which facilitates the provision of figures for this research paper. We also show a grate appreciation to the teachers and the staffs of the 02 schools, have communicated the knowledge and valuable experience throughout the research process.

REFERENCES

[1] Y. F. Wen and C. L. Chang, "Load balancing job assignment for cluster-based cloud computing", Sixth International Conference on Ubiquitous and Future Networks (ICUFN), Shanghai, 2014, pp. 199-204.

[2] G. Shao and J. Chen, "A Load Balancing Strategy Based on Data Correlation in Cloud Computing", 2016, IEEE/ACM 9th International Conference on Utility and Cloud Computing (UCC), Shanghai, 2016, pp. 364-368.

[3] Ho Dac Hung, Tran Cong Hung, Bui Thanh Khiet, Nguyen Thi Nguyet Que, Pham Tran Vu, "A Fair VM Allocation for Cloud Computing based on Game Theory", Proceedings of the 10th National Conference on Fundamental and Applied Information Technology Research (FAIR'10), Da Nang, Viet Nam, 17-18/08/2017.

[4] Shahbaz Afzal, G. Kavitha, "Load balancing in cloud computing - A hierachical taxonomical classification", Journal of Cloud Computing: Advances, Systems and Applications, (2019) 8:22, https://doi.org/10.1186/s13677-019-0146-7.

[5] Tran Cong Hung, Nguyen Xuan Phi, Le Ngoc Hieu, "A Response-Time Based Algorithm of Workload balancing in Cloud Computing", Issue 4 (CS.01) 2018 Journal of information and communication technology.

[6] Rajwinder Kaur & Pawan Luthra, "Load Balancing in Cloud Computing", Recent Trends in Information, Telecommunication and Computing, Association of Computer Electronics and Electrical Engineers 2014, page 374-381.

[7] Md. Firoj Ali, Rafiqul Zaman Khan, "The study on Load Balancing strategies in distributed computing system", International Journal of Computer Science & Engineering Survey (IJCSES), Vol.3, No.2, April 2012, page 21, 22, 23.

[8] Bhatia, Jitendra & Kumhar, Malaram, Perspective Study on Load Balancing Paradigms in Cloud Computing, Vol- 6 • Issue - 1 Sep - Mar 2015pp.112-120 Impact Factor -2.5 available at www.csjournalss.com page 115

[9] Bo Pang and Lillian Lee và Shivakumar Vaithyanathan, Thumbs up Sentiment Classification using Machine Learning Techniques. Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)

[10] Research on k-means Clustering Algorithm: An Improved k-means Clustering Algorithm, Shi Na, Liu Xumin, Guan Yong - Coll. of Inf. Eng., Capital Normal Univ. CNU, Beijing, China, 2010, IEEE - ISBN - 978-1-4244-6730-3

[11] Kmeans Algoritm, 01/04/2017, Retrieved on 21/03/2020 from https://machinelearningcoban.com/2017/01/04/kmeans2/

[12] N. Sasikaladevi – School of Computing, SASTRA University, India sasikalade@gmail.com, “Minimum makespan task scheduling algorithm in cloud computing”, International Journal of Advances in Intelligent Informatics ISSN: 2442-6571 Vol. 2, No. 3, November 2016, pp. 123-130.

[13] Sambit Kumar Mishra, Md Akram Khan, Bibhudatta Sahoo, Deepak Puthal and Mohammad S. Obaidat, Fellow of IEEE, and KF Hsiao, “Time Efficient Dynamic Threshold-based load balancing technique for cloud computing”, 978-1-5090-5957-7/17/$31.00 ©2017 IEEE.

[14] Atyaf Dhari, Khaldun I.Arif , “An Efficient Load Balancing Scheme for Cloud Computing”, Arif Indian Journal of Science and Technology, Vol 10(11), DOI: 10.17485/ijst/2017/v10i11/110107, March 2017, ISSN (Print) : 0974-6846 ISSN (Online) : 0974-5645

[15] Subasish Mohapatra , Ishan Aryendu, Anshuman Panda, Aswini Kumar Padhi, “A Modern Approach For Load Balancing Using Forest Optimization Algorithm”, Proceedings of the Second International Conference on Computing Methodologies and Communication (ICCMC 2018) IEEE Conference Record # 42656; IEEE Xplore ISBN:978-1-5386-3452-3.

[16] Khiet Thanh Bui, Tran Vu Pham, & Hung Cong Tran, “A Load Balancing Game Approach for VM Provision Cloud Computing Based on Ant Colony Optimization”, Context-Aware Systems and Applications International Conference, ICCASA 2016 Thu Dau Mot, Vietnam, pages 52-63.

[17] Nguyen Xuan Phi, Cao Trung Tin, Luu Nguyen Ky Thu, Tran Cong Hung, “Proposed Load Balancing Algorithm To Reduce Response Time And Processing Time On Cloud Computing”, International Journal of Computer Networks & Communications (IJCNC) Vol.10, No.3, May 2018.

[18] Geeta and Shiva Prakash, "A Literature Review of QoS with Load Balancing in Cloud Computing Environment”. (2018).

[19] Sambit Kumar Mishra, Bibhudatta Sahoo, Priti Paramita Parida, "Load balancing in cloud computing: A big picture", Journal of King Saud University – Computer and Information Sciences, Journal of King Saud University – Computer and Information Sciences xxx (2018).

[20] Kothapuli Venkata Subba Reddy, Jagirdar Srinivas and Ahmed Abdul Moiz Qyser, "A Dynamic Hierarchical Load Balancing Service Architecture for Cloud Data Centre Virtual Machine Migration", Smart Intelligent Computing and Applications, Smart Innovation, Systems and Technologies 104, Springer Nature Singapore Pte Ltd. 2019, https://doi.org/10.1007/978-981-13-1921-1_2

AUTHORS

Tran Cong Hung was born in Vietnam in 1961. He received the B.E in electronic and Tel ecommunication engineering with first class honours from HOCHIMINH University of technology in Vietnam, 1987. He received the B.E in informatics and computer engineering from HOCHIMINH University of technology in Vietnam, 1995. He received the Master of Engineering degree in telecommunications engineering course from postgraduate department Hanoi University of technology in Vietnam, 1998. He received PhD. at Hanoi.

Le Ngoc Hieu has been working in IT industry as a IT System Architect since 2010. He is now doctorate student at Post & Telecommunication Institute of Technology. As now, he is working as an IT lecturer for HCMC Open University. His major study is about cloud computing and cloud efficiency for better service; his minor study is about education & economic, especially education in IT line.

Nguyen Thi Huyen Trang was born in Viet Nam in 1987. She received a bachelor's degree in 2011, majoring in computer science at Ho Chi Minh City University of Education, Vietnam. Currently, she is a Master's candidate in Computer Science from Saigon University, Vietnam. She is a Computer teacher at Quang Trung High School, Binh Thuan.

Nguyen Xuan Phi was born in Vietnam in 1980. He received Master in Posts & Telecommunications Institute of Technology in Ho Chi Minh, Vietnam, 2012, major in Networking and Data Transmission. Currently he is a PhD candidate in Information System from Post & Telecommunications Institute of Technology, Vietnam. He is working at the Information Technology Center of AGRIBANK in Ho Chi Minh city, Vietnam. His main research areas are load balancing on cloud computing, optimizing the performance of cloud computing.