A Principal Component Analysis and Clustering based Load Balancing Strategy for Cloud Computing

Law Siew Xue, NazatulAini Abd Majid, Elankovan A. Sundararajan

Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM) Bangi, Selangor Darul Ehsan Malaysia

Abstract **–The objective of the research is to develop a model based on Principal Component Analysis and clustering for batch-processing load balancing in the cloud computing environment. The findings show that the model is able to extract the current computing resources of the physical hosts and cluster the hosts based on their similarity features. The computing resources of the virtual machine for a new requested task is then extracted and matched with the hosts clusters to select the most suitable physical hosts based on the computing resources. Conversely, when compared with the simulation results of the Round Robin and First Come First Serve load balancing models, the proposed method shows a decrease failure rate of task deployment events.**

Keywords **– Load balancing, Principal Component Analysis, Clustering, Cloud Computing, Task Allocation, Physical host pool.**

1. Introduction

The load balancing issue is important for an optimal use of computing resources in cloud computing environments [1].

DOI: 10.18421/TEM91-14
https://dx.doi.org/10.18421/TEM91-14

Corresponding author: NazatulAini Abd Majid,
Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM) Bangi, Selangor Darul Ehsan Malaysia.
Email: nazatulaini@ukm.edu.my

Received: 04 July 2019.
Revised: 21 January 2020.
Accepted: 26 January 2020.
Published: 28 February 2020.

This is a method used for distributing workloads over multiple computers, or other resources to achieve maximum resource utilization, minimize response time, prevent overloading as well as optimizing processing power [2]. There are several researches that have proposed different load balancing models to solve the problem related to the high amount of load as well as carbon emissions. However, al lot of these studies mainly focused on how to attain a load balancing policy with single-processing, which have to do with solving one request at a time. As such, it can lead to a decrease in load efficiency and cause a longer waiting period [3].

Load imbalance in a cloud data centre happens mainly because of ineffective strategy in allocating resources. This is because cloud environment resources can be allocated based on the demand or requests from clients [4], [5], [6]. Thus, during the time a request task is submitted by a user, the allocation of the task is done by the system in the physical host of the cloud data centre. Normally, the cloud data centre randomly selects a physical host to perform a task. But when the number of resources for the task request is higher than the amount of resources readily available in physical hosts, the task would not be able to be carried out by the physical host. Conversely, if the requested number of resources is closer to the amount of available physical hosts, therefore the physical hosts will possess a heavy workload, resulting in a service capacity decline as well as reduction in computing power. This decline or reduction leads to a load imbalance in the cloud data centre as well as service efficiency reduction.

A lot of researches have been carried out towards improving the strategy of load balancing such as Round Robin [7], short job scheduling [8], max-min and min-min Algorithm [9], power aware load balancing [10] fuzzy logic [11], active clustering algorithm [12], honey bee behavior inspired load balancing [13], genetic algorithm [14],[15], stochastic hill climbing [16] progress share-based job scheduling algorithm [17] and hybrid scheduling

algorithm [18]. These load balancing strategies provide short-term allocation that may cause a longer waiting period for every request made by a user. This is due to the fact that every physical host requires to be considered for the allocation process. [3] has considered the problem of short-term allocation for load balancing associated to cloud data centres, in which a set of physical hosts was chosen to maximize load balancing through the use of the Bayes and clustering method. However, there is a need to investigate more potential techniques to provide long-term allocation strategy, like using more than two variables of the physical hosts. This is because the variability of the cloud computing environment is dynamically increasing as stated by [17], in order for PCA to be one of the solutions for datasets that have heterogeneous variables [19].

Therefore, the objectives of this study are to: (i) develop a load balancing model based on PCA and clustering for heterogeneous cloud computing, (ii) develop CloudSim simulations for load balancing model, and (iii) evaluate the proposed model with parameters for load balancing, such as resource utilization, completion time (makespan), throughput, waiting as well as response time, and thereafter compare them with the Round Robin and the First Come First Serve (FCFS) technique. The aim of the present research is to show that the proposed model which is based on multivariate statistical technique is able to extract the current computing resources of the physical hosts and cluster the hosts based on their similarity features.

In terms of the organisation of the present study, the related researches are presented in Section 2, while the proposed PCA and clustering model are shown in Section 3. A detailed description is shown with the use of simulation for the proposed model in Section 4. The result, as well as discussions are presented in Section 5 and Section 6 finally presents the conclusion.

2. Method

The main idea of the proposed load balancing model (PCAC-LB) is shown in Figure 1. and described as follows; the PCA technique reduces the variation dimensions of the physical hosts in Infrastructure as a Service (IaaS) by obtaining the main components which have the most variation from the collected data set. The main components that form a PCA model are then used to transform the variability of each physical host as a score. Furthermore, the K-means clustering technique divides the scores into a number of clusters based on their similarities. Thereafter, the PCA model with three distinct clusters (minimum resources, medium resources and optimum resources) are then used for the next allocated task, which is load balancing. After the arrival of the new task request, it is placed in a selected VM. As such, computing sources from the VM are first extracted using the PCA model and matched to the clusters of physical hosts to choose the most suitable physical host in order to perform the requested tasks.

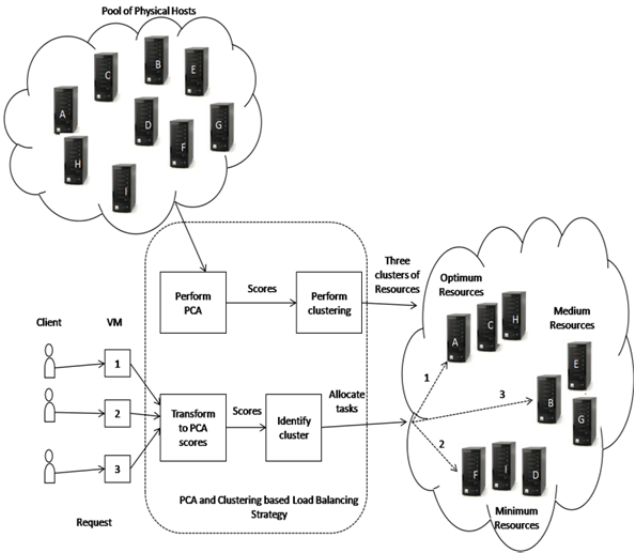

Figure 1. An overview of load balancing strategy using PCA and clustering

The model of the load balancing using PCA and clustering as shown in Figure 2. is divided into three phases: (i) data training, (ii) developing reference model, and (iii) load balancing. The details of the framework are given in Sections 3.1, 3.2 and 3.3, respectively.

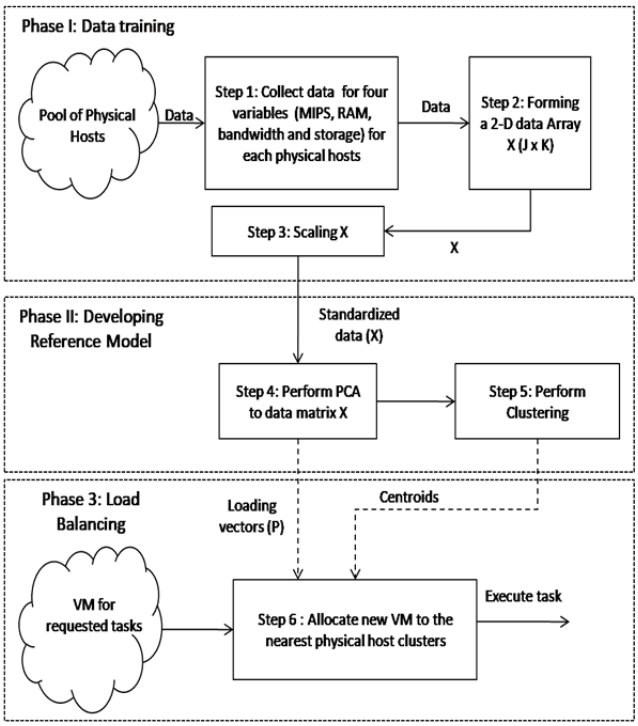

Figure 2. Model of the load balancing using PCA and clustering

Phase I. Data Training

In order to prepare a data, set for data training, three steps were involved: (i) collecting data for four variables, (ii) forming a data array, and (iii) scaling the data. Each step is described below:

1) Step 1: Collecting data for four variables (MIPS, RAM, bandwidth and storage) for each physical host
In a cloud data centre, there are a large number of physical hosts that can be measured based on certain variables, such as MIPS, RAM, bandwidth and storage. With regards to data training, the data sets that are related to minimum, medium and optimum physical hosts are collected from a data centre.
2) Step 2: Forming a 2-D data array X (J x K)
The data sets were rearranged in the form of a 2-D data array X (J=variables, K=data) as shown in Table I. Each column represents a variable: J(1)= MIPS, J(2): RAM, J(3)= Storage and J(4)=Bandwidth. Each row in the data matrix represents a specific host in the cloud data centre.

Table 1. Data for physical hosts

Host ID	J(1): MIPS	J(2): RAM (MB)	J(3): Storage (MB)	J(4): Bandwidth (Mbps)
0	1023	512	5000	5000
1	2048	1800	8000	8000
2	250	124	1000	1000
3	2048	1600	7000	7000
4	2600	1240	6000	6000
5	2500	1530	5500	5800
6	3300	2500	6000	6000
7	1200	980	6000	6000
8	2272	1792	8482	7392
9	3288	2048	9500	9215
10	780	850	1300	1200
11	2000	1300	6000	7500
12	2900	1850	3500	8000
13	1738	1524	5781	9200
14	1900	1358	7200	4402
15	600	2000	2400	3000
16	2500	1000	2679	2815
17	1312	1024	8000	2250
18	900	952	3142	8473
19	1000	2048	4336	4704

3) Step 3: Scaling X
The data matrix X was then pre-processed to mean centred. The values of each variable have zero-mean (when subtracting the mean in the numerator) and unit-variance. Additionally, the scaled data (X) as shown in Table II was used to develop the reference model in Phase II.

Phase II. Developing Reference Model

In order to develop a reference model, two steps were performed: (i) PCA to a data matrix X, and (ii) clustering to score. Each step is described below:

Table 2. Scaled data

Host ID	MIPS	RAM (MB)	Storage (MB)	Bandwidth (Mbps)
0	-784.95	-889.6	-341	-647.55
1	240.05	398.4	2659	2352.45
2	-1557.95	-1277.6	-4341	-4647.55
3	240.05	198.4	1659	1352.45
4	792.05	-161.6	659	352.45
5	692.05	128.4	159	152.45
6	1492.05	1098.4	659	352.45
7	-607.95	-421.6	659	352.45
8	464.05	390.4	3141	1744.45
9	1480.05	646.4	4159	3567.45
10	-1027.95	-551.6	-4041	-4447.55
11	192.05	-101.6	659	1852.45
12	1092.05	448.4	-1841	2352.45
13	-69.95	122.4	440	3552.45
14	92.05	-43.6	1859	-1245.55
15	-1207.95	598.4	-2941	-2647.55
16	692.05	-401.6	-2662	-2832.55
17	-495.95	-377.6	2659	-3397.55
18	-907.95	-449.6	-2199	2825.45
19	-807.95	646.4	-1005	-943.55

4) Step 4: Perform PCA to data matrix X
Regular PCA is applied to the two-dimensional data matrix (X) using:

$$X = TP^T = \sum_{r=1}^{R} t_r p_r^T + E \quad (1)$$

where T (K x R) is a matrix of latent variables scores, P(K x R) is a loading matrix for R latent variables, and E is an error terms matrix. The broken stick rule is used to identify the number of latent that should be retained to form feature vectors (P) as shown in Figure 3. The total variance for PC1 and PC2 is 94.68% in the overall percentage of variance. Based on this, a vector feature was formed by choosing two components, which are: (i) Principal Component 1 (PC1), and (ii) Principal Component 2 (PC2). Figure 4. shows the loading vectors for the reference model.
5) Step 5: Perform Clustering to Scores
K-means clustering was employed to cluster the scores in a score plot based on their pre-identified class of computing resources. The reason why K-means clustering was used is to improve the capability of PCA to divide data accurately into a number of clusters. However, in order to determine a

set of k centroids in n-dimensional space for the minimization of the sum-of-squares criterion, a general algorithm for k-means clustering is stated as follows:

$$J = \sum_{c=1}^{k} \sum_{j=1}^{n} \left\| x_j - \mu_c \right\|^2 \tag{2}$$

a.) K samples were randomly selected from the data set as the initial cluster centroids, , ,…. . Set iteration i=0.
b.) Every sample x_j was assigned to the cluster with the nearest centroid.
c.) After all samples were assigned, positions of the k centroids were calculated again, as follows

$$\mu_c^{(i+1)} = E\left\{x_j\right\}_{x_j \in \mu_c^{(i)}} \tag{3}$$

Process b was repeated as well as c, until the centroids were found to be no longer moving. The set of k centroids were noted as the algorithm output: μ_1 , μ_2 ,…. μ_k .

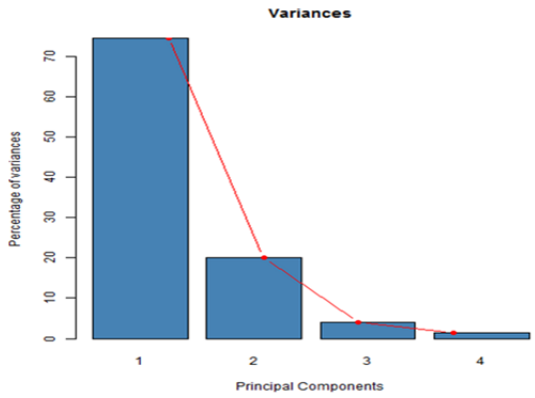

Figure 3. Identification of the number of principal components using broken stick rule

$$\text{Loading vectors} = \begin{bmatrix} 0.1857 & -0.0209 \\ 0.1008 & 0.0119 \\ 0.6532 & -0.7418 \\ 0.7271 & 0.6701 \end{bmatrix}$$

Figure 4. Loading vectors for reference model

After performing k-means clustering to the data matrix X, the pool of physical hosts is then divided into three clusters based on computing resources, which are minimum (cluster 1), medium (cluster 2) and optimum computing resources (cluster 3) as shown in Figure 5. Table III also shows ID for the physical hosts for each cluster.

Table 3. Host id in clustering

Clusters	Host ID
1	0, 4, 5, 6, 7, 14, 17, 19
2	1, 3, 8, 9, 11, 12, 13, 18
3	2, 10, 15, 16

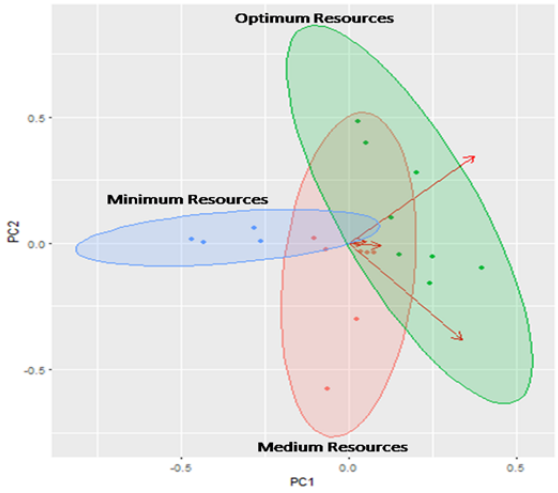

Figure 5. Clusters of physical hosts based on computing resources

Phase III. Load Balancing

This phase uses loading vectors and centroids from Phase II. It identifies the appropriate host cluster and matches it with the new VMs. In this step, the main function is to allocate the new VM to the nearest physical host clusters as shown below:

2) Step 6: Allocate the new VM to the nearest physical host clusters

Table IV describes the new data (X_{new}), in order to allocate the new VM to the nearest physical host clusters, for each VM that are transformed into scores (T_{new}) by employing the use of loading vectors from Phase II.

$$T_{new} = P^T X_{new} \tag{4}$$

These new scores were compared with the list of centroids in order to identify the clusters of the VM which represented the scores. Table V shows the allocation of all new VMs for the requested tasks after the calculation of distance between the scores and the list of centroids. For example, the most suitable physical hosts for the VM with ID 3 is from minimum computing resources, which is physical host no 5.

3. Result and Discussion

Model PCAC-LB is simulated and evaluated based on the requirements as well as evaluation criteria. A scenario for data centre ID 2 with 30 tasks called Cloudlet is also evaluated by four parameters; makespan, throughput, waiting times, response time and resource utilization. The results from CloudSim console with the use of PCAC-LB model as shown in Table 7., where the 30 Cloudlets ran successfully in the scenario and the complete time for all the Cloudlets is 175.15 ms.

a. Makespan

Be brief and state the most important conclusions from your paper. Do not use equations and figures here.

Makespan is the amount of time used to complete all tasks. As such, in order to increase the load balancing performance, it is required that the parameter should be minimized. This is because, if the makespan from Cloudlet or tasks is not reduced, then the request will not be resolved on time.

$$Makespan = max_i(Time_{Finish}) \quad (5)$$

where $Time_{Finish}$shows the finishing time of i^{th} task.

Results in Table VI show that the PCAC-LB model has better performance in terms of makespan, as it requires less time to complete all tasks because the number of built-in VM failures is reduced. Nevertheless, the PCAC-LB model successfully created 25 VM of 30 VM and all the VMs executed Cloudlets simultaneously. This is because the PCA-LB model focuses on allocating the required task based on its computing resources requirement. For example, if the required task needs big computing resources, physical hosts from optimum computing resources will be selected. Hence, it can reduce the problem in task allocation.

Table 4. Data for VM

VM ID	MIPS VM	RAM VM (MB)	Storage VM (MB)	Bandwidth VM (Mbps)
0	457	363	3000	3000
1	566	149	1900	1800
2	820	711	3200	2500
3	715	609	2800	3023
4	463	478	1989	2380
5	235	121	870	939
6	2038	1574	6842	6931
7	2528	1092	5906	5973
8	1657	975	2931	3186
9	842	592	2471	2769
10	1500	1620	3488	2926
11	1759	880	2509	3072
12	512	330	1800	2484
13	412	283	2658	2142
14	268	357	1439	1368
15	1421	1024	4500	3229
16	617	512	2183	1932
17	234	256	1700	2125
18	2125	1920	9422	8136
19	753	842	1256	1147
20	1916	1274	5982	4435
21	2857	1833	3344	4974
22	1392	1130	3687	4625
23	340	233	2090	2567
24	1862	1321	7176	3389
25	596	1830	2380	2947

Table 5. Allocation of the new VM for the specific physical hosts

ID for New VM		Allocation for physical host
0	→	0
1	→	0
2	→	4
3	→	5
4	→	15
5	→	2
6	→	1
7	→	Unsuccessful
8	→	12
9	→	16
10	→	6
11	→	6
12	→	3
13	→	4
14	→	Unsuccessful
15	→	3
16	→	5
17	→	18
18	→	9
19	→	10
20	→	8
21	→	Unsuccessful
22	→	11
23	→	8
24	→	14
25	→	19
26	→	Unsuccessful
27	→	17
28	→	7
29	→	Unsuccessful

Table 6. Comparison of round robin, FCFS, and PCAC-LB on makespan parameter

	VM that successfully created	Makespan (milliseconds)
Round Robin	17	341.98 ms
FCFS	24	340.52 ms
PCAC-LB	25	175.15 s

b. Throughput

Throughput is the total number of tasks, which is successfully implemented in a certain time period. Thus, the parameter should be at maximum for good performance. The formula for throughput is:

$$Throughput = \sum_{task}(Exe_{time}) \quad (6)$$

where Exe_{time}show a certain time period for execution.

Based on this, the total number of Cloudlets completed by the PCAC-LB, FCFS and Round Robin models are recorded every 10 ms as shown in Figure 6. It can be seen that at 100ms, the throughput for the PCAC-LB model is 23, while the processing result for FCFS is 21 and the result for Round Robin is 16. This shows that PCAC-LB improves load balancing performance in terms of throughput, as it can complete more tasks simultaneously. This is because

the PCA-LB model focuses on allocating the required task in batch instead of single allocation.

c. Waiting time and Response time

Waiting time is the time taken by a process in rest of getting an opportunity for execution after finding a source or machine. In Cloudsim, the cloudlet (task) will wait for the source before it is executed. Response time is the time for the task to respond. The response time for Cloudlet is the time to react and it ends when the Cloudlet's task is completed in CloudSim.

$$Response\ time = Time_{finish} - Time_{submission} \quad (7)$$

$$Waiting\ Time = Time_{Start} - Time_{submission} \quad (8)$$

where $Time_{Start}$ is the starting time to execute a cloudlet, and $Time_{submission}$ is the time a cloudlet request is submitted by users, which is 0 ms.

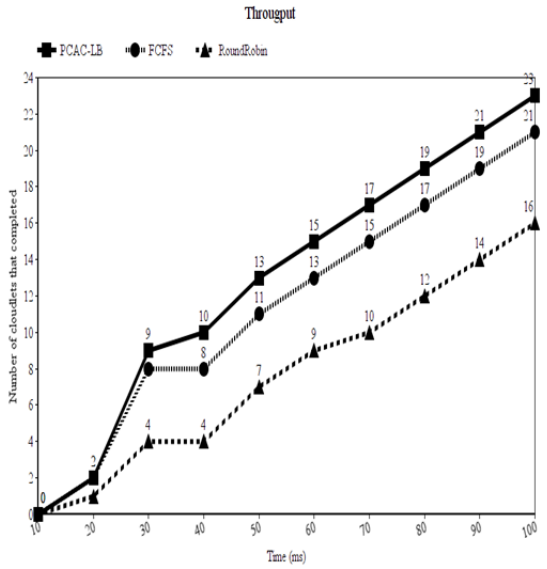

Figure 6. Comparison of Round Robin, FCFS, and PCAC-LB for throughput parameter

Table VII shows that PCAC-LB has better performance in waiting time as well as in response time because PCAC-LB model focuses on batch-processing algorithm, which is using clustering to complete a set of requests (not one after another request). This has reduced the waiting time and response time for each request.

Table 7. Comparison of round robin, FCFS, and PCAC-LB on waiting time and response time

	Average waiting time	Average response time
RoundRobin	39.280 ms	116.591 ms
FCFS	17.972 ms	87.996 ms
PCAC-LB	11.643 ms	73.501

d. Resource utilization

The optimal use of resources can prevent excessive load in certain physical hosts and waste the physical host source in cloud computing. Thus, the average resource utilization can use the following equation.

$$resourceutilization \frac{= (initialresources - availablesource)}{Initialresources} \times 100 \quad (8)$$

Resource utilization for every physical host for Round Robin, FCFS and PCAC model are recorded. The resource utilization of host is divided into three categories: less optimal, middle and optimal resource utilization respectively. The results in Table VIII show that the PCAC-LB model is better in terms of resource utilization as there are more physical hosts that have optimal resource utilization. The improvement of resource utilization in the proposed load balancing is because the PCAC-LB model considers the variations and variables of physical hosts. This shows the positive impact of using higher variables as it can represent the current status of the physical hosts in terms of computing resources.

The use of the PCA method has been investigated in other fields [20], [21], [22] for modelling a complex process. Results in Table IV, V and VI show that PCA is a practical option to model multiple servers that are used in a load balancing strategy. Conversely, the multiple servers or physical hosts can also be modelled based on their computing resources by using this data-based multivariate statistical method. As such, the present study main variance for each host is captured using PCA based on four variables. This is essentially because the PCA holds the capability as a data reduction technique. Hence, high-dimensional data are projected onto a low-dimensional model, in order for it to be much easier to make a comparison between the new and physical host data from clients.

Table 8. Comparison of Round Robin, FCFS, and PCAC-LB on resource utilization parameter

Category	Number of hosts		
	Less optimal resource utilization (<33.33%)	Middle resource utilization (33.34%-66.67%)	Optimal resource utilization (>66.67%)
Round Robin	14	2	4
FCFS	5	7	8
PCAC-LB	**3**	**5**	**12**

The performance of the PCA and clustering appears to be effective in terms of reducing makespan, increasing throughput, reducing waiting and response time and improving resource utilization

when compared with Round Robin and FCFS. One of the factors that contribute to this state of performance is the ability of the PCA to capture the main variance that represents VM based on MIPS, RAM, bandwidth and storage. Also, with regards to the variance, the tasks allocated for each VM as shown in Figure 7. are executed by a physical host that has the most similar features in terms of computing resources. The reason is that, if the variance falls under the category of medium resources, the task allocated for the VM can be executed by one of the physical hosts in this category. For example, VM number four is assigned to physical host E, as it is the suitable physical host to run the task. As such, it will optimize the resource utilization since the physical host is selected based on its capability to run the specified task.

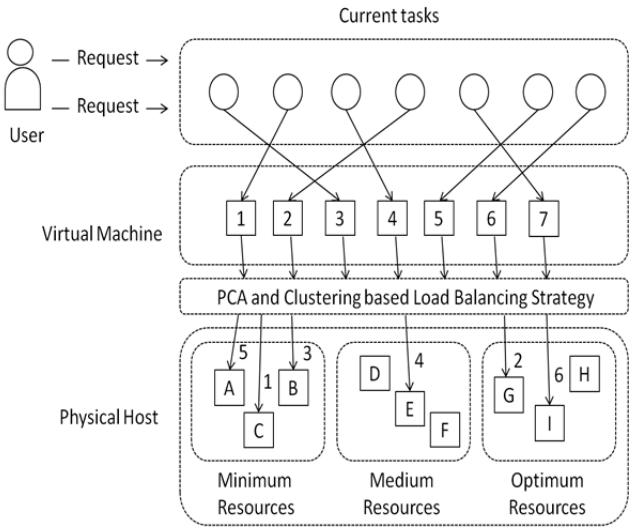

Figure 7. Selection of physical hosts based on three computing resources: minimum, medium and optimum.

Another factor that contributes to the good performance, is due to a set of requests which is processed in batches and not in singles of one request after another. For example, Figure 7. shows that seven VMs were assigned for seven requests. These VMs were then dedicated to the suitable physical hosts based on three categories. Nevertheless, the waiting and response time, respectively can be reduced because every new task was assigned to the physical host collectively. This is possible as the physical hosts have been categorised into three forms based on the computing resource. Therefore, the waiting time to be executed for each task can be reduced because the proposed strategy narrows down the selection of physical hosts.

4. Conclusions

This research has contributed to the development of load balancing strategy by investigating the application of multivariate statistical techniques. As such, a new strategy based on PCA has created a new way of considering heterogeneous variables that represent physical hosts. The differences in computing resources for the physical host have also been determined further by using k-means clustering. The developed strategy based on PCA and clustering, has therefore enabled the load balancing strategy to focus on processing a set of tasks as a batch, since the candidates for the physical hosts are suggested by the proposed model. More so, the strategy has been tested and shows an increase in the load balancing performance in terms of resource utilization, makespan, throughput, waiting time and response time compared to the Round Robin model and FCFS load balancing model. Based on the positive results, the model is expected to assist in developing a data-based system for load balancing for a data centre in a cloud computing environment.

Acknowledgements

We would like to thank FRGS/1/2017/ICT04/UKM/1 and GUP-2015-008 supported by Ministry of Higher Education of Malaysia for their financial assistance.

References

[1] Sreenivas, V., Prathap, M., & Kemal, M. (2014, February). Load balancing techniques: Major challenge in Cloud Computing-a systematic review. In *2014 International Conference on Electronics and Communication Systems (ICECS)* (pp. 1-6). IEEE.

[2] Mahdi, A. S., & Muniyandih, R. C. (2016). Enhancement Of Cloud Performance And Storage Consumption Using Adaptive Replacement Cache And Probabilistic Content Placement Algorithms. *Journal of Theoretical & Applied Information Technology*, *84*(3).

[3] Zhao, J., Yang, K., Wei, X., Ding, Y., Hu, L., & Xu, G. (2015). A heuristic clustering-based task deployment approach for load balancing using Bayes theorem in cloud environment. *IEEE Transactions on Parallel and Distributed Systems*, *27*(2), 305-316.

[4] Kadhum, A. M., & Hasan, M. K. (2017). Assessing the determinants of cloud computing services for utilizing health information systems: A case study. *International Journal on Advanced Science, Engineering and Information Technology*, *7*(2), 503-510.

[5] Samimi, P., Teimouri, Y., & Mukhtar, M. (2016). A combinatorial double auction resource allocation model in cloud computing. *Information Sciences*, *357*, 201-216.

[6] Uddin, M., Memon, J., Alsaqour, R., Shah, A., & Rozan, M. Z. A. (2015). Mobile agent based multi-layer security framework for cloud data centers. *Indian Journal of Science and Technology*, *8*(12), 1.

[7] Pasha, N., Agarwal, A., & Rastogi, R. (2014). Round robin approach for VM load balancing algorithm in cloud computing environment. *International Journal of Advanced Research in Computer Science and Software Engineering*, *4*(5).

[8] Devi, P., & Gaba, T. (2013). Implementation of cloud computing by using short job scheduling. *International journal of advanced research in computer science and software engineering*, *3*(7), 178-183.

[9] Kaur, R., & Luthra, P. (2014). Load balancing in cloud system using max min and min min algorithm. *International Journal of Computer Applications*, *975*, 8887.

[10] Galloway, J. M., Smith, K. L., & Vrbsky, S. S. (2011, October). Power aware load balancing for cloud computing. In *Proceedings of the World Congress on Engineering and Computer Science* (Vol. 1, pp. 19-21).

[11] Sethi, S., Sahu, A., & Jena, S. K. (2012). Efficient load balancing in cloud computing using fuzzy logic. *IOSR Journal of Engineering*, *2*(7), 65-71.

[12] Nitika, M., Shaveta, M., & Raj, M. G. (2012). Comparative analysis of load balancing algorithms in cloud computing. *International Journal of Advanced Research in Computer Engineering & Technology*, *1*(3), 120-124.

[13] LD, D. B., & Krishna, P. V. (2013). Honey bee behavior inspired load balancing of tasks in cloud computing environments. *Applied Soft Computing*, *13*(5), 2292-2303.

[14] Dasgupta, K., Mandal, B., Dutta, P., Mandal, J. K., & Dam, S. (2013). A genetic algorithm (ga) based load balancing strategy for cloud computing. *Procedia Technology*, *10*(2), 340-347.

[15] Hu, J., Gu, J., Sun, G., & Zhao, T. (2010, December). A scheduling strategy on load balancing of virtual machine resources in cloud computing environment. In *2010 3rd International symposium on parallel architectures, algorithms and programming* (pp. 89-96). IEEE.

[16] Mondal, B., Dasgupta, K., & Dutta, P. (2012). Load balancing in cloud computing using stochastic hill climbing-a soft computing approach. *Procedia Technology*, *4*, 783-789.

[17] Lee, G. (2012). *Resource allocation and scheduling in heterogeneous cloud environments*, PhD Thesis, University of California, Berkeley.

[18] Domanal, S. G., & Reddy, G. R. M. (2015, November). Load balancing in cloud environment using a novel hybrid scheduling algorithm. In *2015 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)* (pp. 37-42). IEEE.

[19] Wold, S., Esbensen, K., & Geladi, P. (1987). Principal component analysis. *Chemometrics and intelligent laboratory systems*, *2*(1-3), 37-52.

[20] Majid, N. A. A., Taylor, M. P., Chen, J. J., Stam, M. A., Mulder, A., & Young, B. R. (2011). Aluminium process fault detection by multiway principal component analysis. *Control Engineering Practice*, *19*(4), 367-379.

[21] Majid, N. A. A., Taylor, M. P., Chen, J. J., Yu, W., & Young, B. R. (2012). Diagnosing faults in aluminium processing by using multivariate statistical approaches. *Journal of Materials Science*, *47*(3), 1268-1279.

[22] Majid, N. A. A., Taylor, M. P., Chen, J. J., & Young, B. R. (2011). Multivariate statistical monitoring of the aluminium smelting process. *Computers & chemical engineering*, *35*(11), 2457-2468.