Equally Spread Current Execution Load Modelling with Optimize Response Time Brokerage Policy for Cloud Computing

Anisah Hamimi Zamri[1], Nor Syazwani Mohd Pakhrudin[2], Shuria Saaidin[3], Murizah Kassim[4]*
School of Mechanical Engineering-College of Engineering, Universiti Teknologi MARA, Shah Alam, Selangor, Malaysia[1]
School of Electrical Engineering-College of Engineering, Universiti Teknologi MARA, Shah Alam, Selangor, Malaysia[2, 3, 4]
Institute for Big Data Analytics and Artificial Intelligence (IBDAAI), Universiti Teknologi MARA, Shah Alam, Selangor, Malaysia[4]

***Abstract*—Cloud computing is one of the significant technologies that is used to provide seamless internet surfing for large-scale applications and data storing purposes. The cloud is described as a large platform that enables users to access data from the internet without needing to buy storing space in their equipment such as a computer. Many studies have analysed the load-balancing technique on the cloud to distribute tasks equally between servers using the Equally Spread Current Execution (ESCE) algorithm. ESCE, which is a dynamic load balancer, has quite a few problems such as average level performance and too long of response time which affected the Quality of Services. This research has simulated a cloud computing concept using the ESCE Load Modelling technique with the CloudAnalyst simulator for three servers of Data Center (DC) locations. The ESCE was simulated to enhance its algorithm's performance as a load balancer and higher throughput in the cloud environment. The result shows that ESCE average overall response time is shortest when the DC is located at R0 with response times of 15.05s, 13.05s with 10 VMs, and 8.631s with the Optimize Response Time brokerage policy. This research is significant to promote notable load-balancing technique testing for virtualized cloud machines data centers on Quality of Services (QoS) aware tasks for Internet of Things (IoT) services.**

Keywords*—Equally spread current execution (ESCE); optimize response time brokerage; cloud computing; load balancer; data modelling

I. INTRODUCTION

Pay-as-you-go online computing services, such as apps, storage, and processing power, are referred to as cloud computing. Due to how simple and affordable it is to use large-scale applications, the demand for this technology has greatly increased [1]. A significant aspect of this environment is the scalable delivery of IT infrastructure and applications as a service, according to the definition of cloud computing, which is end user-focused on how they may experience the cloud environment such as data storage for the information technology for education, remote monitoring and mobile robot for surveillance application [2-5].

Fig. 1 presents the illustration of cloud computing as a user device accessing the internet and communication exists between servers, applications, and databases. Cloud computing may be utilised as an interface for the integration to quickly complete a variety of client requirements, in just one click. The biggest issue with this system is managing a large number of client requests that could total millions at one time[6, 7]. Mainly maintaining and enhancing prior software investments while responding to environmental changes is the major objective of current application adaptation[8]. The private cloud, the public cloud, the hybrid cloud, and the community cloud are the four cloud computing deployment methods that are based on research and are grouped by their distribution and physical location[9]. Three types of cloud service models are Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS).

Concerning effectively managing user requests, load-balancing methods must be used [10]. Load-balancing serves as a tool for assignation and also controls the system's overall rendering [11]. The Equally Spread Current Execution (ESCE) is one of the algorithms that act as a load balancer to process user requests. ESCE algorithm (LBA) allocates user requests equally to all virtual machines (VM) bound to the data centre [12]. This kind of load balancer keeps track of the inventory of virtual machines and their current availability. However, depending on the suggested work or goal of the cloud environment, the algorithms' efficacy is unclear [13]. Each algorithm varies depending on the type of load-balancing within the bounds of cloud computing [14]. The turn-based Round Robin load-balancing solution has its unique algorithm [15]. The first server will receive the first request, the second server will receive the second request, and so on. Following distributing tasks after determining their size, the Equally Spread Current Execution algorithm uses a spread spectrum technique.

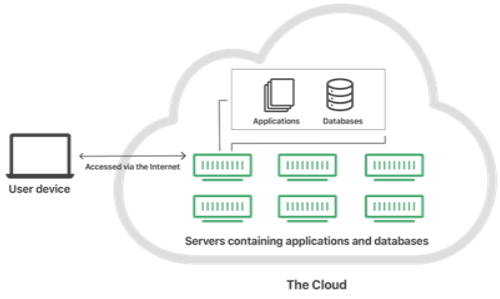

Fig. 1. Illustration of cloud computing

Additionally, this algorithm performs average in terms of response rate and has excessively long mediocre minimum and maximum response times [16]. Response time-wise, the Equally Spread Current Execution algorithm is excessively slow. This is a result of the algorithm's lengthy check of the request size. The algorithms must first successfully assess the availability status of each machine before allocating them to the available virtual machines. This causes system delays and reduces the effectiveness of the entire load-balancing strategy. When compared to other load balancers now in use, the performance of this method is in the middle of the pack [17]. Equally Spread Current Execution was not the best load-balancing strategy when compared to the others. An earlier study compared it to other load-balancing methods including Round Robin and Throttled Load-balancing [18]. It does not, however, qualify as the worst technique either. The issue is that it is not the best technique to employ at the moment.

Cloud computing is Internet-based computing that offers a pool of customizable computing resources such as networks, storage, servers, applications, and services without requiring interaction with the service provider and with little administration effort[19]. The RR protocol's moving horizon estimation problem for a class of discrete time-delay systems. To avoid data collisions, communication between the sensor nodes and the remote state estimator is carried out over a shared network, with only one sensor node able to transmit data at any given time. The RR protocol organizes the transmission order of sensor nodes, with the selected node gaining network access modelled as a periodic function. The device model is reformulated into a linear system without delays due to lifting technology. The problem at hand aims to construct a moving horizon estimator that minimizes the estimation error. In matrix inequality, a proper condition is defined to ensure ultimate boundedness [16]. In cloud computing, the cloud infrastructure cannot handle the flow of information independently with the profusion of data, devices, and interactions [7]. Load-balancing distributes all workloads across each node in a shared or mutual system to maximize resource utilization and reduce job response time. The most important aspect of cloud computing is scheduling, which includes workloads and workflow scheduling under the platform as a service model. The infrastructure as service task to VMs scheduling which machine is decided by the scheduler should go on which job or VMs[20]. Evaluating the energy required for communication between the devices participating in this process and the suggested method's appropriateness for handling optimization problems like VM placement is necessary[21]. Load-balancing is a method of reassigning the entire load to the various nodes of a collaborative system to improve resource efficiency and the job's response time while avoiding a situation where specific nodes are overloaded and others are underloaded. A load-balancing algorithm involved in identity ignores the system's previous state or behavior, relying instead on the system's neighboring behavior. This load can be measured in terms of CPU, memory use, sluggishness, or network load[22]. Load-balancing's primary aim is to enhance device efficiency and functionality for today's QoS for IoT services in communication and data transfer. It is used in cloud computing systems to provide successful alternative solutions in the event of a system failure and to ensure the best possible use of system components. Load-balancing techniques are divided into two main models Static Model and Dynamic Model. First is Static Load-balancing: Static load-balancing occurs in a static environment where the output of algorithms is unaffected by the system's current state. As a result, user expectations do not change over time [23]. Second is Dynamic Load-balancing: The system's state has a significant impact on balancing the efficiency of the algorithms. Since resources are versatile in a dynamic environment, algorithms efficiently perform load-balancing[24].

An increasing number of computation-intensive and delay-sensitive mobile apps keep appearing alongside the evolution of smart Mobile Devices (MD). The task migration issue in the context of cloudlet federation is the primary topic of this research[25]. Considered a cloudlet federation scenario involving three distinct Clouldlet service Providers (CLP) and a remote cloud. Cloudlet federation can efficiently minimise cloudlet deployment and management expenses by pooling resources among CLPs. Task migration faces new obstacles because CLPs vary in their user and resource counts.

Proactive dynamic VM consolidation is used in this research to improve resource utilisation and performance without sacrificing the energy economy[26]. The suggested algorithm creates a fine-grained categorization that takes workload considerations into account by utilising machine learning techniques to create complementary profiles that reduce cross-application interference by strategically colocating HPC and non-HPC workloads. Real HPC workloads were simulated for the study using CloudSim. The outcomes proved that, in terms of the metrics in important areas, the suggested algorithm beats all heuristic methods[26-28].

This research examined the performance of the Equally Spread Current Execution load modelling method for Quality of Service, QoS aware task placement for the Internet of Things (IoT). Since the ESCE load modelling technique requires communication between the data centre and load balancer, the procedure results in overhead [29]. Intended to examine the behaviour of a large distributed system, this project investigated the study of ESCE load modelling using the CloudAnalyst simulator. The user-friendly interface of the CloudAnalyst simulator made it easier for analysts to set up simulations. This research has successfully analysed the performance of ESCE in various situations and proven that it works efficiently by implementing the correct configuration according to the scale of the application.

The paper is divided into five main sections. The Introduction section provides a brief overview of the problem and research question, as well as an introduction to the relevant literature. The theoretical and proposed work section brief the overview of the research gap and the literature review, including the proposed work and some of the mathematical modelling techniques. The Methodology section outlines the research methods used, including the data sources and analysis technique. The Results section presents the findings of the research, while the Discussion section discusses the implications of the research and its implications for future research. Finally, the Conclusion section summarizes the research and provides suggestions for further research.

II. THEORETICAL AND PROPOSED WORK

Critical analysis of vast amounts of data, including energy production and consumption, is required to develop a secure and sustainable energy system [30]. Due to the high demand for cloud services, much research has been done to improve the performance of the current existing load-balancing. Upon enabling the stability of cloud environments, load-balancing is a key role to ensure no node in the load-balancing is unequally utilized than the other [31]. A study has proposed a genetic algorithm-based task scheduling for load-balancing which demonstrated that the proposed method results exceed the performance of current load-balancing techniques [32]. Proposed algorithms are known to surpass basic static and dynamic load-balancing since their method has attempted to solve problems by creating an alternative to minimize the response time such as designing hybrid algorithms in taking a part of existing algorithms and attaching additional processes in the algorithm.

Another study analyzed the response time of dynamic load-balancing, their results showed that all algorithms used in the simulation, THR, ESCE and Particle Swarm Optimization (PSO) gave the same results when the resource is homogenous. Otherwise, entitled to swarm intelligence ability that the task can be executed more efficiently[33]. Most proposed work has proven that the existence of a variety of cloud computing aspects needs more systematic and advanced algorithms to cater for the complicated environment [34]. According to a study, analysis is done to compare the performance between three existing load-balancing types namely the Round Robin (RR), Equally Spread Current Execution (ESCE) and Throttle (THR) using CloudAnalyst by broker policy grouping and load-balancing which results showed that THR has the best performance compared to the other two as prevents overload efficiently by thresholding the available VMs list [35]. The analysis is done by different data centre policies in the same simulation scenario with nine possible load-balancing approaches and five different workloads which have obtained 45 different results. Their work is quite similar to this analysis as they reviewed the performance of the load-balancing method to compare the results and determine which one has the best performance analytically.

In another comparative study done to evaluate the performance of algorithms, the increasing number of users has been used as a variable as they claimed much previous research has not considered the increasing number of users as a contributor to the improvement of algorithms whereas in the simulation to compare three algorithms, RR, ESCE, and THR they have placed UB user base and data centres in the same region by neglecting the effect of geographical distance [36]. Their results proved that algorithms performance does increase with user numbers as initially in the simulation with 5000 users, no significant difference can be seen between the three algorithms' results. Another simulation has been done to analyze performance by using different service broker policies using four data centres all located in different regions but having the same amount of VMs which is 50 [37]. Simulation is done alternately for three different broker policies to analyze performance based on response time, request processing time and cost the results given THR has the best performance with a small processing cost compared to the others. While another study has focused on various policies utilized for load-balancing, a simulation was done between all six different user bases that are utilized with four data centres and in these data centres 25 VMs, 50 VMs 75 VMs, and 100 VMs respectively announces to the 4 DCs [38]. The load-balancing algorithms compared are RR, ESCE, THR and First Come First Serve (FCFS) algorithms. The peak hour users set for those 6 user bases vary with each other and concluded that RR has the best integrated performance. Next is a comparative analysis comparing THR, RR, ESCE, FCFS, and Shortest Job First (SJF) [39]. In this analysis, they optimized 6 user bases in all 6 regions respectively with 4 data centres in R0, R4, R2, and R3 containing 15 physical machines consolidated with 10 VMs each. The results showed that FCFS performed the best when it is compared on a data centre processing time basis while ESCE performed best when it comes to the lowest total cost. They showed that a different basis can give different results.

Analysis of the load modelling utilizing Equally Spread Current Execution (ESCE) load-balancing is the goal of this study. The CloudAnalyst simulator, an expanded piece of software created from the CloudSim toolkit, will be used as the approach abandoned for data collecting. A programmer called CloudSim toolkit is used to track internet behavior based on settings made in the simulation configuration. The CloudAnalyst simulator separates the user's global location into a few zones. Any region can be selected for the simulation, and the number of data centres can be adjusted correspondingly. The initial configuration setup will be the subject of the simulation.

A. Modelling Technique

This algorithm works based on load sets assigned to virtual machine sets at a given time. The mathematical model is defined by the following equations to calculate the processing time for a task to be executed. Eq. (1) is given by n as the number of sets for the load (L) or requests that need to be scheduled to servers.

$$L = \{L_1, L_2, \dots, L_n\} \quad (1)$$

Eq. (2) is given by k as the number of sets for virtual machines (V) in a particular data centre (D).

$$V = \{V_1, V_2, \dots, V_k\} \quad (2)$$

Eq. (3) is given by DL as the current data centre load.

$$DL = \{VL_1, VL_2, \dots, VL_k\} \quad (3)$$

Eq. (4) shows to find a function where the load set can be mapped (multiply) into virtual machine set (VL) and f(L), forming load VL_i of each virtual machine V_i to be essentially equal.

$$VL_1 \approx VL_2 \approx \dots \approx VL_k \quad (4)$$

Eq. (5) shows to calculate the time needed to allocate all tasks to virtual machine V_i, take τ_0 as the time to execute task L_0.

$$t_i = \sum_{0 \epsilon f(L_i)(i=1,\dots,n)} \tau_0 \quad (5)$$

When there is only one virtual machine, which means k=1 and all available tasks will be executed serially on that one virtual machine, Eq. (6) shows the execution time will be the summation of all time and can be calculated as T_1.

$$T_1 = \sum T_0(0 = 1, \ldots, n) \quad (6)$$

And when k>1 which means there are more virtual machines available, all available tasks can be allocated (shared equally) to multiple servers. Making the serially executed task before becoming parallel since multiple virtual machines are working simultaneously. Thus, Eq. (7) shows the time needed to execute a task is calculated as T_k.

$$T_k = max_{i=1,\ldots,n} t_i \quad (7)$$

In conclusion, the goal is to solve the function to get the minimum of time needed to execute a task.

III. Methodology

This section explains the details of the flow chart of the proposed study and the simulation for the analysis of Equally Spread Current Execution Load Modelling for Cloud Computing.

A. Flow Chart

Fig. 2 shows the ESCE algorithm flowchart which depicts how the algorithm handles user requests. To ensure equal loads for every virtual machine (VM) involved in the algorithm, ESCE uses the spread spectrum approach. Receiving a request, the load balancer updates the index table with the VM status count and scans the request in the index table before allocating it to an available VM. The parallelogram represents the beginning of the user request, the square shape of the process carried out by the simulator to evaluate the request before sending it to the virtual machine, and the diamond is the choice made by the load balancer of which virtual machine to send the user request to be processed. The virtual machine count and availability are stored in data storage with a cylinder shape.

B. Simulation

Equally Spread Current Execution (ESCE) is a cloud analytic methodology that is used to optimize the performance of cloud services. It is a cost-effective method of achieving higher throughput in the cloud. ESCE works by dynamically spreading the current execution load across multiple servers in order to evenly distribute the load. This helps to reduce the load on any one server, thus increasing the overall performance of the cloud. Additionally, ESCE can also be used to improve the scalability of the cloud environment. By leveraging multiple servers, the cloud can handle larger workloads without experiencing any performance degradation. ESCE is an effective way to improve the performance and scalability of cloud services and can be implemented in a variety of cloud environments.

The simulation is started by identifying the desired system components such as virtual machines, storage, network components and the desired number of resources for each component. A simulation tool that will be used to measure the performance of the system was developed using CloudAnalyst. This tool should include the ability to measure system performance and resource utilization. The simulation began with powering the software application. The user base (UB), data centre (DC), and load-balancing policy were identified in the simulation configuration, in this example, the Equally Spread Current Execution (ESCE). Execute the simulation tool and analyze the results. This includes measuring the average response time, resource utilization, throughput, and scalability of the system. The entire simulation lasted for 60 minutes. The number of data centres and virtual machines is a controlled variable in this simulation. The changes in outcome were determined by both variables. Based on a few initial setups, this software was used to simulate and study the cloud environment. The Equally Spread Current Execution (ESCE) load balancer paradigm was employed in this study to examine load-balancing performance. Response time, data centre processing time, and virtual machine (VM) cost are the outcomes tracked. The system configuration and workload have been optimized to achieve the desired performance and re-run the simulation tool and analyze the results. Fig. 3 illustrates how the CloudAnalyst simulator, an extension of the CloudSim toolkit, simulates scenarios based on internet behavior [40]. Build with a clear graphical user interface, the simulator's main objective is to separate experimentation activity from mapping exercises so the modeler may focus on the issues rather than the technicalities.

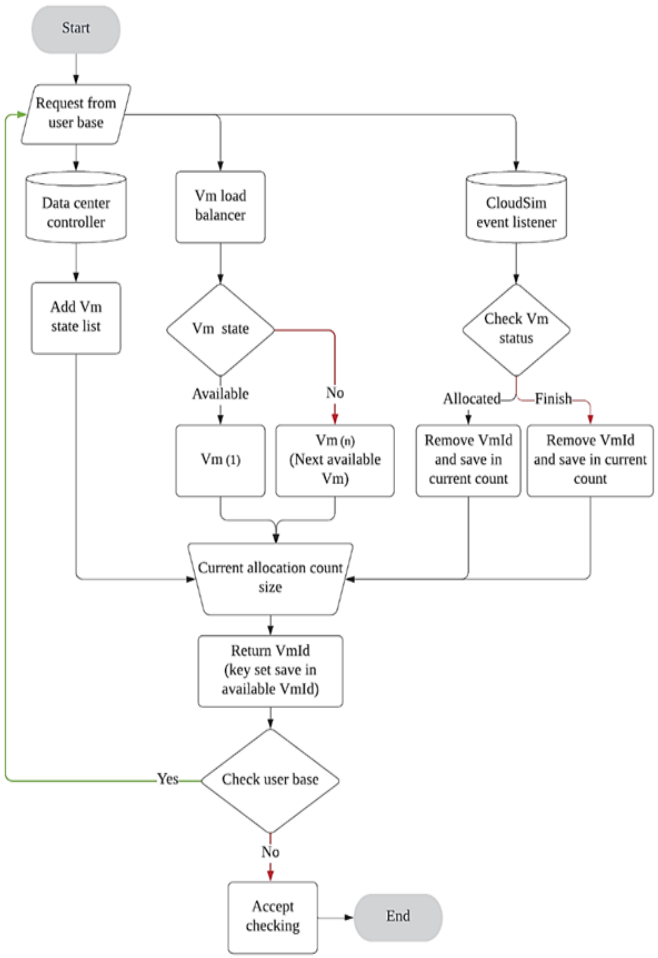

Fig. 2. ESCE execution process flowchart

Fig. 3. Graphical user interface (GUI) of CloudAnalyst

IV. Result and Discussions

This section describes the result of the simulation and the discussions for the simulation in the CloudAnalyst toolkit. This analysis was run in the CloudAnalyst simulator, the data is worked from an application "Facebook" known to be one of the examples of cloud computing applications. North America, Africa, and Oceania have been chosen to be the regions where user bases are assigned. The software identified these regions as R0, R4 and R5 respectively. There are more than these three regions in the software to be selected from, but they are chosen based on the approximate distance from each other to balance the outcome of simulation scenarios. The statistics of Facebook users in 2022 showed that R0 has 201.3 million users, R4 is the region with the highest number of users at 242.2 million and lastly, and R5 with 21.0 million.

A. Configurations

Table I presents the user number details for peak and off-peak hours. Table II presents the detailed settings for different simulation scenarios. There are a few different scenarios for simulation testing to get the desired results. Based on the statistical information, each region was set to only 10% of the total user number for peak-hour users and 5% for off-peak hours. The number of requests coming in is assumed to be once per five minutes for R0 and R5, while once per ten minutes for R4. The data size for a request is set to 100b. All simulation scenarios were run for 60 minutes with user grouping factor in user bases at 1000, request grouping factor in data centres at 100 and executable instructions length per request at 500 bytes. The load-balancing policy used is the Equally Spread Current Execution Load (ESCE) since it is the main objective of this analysis. The data centres are set to a memory of 1024 Mb. Other unmentioned configurations are unchanged in their default settings.

B. Response Time

Response time is the time taken to respond to requests coming in from clients. Lower response time means the load-balancing algorithm is doing a good job. Fig. 4 shows the analysis of the DC location as the variable. Three different simulation scenarios run with one DC in each simulation with the DC being located at R0, R4 and R5 alternately. Average overall response time showed DC located at R0 having the shortest response time compared to DC located at R4 and R5.

TABLE I. User Number for Peak and Off-Peak Hours

Region	User number	
	Peak Hour(million)	*Off-Peak Hour (million)*
R0	4.026	2.013
R4	4.844	2.422
R5	0.420	0.210

TABLE II. Detailed Settings for Different Scenarios

Simulation	DC Number	Different scenarios		
		VM Number	*DC Location (Region)*	*Service Broker Policy*
1	1	100	R0	Closest Data Centre
2	1	100	R4	Closest Data Centre
3	1	100	R5	Closest Data Centre
4	1	1000	R0	Closest Data Centre
5	1	10	R0	Closest Data Centre
6	3	100	R0, R4, R5	Closest Data Centre
7	3	100	R0, R4, R5	Optimise Response Time
8	3	100	R0, R4, R5	Reconfigure Dynamically with Load

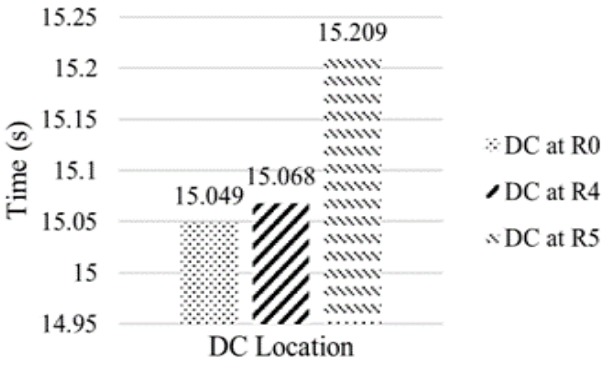

Fig. 4. Average overall response time analysis based on one data centre (a) at a different location

Fig. 5 shows the result of the simulation run with three different numbers of VMs. 10 VMs gives the shortest response time analysis at 13.05 seconds among all three followed by 100 VMs at 15.05 seconds and 1000 VMs at 15.25 seconds. 100 and 1000 VMs gave only slightly different result from each other but is still significant with the difference being approximately 0.2 seconds which is quite big since original data are analyzed in milliseconds.

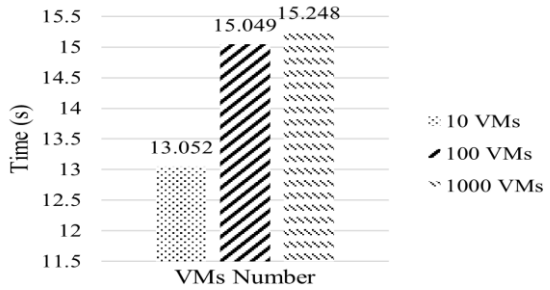

Fig. 5. Average overall response time analysis based on one data centre at a different number of VM

Fig. 6 presents the result given by applying different brokerage policies in three different simulations to analyze the response time given by each brokerage policy. The Optimize Response Time Brokerage policy has the shortest response time at 8.63 seconds, followed by reconfigure dynamically with load at 13.31 seconds response time and the closest data centre at 13.40 seconds. Response time is also observed by region and all three regions gave different results depending on the variables carried out for each simulation. Fig. 7 illustrates the result of response time with different DC locations. This result showed that all three locations gave R5 the shortest response time.

Fig. 8 presents all VMs number categories that had the same result with R5 being the region with the shortest average response time. Fig. 9 shows that given three DC in total with one at each region respectively, R5 also have the shortest response time among the three regions which is small compared in number.

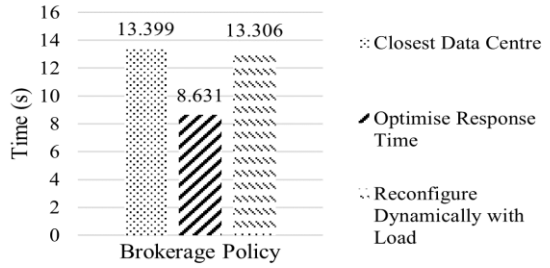

Fig. 6. Average overall response time analysis based on three data centres for different brokerage policy

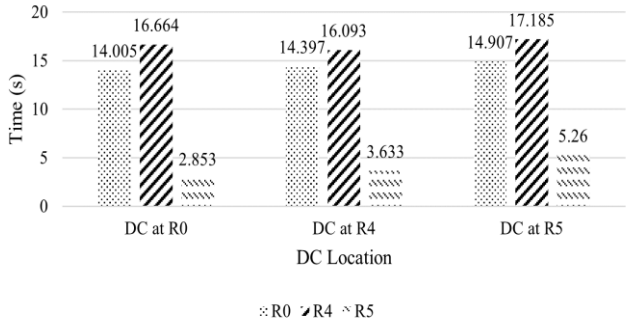

Fig. 7. Average response time by region based on one data centre at a different location

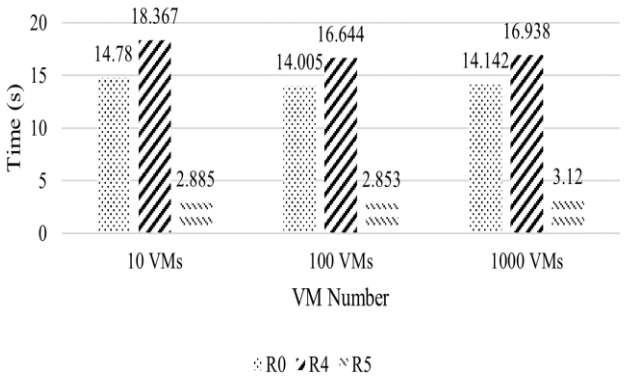

Fig. 8. Average response time analysis by region based on one data centre with a different number of VM

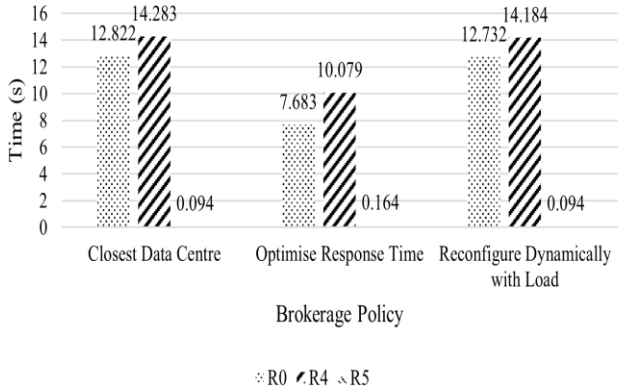

Fig. 9. Average response time analysis by region based on three data centres for different brokerage policies

C. Average Data Centre Processing Time

The data centre processing time analyzed the DC capability to work under different conditions. Fig. 10 shows one DC placed at different locations giving the shortest average processing time of 14.55 seconds when located at R4. Fig. 11 shows processing time took only 12.38 seconds for 10 VMs in one DC compared to 14.71 seconds for 100 VMs and 14.90 seconds for 1000 VMs.

Fig. 10. Average DC processing time analysis based on one data centre at a different DC location

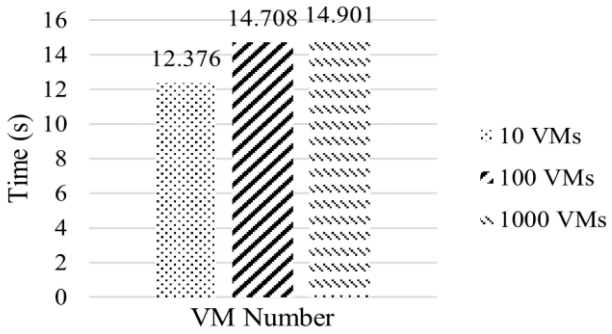

Fig. 11. Average DC processing time analysis based on one data centre with a different number of VM

Fig. 12 describes the result of DC processing time for the simulation with different brokerage policies. Results showed that Optimize Response Time Brokerage policy gave the shortest processing time for the data centre at 8.36 seconds compared to the other two brokerage policies.

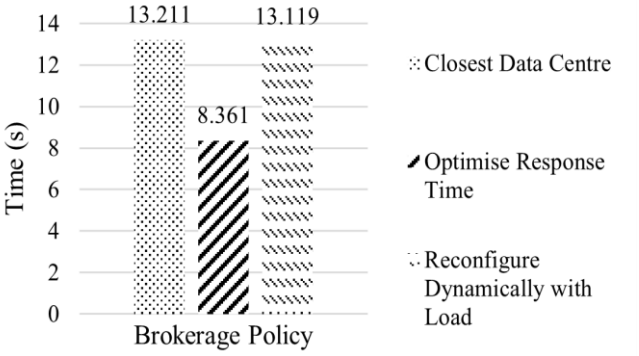

Fig. 12. Average DC processing time analysis based on three data centres for different brokerage policy

D. Resource Utilization (Cost)

Defined to check the utilization of resources. It is related to cost. Resource utilization in a system should be maximized to avoid clients paying for any unused resources. All cost used in the simulation is as shown in Table III.

One data centre with 100 VMs would cost approximately 10$ each. Brokerage policy simulation only used one data centre for each different brokerage policy. Fig. 13 shows that all cost does not differ much from each other.

TABLE III. COST-DETAILED SETTING FOR EACH DC

Category	Price ($)
Cost per VM ($/hr)	0.1
Memory Cost ($/s)	0.05
Storage cost ($/s)	0.1

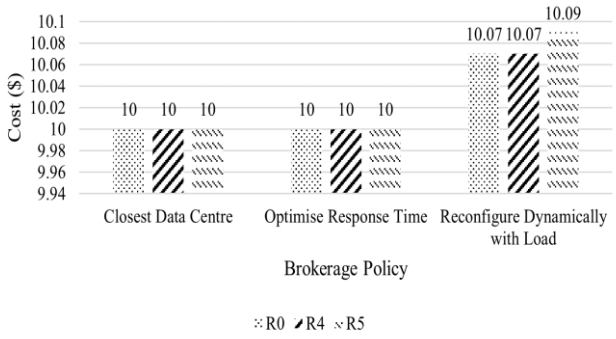

Fig. 13. VM cost analysis by region based on three data centres for different brokerage policy

VM cost would differ depending on how much VM is assigned in a data centre. Fig. 14 shows the result for the simulation of different DC locations but all having one data centre at a time and 100 VMs. Each DC cost the same since all have the same VM number. Fig. 15 shows a clear difference since the three simulation uses a different amount of VM in one data centre. 10 VMs is the cheapest since 1 VM used per hour only cost 0.1$. 1000 VMs cost only 80$ and not 100$ since normally cloud plan offers some discount for certain packages subscribed.

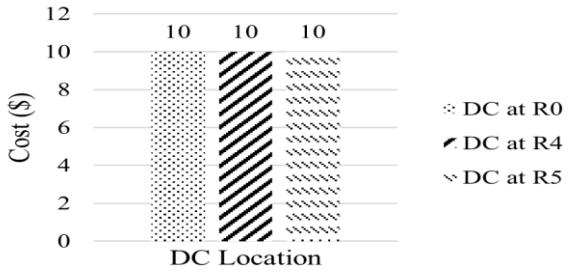

Fig. 14. VM cost analysis based on one data centre at a different DC location

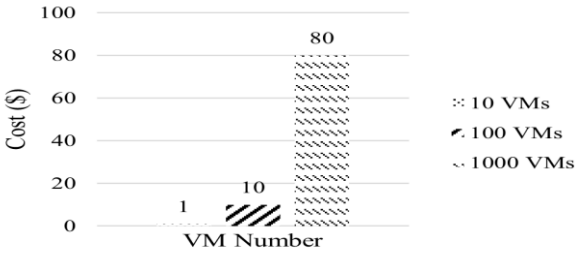

Fig. 15. VM cost analysis based on one data centre with a different number of VM

Fig. 16 shows three data centres would cost approximately 30$ since one data centre costs 10$. The reconfiguring dynamically with load brokerage policy gave off 30.22$ since the machine worked according to load changes. The 0.22$ difference is the result of algorithm work to be reconfigured according to load.

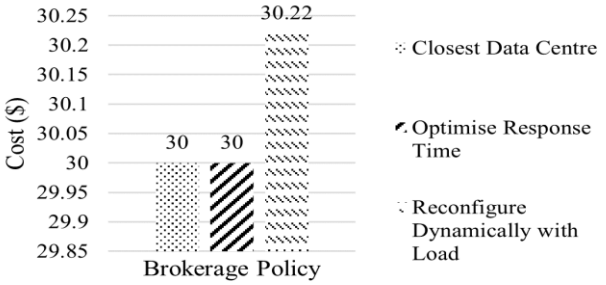

Fig. 16. VM cost analysis based on three data centres for different brokerage policy

Data transfer costs depended on the DC location. Data transfer costs would be higher if the distance between the DC location and the region where requests come from is quite far. Fig. 17 shows that in the closest data centre and reconfigure dynamically with load brokerage policy the region with the cheapest data transfer cost is R5 since the DC location is closer to any of the other two regions but, in optimizing Response Time Brokerage policy, the algorithm had to work based on the policy despite the location of DC. Hence, the cheapest cost for this brokerage policy only went as low as 106.91$ at R0. The total data transfer cost for one data centre at a different location is lowest when DC is located at R0 and R5 with both being 385.11$. Data transfer cost is calculated at a total of DC transferring data to all assigned regions. Fig. 18 illustrates there is only one data centre at a time which would make all three regions assigned to the one DC to carry out task execution work.

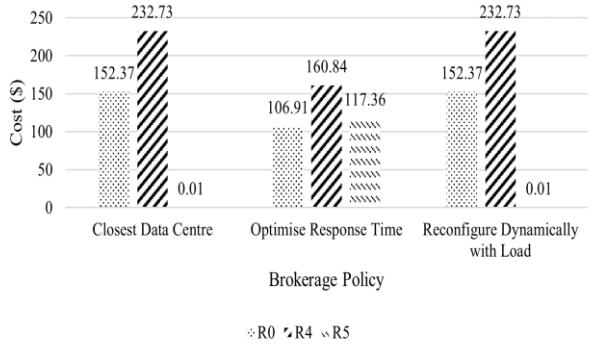

Fig. 17. DT cost analysis by region based on (a) three data centres for different brokerage policy

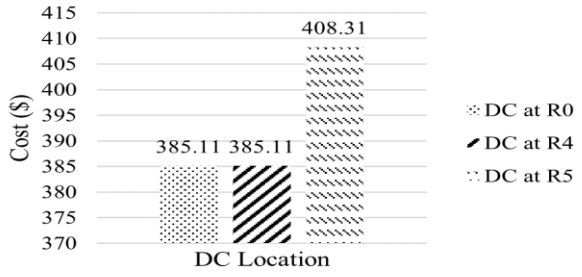

Fig. 18. DT cost analysis by region based on one data centre at a different DC location

Fig. 19 shows 100 VMs and 1000 VMs having the same results of 385.11$ while 10 VMs costs 504.3$. This is because the lower number of VMs had to do more work to execute data before transferring it to the user in their respective regions.

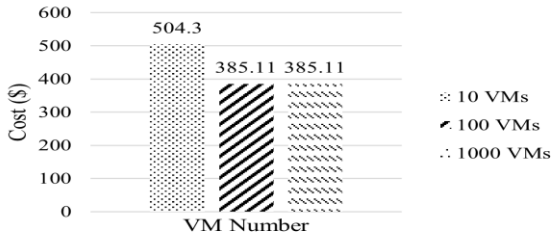

Fig. 19. DT cost analysis based on one data centre with a different number of VM

Fig. 20 illustrates the brokerage policies have equal same results of 385.11$ in data transfer cost since simulations were carried out with one data centre located respectively at each region. All regions have their own data centre to process tasks.

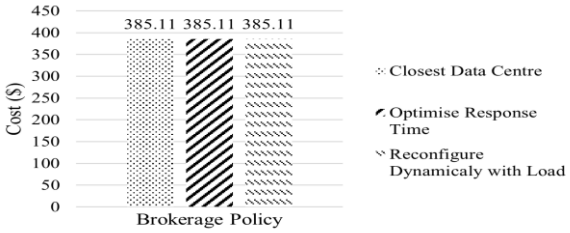

Fig. 20. DT cost analysis by region based on three data centres for a different brokerage policy

V. Conclusions

This study has successfully analysed the three main factors of cloud computing which are off-peak resource utilisation, minimum data processing time, and minimal average response time. To optimize the response time brokerage policy for cloud computing, a simulation for the equally spread current execution load has been developed. This simulation has considered the current workload of each cloud system, the expected workload in the near future, and the existing resources of each system. The goal of the model is to ensure that the workload is evenly balanced across the cloud systems, to ensure that no one system is overloaded while another is underutilized. The result of this simulation was able to allocate the workload among the cloud systems in a way that optimizes response time and meets the goals of the system. This can be done by assigning the workload to the system that can best handle it, or by using techniques such as load-balancing or resource scheduling. It should also consider the future growth of the cloud system. This can be done by allowing for future increases in workloads.

Acknowledgment

Authors would like to thank Research Management Centre (RMC), Universiti Teknologi MARA, Shah Alam, Selangor for the support funding in this research from grant no. 600-RMC/GPMST5/3(038/2021) and Institute for Big Data

Analytics and Artificial Intelligence (IBDAAI) in supporting this research.

REFERENCES

[1] T. Azmat, D. Kumar, and V. K. Dwivedi, "A Novel Approach towards an Efficient Load-balancing Algorithm in Cloud Computing," in 2019 4th International Conference on Information Systems and Computer Networks, ISCON 2019, 2019, pp. 572-577, doi: 10.1109/ISCON47742.2019.9036309.

[2] A. S. M. Al-Obaidi, A. Al-Qassar, A. R. Nasser, A. Alkhayyat, A. J. Humaidi, and I. K. Ibraheem, "Embedded design and implementation of mobile robot for surveillance applications," Indonesian Journal of Science and Technology, Article vol. 6, no. 2, pp. 427-440, 2021, doi: 10.17509/ijost.v2i2.

[3] L. Hasanah, W. L. Hakim, A. Aminudin, S. K. Sahari, and B. Mulyanti, "A design and performance analysis of a telemetry system for remote monitoring of turbidity of water during the covid-19 pandemic," Indonesian Journal of Science and Technology, Article vol. 5, no. 2, pp. 299-307, 2020, doi: 10.17509/ijost.v5i2.24705.

[4] Saripudin, A. Djohar, D. Rohendi, and A. G. Abdullah, "Developing information technology in opencourseware: From movements to opportunities in Asia," Indonesian Journal of Science and Technology, Article vol. 5, no. 3, pp. 308-320, 2020, doi: 10.17509/ijost.v5i3.24886.

[5] K. Stanoevska-Slabeva and T. Wozniak, "Cloud basics-An introduction to cloud computing," in Grid and Cloud Computing: A Business Perspective on Technology and Applications, 2010, pp. 47-61.

[6] S. M. Shetty and S. Shetty, "Analysis of load-balancing in cloud data centers," Journal of Ambient Intelligence and Humanized Computing, pp. 1-9, 2019, doi: 10.1007/s12652-018-1106-7.

[7] N. S. M. Pakhrudin, M. Kassim, and A. Idris, "A review on orchestration distributed systems for IoT smart services in fog computing," International Journal of Electrical and Computer Engineering, vol. 11, no. 2, p. 1812, 2021. [Online]. Available: 10.11591/ijece.v11i2.pp1812-1822.

[8] V. Andrikopoulos, T. Binz, F. Leymann, and S. Strauch, "How to adapt applications for the Cloud environment: Challenges and solutions in migrating applications to the Cloud," Computing, Article vol. 95, no. 6, pp. 493-535, 2013, doi: 10.1007/s00607-012-0248-2.

[9] T. Diaby and B. B. Rad, "Cloud Computing: A review of the Concepts and Deployment Models," International Journal of Information Technology and Computer Science, vol. 9, pp. 50-58, 2017, doi: 10.5815/IJITCS.2017.06.07.

[10] A. Mishra and D. Tiwari, "A Proficient Load-balancing Using Priority Algorithm in Cloud Computing," in Proceedings of the 2020 IEEE International Conference on Machine Learning and Applied Network Technologies, ICMLANT 2020, 2020, doi: 10.1109/ICMLANT50963.2020.9355972.

[11] M. Hamdani, Y. Aklouf, and H. Chaalal, "A Comparative Study on Load-balancing Algorithms in Cloud Environment," in ACM International Conference Proceeding Series, 2020, doi: 10.1145/3447568.3448466.

[12] V. S. Handur, S. Belkar, S. Deshpande, and P. R. Marakumbi, "Study of load-balancing algorithms for Cloud Computing," in Proceedings of the 2nd International Conference on Green Computing and Internet of Things, ICGCIoT 2018, 2018, pp. 173-176, doi: 10.1109/ICGCIoT.2018.8753091.

[13] A. Singh and R. Kumar, "Performance evaluation of load-balancing algorithms using cloud analyst," in Proceedings of the Confluence 2020 - 10th International Conference on Cloud Computing, Data Science and Engineering, 2020, pp. 156-162, doi: 10.1109/Confluence47617.2020.9058017.

[14] H. Rai, S. K. Ojha, and A. Nazarov, "Cloud Load-balancing Algorithm," in Proceedings - IEEE 2020 2nd International Conference on Advances in Computing, Communication Control and Networking, ICACCCN 2020, 2020, pp. 861-865, doi: 10.1109/ICACCCN51052.2020.9362810.

[15] A. Markandey, P. Dhamdhere, and Y. Gajmal, "Data access security in cloud computing: A review," in 2018 International Conference on Computing, Power and Communication Technologies, GUCON 2018, 2019, pp. 633-636, doi: 10.1109/GUCON.2018.8675033.

[16] I. N. Falisha, T. W. Purboyo, and R. Latuconsina, "Experimental Model for Load-balancing in Cloud Computing Using Equally Spread Current Execution Load Algorithm," International Journal of Applied Engineering Research, vol. 13, no. 2, pp. 1134-1138, 2018, doi: 10.37622/000000.

[17] K. Kaur and R. Mahajan, "Equally spread current execution load algorithm-a novel approach for improving data centre's performance in cloud computing," International Journal on Future Revolution in Computer Science & Communication Engineering, vol. 4, no. 8, pp. 08–10-08–10, 2018.

[18] C. Xue, "Method and implementation of server load-balancing in cloud computing," in Proceedings - 2018 3rd International Conference on Mechanical, Control and Computer Engineering, ICMCCE 2018, 2018, pp. 511-513, doi: 10.1109/ICMCCE.2018.00113.

[19] F. Alhaidari and T. Z. Balharith, "Enhanced Round-Robin Algorithm in the Cloud Computing Environment for Optimal Task Scheduling," Computers, vol. 10, no. 5, p. 63, 2021. doi: https://doi.org/10.3390/computers10050063.

[20] S. Kumar and A. Dumka, "Load-balancing with the Help of Round Robin and Shortest Job First Scheduling Algorithm in Cloud Computing," in Proceedings of International Conference on Machine Intelligence and Data Science Applications, Singapore, M. Prateek, T. P. Singh, T. Choudhury, H. M. Pandey, and N. Gia Nhu, Eds., 2021// 2021: Springer Singapore, pp. 213-223. [Online]. Available: https://doi.org/10.1007/978-981-33-4087-9_19. [Online]. Available: https://doi.org/10.1007/978-981-33-4087-9_19.

[21] S. I. Suliman et al., "An effective energy-efficient virtual machine placement using clonal selection algorithm," International Journal of Advanced Technology and Engineering Exploration, vol. 8, no. 75, p. 412, 2021. doi: 10.19101/IJATEE.2020.762129.

[22] S. Kaur and T. Sharma, "Efficient load-balancing using improved central load-balancing technique," in 2018 2nd International Conference on Inventive Systems and Control (ICISC), 19-20 Jan. 2018 2018, pp. 1-5, doi: 10.1109/icisc.2018.8398857.

[23] V. Joshi, "Load-balancing Algorithms in Cloud Computing," (in English), International Journal of Research in Engineering and Innovation, vol. 3, pp. 530 - 532, 2019-12-14 2019.

[24] T. Deepa and D. Cheelu, "A comparative study of static and dynamic load-balancing algorithms in cloud computing," in 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS), 1-2 Aug. 2017 2017, pp. 3375-3378, doi: 10.1109/icecds.2017.8390086. [Online]. Available: 10.1109/icecds.2017.8390086.

[25] H. Ye, J. Guo, and X. Li, "Delay-Aware and Profit-Maximizing Task Migration for the Cloudlet Federation," International Journal of Advanced Computer Science and Applications, Article vol. 13, no. 10, pp. 420-428, 2022, doi: 10.14569/ijacsa.2022.0131050.

[26] R. Kamran, A. A. El-Moursy, and A. Abdelsamea, "Efficient HPC and Energy-Aware Proactive Dynamic VM Consolidation in Cloud Computing," International Journal of Advanced Computer Science and Applications, Article vol. 13, no. 10, pp. 858-869, 2022, doi: 10.14569/ijacsa.2022.01310102.

[27] R. Alanazi, "Analysis of Privacy and Security Challenges in e-Health Clouds," International Journal of Advanced Computer Science and Applications, Article vol. 13, no. 9, pp. 484-489, 2022, doi: 10.14569/ijacsa.2022.0130955.

[28] Y. Qiu, B. Sun, Q. Dang, C. Du, and N. Li, "Fine-grained Access Control Method for Blockchain Data Sharing based on Cloud Platform Big Data," International Journal of Advanced Computer Science and Applications, Article vol. 13, no. 10, pp. 24-31, 2022, doi: 10.14569/ijacsa.2022.0131004.

[29] T. Saxena and V. Chourey, "A survey paper on cloud security issues and challenges," in Proceedings of the 2014 Conference on IT in Business, Industry and Government: An International Conference by CSI on Big Data, CSIBIG 2014, 2014, doi: 10.1109/CSIBIG.2014.7056957.

[30] M. Aziz, "Advanced green technologies toward future sustainable energy systems," Indonesian Journal of Science and Technology, Article vol. 4, no. 1, pp. 89-96, 2019, doi: 10.17509/ijost.v4i1.15805.

[31] M. Mesbahi and A. M. Rahmani, "Load-balancing in cloud computing: a state of the art survey," Int. J. Mod. Educ. Comput. Sci, vol. 8, no. 3, p. 64, 2016, doi: 10.5815/ijmecs.2016.03.08.

[32] T. Nakrani, D. Hiran, C. Sindhi, and M. I. Sandhi. Genetic Algorithm Based Task Scheduling for Load-balancing in Cloud, Lecture Notes on Data Engineering and Communications Technologies, vol. 52, pp. 283-293, 2021.

[33] S. Handur Vidya and R. Marakumbi Prakash, "Response time analysis of dynamic load-balancing algorithms in Cloud Computing," in Proceedings of the World Conference on Smart Trends in Systems, Security and Sustainability, WS4 2020, 2020, pp. 371-375, doi: 10.1109/WorldS450073.2020.9210305.

[34] A. Kaur, B. Kaur, and D. Singh, "Optimization techniques for resource provisioning and load-balancing in cloud environment: a review," International Journal of Information Engineering and Electronic Business, vol. 9, no. 1, p. 28, 2017, doi: 10.5815/ijieeb.2017.01.04.

[35] M. Hashemi and A. Masoud, "Load-balancing Algorithms in Cloud Computing Analysis and Performance Evaluation," IEEE, vol. 3, no. 4, 2020.

[36] A. Y. Ahmad and A. Y. Hammo, "A Comparative Study of the Performance of Load-balancing Algorithms Using Cloud Analyst," Webology, vol. 19, no. 1, 2022, doi: 10.14704/WEB/V19I1/WEB19328.

[37] S. Lamba and D. Kumar, "A comparative study on load-balancing algorithms with different service broker policies in cloud computing," International Journal of Computer Science and Information Technologies, vol. 5, no. 4, pp. 5671-5677, 2014.

[38] S. Mohapatra, K. S. Rekha, and S. Mohanty, "A comparison of four popular heuristics for load-balancing of virtual machines in cloud computing," International Journal of Computer Applications, vol. 68, no. 6, 2013, doi: 10.5120/11586-6922.

[39] J. Rathore, B. Keswani, and V. S. Rathore. Analysis of Load-balancing Algorithms Using Cloud Analyst, Advances in Intelligent Systems and Computing, vol. 841, pp. 291-298, 2019.

[40] B. Wickremasinghe, R. N. Calheiros, and R. Buyya, "CloudAnalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications," in Proceedings - International Conference on Advanced Information Networking and Applications, AINA, 2010, pp. 446-452, doi: 10.1109/AINA.2010.32.