Effect of Service Broker Policies and Load Balancing Algorithms on the Performance of Large Scale Internet Applications in Cloud Datacenters

Ali Meftah
College of Computer and Information Sciences
King Saud University
Riyadh, Saudi Arabia

Ahmed E. Youssef[1,2] and Mohammad Zakariah[1]
[1]College of Computer and Information Sciences
King Saud University, Riyadh, Saudi Arabia
[2]Faculty of Engineering, Helwan University, Cairo, Egypt

***Abstract*—Cloud computing is advancing rapidly. With such advancement, it has become possible to develop and host large scale distributed applications on the Internet more economically and more flexibly. However, the geographical distribution of user bases, the available Internet infrastructure within those geographical areas, and the dynamic nature of usage patterns of the user bases are critical factors that affect the performance of these applications. Therefore, it is necessary to compromise between datacenters, service broker policies, and load balancing algorithms to optimize the performance of the application and the cost to the owners. This paper aims at studying the effect of service broker policies and load balancing algorithms on the performance of large-scale Internet applications under different configurations of datacenters. To achieve this goal, we modeled the behavior of the popular Facebook application with the most recent worldwide users' statistics. Then, we evaluated the performance of this application under different configurations of datacenters using: 1) two different service broker policies, namely, closest datacenter and optimum response time; and 2) three load-balancing algorithms, namely, round robin, equally spread current execution, and throttled load balancer. The overall average response time of the application and the overall average time spent for processing a user request by a datacenter are measured and the results are discussed. This study would help service providers generate valuable insights on coordination between datacenters, service policies, and load balancing algorithms when designing Cloud infrastructure services in geographically distributed areas. In addition, application designers would benefit greatly from this study in identifying the optimal arrangement for their applications.**

Keywords*—Cloud computing; datacenters; load balancing algorithms; service broker policies; CloudAnalyst

I. Introduction

Cloud computing (CC) has become a prevalent technology in recent years. It provides a flexible and straightforward approach for maintaining and recovering information. Furthermore, it facilitates the collection of extensive information and the dissemination of records to various clients around the globe. Dealing with these vast information collections requires several strategies to enhance and streamline operations and provide attractive levels of execution to clients. CC incorporates computational and capacity benefits through a pay-per-use business model. Thus, it is exceptionally desirable to business holders as it eliminates the provisioning overhead and enables organizations to begin very small and extend their assets only when needed [1], [2]. It is likely the main innovation that totally supplements the web, "cloud computing alludes to registering on the Internet, instead of processing on a desktop" [3].

With the advancement of the Cloud, it is now possible to build and host large scale applications such as social networking sites and e-commerce on the Internet more economically and more flexibly. Cloud Service Providers (CSP) are willing to provide large scaled computing infrastructure at a cheaper price (i.e., on pay per use basis) and in a very flexible manner (i.e., the users can scale up or down at will). However, several issues must be addressed to optimize the performance of applications such as the geographic distribution of the user bases, the available Internet infrastructure within those geographic areas, the dynamic nature of the usage patterns of the user base and how well the cloud services can adapt itself [21].

In practice, cloud computing clients request specific services and require that their demands be fulfilled ahead of schedule at limited costs. As shown in Fig. 1, the traffic routing between user bases (UB) and datacenters is controlled by a service broker that decides which datacenter should provide the service to the requests coming from each user base. Thus, the service broker controls traffic routing between user bases and datacenters. The load balancing algorithm determines which VM should be assigned to the next client request for processing according to different policies. For example, the round robin algorithm processes in a circular order by handling the process without priority, but equally spread current execution algorithm processes using priority. With the throttled algorithm, the client first requests the load balancer to find a suitable VM to perform the required operation. VMs should be assigned in a way that guarantees low response time and minimum transfer delay. Accordingly, service broker polices, load balance algorithms, and datacenters configuration are critical factors that influence the performance of applications.

This paper aims at studying the effect of service broker policies and load balancing algorithms on the performance of distributed large scale applications in cloud computing environments.

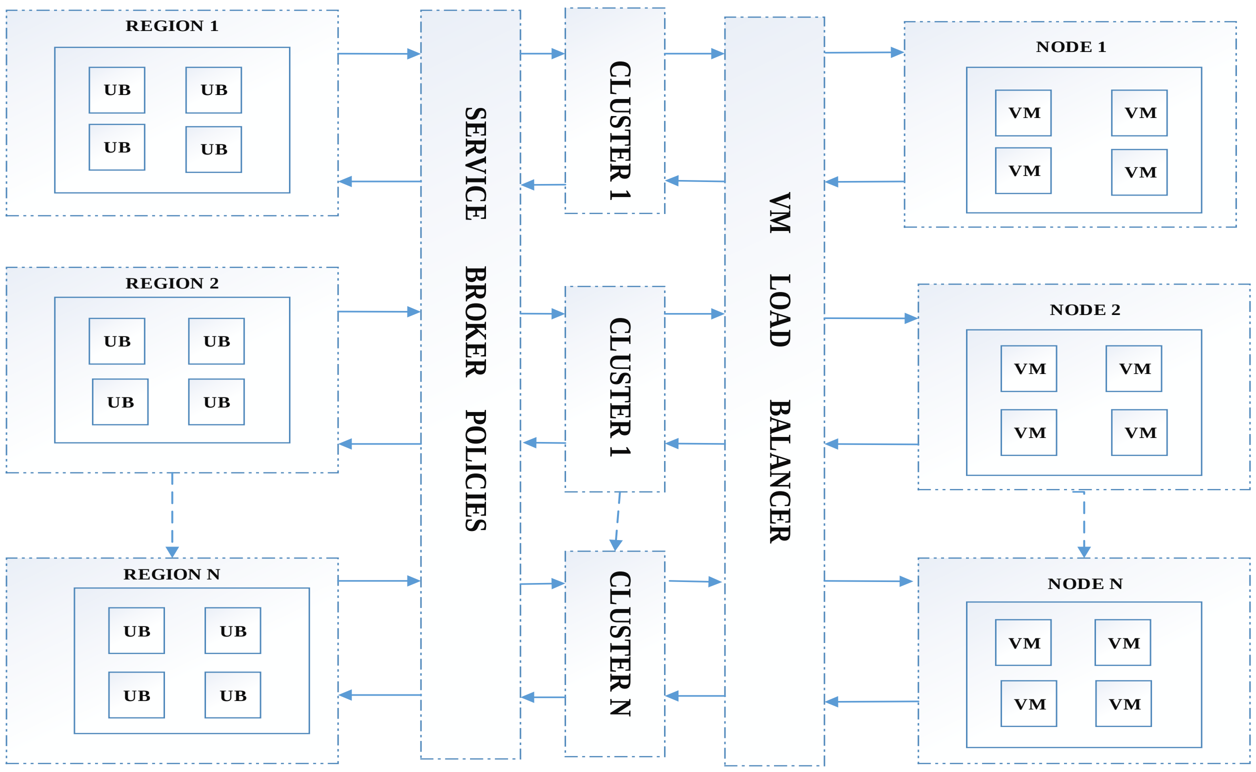

Fig. 1. Example cloud computing architecture [4].

To achieve this goal, we simulated the behavior of the popular Facebook application with the most recent worldwide users' statistics using CloudAnalyst tool. We examined the performance of this application in different scenarios (i.e., with different configurations of datacenters) and using: 1) two different service broker policies; namely, closest datacenter and optimum response time; and 2) three load balancing algorithms, namely, round robin, equally spread current execution, and throttled load balancer. The overall average response time of the application and the overall average time spent for processing a user request by a datacenter are recorded and the results are discussed.

This study helps CSP generate valuable insights on coordination between datacenters, service broker policies, and load balancing algorithms when designing Cloud infrastructure services in geographically distributed areas to optimize the application performance and the cost to the owners. In addition, application designers may use this study in identifying the optimal arrangement for their applications.

The rest of this paper is organized as follows: Section 2 provides background information on cloud computing and the most popular service broker policies and load balance algorithms; Section 3 describes the CloudAnalyst simulation tool; Section 4 shows and discusses our experimental results; Section 5 lists some related work; and finally, in Section 6, we give our conclusion and future work.

II. Essential Concepts

In this section, we review cloud computing and the most popular service broker policies and load balancing algorithms.

A. Cloud Computing

A cloud is a collection of IT resources, including hardware and software, deployed in a datacenter. The data center is built, operated, and managed by a Cloud Service Provider (CSP) which is an organization that provides cloud services. The provider may be an external (i.e., off-premise) provider to the consumer organization such as Amazon and Microsoft, or internal to the consumer organization (i.e., on premise), for example, the IT department. Cloud computing service models enable consumers to conveniently provision IT assets from a CSP in a way similar to a utility service such as electricity, wherein a consumer simply plugs in an electrical appliance to a socket, turns it on, and only pays for the amount of electricity used. The services provided by the cloud ranged from network-accessible data storage and processing, software development and deployment tools, to fully-featured software applications. CC paradigm brings uncountable benefits to users such as high scalability, rapid elasticity, and excellent availability of computing resources.

The National Institute of Standards and Technology (NIST) [23] defines different cloud service models. The first type of cloud service model is called Infrastructure-as-a-Service (IaaS) where the capability provided by CSP to the consumer is to provision processing, storage and networks. The underlying infrastructure (hardware) is managed solely by CSP. However, the consumer has control over operating systems and is able to run and deploy software applications. The second type of service model is called Platform-as-a-Service (PaaS) where the consumer is able to develop and deploy onto the cloud infrastructure applications using programming languages, libraries, services, and tools provided by CSP. The consumer does not manage or control the underlying cloud infrastructure (i.e., network, storage, compute system) or operating systems, but has control over the deployed applications. The last cloud service model is called Software-as-a-Service (SaaS) where the capability provided to the consumer is to use the CSP's applications running on a cloud infrastructure. The applications are accessible from various client devices through either a thin client interface, such as a web browser (for example, web-based email), or a program interface. The consumer does not manage or control the underlying cloud infrastructure, operating systems, or even individual application capabilities.

B. Service Broker Policies

Service brokers handle traffic routing between user bases and datacenters by employing different policies such as:

Closest datacenter: The default routing policy routes traffic to the closest datacenter in terms of network latency from the source user base. This policy utilizes the concept of region proximity in selecting the datacenter to which the user request has to be directed. A region proximity list is maintained using the "lowest network latency first" criterion to set the order of occurrence of datacenters in the list. The datacenter that occurs first in the list, i.e., the closest data center, is selected to fulfill the request using this policy. In cases where more than one datacenter with the same latency are available, a random selection of the datacenters is made. This policy is, therefore, beneficial in cases where the request can be satisfied by a datacenter that is quite close or within the same region.

Optimal response time policy: This policy calls for the service broker to first identify the closest datacenter by making use of the network latency parameter, as in the previous policy. Then, the current response time is estimated for each datacenter. If the estimated response time is the one for the closest datacenter, then the closest datacenter is selected. Otherwise, the closest datacenter or the datacenter with the lowest response time is selected with a 50:50 chance.

C. Load Balancing Algorithms

Load balancing algorithms distribute the workload among server hubs within a datacenter as shown in Fig. 2. The primary difference between load balancing algorithms lies in the manner in which they choose the server hubs and direct new demands to those particular hubs. In fact, load balancing consists of distributing the workload to individual centers that may have fewer loads than others. Load balancing is applied to enable successful utilization of assets, to improve the reaction time of the assignment, and to eliminate situations in which some VMs are heavily loaded while others are only marginally loaded [5], [6]. Load balancing strategies are utilized by different server farms to adjust the workload among available VMs. Currently, existing load balancing algorithms include the Round Robin (RR) algorithm, equally spread current execution algorithm and the throttled algorithm.

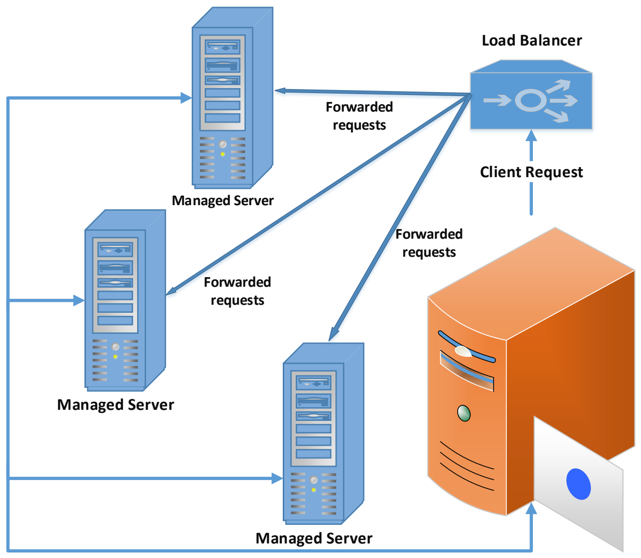

Fig. 2. Load balancing scenario [26].

Round Robin (RR) Algorithm: The RR algorithm, shown in Fig. 3, is one of the conventionally utilized algorithms that arbitrarily selects VMs. In the RR approach, time cuts are allocated to each assignment in a roundabout manner. Each undertaking is apportioned to available VMs in a roundabout request. However, there may be instances in which a few hubs are significantly loaded while others are only somewhat loaded. Thus, there are instances where the framework stack becomes irregular [7]. The RR algorithm is also a clear and static planning system that uses the guideline of time cuts, in which time is divided into various interims, and each VM is given a particular time cut or time interim [8], [9].

Equally spread current execution algorithm: The name of this algorithm suggests that it operates by equally spreading the execution load across different VMs [10]. It distributes the load randomly by first checking the size of the process and transferring the load to VMs that are only lightly loaded or which can handle the task easily in a small amount of time while maximizing throughput. This algorithm is a dynamic load balancing algorithm that determines the priority by checking the size of the process. It requires a load balancer that monitors the jobs to be executed.

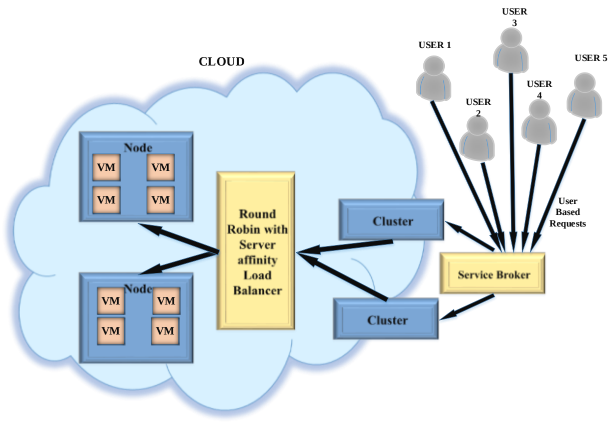

Fig. 3. Round robin load balancing [24].

The task of the load balancer is to queue the jobs and hand them to different VMs. The load balancer analyzes the queue frequently for new jobs and then allots them to the list of free virtual servers. The load balancer also maintains a list of tasks allotted to the virtual servers, which helps them to identify which VMs are free and need to be allotted new jobs.

Throttled Algorithm: The throttled algorithm, shown in Fig. 4, begins by allocating suitable VMs when the client sends a request to the load balancer. The load balancer maintains a file table of all the VMs together with their states, occupied or open mode. At the beginning, every VM is set to accessible mode. Then, the server controller guides the balancer for the next VM allotment, before it gets another demand. The balancer checks the table altogether until an important match of the VM was found. On the off chance that the ideal VM is discovered, then the balancer returns the ID of that particular VM to the server controller. Immediately, the server controller sends a demand to the VM with the specified ID. From that point onwards, the server controller sends a warning to balance the new distribution with the goal of refreshing the table. On the off chance there exists a case, when the VM is present, the balancer returns with that server information. When the VM is finished handling the doled out demand, the server controller gets a reaction cloudlet and it sends a message to the load balancer for VM de-assignment [11], [12].

III. CloudAnalyst Simulator

CloudAnalyst [12], [21] is a GUI-simulator developed to study the behavior of large scaled Internet applications in CC environment. Using CloudAnalyst, application developers or designers can determine the best strategies for selecting datacenters to serve specific requests, allocating resources among available datacenters, and the costs related to such operations. It is an open source simulation tool [19] that enables the recreation and assessment of the execution of different cloud administrations.

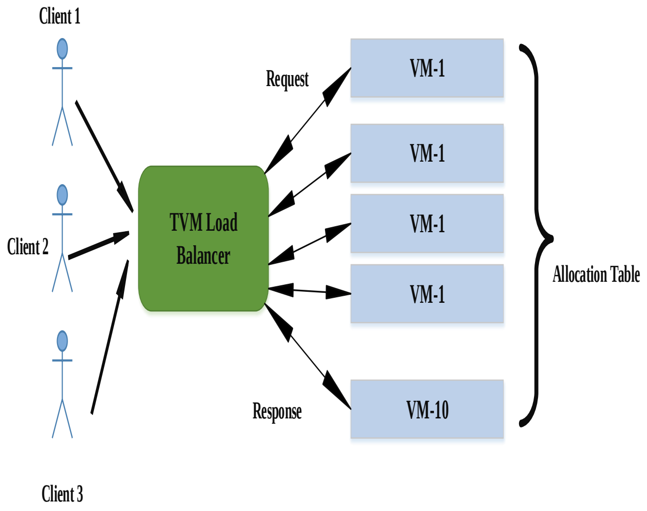

Fig. 4. Throttled algorithm load balancing [25].

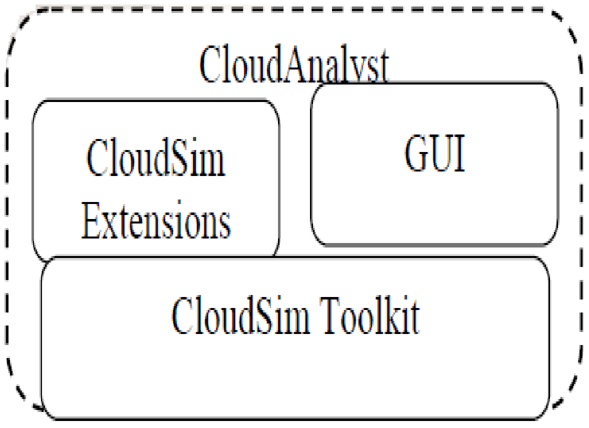

Fig. 5. CloudAnalyst architecture [12].

The CloudAnalyst is built on top of CloudSim tool kit [20], as shown in Fig. 5, by extending CloudSim functionality with the introduction of concepts that model Internet and Internet Application behaviors. It separates the simulation experimentations from programming tasks to enable users to quickly set up simulations and summarize results in useful formats. Different measures can be produced as an output of CloudAnalyst, such as, response time of the simulated application, usage patterns of the application, time taken by datacenters to service a user request, and the cost of operation.

IV. Experiments

The purpose of our experiments.is to show how different service broker policies, load balancing algorithms, and datacenter settings affect the performance of Internet large scale applications hosted on cloud datacenters. Social network services are one of the most popular Internet applications that vary with geographic location, sources of service requests, and time of day. They are large scaled applications that can benefit from cloud technology because they typically present non-uniform usage patterns [12]. One popular social network application today is Facebook, which has over 1.67 billion registered users around the world [22]. Table I shows Facebook usage and Facebook growth statistics by world geographic regions as of 30 June 2016. In our experiments, CloudAnalyst simulation tool was used to model and analyze the behavior of the Facebook application with the most recent worldwide users' statistical distribution.

A. Experiments Setup

We characterized six user bases representing the six primary areas of the world with the parameters depicted in Table II. For our simulation, we utilized a comparable speculative application with size equals 1/10 the size of Facebook. For simplicity, every client base was contained inside a solitary time zone and it was assumed that most clients utilized the application at night after working for approximately 2 hours. It was also assumed that 1% of the enrolled clients are web-based during peak hours at the same time and only a single tenth of that number of clients is online during off-peak hours. Moreover, every client makes another demand every 5 minutes when he or she is on the web. Table III shows the diverse CloudAnalyst parameter values used in our experiments.

TABLE I. FACEBOOK SUBSCRIBERS AND WORLD POPULATION STATISTICS AT JUNE 30, 2016 – UPDATE [22]

World Regions	Population (2016 Est.)	FACEBOOK users (as at 30 June 2016)
Africa	1,185,529,578	146,637,000
Asia	4,052,652,889	559,003,000
Europe	832,073,224	328,273,740
Latin America / Caribbean	626,054,392	326,975,340
Middle East	246,700,900	76,000,000
North America	359,492,293	223,081,200
Oceania / Australia	37,590,820	19,463,250
WORLD TOTAL	**7,340,094,096**	**1,679,433,530**

TABLE II. USER BASES USED IN THE EXPERIMENTS

User Base	Region	Time Zone	Peak Hours (Local time)	Peak Hours (GMT)	Simultaneous Online Users During Peak Hours	Simultaneous Online Users During Off-peak Hours
N. America	0	GMT - 6:00	7:00–9:00 pm	13:00–15:00	2230812	223081
S. America	1	GMT - 4:00	7:00–9:00 pm	15:00–17:00	3269753	326975
Europe	2	GMT + 1:00	7:00–9:00 pm	20:00–22:00	3282737	328274
Asia	3	GMT + 6:00	7:00–9:00 pm	01:00–03:00	5590030	559003
Africa	4	GMT + 2:00	7:00–9:00 pm	21:00–23:00	1466370	146637
Oceania/Australia	5	GMT + 10:00	7:00–9:00 pm	09:00–11:00	194633	19463

TABLE III. CLOUDANALYST PARAMETER VALUES USED IN THE EXPERIMENTS

Parameters		Assigned Value
Simulation Duration		60 Min
Virtual Machine (VM)	No. of VMs	Dependent on scenario
	Image size	10,000
	Memory	512 MB
	Bandwidth	1000 MB
Datacenter	Region	0
	Architecture	x86
	Operating system	Linux
	Virtual Machine Monitor (VMM)	Xen
	Memory per machine	2 GB
	Storage per machine	100 TB
	Available bandwidth per machine	1000000 MB
	Number of processors	4
	Processor Speed	10000 MB
	VM Policy	Time shared
User grouping factor in user bases		1000
Request grouping factor in datacenters		100
Executable instruction length		250

B. Simulation Senarios

We used the CloudAnalyst simulation tool to evaluate the performance of Facebook application described above in six different scenarios using: 1) two different service brokers policies; namely, closest datacenter (CDC) and optimum response time (ORT) and 2) three load balancing algorithms; namely; round robin, equally spread current execution, and throttled load balancer. Each scenario represents a different configuration for the datacenters. These scenarios are shown in Table IV.

TABLE IV. SIMULATED SCENARIOS

Scenario ID	Datacenter Settings
scenario1	One datacenter with 50 VMs, located at location 0
scenario2	Two datacenters with 25 VMs each, both located at location 0
scenario3	Three datacenters with 50, 75, and 100 VMs respectively, all located at location 0
scenario4	Three datacenters with 50, 75, 100 VMs, located at three different locations: 0, 1, and 2
scenario5	Six datacenters with 50 VMs, located at different locations: 0, 1, 2, 3, 4, and 5
scenario6	Six datacenters with 25, 25, 50, 50, 75, 100 VMs, located at locations: 0, 1, 2, 3, 4, and 5

In the first scenario, a single datacenter containing 50 VMs was employed to process all user requests from around the world. In scenario2, we added a second datacenter and reduced the number of VMs to 25 for each datacenter. In the third scenario, three datacenters with different numbers of VMs (50, 75, and 100) were used. In all these scenarios, the geographical location of the datacenter was location 0 (i.e., North America). The fourth scenario was the same as third one except that the datacenters were distributed over three locations, North America, South America, and Europe. In the 5th and 6th scenarios, we used six datacenters distributed over the six locations described in Table II. All datacenters in scenario5 have the same size (50 VM), while in scenario6, datacenters have sizes 25,25,50,50,75,100VM respectively.

C. Results and Discussion

During simulation of each scenario, readings during the 24 hours of the day for the response time of the application and the time spent for processing a user request by each datacenter are measured and the Min, Max, and overall average values are recorded as shown in Table V. A quick inspection to the results revealed that, in general, the throttled load balancing algorithm outperforms the other two algorithms since its recorded overall average response time (ART) and overall average processing time (APT) are the shortest in all scenarios, this is also shown in Fig. 6 and 7. Table VI shows ART and APT of the throttled load balancing algorithm in all scenarios.

Table VI and Fig. 8 clearly show that the best values for ART and APT are obtained in scenario3 with the CDC service broker policy and in scenario6 with the ORT service broker policy. However, the cost of scenario3 is much less than scenario6 as can be noticed from Table IV. Finally, Fig. 9 shows a snapshot of the simulation results.

V. Related Work

Various studies have been conducted on load balancing in the cloud computing environment. Randles et. al. [13] investigated the operation of three load balancing algorithms and discussed their disadvantages. They proposed arrangements for load adjusting. Khiyaita et. al. [14] reviewed cloud computing in depth and grouped load balancers in terms of their execution in a commonly circulated framework. They also discussed virtualization and examined the different interfaces in detail. Smith and Nayar [15] addressed the reasons for cloud appropriation and discussed how cloud computing helps different endeavors. Nandgaonkar and Raut [16] focused on the critical documentation required in distributed computing, administration from cloud suppliers, and operations in the cloud.

Padhy [11] assessed several outstanding current load adjusting algorithms that can be utilized. Jadeja and Modi [17] explored the building outline of distributed computing, its advantages, and a few drawbacks, for example, security, protection, and genuineness. Wickremasinghe et. al. [12] discussed the operation of a GUI-based apparatus called CloudAnalyst that can be utilized for concentrating and executing gigantic scaled web applications. Finally, Mohialdeen [18] discussed various booking strategies, reviewed various scheduling algorithms in distributed computing, and discussed their application in cloud computing.

VI. Conclusions and Future Work

This paper has studied the effect of service broker policies and load balancing algorithms on the performance of large scale applications in cloud computing environments. In order to accomplish this study, we have created different operation scenarios under different settings of datacenters and using two services broker policies; namely, closest datacenter and optimum response time and three load balancing algorithms; namely; round robin, equally spread current execution, and throttled load balancer. In our experiments, we modeled and analyzed the behavior of the popular Facebook application and evaluated its performance in each scenario using the CloudAnalyst simulation tools. The overall average response time of the application (ART) and the overall average time spent for processing a user request by a datacenter (APT) are measured. The results showed that the throttled load balancing algorithm outperforms the other two algorithms since its recorded readings (i.e., ART and APT) are the shortest.

This study would benefit CSP generate valuable insights on coordination between datacenters, service brokers policies, and load balancing algorithms when designing Cloud infrastructure services in geographically distributed areas to optimize the application performance and the cost to the owners. In addition, application designers may use this study to identify the optimal configurations for their applications. In the future, we plan to extent this work to include more parameters that may impact applications' performance and to examine other types of large scale applications.

TABLE V. Experimental Results

			Response time of the application (ms)			**Time spent in processing a request by a datacenter (ms)**		
			Min	**Max**	**Average**	**Min**	**Max**	**Average**
Scenario 1	Closest Datacenter (CDC)	Round Robin	73.34	5398.37	1598	0.81	4770.27	1235.31
		Equally Spread	68	5396.20	1601.02	081	4771.94	1238.33
		Throttled	55.55	5279.04	**970.86**	0.81	4647.32	**620.41**
	Optimum Response Time (ORT)	Round Robin	71.25	5399.99	1597.85	1.47	4771.93	1235.05
		Equally Spread	72.13	5398.33	1601.43	1.47	4773.17	1238.65
		Throttled	55.46	5277.15	**970.77**	1.47	4660.90	**620.27**
Scenario 2	Closest Datacenter (CDC)	Round Robin	66.88	4450.79	1096.92	0.39	3851.89	740.94
		Equally Spread	59.77	4068.73	1096.53	0.26	3479.98	740.63
		Throttled	52.12	4295.62	**730.02**	0.82	3665.02	**377.56**
	Optimum Response Time (ORT)	Round Robin	58.66	4223.89	1277.05	0.41	3634.81	919.31
		Equally Spread	58.49	4487.76	1258.84	0.63	3909.48	901.27
		Throttled	53.27	4302.81	**816.06**	0.51	3686.20	**463.70**
Scenario 3	Closest Datacenter (CDC)	Round Robin	63.96	3038.29	785.29	0.96	2451.74	428.97
		Equally Spread	64.84	2150.56	716.48	0.95	1566.29	354.80
		Throttled	52.82	2821.96	**577.52**	0.78	2200.35	**221.73**
	Optimum Response Time (ORT)	Round Robin	58.21	3362.37	943.18	0.76	2762.19	587.08
		Equally Spread	64.64	3062.81	931.59	0.61	2502.19	575.49
		Throttled	52.15	3322.85	**647.49**	0.63	2687.67	**292.63**
Scenario 4	Closest Datacenter (CDC)	Round Robin	68.43	6507.21	1992.83	0.39	6111.12	1803.72
		Equally Spread	68.43	6605.26	1993.34	0.39	6209.48	1804.23
		Throttled	51.05	5364	**1046.84**	1.56	4973.69	**868.30**
	Optimum Response Time (ORT)	Round Robin	70.16	6561.78	1405.88	1.19	6159.04	1151.80
		Equally Spread	72.58	5460.62	1345.82	1.49	5044.94	1088.67

		Throttled	46.51	5370.93	**778.58**	1.81	4973.84	**531.64**
Scenario 5	Closest Datacenter (CDC)	Round Robin	47.71	2215.77	1209.78	1.48	2125.94	1141.38
		Equally Spread	47.71	2215.77	1209.81	1.48	2125.94	1141.40
		Throttled	47.71	2190.40	**640.11**	1.48	2087.26	**576.26**
	Optimum Response Time (ORT)	Round Robin	48.22	2129.41	842.68	0.95	1770.83	622.63
		Equally Spread	48.22	2214.30	835.95	0.67	1771.83	618.39
		Throttled	48.22	1976.28	**520.21**	0.95	1736.17	**306.42**
Scenario 6	Closest Datacenter (CDC)	Round Robin	48.49	2215.77	1194.19	2.26	2125.94	1125.31
		Equally Spread	48.49	2215.77	1194.24	2.26	2125.94	1125.35
		Throttled	48.49	2190.40	**638.49**	2.26	2087.26	**574.66**
	Optimum Response Time (ORT)	Round Robin	49.40	2215.75	825.95	1.25	1848.08	617.19
		Equally Spread	49.40	2324.31	818.39	1.25	1772.83	604.11
		Throttled	49.40	2070.27	**517.48**	1.42	1811.29	**306.11**

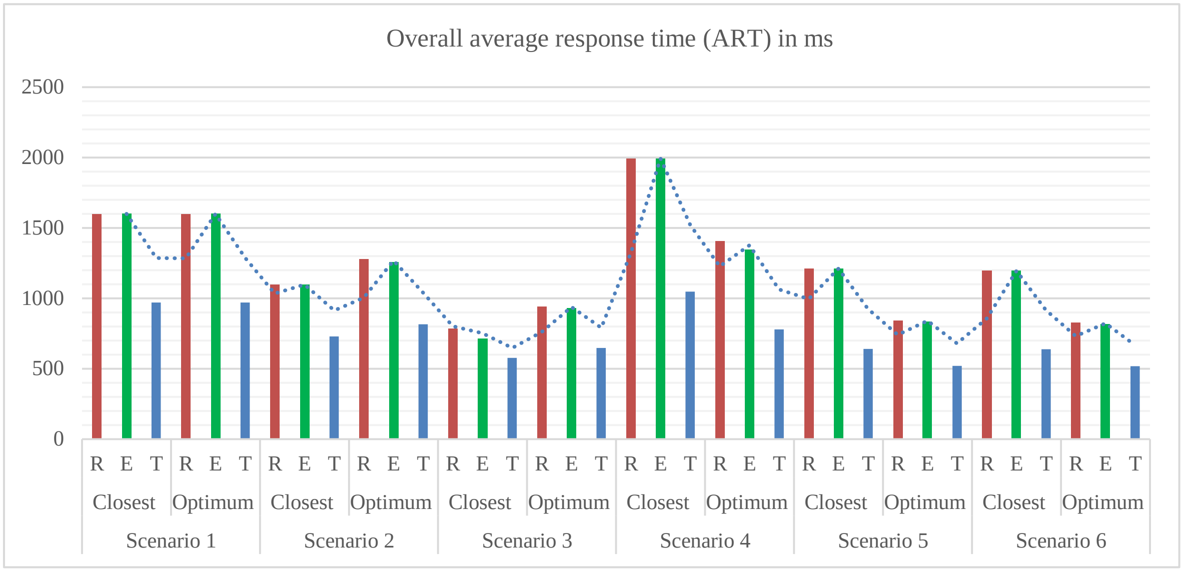

Fig. 6. Overall average response time (R: Round Robin algorithm, E: Equally Spread Current Execution algorithm and T: Throttled load balancing policy algorithm).

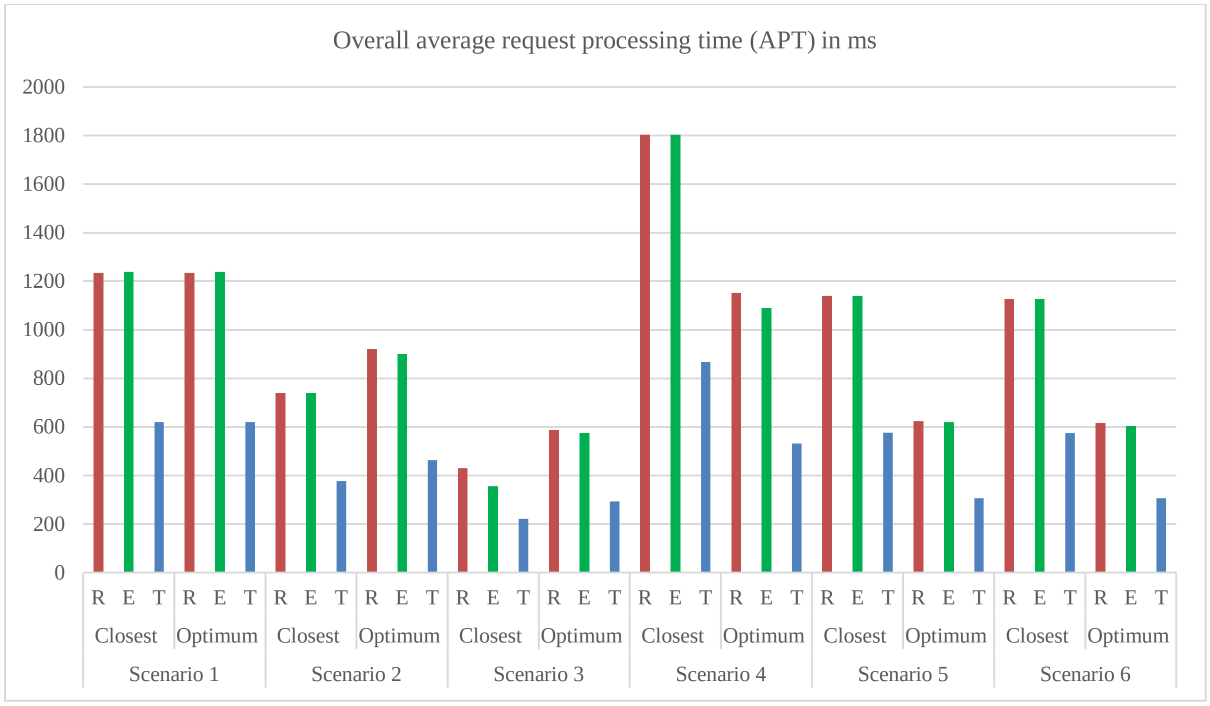

Fig. 7. Overall average request processing time (R: Round Robin algorithm, E: Equally Spread Current Execution algorithm, and T: Throttled load balancing policy algorithm).

TABLE VI. RESULTS FOR THE THROTTLED LOAD BALANCING ALGORITHM

Scenario ID	**Scenario1**		**Scenario2**		**Scenario3**		**Scenario4**		**Scenario5**		**Scenario6**	
Broker Policy	CDC	ORT	CDC	ORT	**CDC**	ORT	CDC	ORT	CDC	ORT	CDC	**ORT**
ART	970.86	970.77	730.02	816.06	**577.52**	647.49	1046.84	778.58	640.11	520.21	638.49	**517.48**
APT	620.41	620.27	377.56	463.7	**221.73**	292.63	868.3	531.64	576.26	306.42	574.66	**306.11**

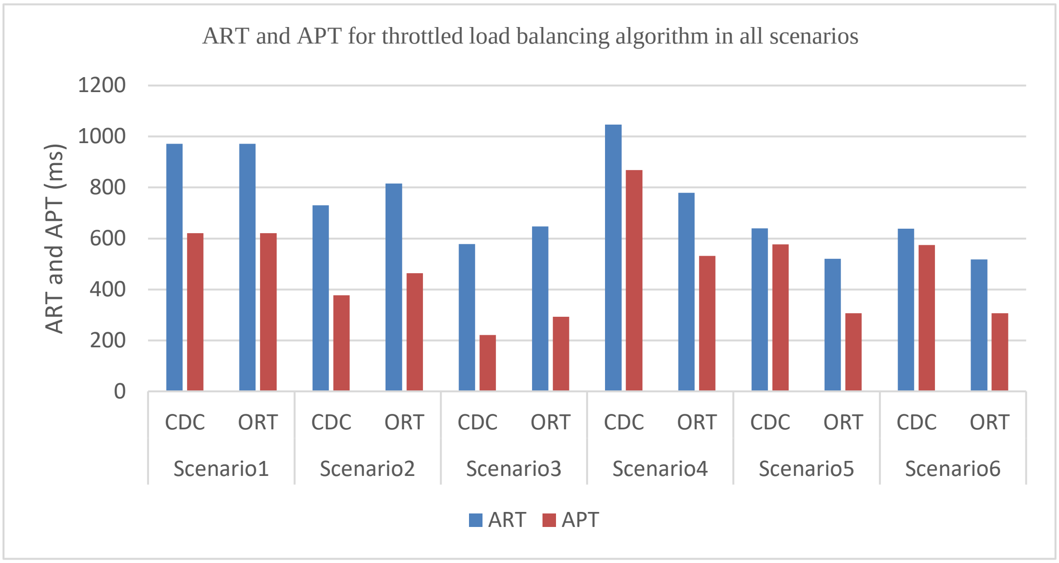

Fig. 8. ART and APT for throttled load balancing algorithm in all scenarios.

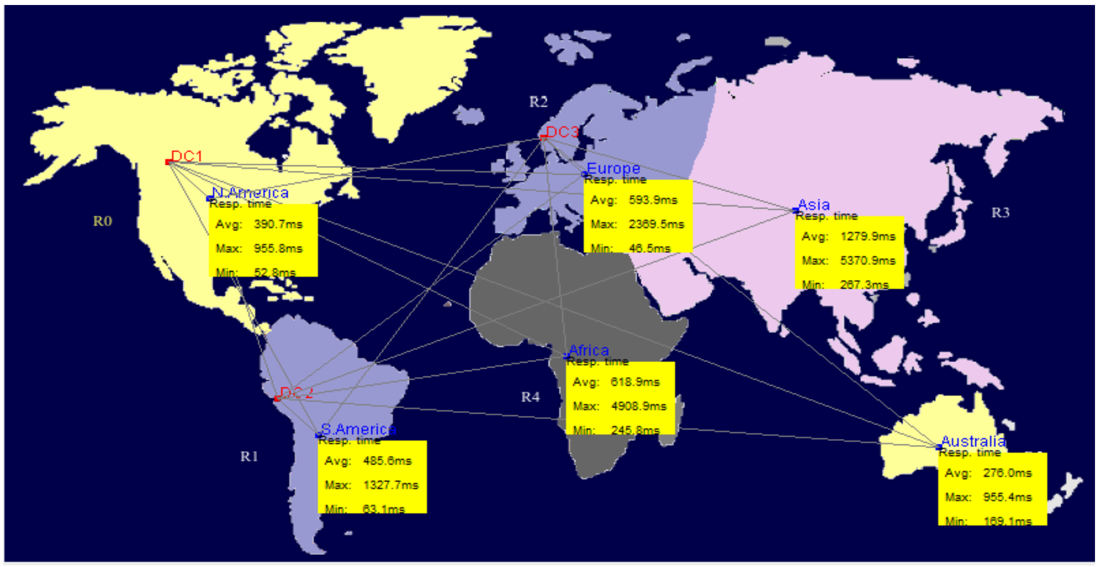

Fig. 9. Snapshot of simulation results.

ACKNOWLEDGMENT

This work is supported by the Research Center of College of Computer and Information Sciences (CRC) at King Saud University. The authors are grateful for this support.

REFERENCES

[1] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges", Journal of internet services and applications, vol. 1, pp. 7-18, 2010.

[2] M. D. Dikaiakos, D. Katsaros, P. Mehra, G. Pallis, and A. Vakali, "Cloud computing: Distributed internet computing for IT and scientific research," IEEE Internet Computing, vol. 13, 2009.

[3] D. Blacharski and C. Landis, Cloud Computing Made Easy: Cary Landis, 2010.

[4] S. Patel, R. Patel, H. Patel, and S. Vahora, "CloudAnalyst: "A Survey of Load Balancing Policies", International Journal of Computer Applications, vol. 117, 2015.

[5] Z. Chaczko, V. Mahadevan, S. Aslanzadeh, and C. Mcdermid, "Availability and load balancing in cloud computing," in International Conference on Computer and Software Modeling, Singapore, 2011.

[6] R. Lee and B. Jeng, "Load-balancing tactics in cloud," in Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2011 International Conference on, 2011, pp. 447-454.

[7] A. Singh, P. Goyal, and S. Batra, "An optimized round robin scheduling algorithm for CPU scheduling," International Journal on Computer Science and Engineering, vol. 2, pp. 2383-2385, 2010.

[8] D. Nusrat Pasha, A. Agarwal, and R. Rastogi, "Round Robin Approach for VM Load Balancing Algorithm in Cloud Computing Environment," International Journal, vol. 4, 2014.

[9] M. M. D. Shah, M. A. A. Kariyani, and M. D. L. Agrawal, "Allocation of virtual machines in cloud computing using load balancing algorithm," International Journal of Computer Science and Information Technology & Security (IJCSITS), vol. 3, pp. 2249-9555, 2013.

[10] M. Nitika, M. Shaveta, and M. G. Raj, "Comparative analysis of load balancing algorithms in cloud computing," International Journal of Advanced Research in Computer Engineering & Technology, vol. 1, pp. 120-124, 2012.

[11] R. P. Padhy, "Load balancing in cloud computing systems," National Institute of Technology, Rourkela, 2011.

[12] B. Wickremasinghe, R. N. Calheiros, and R. Buyya, "Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications," in Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on, 2010, pp. 446-452.

[13] M. Randles, D. Lamb, and A. Taleb-Bendiab, "A comparative study into distributed load balancing algorithms for cloud computing," in Advanced information Networking and applications Workshops (WAINA), 2010 IEEE 24th International Conference on, 2010, pp. 551-556.

[14] A. Khiyaita, H. El Bakkali, M. Zbakh, and D. El Kettani, "Load balancing cloud computing: State of art," in Network Security and Systems (JNS2), 2012 National Days of, 2012, pp. 106-109.

[15] J. E. Smith and R. Nayar, "Introduction to virtual Machines," Virtual machines: versatile platforms for systems and processes, pp. 9-10, 2004.

[16] S. V. Nandgaonkar and A. Raut, "A comprehensive study on cloud computing," International Journal of Computer Science and Mobile Computing, vol. 3, pp. 733-738, 2014.

[17] Y. Jadeja and K. Modi, "Cloud computing-concepts, architecture and challenges," in Computing, Electronics and Electrical Technologies (ICCEET), 2012 International Conference on, 2012, pp. 877-880.

[18] I. A. Mohialdeen, "Comparative study of scheduling al-gorithms in cloud computing environment," Journal of Computer Science, vol. 9, pp. 252-263, 2013.

[19] http://opensourceforu.com/2016/11/best-open-source-cloud-computing-simulators/

[20] R. N. Calheiros, R. Ranjan, C. A. De Rose, and R. Buyya, "Cloudsim: A novel framework for modeling and simulation of cloud computing infrastructures and services," The 38th International Conference on Parallel Processing (ICPP), Vienna, Austria, 2009.

[21] B. Wickremasinghe, “CloudAnalyst: A CloudSim-based Tool for Modelling and Analysis of Large Scale Cloud Computing Environments”, MEDC Project Report, distributed computing project, CSSE dept., University of Melbourne. 2009.

[22] http://www.internetworldstats.com/.

[23] http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

[24] Mahajan, Komal, A. Makroo, and D. Dahiya. "Round robin with server affinity: a VM load balancing algorithm for cloud based infrastructure," Journal of information processing systems 9, no. 3 (2013): 379-394.

[25] N. Pasha, A. Agrawal, R. Rastogi, “Round Robin Approach for VM Load Balancing Algorithm in Cloud Computing Environment”, IJARCSSE, Volume 4, pages 34-39 Issue 5, May 2014.

[26] https://www.codeproject.com/Articles/430701/Singleton-Pattern-Load-Balancer-Demonstration