An Efficient Distributed Dynamic Load Balancing Method based on Hybrid Approach in Cloud Computing

Sandeep Rai
Asst. Professor
Computer Science & Engg.
TIT (Excellence), Bhopal, India

Neha Sagar
M.Tech Scholar
Computer Science & Engg.
TIT (Excellence), Bhopal, India

Ram Sahu
Asst. Professor,
Computer Science & Engg.
TIT (Excellence), Bhopal, India

ABSTRACT
Cloud Computing shares data between cloud users over the internet and provides it to different resources. There are many challenges faced in cloud computing. The main one is Load balancing .Many researchers have been working on load balancing but still a lot of work has to be done to provide efficient & optimal load balancing method. In this dissertation we are proposing a new hybrid algorithm, which is based on distributed and dynamic load balancing. The basic idea behind the EDLBHA is to identify unused machines and resources earlier in shortest access time and also adding a new dynamic balancing parameter, which is based on distributed priority load distribution with dynamic load partitioning. The proposed EDLBHA method is compared with existing Load balancing algorithm with Cloud Simulator. Simulation results clearly shows that our proposed EDLBHA perform outstanding over existing algorithms in term of various performance measurement parameters such as waiting time, turnaround time and throughput

General Terms
Cloud computing, Load balancing, Algorithms.

Keywords
Cloud computing, Load balancing, Virtualization, Honey bee, distributed and dynamic load balancing, Resource utilization.

1. INTRODUCTION
Cloud computing is a connection of networks, which aims on sharing computations or resources (e.g., networks, servers, storage, applications, and services) and works on pay per use basis. It enables cloud users to access or store their data anytime and anywhere. It is based on virtualization concept. Virtualization is a method for creating virtual servers that run on a group of real servers. A Cloud system consists of 3 major components: (1) Clients- End users who interact with the clients to manage information related to the cloud [4]. (2) Data Center- It is a collection of servers hosting different applications. An end user connects to the datacenter to use different applications.[2] (3) Distributed Servers-They are the parts of a cloud which are present throughout the Internet hosting different applications

Load Balancing - It is a method in which the workload on the resource nodes is shifted to respective other resource node in a network without disturbing the running task.[5] Load balancing in cloud improves the performance, have a backup even in case the system fails partially to maintain the stability of system, to increase user satisfaction to improve resource utilization ratio. Types of Load balancing algorithms are:

(i) Based on the process, load balancing algorithms can be divided as (1) Sender Initiated: Load balancing algorithm initiated by sender side. (2) Receiver Initiated: Load balancing algorithm initiated by the receiver side. (3) Symmetric: It is the combination of sender initiated and receiver initiated algorithms.

(ii) Based on the current state of the system, load balancing algorithms can be divided into 2 types as (1) Static algorithm- It doesn't depend on the present state of the system. Prior knowledge of the system is needed. (2) Dynamic algorithm: Load balancing decisions are based on current status of the system. Here no prior knowledge is needed, so it is better than static approach.

2. LITERATURE SURVEY
Load balancing algorithms is one of the most challenging concerns in cloud computing. Many researches have been done in the field of Load balancing of cloud computing. Jobs are queued and collected into a set then scheduling algorithm will start after a fixed period of time.

Ms. Nitika et al [20] presented an FCFS (First come first serve) algorithm in which jobs are executed according to the order of their arriving. The job which comes first is executed first and the next job will be executed after that in its turn. The FCFS algorithm may have a ‘‘convoy effect’’ which happens when there is a job with a large amount of workload in the job queue. Then all the queued jobs behind it must wait for a long time for the longer jobs to end. The main drawback of FCFS is that it is non-preemptive. The shortest job at queue end have to wait for the lengthy jobs at the front to finish. Its response time is also low.

Subhadra Bose Shaw et al [3] presented Round Robin scheduling algorithm (RR) defines its queue in the ring form and also defines a fixed time quantum. Each job can be executed only within this quantum, and in its turn. If the job cannot be completed in single quantum, it will return back to the queue and wait for the next round. The main advantage of RR algorithm is that jobs are executed in their turn and do not have to wait for the previous job completion. Here the time is divided into multiple slots, each node is assigned a particular time quantum and the node will perform its execution in this quantum. It does not suffer from a starvation problem. However, if the job queue is fully loaded or workload is heavy, it will take a lot of time to complete all the jobs. It is static in nature. Furthermore, a suitable time quantum is difficult to decide.

Subhadra Bose Shaw et al [3] presented Honey Bee Colony optimization algorithm is derived from the honey bee behavior for finding and collecting food. This algorithm stimulates the foraging behavior of honey bees and has three phases. First, employee bees which stays on a food source and provides the neighborhood of the source, second onlooker bees which gets the information of food sources from the employee bees through waggle dance and select one of the food sources to collect food and third are scout bees which are translated from employee bees whose food has been exhausted and responsible for finding new food source.

3. PROBLEMS IN EXISTING METHODS

3.1 Job arrival patterns are not predictable- Jobs are arrived from various nodes in cloud environment, so it is quite difficult to identify exact arrival patterns.

3.2 Load balancing methods depend on type of algorithm whether it is static or dynamic.

3.3 Static Load Balancing-Existing load balancing methods are static which are less complex in comparison to dynamic schemes 1.1which brings extra expenses but are flexible and can be changed as the system status changes.

3.4 Performance parameters- Results of existing methods can be improved in terms of performance calculation parameters such as Response Time, Waiting Time, Turnaround time and Process Time.

3.5 Quality of service-Better QOS for cloud environment is also a big challenge.

3.6 Slow response time-In existing load balancing methods response time is slower which leads to poor system performance.

3.7 Priority is not assigned- In existing load balancing methods priority is not used in distribution of jobs/resources.

4. 4. PROPOSED METHODOLOGY

The proposed methodology EDLBHA is based on efficient loads balancing. The basic idea behind the EDLBHA is to identify unused machines and resources earlier in shortest access time and also adding a new dynamic balancing parameter, which is based on distributed priority load distribution with dynamic load partitioning. Random sampling redesigned the queue size, as per the load. In EDLBHA queue length and processing time are used as parameters. On arriving at the public cloud, first the right partition is selected. The cloud partition status can be divided into three types-

(1) Idle: if the percentage of idle nodes exceeds A, change status to idle.
(2) Normal: if the percentage of the normal nodes exceeds B, change status to normal.
(3) Overload: if the percentage of the overloaded nodes exceeds C, change status to overloaded .

The cloud partition balancers sets the parameters A, B, and C. The main controller frequently communicates the balancers to refresh the parameters status information. The proposed methodology dynamic distributed load balancing with random sampling (EDLBHA) use following steps-

Inputs**:** Number of server, number of Cloudlets, brokers, VM's, Data Centers, 'I' number of resources on data centers to be accessed by the cloud Access/burst time for each of the cloudlet Priority of each of the cloudlet.

Output**:** Improved turnaround time, process time, waiting time and performance

Step-1 Create Cloud environment

1.1: Cloud Server or Virtual Machines are created $(S_1....S_n)$.
1.2: Number of users or cloudlets are created $(U_1....U_n)$.
1.3: Cloud Broker B is created.
1.4: Data centers DC_i are created.
1.5: User send number of jobs $(J_1....Jn)$ having burst time and priority to server through Cloud broker.

Step-2 Define a load parameter

2.1 Set: $P = \{P_1;P_2,.. Pm\}$ with each node
2.2 Pi (1<=i<=m, Pi € [0 or 1]), parameter either static or dynamic. Where m represents the total number of the parameters

Step-3 Calculate capacity of the cloud system capacity

3.1. Assign priority for task parameter-$_{Thigh}$, $_{Tmed}$, and $_{Tlow}$ represents high, medium and low priority tasks respectively.
3.2. Assign instruction of task $_{Thigh,\ Tmed}$, and $_{Tlow}$ to $_{Ihigh}$, $_{Imedium}$, and $_{Ilow}$, respectively to a VM.
3.3. Calculate Capacity Cvm of a virtual machine VM.
Cvm = (Number of processors in VM) * (Number of instructions of all processors) * (communication bandwidth ability)
3.4. Calculate Capacity C of all VMs or Capacity of data center

$$C=\sum_{i=1}^{m} Ci$$

Step-4 Calculate the load degree

4.1 Load degree (N)= $\sum_{i=1} \alpha_i F_i$,
Where $\alpha_i F_i$ $\sum \alpha_i = 1$ shows weights that represents different values for or different kinds of jobs.
N represents the current node.

Step-5 Calculate the average cloud partition degree using load degree of node -

$$5.1\ Load\_degree_{avg} = \sum_{i=1}^{n} Load\ degree\ (N_i) / n$$

if Value for Load_degree $_{high}$ is high ,then set different values for different situations based on Load degree $_{avg}$

Step-6 Calculate processing time of a Virtual machine and all Virtual machines.

6.1. Processing time PTi, of all Virtual machine
PTi = Load of all VMs in a data center / Capacity of all VMs

Step-7 Cloud partitioning (End User requests are assign in to queue)-

7.1 While job in the queue
7.2 Search Best Partition (job);
7.3if partition State = = idle ||
partition State == normal, then
7.4 Send Job for Partition; else
7.5 Search other Partition;

Step 8: (Assign priority to job)

8.1 Compute Priority vector for all d matrices using, $A_W = y_{maxw}$
8**.**2 Make a matrix with priority vector using
$P[N] = [W_1, W2...Wi]$, (N=size)
8.3 Check if the Job 'J_i' having highest priority from 'Pi'.
8.4 Assign quantum time to each node.
8.5 Now also check the burst time of the job having highest priority.
If burst time is less then execute the scheduling of the current job or Task with minimum length has higher priority.

Step-9 Check nodes load status levels

9.1 Idle if - Load_degree (N) = 0, no job is being processed by this node so the status is Idle.

9.2 Normal – if $0<Load\_degree(N)<=Load\_degree_{high}$ it can process other jobs.

9.3 Overloaded if - $Load\_degree_{high}<= Load\_degree(N)$ the node is unavailable and cannot receive jobs until it gets normal.

Step-10 Proceed the jobs which are in ready queue

10.1 Submit the list of tasks $T=T_1, T_2…T_n$ by the user.

10.2 Get the available virtual resources from data center. (for i.e., $VM_1, VM_2…VM_m$.)

10.3 Check if (Standard deviation< Threshold time) System load is balanced and Exit

Where $$\Omega = [(1/3* \sum_{i=1}^{m} (PT_i-PT)^2]^{1/2}$$

m: No of Virtual machine V_m

Ω: Standard deviation

PT=Load / Capacity

T_s=Threshold value

10.4(Job Processing)

10.4.1 Select each jobs one by one

10.4.2 proceed it into the VM_s by considering the assign priority and load .

10.5 Compute the fitness value, $\Omega \leq T_s$, where threshold value T_s is in between 0 and 1.

10.6 Based on Fitness value, employed bees update the available source position by-

$$E_{ij}=(X_{ij} * W_{ij})+2*(L_{ij} – 0.5) * (X_{ij}-X_{kj})L_1 + Q_{ij}(X_{ij} – X_{kj}) L_2$$

Where-

$Wij=L_1$= 1 /(1+ exp*(- Fitness(i) / ap))

L_2=1 if bee is onlooker one

L_1, L_2 – are fixed number, 0 or 1

L_2=1 /(1+ exp*(- Fitness(i) / ap)) ,if a bee is employed one

Where -

X_{ij}=nearest neighborhood search solution of employed bees.

W_{ij} =initial weight

X_{kj}=nearest search solution of onlooker bees.

A_p= Fitness value in first iteration

L_{ij}=Random numbers between [0,1] for employed bees.

Qij=Random numbers between [0,1] for onlooker bees.

10.7 Employed bee share his information related to neighborhood position with onlookers bee and also scout bees.

Step- 11 Repeat step 2 to 8 for each iterations unless a best solution is not found.

Step- 12 After allocating all tasks, check the load of the Vms.

12.1 If any virtual machine V_M is overloaded, it goes for next under loaded V_M and assigns the task.

12.2 After completing the each task, repeat the process for all the available jobs/tasks till the system become balanced

Step-13 End

5. EXPERIMENTAL SETUP & IMPLEMENTATION

Cloud Sim Simulator - It is a simulator tool which provides a framework for the simulation and modelling of cloud computing services and infrastructure. We are using Cloud Sim 3.0 tool for simulation with Net-Beans IDE 6.9.This tool provides packages and classes which helps in creating simulation environment of cloud.

Cloud Setup - Create Cloudlets, Data Centers, Virtual Machines and Brokers using the library of CloudSim3.0. Following parameters were used for simulation for Round robin, existing method (Honey bee) and proposed method EDLBHA. And various results are calculated for each method.

Table 1 . Simulation parameters

S.No.	Cloud devices	No. of Used devices	Physical Characteristics
1.	Cloudlets	100-1000	Length=150000bits Per Number = 1 File Size = 300 Mips output Size = 300 Mips
2.	Cloud broker	10-20	NA
3.	Data center	10-100	Max Power = 250 Static Power Percent = 0.7 RAM = 10000 Storage = 1000000 BW = 100000
4.	Virtual machine	20-100	Pes Number = 1 RAM = 128 BW = 2500 size = 2500 Vmm = "Xen"

6. RESULTS AND ANALYSIS

Result Comparison- Following results are calculated for Round Robin method, Existing Honey bee method and proposed EDLBHA techniques.

6.1 Data Transmission Rate

- How much amount of data is transferred in a particular time period. The table shown below is the analysis of the total bits to be transmitted over a certain time of the communication. It clearly shows that proposed method EDLBHA has better data transmission rate than existing methods.

Table 2. Data Transmission Rate

Time (in ms)	RR Method (bits)	Honey bee Method (bits)	Proposed EDLBHA (bits)
100	331	401	477
150	441	521	589
200	557	688	721
250	699	876	989
350	875	998	1105
400	1089	1159	1320

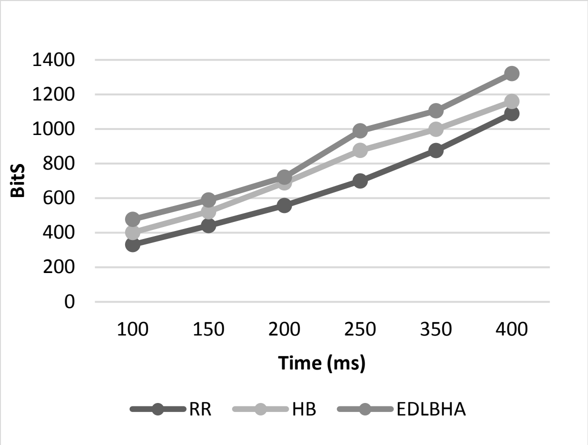

Figure 1. Comparison of Data Transmission Rate

6.2 Response Time

- The amount of time taken from a process submission till the first response received. Less response time shows better efficient performance.

Table 3. Response time

No. of Resources	Response Time(in ms)		
	Round Robin	Honey Bee	Proposed EDLBHA
20	4.14	3.50	2.88
40	16.2	15.4	14.23
60	56.4	50.7	44.2
80	68.44	60.1	34.2
100	74.2	64.4	54.2

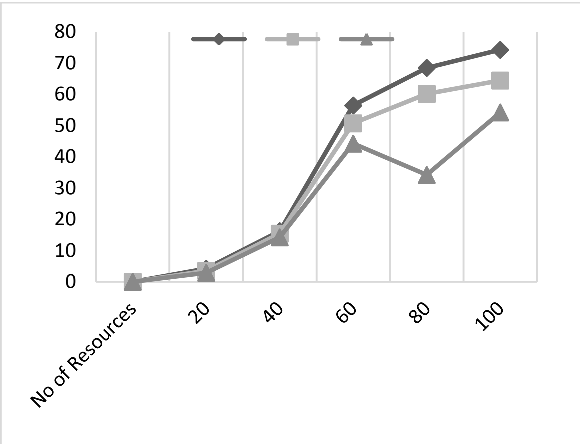

Figure 2. Response time (in ms) RR, HB and proposed EDLBHA

6.3 Waiting time

-It is defined as how long each process has to wait before it gets it's time slice.

Table 4. Waiting time

No.of Server	Waiting Time (in ms)		
	Round Robin	Honey bee	Proposed EDLBHA
10	13.0	10.8	8.4
20	19.77	13.7	11.2
30	54.2	46.5	32.5
40	67.44	40.95	33.4
50	82.1	49.7	38.9

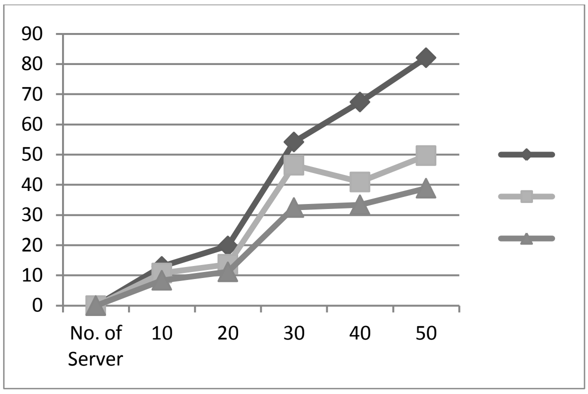

Figure 3. Comparison of Waiting Time for RR , HB and EDLBHA

6.4 Turnaround time

– Time required for completion of a process.

Table 5. Turnaround time

No.of Resources	Turn Around Time(in ms)		
	Round Robin	Honey bee	Proposed EDLBHA
20	50.2	49.7	48.4
40	72.8	68.9	61.2
30	87.7	81.2	75.4
40	73.8	64.5	62.1
50	66.4	58.9	53.33

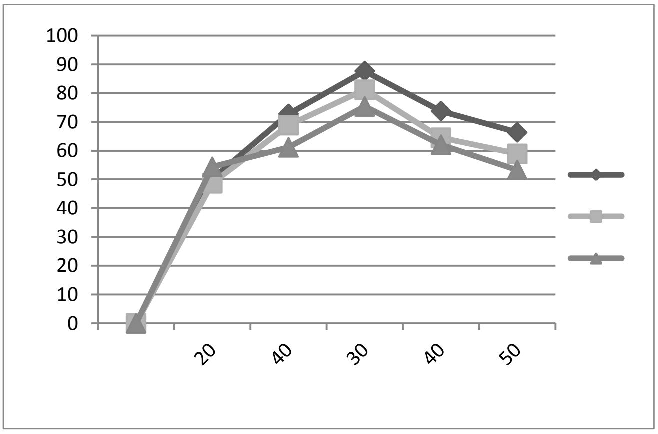

Figure 4. Comparison of Turn Around Time for RR , HB and EDLBHA

Table 6. Overall Response Summary of EDLBHA (Proposed) & Honey Bee (Existing)-

Summary for Proposed method EDLBHA	Average (ms)	Min (ms)	Max (ms)
Overall response Time	300.08	207.06	310.12
Data Center Processing Time	0.34	0.04	0.62

Table 7. Overall response time of Honey Bee

Summary for Existing method Honey Bee	Average (ms)	Min (ms)	Max (ms)
Overall response Time	322.66	207.26	344.22
Data Center Processing Time	0.44	0.10	0.88

6.5 Percentage of Overloaded & Non-Overloaded VMs after applying HBBLB and EDLBHA

Table 8. VM status in % after Load Balancing

HBBLB		EDLBHA	
Non - Overloaded	Overloaded	Non - Overloaded	Overloaded
8020	100	8120	0
80%	20%	100%	0%

7. CONCLUSION AND FUTURE WORK

Cloud Computing offers a very large number of opportunities of using IT infrastructure as a utility with many possibilities like scaling down and scaling up depending upon the needs of the organization. However, similar to most rising technologies cloud computing is also having issues that need to be resolved. This work gives an introduction to cloud computing , challenges that need to be focused in future also covers various load balancing methods, its need. Load Balancing problem is considered as the main problem here. The proposed method EDLBHA overcomes the load balancing problem for cloud computing. The result analysis shows the performance and comparison of the proposed and existing methodology. Simulation results clearly shows that our proposed methodology EDLBHA, having outstanding results in terms of efficient energy utilization, waiting time, response time and turnaround time.

In future further enhancements can be done for the improvement of scheduling in cloud resources in distributed framework or virtualization of data centers, for the capacity management of the cloud computing where the capacity of datacenters can be increased or decreased depends on the request of the users. In future proposed EDLBHA method can be implemented in real time environment instead of a simulator.

8. REFERENCES

[1] Shridhar G.Damanal and G. Ram Mahana Reddy,” Optimal Load Balancing in Cloud Computing By Efficient Utilization of Virtual Machines”,978-1-4799-3635-9/14,IEEE 2014,PP 97-103.

[2] Gulshan Soni, Mala Kalra” A Novel Approach for Load Balancing in Cloud Data Center”, 978-1-4799-2572-8/14, IEEE 2014, PP 807-813.

[3] Subhadra Bose Shaw, Dr. A.K. Singh,” A Survey on Scheduling and Load Balancing Techniques in Cloud Computing Environment”, Published in 5th International Conference on Computer and Communication Technology (ICCCT), 978-1-4799-6758-2/14,IEEE 2014, PP 87-96.

[4] Tushar Desai, Jignesh Prajapati,” A Survey of Various Load Balancing Techniques and Challenges in Cloud Computing”, published in International journal of scientific & technology research volume 2, issue 11, November 2013,PP 158-162.

[5] Soumya Ray and Ajanta De Sarkar,” Execution analysis of load balancing Algorithms in cloud computing Environment”, International Journal on Cloud Computing: Services and Architecture (IJCCSA),Vol.2, No.5, October 2012,pp 78-98.

[6] Ajay Gulati, Ranjeev. K. Chopra,” Dynamic Round Robin for Load Balancing in a Cloud Computing”, IJCSMC, Vol. 2, Issue. 6, June 2013, pp.274 – 278.

[7] Ranjan Kumar , G.Sahoo, “Cloud Computing Simulation Using CloudSim”, International Journal of Engineering Trends and Technology (IJETT) – Volume 8 Number 2- Feb 2014,pp 82-87.

[8] Soumya Ray and Ajanta De Sarkar,” Execution analysis of load balancing Algorithms in cloud computing Environment”, International Journal on Cloud Computing: Services and Architecture (IJCCSA), Vol.2, No.5, October 2012, pp 183-196.

[9] Tushar Desai, Jignesh Prajapati,” A survey of various load balancing techniques and challenges in cloud computing”, International journal of scientific &technology research volume 2, issue 11, November 2013,PP 158-162.

[10] Rakesh Kumar Mishra ,Sreenu Naik Bhukya,” Service Broker Algorithm for Cloud-Analyst”,International Journal of Computer Science and Information Technologies, Vol. 5 (3), 2014,3957-3962.

[11] Ms Jayshri Damodar Pagare, Dr. Nitin A Koli,” Design and simulate cloud computing environment using cloudsim”, Int.J. Computer Technology &Applications, Vol 6 (1),35-42.

[12] Klaithem Al Nuaimi, Nader Mohamed, Mariam Al Nuaimi and Jameela AI-Jaroodi, "A Survey of Load Balancing in Cloud Computing: Challenges and Algorithms", in Second Symposium on Network Cloud Computing and Applications, 978-0-7695-4943-9/12, IEEE 2012.

[13] Zhen Xiao, Weijia Song and Qi Chen, "Dynamic Resource Allocation Using Virtual Machines for Cloud Computing Environment" ,IEEE transactions on parallel and distributed systems, vol. 24, no. 6, june 2013.

[14] Hemant S. Mahalle, Parag R. Kaveri and Vinay Chavan, "Load Balancing On Cloud Data Centres" in International Journal of Advanced Research in Computer Science and Software Engineering,Volume 3, issue 1, January 2013.

[15] Shridhar G. Domanal and G. Ram Mohana Reddy," Load Balancing in Cloud Computing Using Modified Throttled Algorithm",IEEE, International conference. CCEM 2013. In press.

[16] Brototi Mondal, Kousik Dasgupta and Paramartha Dutta, "Load balancing in cloud computing using stochastic hill climbing a soft computing approach" in Procedia Technology 4 (2012) 783 - 789, ELSEVIER C3IT-2012.

[17] B. Wickremasinghe, R.N.Calheiros, R.Buyya, Cloud analyst: A cloudsim-based visual modeller for analysing cloud computing in: Proceedings of the 24th International Conference on Advanced Information Networking and Applications (AINA 2010), Perth, Australia" 2010.

[18] Mr. Manan D. Shah, "Allocation Of Virtual Machines In Cloud Computing Using Load Balancing Algorithm" in International Journal of Computer Science and Information Technology & Security (lJCSlTS), ISSN: 2249-9555 Vol. 3, No.1, February 2013.

[19] Ms. Nitika, Ms. Shaveta, Mr. Gaurav Raj, "Comparative Analysis of Load Balancing Algorithms in Cloud Computing" in International Journal of Advanced Research in Computer Engineering & Technology Volume 1, Issue 3, May 2012.

[20] M. Armbrust et aI., "Above the Clouds: A Berkeley View of Cloud Computing," technical report, Univ. of California, Berkeley, Feb. 2009.

[21] Hung-Chang Hsiao, Hsueh-Yi Chung, Haiying Shen, and Yu Chang Chao, "Load Rebalancing for Distributed File Systems in Clouds" IEEE transactions on parallel and distributed systems, vol. 24, no. 5, May 2013.