Performance Optimization in Cloud Computing Through Cloud Partitioning-Based Load Balancing

Sonam Srivastava and Sarvpal Singh

Abstract Cloud computing is one of the today's largest hearing fields and exciting technologies, because it is flexible and scalable, also it reduces the cost and complexity of applications. Consequently, arises the concept of large scale computing where there is geographical distribution of data centres and end users all across the globe. Thus, it becomes challenging issue for these data centres how to efficiently handle the huge number of incoming requests from different user bases. So, evaluation of performance of the cloud and its proper analysis is of utmost importance, so that the users can take right decisions. One of the very important issues in association with this domain is that of load balancing. The major object of algorithms concerned with balancing of load is how to efficiently designate the task to nodes, such that the overall response time of request is reduced and processing of request is brought about effectively. Thus, in this paper we have integrated the concept of Cloud partitioning along with the central load balancing algorithm so as to balance the load effectively. Due to the existence of partitions in the system, it is possible to apply good load balancing strategy and use optimal algorithms based on the state of the partition. We have carried out the implementation and simulation of our proposed work on CloudSim simulator. Thus, the task execution is done effectively and the results are better for the proposed modified central load balancing algorithm as compared to previous algorithm in large scale cloud computing environment.

Keywords Cloud computing · Load balancing · Data centre · Virtual machine

S. Srivastava
Department of Computer Science and Engineering, Madan Mohan Malaviya University of Technology, Gorakhpur, Uttar Pradesh, India
e-mail: sonamsrivastava114@gmail.com

S. Singh (✉)
MMMUT, Gorakhpur, Uttar Pradesh, India
e-mail: spsingh@mmmut.ac.in

S.K. Bhatia et al. (eds.), *Advances in Computer and Computational Sciences*, Advances in Intelligent Systems and Computing 554,
https://doi.org/10.1007/978-981-10-3773-3_30

1 Introduction

Cloud computing basically refers to various services and applications that are delivered from distinguished data centres throughout the world. Such application services are delivered via Internet. Three types of services that are rendered by cloud computing are IaaS, PaaS and SaaS [1]. Cloud computing is a rising field and due to its emerging status there are a vast number of organisations and deluge of enterprises that are drifting towards using this environment [2]. Thus, comes into picture the concept of large scale computing where there is vast number of user bases and data centres geographically distributed all across the world. Thus, it becomes a great challenge for these data centres to effectively handle the huge number of incoming requests from different user bases. So, evaluation of performance of the cloud and its proper analysis is of utmost importance, so that the users can take appropriate decisions. The key technology in support of cloud computing is virtualization. Here the virtual machines are particularly abstract form machine running on physical machines. Since, the tasks that are co-located do not intervene with each other and ingress only their own data, it feels to the user as working in a completely isolated environment. There are several factors that can be effective on cloud performance. As far as cloud environment is concerned there are several techniques to take care of the huge services along with the various operations that are performed upon them. One of the very important issues in association with this domain is that of load balancing. There exist distinctive algorithms for balancing of load in distinctive environments. The major object of algorithms concerned with balancing of load is how to efficiently designate the task to nodes, such that the overall response time of request is reduced and processing of request is brought about effectively. Although virtualization has made remarkable attempts for balancing the load of the entire system there still occur a possibility of the underutilisation or overutilisation of resources [3]. Underutilisation of server results in the increase in the power consumption of entire system thereby raising the operational cost and is also not environment friendly. Overloaded servers bring about degradation of performance. Due to imbalance in load distribution excess heat would be produced by heavily loaded servers which would affect the cooling system cost. So, it is mandatory to properly balance the load on different servers by choosing appropriate strategy of assigning the incoming requests from end users to existing VMs. The performance of cloud scenario depends largely on execution time of tasks. The observed limitation of the existing central load balancing algorithm considered here in this thesis is that it does not state any remedy to the fact that when the status of all the VMs is busy what can be done instead of placing all the cloudlets in the queue one by one and then allocating the cloudlets when any one of them is available. So, this would take a lot of time for execution. Our major objective is to design and develop a modified central load balancing algorithm that ameliorates the identified problem and optimises the performance based on the parameter values of the overall response time.

2 Load Balancing

Load balancing in cloud is concerned primarily with job of distributing workloads across various computing resources. The very prominent significance of balancing load in cloud computing is the resource availability is ameliorated and also the document management systems are benefitted with reduced costs. Through load balancing in cloud the transfer of loads can be brought about at a global level. Load balancing strategically helps resources and networks by providing reduced response time and hike in throughput [4, 5]. It also divides the traffic between various servers so that no delay is incurred while sending and receiving the data. It fairly allocates the resources to a computer resulting in ultimate user satisfaction and appropriate resource utilisation which consequently minimises resource consumption and minimises bottleneck problem.

A. *Types of Load Balancing*

The two broad classifications of load balancing algorithms are static and dynamic load balancing algorithms.

Static load balancing algorithms: The allocation of tasks to processors is done during compile time that is prior to program execution. It is done based on processing capabilities of machines. The static scheduling algorithms are non-pre-emptive and their major objective is to reduce overall execution time. They have nothing to do with dynamic modifications at run time.

Dynamic load balancing algorithms: They deal with redistributing processes amongst the processors during the run time of execution. It dynamically re-designates the load amongst machines by collecting information, knowing the run-time conditions and gathered characteristics. The communication dropdown associated with the task is the main reason behind the run time overhead in balancing load.

The dynamic load balancing further has two classifications **centralised** and **distributed**. If the single node is responsible for handling the distribution within the entire system (centralised). On the other hand if work involved in making decisions is distributed amongst different processors (distributed). Although centralised approach is simple, it involves single point of failure and bottleneck problems.

Distributed algorithms are free from it. Distributed dynamic scheduling can again be either **cooperative** or **non-cooperative**. In the former, every processor is responsible for carrying out its own part of task scheduling whereas in the latter one the independent processors independently arrive at a decision of their resource usage, without affecting the rest of the system.

3 Literature Review

Sidhu et al. [6] have discussed the Round Robin strategy in this paper. Here the data centre controller handles all the requests and allocates them to the Virtual machines in rotating manner. The very first request is assigned to randomly picked

Virtual machine. Subsequently, the requests are allocated in the circular fashion. Even though the authors distribute the load of work amongst processors equally but still the processing time of different processes is not identical. Although this approach does the equal distribution of load, there appears the heavy loading on certain virtual machines whereas on the other hand remaining is idle.

Zhang et al. [7] have discussed the Weighed Round Robin strategy in this paper. It is the updated Round Robin version in which every virtual machine is assigned weight according to their processing capacity. In this manner the powerful virtual machine is assigned more requests. In such a situation the two requests are assigned by the controller to the one which is the powerful virtual machine in comparison to one request assigned to the weaker one. It provides better resource utilisation but at the same time it does pay heed to the time of processing each request.

Ahmad et al. [8] have proposed an ESCEW algorithm which scans the queue of jobs and virtual machine list. In this algorithm any available VM is allocated a request which can be handled by it. If the VM is overloaded then it is the responsibility of the balancer to distribute some tasks to the idle VM. This would ameliorate the response time and the overall time of processing tasks.

James et al. [9] gave a Throttled algorithm of balancing load that tries to evenly scatter the load amongst several virtual machines. When the request arrives, the status of virtual machines is consulted from recorded list, if it is idle then the request is accepted otherwise it returns -1 and request gets queued up.

Lua et al. [10] propose a Join-Idle-Queue distributed algorithm for large systems. Here, firstly idle processors are load balanced across dispatchers and then decrease average length of queue at each processor jobs are designated to processors. Thus, JIQ algorithm does not incur any kind of communication overhead between processors and dispatchers at job arrivals.

Wang et al. [11] have proposed a two phase algorithm of balancing load by amalgamating the two phase load balancing algorithms namely opportunistic load balancing algorithm and load balancing min-min algorithm to have better execution efficiency and load of the system is well maintained. The phases of the algorithm are that the job is allocated to the service manager by the Opportunistic Load Balancing scheduling manager in the first phase. Service manager uses the Load balancing min-min to select the appropriate service node for executing the subtask in the second phase. However, the algorithm is only applicable in case of static environment.

Ghafari et al. [12] propose a decentralised nature inspired honey-bee-based load balancing technique for self-organisation which works best under the availability of heterogeneous resources. Local server action is something through which it achieves the global balancing of load. The approach is similar and adopted from real life behaviour of honey bees for harvesting their food and foraging. In the algorithm the servers are set certain virtual servers where each of it has its own virtual server queues. As per the approach as specified and implemented in the paper works very well as far as the resources under consideration are heterogeneous. The problem related to this approach is that it does not improve throughput equally with the increase in the resources.

Zhang et al. [13] have proposed an ant colony and complex network theory-based mechanism of balancing load in open federation of cloud computing. It enhances numerous aspects of ant colony related algorithms which dealt with only balancing loads in distributed environment.

Ren et al. [14] have proposed a Fractal algorithm of balancing load. It is dynamic in nature and assures that the unnecessary migrations are not triggered by the peak instantaneous load. Timing of virtual machine migration is decided by the load forecasting method. The results in the paper demonstrate that the proposed achieves considerably better load balancing. The comparison of the algorithm has been done with existing honey-bee and ant-colony algorithms. The authors conduct the testing of the existing algorithm and experimentally conclude that the algorithms upgrades system performance and achieve load balancing.

Achar et al. [15] have proposed a weighted least connection algorithm that assigns the load to the nodes with the minimum that is the least number of connections. Although it efficiently balances the load, it still does not consider the storage capacity and processing speed.

Randles et al. [16] have stated that the complexity and scalability of the systems brings about the infeasibility in the assignment of jobs to specific servers in a centralised manner. Thus, the paper examines the three possible solutions proposed for balancing load. So, the authors introduce a self aggregation using the concept of active clustering. It works on the tenet of aggregating together similar nodes and then working on them by efficient load balancing. The limitation with this algorithm is that it works poorly in heterogeneous environment.

4 Proposed Approach for Load Balancing

We have adopted cloud partitioning strategy to balance the load. The load balancing then starts, immediately after the cloud partitioning has been done. Once the job arrives at the appropriate system, it is the responsibility of the main controller to decide which job will be moved to which cloud partition. The load balancer of that partition then decides as to how the job has to be assigned to the nodes. If the load status of a cloud partition is detected as normal, this partitioning can be achieved locally. If the load status of cloud partition is not normal, this job should be transferred to another partition. In our proposed approach each and every cloudlet will pass through different stages. And ultimately each cloudlet will be assigned to the suitable virtual machine.

- Cloudlet to broker.
- From broker to the most suitable data centre.
- From data centre to its controller.

- From controller to best partition.
- Finally cloudlet is assigned to the virtual machine.

Figure 1 shown is the diagrammatic representation of how our proposed approach works. In our proposed approach the tasks or the requests or jobs from the users firstly arrive at a broker, from there the requests are forwarded to appropriate data centre. The data centre has a data centre controller which decides on which partition the requests will be sent to. Every data centre is divided into partitions and each partition has a number of virtual machines. The activities of each partition are handled by its own partition balancer. Thus, the Data centre controller sends the request to particular partition on the basis of their status. If the status of first partition is normal the request is assigned to the available node under that partition otherwise it checks for the next partition and repeats the process for the next partition.

A. *Proposed Algorithm*
The flowchart of the proposed algorithm has been shown in Fig. 2. The step-wise description of the proposed algorithm is given below

1. User sent a request.
2. All user requests are collected at the point of the broker and broker sends these requests to the Data centre controller.
3. A table is maintained with the central load balancer that contains id of virtual machine (VMid), availability of status of states like (BUSY or AVAILABLE) and VMs priority.

These states are completely dependent on the cloud partition, like if one cloud partition is normal then other two partitions are idle and overloaded, we have set this condition inside the cloud. Initially, all Virtual Machines are in available state.

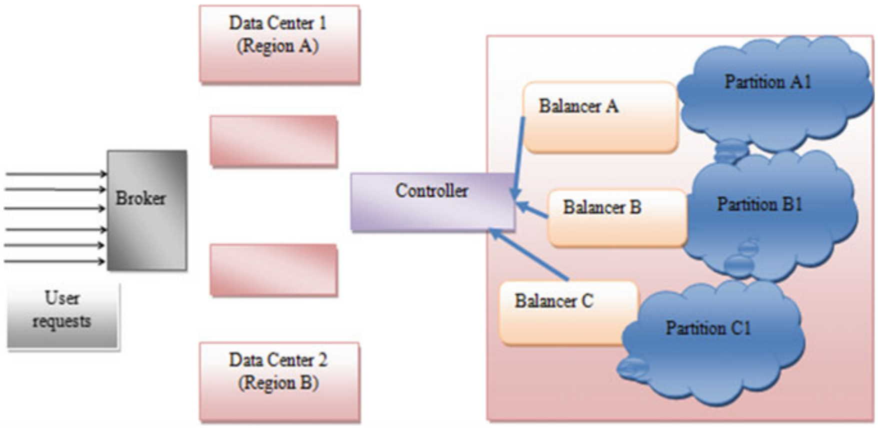

Fig. 1 Working of proposed algorithm

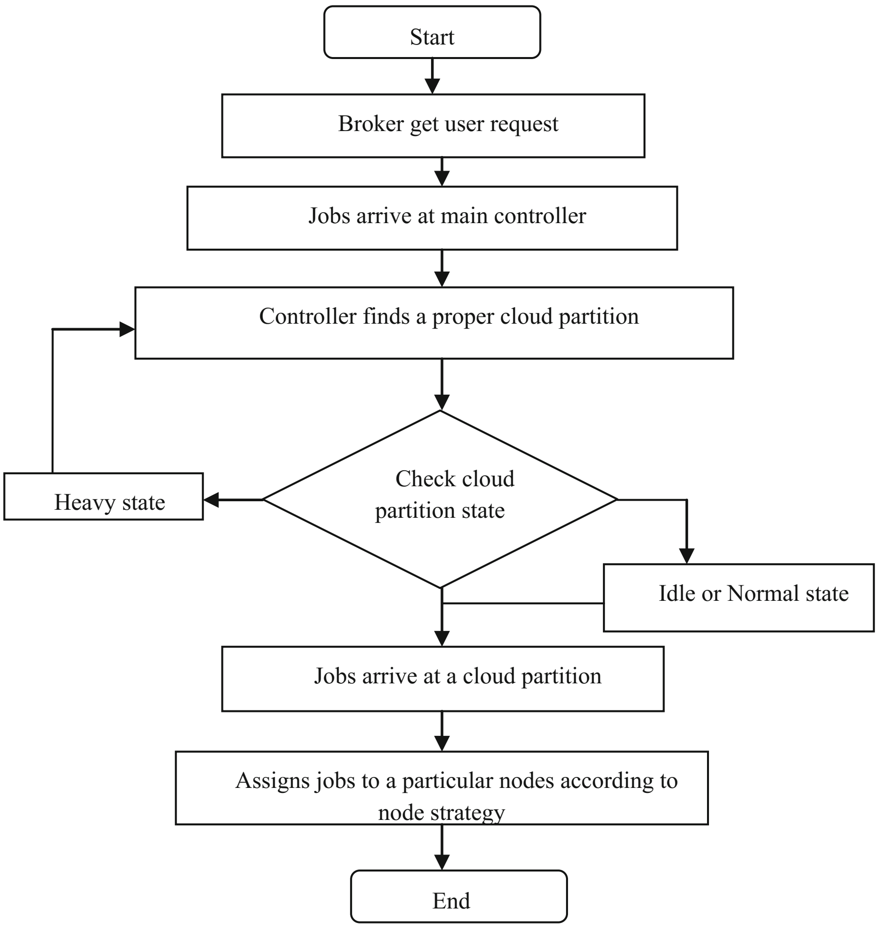
Start
Broker get user request
Jobs arrive at main controller
Controller finds a proper cloud partition
Check cloud partition state
Heavy state
Idle or Normal state
Jobs arrive at a cloud partition
Assigns jobs to a particular nodes according to node strategy
End

Fig. 2 Proposed algorithm

4. Data Centre Controller receives the new request from the broker.
5. Data Centre Controller finds a proper cloud partition for next allocations, every cloud partition has its own balancer.
6. Partition Balancer parses the table from top in order to find the highest priority partition virtual machine and the state of that cloud partitions is available.
   If found:
   (a) The Partition Balancer returns the VMid to the Data Centre Controller.
   (b) The Data Centre Controller forwards the request to the VM identified by that VMid.
   (c) Data Centre Controller notifies the Partition Balancer of newer allocation. Central Load Balancer refreshes the table accordingly.

(d) Partition Balancer refreshes the table accordingly.

If not found:

(a) The Partition Balancer not accepting request and process again back to step 5, that means partition states is heavy.
(b) The Data Centre Controller queues the again request to find the proper partitions.

7. When the partition finishes the processing of requests, and Data Centre Controller receives the response cloudlet, it notifies the Partition Balancer of the VM de-allocation.
8. The Data Centre Controller checks if there are any waiting requests in the queue. If there are, it continues from step 5.
9. End the process.

5 Performance Evaluation and Results

A. *System Requirements*

The proposed algorithm is implemented in CloudSim simulator. It is used for modelling and analysis of large scale cloud computing environment. To set up the simulation environment we need Java development kit which is JDK 1.6 and Eclipse IDE. The language used in the Eclipse IDE is Java. The database used here is Microsoft access. With the proposed algorithm the database connectivity is done with the system and window 7 is used.

B. *Performance analysis*

The certain metrics of performance have to be considered while balancing load in cloud environment. The scalability, utilisation of resources, response time, etc. are certain parameters on the basis of which performance can be computed and analysed. Here, in our work we have seen that the performance of our proposed approach is better than the existing algorithm. The considered existing algorithm is central load balancer [4]. We have modified the existing algorithm by introducing the concept of partitioning [5]. The algorithm in the considered base paper is compared with Round Robin and Throttled. Round Robin and Throttled algorithms give good performance with smaller size requests but when huge data is arriving for allocation the central load balancer would give better results. In addition to this if very large scale computing is to be handled, that can now be done with the help of modified central load balancing algorithm.

Table 1 shown above gives the parametric considerations that have been taken for evaluating the overall performance of modified central load balancer as compared to central load balancer. Figure 3 clearly portrays the overall response time

Table 1 Parametric considerations

S. No.	Parameters	Central load balancer	Modified central load balancer
1	Data centres	2	2
2	Service broker policy	Closest data centre	Closest data centre
3	User base	2	2
4	Request per user per hour	60	60
5	VM	5	5
6	Average RT (ms)	0.35	0.25
7	Minimum RT (ms)	0.01	0.01
8	Maximum RT (ms)	0.62	0.48

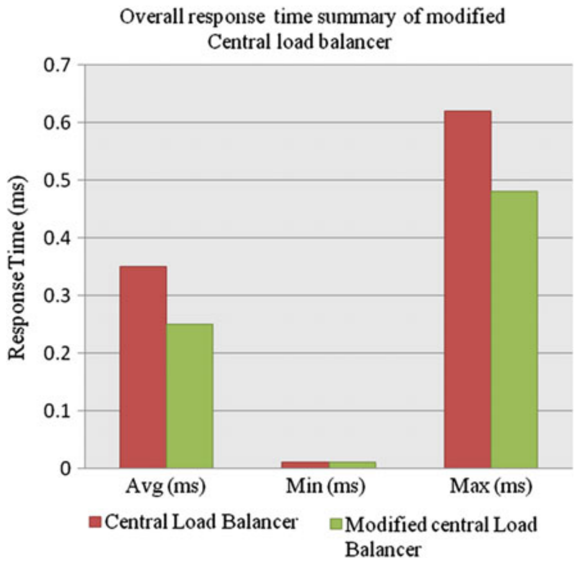

Fig. 3 Comparison of two algorithms on the basis of closest data centre policy

summary of modified central load balancer and thus from these results we conclude that the performance is better optimised with the help of our algorithm.

6 Conclusion

The performance of a cloud computing system depends on the task execution time. In our proposed approach we have integrated the concept of cloud partitioning with central load balancing algorithm to achieve efficient load balancing. In the proposed algorithm prior to the allocation of request to virtual machine the request has to move to suitable partition which is decided by the data centre controller.

Thus, the future work is to develop an enhanced load balancing algorithm considering the fact that existence of partitions in the system, it is possible to apply good load balancing strategy and use optimal algorithms based on the state of the partition. Thus, the task execution is done effectively.

Acknowledgements The completion of this paper would be incomplete unless we express our deep gratitude to the authors whose work we have taken for reference. Their papers and books have provided us with constant guidance and encouragement without which the successful completion of this paper could not have been made possible.

References

1. Shyam Patidar, Dheeraj Rane and Pritesh Jain "A survey paper on cloud computing" 2012 Second International Conference on Advanced Computing and Communication Technologies on 7–8 January 2012 pp. 394-398 ISBN: 978-1-4673-0471-9 doi:10.1109/ACCT.2012.15 2012 IEEE.
2. Niloofar Khanghahi and Reza Ravanmehr "Cloud computing performance evaluation: Issues and challenges" International Journal on Cloud Computing: Services and Architecture (IJCCSA) October 2013 Volume 3 Number 5 doi:10.5121/ijccsa.2013.3503.
3. Drnitry Drutskoy, Eric Keller and Jennifer Rexford "Scalable Network Virtualization in software-defined networks" Internet Computing IEEE on 27 November 2012 pp. 20–27 Volume 17 Issue 2 ISSN: 1089-7801 doi:10.1109/MIC.2012.144 IEEE.
4. Soni, Gaurav, and Mala Kalra. "A novel approach for load balancing in cloud data centre." 2014 IEEE International Advance Computing Conference (IACC), on 21–22 February 2014 pp. 807–812 ISBN: 978-1-4799-2571-1 doi:10.1109/IAdCC.2014.6779427 IEEE.
5. Antony Thomas and Krishnalal G, "A Novel Approach of Load Balancing Strategy in Cloud Computing" International Journal of Innovative Research in Science, Engineering and Technology on 16–18 July 2014 ISSN: 2319-8753.
6. Amandeep Kaur Sidhu and Supriya Kinger, "Analysis of load balancing techniques in cloud computing" International Journal of Computers and Technology March-April 2013 Volume 4 Number 2, ISSN 2277-3061.
7. Qi Zhang, Lu Cheng and Raouf Boutaba, "Cloud computing:State of art and research challenges" Journal of Internet Services and applications May 2010 Volume 1 Issue 1 pp. 7–18.
8. Tanvee Ahmad and Yogendra Singh, "Analytic study of load balancing techniques using tool cloud analyst" International Journal of Engineering Research and Applications (IJERA) March-April 2012 pp. 1027–1030 Volume 2 Issue 2 ISSN: 2248-9622.
9. Jasmin James and Dr. Bhupendra Verma, "Efficient VM load balancing Algorithm for a cloud computing environment" International Journal on Computer Science and Engineering September 2012 Volume 4 Issue 9 pp. 1658–1663.
10. Y. Lua, Q. Xiea, G. Kliotb, A. Gellerb, J.R. Larusb and A. Greenber, "Join-Idle-Queue: A novel load balancing algorithm for dynamically scalable web services" Performance Evaluation November 2011 Volume 68 Issue 11 pp. 1056–1071 doi:10.1016/j.peva.2011.07.015.
11. Shu-Ching Wang, Kuo-Qin Yan, Wen-Pin Liao and Shun-Sheng Wang, "Towards a load balancing in a three level cloud computing network" 2010 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT) on 9–11 July 2010 pp. 108-113 ISBN: 978-1-4244-5537-9 doi:10.1109/ICCSIT.2010.5563889 IEEE.

12. Seyed Mohssen Ghafari, Mahdi Fazeli, Ahmad Patooghy and Leila Rickhtechi, “Bee-MMT: A load balancing method for power consumption management in cloud computing” 2013 Sixth International Conference on Contemporary Computing on 8–10 August 2013 pp. 76-80 ISBN: 978-1-4799-0190-6 doi:10.1109/IC3.2013.6612165 IEEE.
13. Z. Zhang and Xu Zhang, “A load balancing mechanism based on ant-colony and complex network theory in open cloud computing federation” 2010 second International Conference on Industrial Mechatronics and Automation (ICIMA) on 30–31 May 2010 pp. 240–243 ISBN: 978-1-4244-7653-4 doi:10.1109/ICINDMA.2010.5538385 IEEE.
14. Haozheng Ren, Yihua Lan and Chao Yin, “The load balancing algorithm in cloud computing environment” 2012 Second International Conference on Computer Science and Network Technology (ICCSNT) on 29–31 December 2012 pp. 925–928 ISBN: 978-1-4673-2963-7 doi:10.1109/ICCSNT.2012.65226078 IEEE.
15. Raghavendra Achar, P. Santhi Silagam, Nihal Soans, P. V. Vikyath, Sathvik Rao and Vijeth A. M., “Load balancing in cloud based and on live migration of virtual machines” 2013 Annual IEEE India Conference (INDICON) on 13–15 December 2013 pp. 1–3 ISBN: 978-1-4799-2274-1 doi:10.1109/INDICON.2013.6726147 IEEE.
16. Martin Randles, David Lamb, A. Taleb Bendiab, “A comparative study into distributed load balancing algorithms for cloud computing” 2010 IEEE 24th International Conference on Advanced Information Networking and Application Workshop (WAINA) on 20–23 April 2010 pp. 551–556 ISBN: 978-1-4244-6701-3 doi:10.1109/WAINA.2010.85 IEEE.