The Impact On Load-Balancing In Cloud Computing

R.GowriPrakash, R.Shankar, S.Duraisamy

Abstract: Cloud computing is a current model for accessing services via web. In such model, resources are circulated to clients in all around the globe for accessing services more rapidly. This model has several difficulties such as load-balancing, safety measures, resource scheduling scaling, Quality of Service (QoS) control, service accessibility and data center energy utilization. Among these, one of the most fundamental difficulties is load-balancing. This is a practice of allocating and re-allocating the load among the accessible resources for maximizing the throughput when reducing response time, energy utilization, resource consumption and cost. Therefore, an efficient load-balancing scheme is needed to improving the performance of cloud computing. Several load-balancing algorithms in cloud computing have been proposed by different researchers in the past years. In this paper, some of them are surveyed with those merits and demerits to further enhance the load-balancing in cloud using recent algorithms.

Keywords: Cloud computing, load-balancing, resource scheduling, resource utilization.

————————————◆————————————

1. INTRODUCTION

In modern network models, the cloud computing framework [1] is viewed as an incomparable development due to the volatile utilization of web and progression of communication technology. It offers software and hardware together as resources over a web for the cloud user. Typically, it is a web-based computing framework that allocates data, resources and services to different tools of the user on demand. Effective and scalable characteristics of this model can achieve by sustaining appropriate cloud resource management. The most vital characteristic of this model is virtual cloud resources. The services are provided to the users by the Cloud Service Provider (CSP); but this is more difficult by using virtual cloud resources. As a result, load-balancing has a crucial impact on the system performance [2]. In cloud environment, load-balancing may occur among physical hosts or Virtual Machines (VMs). The main intention of load-balancing technique is allocating an equal load among all VMs or hosts. The load-balancing algorithm is split into static and dynamic-based algorithms [3]. The static-based algorithms are more suitable for stable environment with homogenous system. The dynamic-based algorithms are more efficient and adaptable in both homogeneous and heterogeneous environment. Mostly, load is defined as the allocation of various tasks to VMs. The following are the load-balancing issues:

- ➢ Task allocation: It refers to the random allocation of a fixed number of tasks into many Physical Machines (PMs) and then VMs with respect to the PMs.
- ➢ VM Relocation Management: It refers to the relocate of VM from one PM to the other PM for increasing the resource consumption of the data center in which PM is overloaded.

In this paper, various load-balancing techniques in cloud are analyzed based on their merits and demerits and compared each techniques in terms of total VM cost, average response time, resource utilization, waiting time, running time, response time, memory usage, CPU usage, makespan, energy utilization, fault tolerant level, performance degradation and communication cost reduction.

2. SURVEY ON LOAD-BALANCING TECHNIUQES IN CLOUD COMPUTING

Kumar & Prashar [4] proposed a bio-inspired hybrid algorithm in which the load-balancing among the cloud nodes was achieved by hybridizing the Ant Colony Optimization (ACO) and priority-based Artificial Bee Colony (ABC) algorithms in which ACO was used for balancing the workload and ABC was used for optimizing the resource scheduling. The tasks were scheduled to the best available virtual resources with a trade-off between time and computational cost. Keshvadi & Faghih [5] proposed a Multi Agent based load-balancing (MA) algorithm for load-balancing in Infrastructure as a Service (IaaS) cloud environment. In order to realize dynamic load-balancing across VMs, the MA algorithm shifted the load in the IaaS architecture and it also maximized the utilization of resources. This algorithm performed both sender initiated and receiver initiated approach for minimizing the waiting time of the tasks and ensuring the Service Level Agreement (SLA). It was comprised of three agents are VM Migration (VMM) agent, Datacenter Monitor (DM) and Negotiator Ant (NA). The loads were monitored by VMM by collecting the bandwidth, CPU and memory consumption of each VM hosted by different tasks. The VMM's information was monitored by DM agent through information policy. DCM agents initiated NA agents. In order to obtain the VMs status, DCM moved to other data centers and communicated with the agents of DCM of those datacenters. It also searched for the desired configuration. Naha & Othman [6] proposed State-Based Load-balancing (SBLB) algorithm to balance load among VMs in cloud. In addition to this, three different cloud brokering algorithm called as Cost Aware (CA), Load Aware (LA), Load Aware Over Cost (LAOC) were proposed. Based on states of VMs, SBLB algorithm retained two different tables. It checked whether each VM in cloud reached a usage threshold. If it so, then that VM was placed in the busy state otherwise it is marked as in the available state. The data center controller passed the user requests to the load balancer and it returned the available VMs from the state table based on the user requests. Simultaneously, the table was updated after the allocation of requests to VMs. If the data center controller does not find any available VMs, then data center controller waited for resource availability. The load balancer reallocates the VMs for another task when the processing is finished in a specific VM. Khani et al. [7] proposed a Distributed and scalable Load-balancing (DLB) mechanism for cloud computing by using game theory. Initially, this mechanism was performed locally in each PM and its current state was detected. Then, a selection process was executed if under-used or over-used states were detected. In this process, the moving VMs were selected. At last, the best host for moving VMs was selected by a migration process. Phi & Hung [8] proposed a VM-level load-balancing algorithm for improving the average response time and average processing time of the system in the cloud computing. In this algorithm, the scheduling policies used were time-share and space-share for VM and tasks. Mainly, this algorithm was proposed by improving the Throttled algorithm based on the research and evaluation of three Max-min algorithm and avoiding congestion in load-balancing algorithm. Mousavi et al. [9] proposed a load-balancing algorithm for resource allocation in cloud computing. This algorithm was based on the hybridization of Teaching-Learning-Based Optimization (TLBO) and Grey Wolves Optimization (GWO) algorithms for maximizing the throughput by using well balanced load across VMs and solving the problem of trap into local optimum. Also, the priorities of tasks were well balanced and load-balancing was efficiently considered based on time, cost for minimizing amount of waiting time of the tasks in the queue. Lawanyashri et al. [10] proposed energy aware hybrid fruitfly optimization technique with stimulated annealing to attain the best optimum solution for load-balancing. Initially, each swarm of flies moved in different directions in uniform manner. Then, integrated simulated annealing is performed to update the current locations and solutions to increase the convergence speed of fruitfly optimization algorithm. Shen [11] proposed Resource Intensity Aware Load-balancing (RIAL) method in cloud for load-balancing. RIAL method assigned different weights to different resources for each PMs based on the resource intensities. Then, these weights were used in selecting VMs to migrate and finding destination PMs in each load-balancing operation. Hence, an overloaded PM migrated out its VMs with low consumption on low intensity resources and high consumption on high intensity resources. Moreover, an extended version of RIAL was proposed with three additional algorithms. First algorithm determined the optimal weight. Second algorithm had a more strict migration triggering algorithm to avoid unnecessary migration. Third algorithm selected the destination PMs in a decentralized manner. Adhikari & Amgoth [12] proposed a new Heuristic-Based Load-balancing Algorithm (HBLBA) for IaaS cloud based on two steps such as server configuration and task-VM mapping. In this algorithm, an efficient strategy was formulated to configure the servers based on the number of incoming tasks and their sizes for discovering the appropriate VMs for assignment and maximizing the utilization of computing resources. In task-VM mapping, a queuing model was adopted via tasks as assigned to the VM for minimizing the waiting time and completion time of the tasks. Huang et al. [13] proposed a Feature Weight Preferences with Fuzzy Clustering method for Load-balancing (FWPFC-LB) in cloud computing containing multi-class system resources and achieving an optimal balancing solution by load data fusion. In this method, feature weight preferences were put forward for establishing the relationship between prior knowledge of specific cloud scenario and load-balancing process. Kumar & Sharma [14] proposed a dynamic scheduling algorithm for balancing the workload among all the virtual machines with elastic resource provisioning and deprovisioning based on the last optimal k-interval. This algorithm can distribute the tasks and add the cloud resource if task rejection ratio was higher than the SLA defined threshold value. The load at each VM and data center was continuously monitored. If any VM was over-loaded, then the under-loaded VM was discovered and the task was transferred from over-loaded VM to under-loaded VM by using the task migration policy. Priya et al. [15] proposed an integrated resource scheduling and load-balancing algorithm for efficient cloud service provisioning. In this algorithm, a Fuzzy-based Multidimensional Resource Scheduling and Queuing Network (F-MRSQN) model was constructed for obtaining the resource scheduling efficiency in cloud computing. Then, the utilization of VM was increased via effective load-balancing by dynamically choosing a request from a class using multidimensional queuing load optimization algorithm. After that, a load-balancing algorithm was implemented for avoiding underutilization and overutilization of resources. Haidri et al. [16] proposed Capacity based Deadline Aware Dynamic Load-balancing (CPDALB) algorithm for load-balancing in heterogeneous nature of the cloud. CPDALB focused on the selection of VM for allocation of tasks to guarantee customer satisfaction in terms of cost of running applications and meeting deadline constraints. It utilized the idea of deploying requests to VMs based on their processing capacities to reduce load imbalance with satisfying the deadline. Kong et al. [17] proposed a fast heuristic algorithm based on the zero imbalance approach for load-balancing in heterogeneous environment. It focused on reducing the completion time difference among heterogeneous VMs without priority methods and complex scheduling decision which often subject the heuristic algorithms to the cloud computing configuration. The fast heuristic algorithm described two constraints are earliest finish time and optimal completion time. It considered the task transfer time onto network bandwidth of VM to achieve load-balancing and task scheduling effectively. Mohanty et al. [18] proposed JAYA algorithm for load-balancing in cloud computing. In JAYA algorithms, tasks were mapped into the population of JAYA algorithm for load-balancing with the consideration of different VMs and the completion time of each task. Based on the response time, the best and worst solutions for load-balancing were selected and the remaining populations were updated based on the best solution. After a certain number of iteration, best solution for load-balancing was achieved. Hsieh & Chiang [19] proposed an efficient duplication strategy called three-phase Dynamic Data Replication Algorithm (DDRA) to reduce the workload and enhance the ability of cloud. It was comprised of three phases. For the first two phases, DDRA was designed to find the suitable service nodes to achieve the balance of workload based on the service nodes workloads. Finally, a dynamic duplication deployment scheme was designed for achieving higher performance and load-balancing between the nodes. Mansouri et al. [20] proposed Fuzzy system and Modified Particle Swarm Optimization (FMPSO) algorithm to enhance load-balancing and cloud throughput. The

FMPSO considered four modified velocity updating methods and roulette wheel selection technique to improve the global search capability of PSO. After that, crossover and mutation operators were utilized to solve local optima problem in PSO. This schema was explored fuzzy inference system for calculation of objective function. Sekaran et al. [21] proposed a meta-heuristic algorithm called dominant firefly algorithm for load-balancing in cloud computing. It solved the imbalance problems in cloud servers which enhanced the experiences of mobile-learning (m-learning) users. In dominant firefly algorithm, the less luminescence called submissive fireflies were moved towards the most intense brightness called dominant fireflies to balance the load in cloud computing. Gamal et al. [22] proposed a hybrid meta-heuristic technique called Osmotic Hybrid artificial Bee and Ant Colony optimization (OH-BAC) for load-balancing in cloud computing. The OH-BAC combined the osmotic behavior with bio-inspired algorithms. The osmotic behavior allowed the automatic deployment of VMs that were migrated through cloud infrastructure. Then, OH-BAC exploited the Ant Colony Optimization (ACO) and Artificial Bee Colony (ABC) to choose the best VM to the most suitable PM. Furthermore, OH-BAC made activation to the most suitable osmotic host among all PMs in the system to reduce power consumption. Table 2.1 illustrates an overview of merits, demerits and performance metrics of above discussed load-balancing techniques.

Table 2.1 *Comparison of load-balancing techniques*

Ref. No.	Methods Used	Merits	Demerits	Performance Metrics

[4]	Hybrid ACO and ABC	The overall response time and processing time were reduced to execute the incoming tasks.	Time consumption was high for multiple simulations were carried out at a same time.	Number of User Base (UB)=14:
				Response time=150ms;
				Processing time=9.7ms;
				Total processing cost=$3.50

[5]	MA	Maximizing resource utilization, Reduce migration cost, Avoid or reduce dynamic migration	Datacenter management ants requires a timer for self obliterating and waiting for data from parent	Response Time (s) Number of tasks 100: MA =21 Makespan MA = 39
[6]	SBLB	SBLB improves average response time	CA requires more processing time	Total VM cost (US$) Cloud Scenario SC3: CA- SBLB=1070 LA-SBLB=1150 LAOC-SBLB=1090 Average Response Time (ms) Cloud Scenario SC3: CA- SBLB=150 LA-SBLB=240 LAOC-SBLB=90
[7]	DLB	Negligible overhead.	Energy utilization was high.	Number of VMs=90k: Total energy utilization=14000kWh; Performance Degradation due to Migration (PDM)=0.1%
[8]	VM-level load-balancing algorithm	Less average execution time and average response time	Security of the load was not considered.	Average execution time=284.65ms; Average response time=1686.467ms
[9]	Hybrid TLBO and GWO	Reduced waiting time and efficient load-balancing.	The performance was not analyzed well.	-Nil-
[10]	Hybrid Fruitfly Optimization	Improves convergence rate, improves optimization accuracy	Threshold value influence the workload assignment	Makespan (s) Hybrid Fruitfly Optimization= 26 Energy utilization (kWh) No. of tasks 400: Hybrid Fruitfly Optimization= 3.1
[11]	RIAL	Fast and constant convergence with fewer migrations	Still needs an improvement in terms of effectiveness and efficiency of load-balancing	Performance Degradation $(\times 10^3)$ No. of VMs 2500: RIAL=0.18 Communication cost reduction Time 8 hours: RIAL=80
[12]	HBLBA	Helpful in utilizing the resources efficiently.	The number of Service Level Agreement (SLA) violations was high.	Number of tasks=200: Number of VMs=15: Makespan=390sec; Average waiting time=380sec; Average VM utilization=0.72%; Average CPU utilization=56%

[13]	FWPFC-LB	Better load-balancing and scalability.	It requires adaptive parameter optimization in load data fusion for improving the performance.	Number of tasks=100:
				Migration time=3.6sec;
				Makespan=90sec
				Number of computing nodes=50:

				Throughput=750tasks/min
[14]	Dynamic scheduling algorithm	Better elasticity and reduced rejection ratio of task.	It needs to increase the QoS parameters to ensure the high-priority requests.	Number of task=10: Makespan=564sec;
[15]	F-MRSQN	Computational complexity was less.	It requires privacy-aware efficient resource scheduling to increase the privacy for intermediate sharing data and information.	Number of user requests=35: Average success rate=90%; Response time=11.5msec; Resource scheduling efficiency=95%
[16]	CPDALB	Effectively balance loads in heterogeneous nature of the cloud	Load-balancing for a batch of workflow application could be considered	Turn Around Time (for 500 cloudlets) = 0.12×10^4ms Average Response Time = 250ms Deadline Meet = 490 Total Gain =Rs. 3.8×10^4
[17]	Zero balance approach	Used to highly satisfy cloud users and providers requirements	Failed to consider power consumption which has a significant impact on the efficiency on load-balancing	Running Time (for real workload W02) = 574.5sec Speedup (for real workload W02) = 935 Efficiency (for real workload W02) = 0.468
[18]	JAYA	Better response time	Random initialization of population affects the performance of JAYA	Average Response Time (for 50VMs, UB4 user base) = 50.006ms Data center request servicing time of user base (for 50 VMs , DC4 data centers) =1.241
[19]	DDRA, dynamic duplication deployment	Enhance availability, access efficiency and load-balancing	The distribution of replicas is high when the workload is small	Mean average deviation (for 20000 workloads, 100 nodes) = 655 Number of Replicas (@5ms) = 5.8
[20]	FMPSO	Less execution time, less resource usage	Selection proper membership function in FMSPSO is difficult	Total execution time = 733sec Makespan =172sec Degree of imbalance = 23
[21]	Dominant firefly algorithm	Effectively balance the load in multiple cloud server VMs	Findings could be extended to cost computational methods to utilize maximum CPU that would increase server efficiency	Response Time (for 400 tasks) = 24sec Task Migration Time (@400 tasks) = 250sec
[22]	OH-BAC	Improves the energy utilization	OH_BAC has more Service Level Agreement Violations Time per Active Host (SLATAH)	Energy utilization (for 250 tasks) = 81kWh Service Level Agreement Violation (for 250 tasks) = 80% Performance Degradation due to Migration (for 250 tasks) = 10% SLATAH (for 250 tasks) = 84%

3. CONCLUSION

Recently, Cloud computing is increasing rapidly as a successful paradigm in which load-balancing is one of the most significant issues for distributing the dynamic load uniformly among all the nodes to avoid the status that some nodes are over-loaded while others are under-loaded. Various algorithms have been suggested to solve the load-balancing problem in cloud computing. In this article, the merits and limitations of different load-balancing algorithms in cloud computing are focused and represented them in tabular form. On the basis of comparison between different algorithms, it is obvious that OH-BAC optimization algorithm has achieved efficient load-balancing in terms of minimum energy utilization, less number of VMs migration and number of shutdown hosts compared to the other load-balancing algorithms. However, this algorithm has few limitations. As a result, further research would be required to solve these problems of OH-BAC algorithm and increase the efficiency of load-balancing.

REFERENCES

[1] Kherani, F. F., & Vania, J. (2014). Load-balancing in Cloud Computing. International Journal of Engineering Development and Research, 2(1),907-912.

[2] Mehmood, M., Sattar, K., Khan, A. H., & Afzal, M. (2015). Load-balancing approach in cloud computing. Journal of Information Technology & Software Engineering, 5(03), 1-5.

[3] Karimi, A., Zarafshan, F., Jantan, A., Ramli, A. R., & Saripan, M. (2009). A new fuzzy approach for dynamic load-balancing algorithm. arXiv preprint arXiv:0910.0317.

[4] Kumar, R., & Prashar, T. (2016). A bio-inspired hybrid algorithm for effective load-balancing in cloud computing. International Journal of Cloud Computing, 5(3), 218-246.

[5] Keshvadi, S., & Faghih, B. (2016). A multi-agent based load-balancing system in IaaS cloud environment. International Robotics & Automation Journal, 1(1), 3-8.

[6] Naha, R. K., & Othman, M. (2016). Cost-aware service brokering and performance sentient load-balancing algorithms in the cloud. Journal of Network and Computer Applications, 75, 47-57.

[7] Khani, H., Yazdani, N., & Mohammadi, S. (2017). A self-organized load-balancing mechanism for cloud computing. Concurrency and Computation: Practice and Experience, 29(4), e3897.

[8] Phi, N. X., & Hung, T. C. (2017). Load-balancing algorithm to improve response time on cloud computing. International Journal on Cloud Computing: Services and Architecture, 7(6), 1-12.

[9] Mousavi, S., Mosavi, A., & Varkonyi-Koczy, A. R. (2017, September). A load-balancing algorithm for resource allocation in cloud computing. In International Conference on Global Research and Education (pp. 289-296). Springer, Cham.

[10] Lawanyashri, M., Balusamy, B., & Subha, S. (2017). Energy-aware hybrid fruitfly optimization for load-balancing in cloud environments for EHR applications. Informatics in Medicine Unlocked, 8, 42-50.

[11] Shen, H. (2017). RIAL: Resource intensity aware load-balancing in clouds. IEEE Transactions on Cloud Computing, PP(99), 1-14.

[12] Adhikari, M., & Amgoth, T. (2018). Heuristic-based load-balancing algorithm for IaaS cloud. Future Generation Computer Systems, 81, 156-165.

[13] Huang, W., Ma, Z., Dai, X., Xu, M., & Gao, Y. (2018). Fuzzy Clustering with Feature Weight Preferences for Load-balancing in Cloud. International Journal of Software Engineering and Knowledge Engineering, 28(05), 593-617.

[14] Kumar, M., & Sharma, S. C. (2018). Deadline constrained based dynamic load-balancing algorithm with elasticity in cloud environment. Computers & Electrical Engineering, 69, 395-411.

[15] Priya, V., Kumar, C. S., & Kannan, R. (2019). Resource scheduling algorithm with load-balancing for cloud service provisioning. Applied Soft Computing, 76, 416-424.

[16] Haidri, R. A., Katti, C. P., & Saxena, P. C. (2019). Capacity based deadline aware dynamic load-balancing (CPDALB) model in cloud computing environment. International Journal of Computers and Applications, 1-15.

[17] Kong, L., Mapetu, J. P. B., & Chen, Z. (2019). Heuristic Load-balancing Based Zero Imbalance Mechanism in Cloud Computing. Journal of Grid Computing, 1-26.

[18] Mohanty, S., Patra, P. K., Ray, M., & Mohapatra, S. (2019). An Approach for Load-balancing in Cloud Computing Using JAYA Algorithm. International Journal of Information Technology and Web Engineering (IJITWE), 14(1), 27-41.

[19] Hsieh, H. C., & Chiang, M. L. (2019). The Incremental Load Balance Cloud Algorithm by Using Dynamic Data Deployment. Journal of Grid Computing, 1-23.

[20] Mansouri, N., Zade, B. M. H., & Javidi, M. M. (2019). Hybrid task scheduling strategy for cloud computing by modified particle swarm optimization and fuzzy theory. Computers & Industrial Engineering, 130, 597-633.

[21] Sekaran, K., Khan, M. S., Patan, R., Gandomi, A. H., Krishna, P. V., & Kallam, S. (2019). Improving the Response Time of M-Learning and Cloud Computing Environments Using a Dominant Firefly Approach. IEEE Access, 7, 30203-30212.

[22] Gamal, M., Rizk, R., Mahdi, H., & Elnaghi, B. E. (2019). Osmotic Bio-Inspired Load-balancing Algorithm in Cloud Computing. IEEE Access, 7, 42735-42744.