Cloud Computing Load Balancing Techniques: Retrospect and Recommendations

*[1]Oludayo A. Oduwole, [2]Solomon A. Akinboro, [1]Olusegun G. Lala, [3]Michael A. Fayemiwo and [4]Stephen O. Olabiyisi
[1]Department of Computer Science, Adeleke University, Ede, Nigeria
[2] Department of Computer Science, University of Lagos, Lagos, Nigeria
[3]Department of Computer Science, Redeemers University, Ede, Nigeria
[4]Department of Computer Science, Ladoke Akintola University of Technology, Ogbomoso, Nigeria
{dayooduus|akinboro2002}@yahoo.com|{lalagbenga|mfayemiwo}@gmail.com|soolabiyisi@lautech.edu.ng

REVIEW ARTICLE
Received: 17-DEC-2021; Reviewed: 27-JAN-2022; Accepted: 13-MAR-2022
http://dx.doi.org/10.46792/fuoyejet.v7i1.753

Abstract- Load balancing is a research area that seeks to improve the quality of services provided to various clients in cloud computing environments. As cloud users increase around the world, cloud service providers are challenged to develop strategies for distributing tasks to machines for processing at cloud data centres. This work collected and undertook a thorough review of various load balancing techniques, uncovering the key limitations of existing strategies. The publications were chosen from peer-reviewed papers on Google Scholar. Cloud computing, cloud load balancing techniques, approaches to cloud load balancing, and big-data cloud computing systems were among the terms used in the search. Out of 201 studies, 39 met the criteria for inclusion. 5 of the research focused on cloud computing, 6 on cloud load balancing, 7 on resource scheduling in cloud, 16 on techniques for balancing cloud load, and 5 on big-data cloud computing environments. The study identified some research gaps and recommended a throughput-maximization based central-distributive load balancing architecture as a solution to maximize throughput, minimize response time and processing cost, and optimize load balancing architecture.

Keywords- Centralized, cloud-computing, distributive, load-balancing.

1 INTRODUCTION

Due to the obvious services it provides to different users, cloud computing is a well-developed business strategy for distributed data centres. The cloud computing model provides IT tools that are shared, allocated, and accessed by users based on individual demand (Suresh & Sakthivel, 2017; Adhikari & Amgoth, 2018). Furthermore, cloud computing offers a variety of services such as Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), and Infrastructure-as-a-Service (IaaS). These facilities are helpful in different applications, including scientific, business, and industrial applications (Kumar and Sharma, 2018). In summary, cloud computing platform has three severe challenges: virtualization, distributed framework, and load balancing. The distribution of loads to the processing elements is the load balancing problem.

In a multi-node environment, it is very likely that some nodes will be overloaded while others will be idle (Afzal & Kayitha, 2019). Load unbalancing is an unfavourable occurrence for cloud service providers (CSPs), as it reduces the reliability and efficacy of computing services while also jeopardizing the Quality of Service (QoS) promised under the service level agreement (SLA) between the customer and the provider of cloud services. The necessity for load balancing (LB) emerges in these circumstances, and this is a particular research issue of interest (Mishra, Sahoo & Parida, 2018).

*Corresponding Author

Section B- ELECTRICAL/ COMPUTER ENGINEERING & RELATED SCIENCES
Can be cited as:
Oduwole O.A., Akinboro S.A., Lala O.G., Fayemiwo M.A. and Olabiyisi S.O. (2022): "Cloud Computing Load Balancing Techniques: Retrospect and Recommendations", FUOYE Journal of Engineering and Technology (FUOYEJET), 7(1), 17-22. http://dx.doi.org/10.46792/fuoyejet.v7i1.753

Load unbalancing is an unfavourable occurrence for cloud service providers (CSPs), as it reduces the reliability and efficacy of computing services while also jeopardizing the Quality of Service (QoS) promised under the service level agreement (SLA) between the customer and the provider of cloud services. The necessity for load balancing (LB) emerges in these circumstances, and this is a particular research issue of interest (Mishra, Sahoo & Parida, 2018).

Load balancing entails task redistribution in a distributed network, such as cloud computing, so that there are no overworked, under-burdened, or idle computer machines (Achar et al., 2013; Magalhaes et al., 2015). It boosts cloud performance by attempting to improve restricting parameters such as reaction time, processing time, stability of the system, and job transfer (Dam et al., 2015; Dave et al., 2016). Researchers have proposed different approaches to improve quality of cloud computing services and consumption of resources. These include pre-emptive, responsive, mixed, stable and reactive methods (Afzal & Kayitha, 2019).

This paper provides an in-depth investigation of approaches for improving cloud resource utilization through an analysis of load balancing algorithms, and assessment of their strengths and weaknesses. In an attempt to enhance the performance of cloud in terms of throughput, response time, task rejection ratio and CPU utilization rate, attention of researchers is drawn to the invention of strategies that are based on maximization of throughput and rearrangement of spatial node distribution. The second section of this study discusses strategies for achieving load balancing in cloud networks. Section 3 provides a critique of related research, and Section 4 provides the conclusion.

2 An Overview of Techniques for Balancing Load in the Cloud

2.1 Pre-emptive Approach

A pre-emptive Load Balancing algorithm contemplates action by producing changes rather than merely reacting to changes as they happen. Its goal is to achieve a positive outcome by preventing rather than reacting to a problem. Pre-emptive actions seek to identify and capitalize on opportunities, as well as to take precautions against possible future problems and threats. The disadvantage is that only a few classic pre-emptive procedures with no concepts have been implemented (Afzal & Kayitha, 2019).

Polepally & Chatrapati (2017) demonstrated a cloud computing LB technique based on dragonfly optimization and constraint measures that distributes consistent load among VMs while consuming the least amount of power. Peng et al. (2018) proposed an Ant Colony Optimization (ACO) enhancement for achieving balanced distribution of multidimensional resources by introducing the concept of load imbalance degree and PM selection expectation. To decrease predicted response time and retain fairness. Some known pre-emptive load balancers are shown in Table 1.

2.2 Responsive Load Balancing in Cloud Computing

Instead of managing a situation, a responsive method to load balancing responds to it. Load imbalance is addressed as it arises, with noticeable repercussions. The vast majority of load balancers are of this type. The primary fault in existing work is that the issue of load imbalance is allowed to happen before researchers propose methods for solving it by improving some task scheduling parameter(s) (Afzal & Kayitha, 2019). Table 2 shows various existing load balancing techniques that use responsive methodologies. Preventive approaches are preferable to responsive approaches because the former seeks to prevent a problem before it occurs, whereas the latter seeks to solve a problem after it has occurred (Afzal & Kayitha, 2019).

2.3 Static versus Dynamic Methodologies

Load balancers are generally classified as either static or dynamic, such as in Nuaimi *et al.* (2012) and Alakeel (2010). The static balancer ensures that the system parameters required for job allocation are known ahead of time. These include resource requirements, communication time, server processing capacity, memory capacity, and so on (Alexeev *et al.*, 2012). The major downside of this method is that they do not take into account the system's present status when deciding, making them unsuitable for systems such as distributed systems, where the system's states change frequently (Mesbahi & Rahmani, 2016).

Dynamic methods of balancing load consider the present system's status on which they decide. The key benefit of this method is that tasks can be dynamically transferred from an overburdened to an under-loaded node. However, formulating and developing a dynamic load balancer is far more complex and difficult than uncovering a static solution, but we can achieve better performance and have more easy and timely solutions via dynamic mechanisms (Nuaimi *et al.*, 2012; Alakeel, 2010).

There are two types of dynamic load balancing algorithms: distributed and non-distributed. The load balancing procedure can be implemented by all nodes in the system in distributed approaches, as proposed by Shi *et al.* (2011). Furthermore, in this strategy, all nodes are connected with each other to achieve a global objective in the system, which is known as cooperative, or each node can work independently to achieve a local goal, which is known as non-cooperative. However, in a non-distributed scheme, the burden of stabilizing the system workload is not shared by all system nodes. A single node can only implement the load balancing framework between all nodes in a centralized approach in a non-distributed scheme. In semi-distributed mode, the system is divided into partitions or groups, in each of which a single node does load balancing (Mesbahi & Rahmani, 2016).

2.4 Centralized Approach

In this case, all job allocation and scheduling choices are made by a single node (server). This node contains the knowledge base for the entire cloud network. Its main strength is the reduction in time required to investigate various cloud resources, but it places an excessive burden on the centralized server. Other drawbacks are fault intolerance and a low failure recovery rate (Katyal & Mishra, 2013).

2.5 Distributive Approach

In this arrangement, there is no one node responsible for allocating resources or scheduling jobs. Multiple nodes monitor the cloud network to make precise load balancing decisions. Every node maintains a local knowledge base to ensure efficient load distribution. This architecture relieved a single node of a significant failure burden, and as a result, no single node is overburdened with task scheduling judgments, allowing it to be fault tolerant (Tripathi & Singh, 2017; Katyal & Mishra, 2013).

2.6 Hierarchical Cloud Computing Load Balancing

Load balancing decisions are made at different levels of the cloud hierarchy in the layered architecture to cloud load balancing. This strategy works best in a master-slave situation. This technique can be described using a tree data structure, where the parent node obtains information from the child node and uses that information to apply load distribution for the child node under its supervision (Katyal & Mishra, 2013; Dar & Ravindran, 2017). Table 3 classifies some existing cloud load balancers based on node distribution.

http://dx.doi.org/10.46792/fuoyejet.v7i1.753 http://journal.engineering.fuoye.edu.ng/

Table 1. A Review of Pre-emptive Load Balancing in Cloud Computing

Authors	Algorithm Used	Technique Used	Advantages	Limitations
Kumar *et al.*, (2018)	Conventional Non-Classical	Heuristic, Classical Deterministic	• Designed to accommodate large workloads within a specified time frame. • Improves flexibility • Instant scaling of resources • Task rejection ratio is minimized	• Tasks that take longer than the stipulated deadline are rejected. • Thresholds for determining overloaded and under-loaded VM are set arbitrarily because there is no formula for them.
Polepally *et al.* (2017)	Load balancing using Dragonfly optimization and constraint measures	Swarm optimization	• Task scheduling is accomplished while using less energy.	• Tasks that surpass the threshold limit are unable to be completed. • Task rejection rate is quite high.
Xiao *et al.* (2017)	Fairness Aware Algorithm	Non-cooperative game theory-based optimization	• The Nash equilibrium point yields the best load balancing	• Execution time is high
Li *et al.* (2011)	Ant Colony Optimization	Swarm based optimization	• Reduced makespan	• Tasks are distinct from each other.
Peng *et al.* (2018)	Ant Colony Optimization	Swarm based optimization	• Improved resource utilization	• Cost is not considered

Table 2. Review of Responsive Approaches to Cloud Load Balancing

Authors	Algorithm Used	Technique Used	Advantages	Limitation
Vanitha *et al.* (2017)	Genetic Algorithm	Metaheuristic	• Response time, makespan, and task rejection ratio have all been reduced.	• Reduced throughput, scalability, and resource utilization.
Rajput *et al.* (2016)	Genetic Algorithm and Minmin	Evolutionary based Heuristic	• increased scalability • Response time and execution costs were reduced.	• Minimal resource utilization, a lower level of load balance
Kapur (2015)	Non-classical	Heuristic	• High data rates and scalability, with a shorter response and execution time	• Low resource utilization and degree of balance. • High task rejection ratio and migration time
Dam *et al.* (2015)	Genetic Algorithm	Optimization	• Scalability and fault tolerance have been improved. • Response time, power consumption, and migration time are all low.	• A lack of balance, inefficient use of resources, and a high task rejection ratio
Vasudevan *et al.* (2016)	Honey Bee Algorithm	Optimization	• Minimized execution time, response time and execution cost	• Low throughput, low scalability, low degree of balance and resource usage

Table 3. Categorization of some existing cloud load balancers based on node distribution

Authors	Title of Work	Central	Distributive	Hierarchical
Dave and Maheta, 2014	Utilizing round robin concept for load balancing algorithm at virtual machine level in cloud environment,	Yes	No	No
Dasgupta *et al.* (2013)	A Genetic Algorithm (GA) based load balancing strategy for cloud computing.	Yes	No	No
Radojevic and Zagar (2011)	Analysis of issues with load balancing algorithms in hosted (cloud) environments.	Yes	No	No
Dhinesh and Venkata (2013)	Honey bee behaviour inspired load balancing of tasks in cloud computing environments.	No	Yes	No
Wang *et al.*(2010)	Towards a load balancing in a three-level cloud computing network	No	No	Yes
Miglani and Sharma (2019)	Modified Particle Swarm Optimization based upon Task categorization in Cloud Environment	No	Yes	No
Kargar and Yakili (2015)	Load balancing in Map-Reduce on homogeneous and heterogeneous clusters: an in-depth review.	No	Yes	Yes
Riakiotakis *et al.* (2011)	Distributed dynamic load balancing for pipelined computations on heterogeneous systems.	No	Yes	No

http://dx.doi.org/10.46792/fuoyejet.v7i1.753 http://journal.engineering.fuoye.edu.ng/

3 Review of Related Research

Alkayal *et al.* (2016) developed an effective load balancer in a cloud environment based on Cuckoo Search and Firefly Algorithm (CS-FA). The proposed technique essentially prevents workload imbalances by estimating each virtual machine's capacity and load, and allocating tasks to the best machine as determined by the CS-FA algorithm. The CS-FA outperformed existing Hybrid Dynamic LB (HDLB) by migrating a significantly fewer number of tasks, indicating superior load balancing. However, topology optimization via node rearrangement were not taken into account. Various load balancing approaches in different cloud systems were investigated by Mishra *et al.* (2018). A system architecture was provided, along with different models for the host Virtual machine and numerous performance criteria. The method used in calculating the system's makespan and energy consumption was outlined, and a taxonomy for the prevention of imbalance of cloud load was provided.

Deepa et al. (2018) explored cloud computing and its various service categories, deployment models, and architecture. Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) are the 3 key service classes explained in this paper. The cloud architecture's front end and back-end components were examined. Minimal costs, limitless storage, backup and recovery, automatic software integration, easy access to information, and speedy implementation were also identified as benefits, while technical issues, cloud security, and cyber threats were highlighted as downsides. The study has provided sufficient information to alleviate the uncertainty that is often associated with cloud computing terms.

Afzal and Kayitha (2019) evaluated past work on cloud load balancing and discussed its benefits and drawbacks. The literature review followed a wide research strategy that explains how the load unbalancing problem is approached and specifies the methodology, theories, algorithms, approaches, and paradigms that are used. The load unbalancing problem was investigated using the constructive generic framework (CGF) methodology. The study also includes a taxonomy of algorithms that can help future investigators cope efficiently with load unbalancing issues, such as nature-inspired algorithms, machine learning, and mathematically derived algorithms.

Ngharamike et al. (2018) looked at different cloud simulation models for assessing cloud infrastructure before being implemented in the real world. CloudSim, GreenCloud, NetworkCloudSim, iCancloud, CloudAnalyst, MDCSim, EMUSIM, and CloudSched were studied in terms of their retrospect and limitations. In addition, they were compared in terms of the underlying framework, programming language, graphical user interface, availability, cost modelling, and energy modelling. It was discovered that none of the tools could completely model a true cloud environment, and that they were more efficient at describing one aspect of the cloud than the other. GreenCloud spends more time simulating than others, but it is the most ideal for modelling data centre energy use. CloudAnalyst excels at modelling federation policy, cost, and simulation time (response and execution time), while iCancloud excels at large data centre cost and component modelling. CloudSched outperformed others in the analysis of computer hardware utilization by applications, while NetworkCloudSim was the best at portraying network components of cloud centres.

Jayaraj et al. (2019) presented a process optimization of big-data cloud centres using the nature-inspired Firefly Algorithm and K-Means Clustering. The proposed optimization method was compared to state-of-the-art algorithms such as Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), and Ant Colony Optimization (ACO) using response time, throughput, and latency as metrics. The proposed balancer reduces latency, time of response and throughput multiple times, but does not take into account CPU utilization rate, which reveals degree of load balancing reached, and does not consider topology optimization required for disperse nature of big data characterized cloud.

3.1 Peculiar Challenges in Previous Work on Cloud-Based Load Balancing

When cloud systems are designed to handle large volumes of requests from dispersed sources at high transmission rates, mechanisms for achieving load balancing must be improved further. This, according to prior studies can be accomplished by incorporating strategies that maximize throughput while significantly reducing response time. Furthermore, improvements to established methods of balancing load are required to address minimization of processing costs and cloud topology (spatial arrangement of nodes), as previously reported. Previous work emphasized improving response time but does little to reduce processing costs (Aswini *et al.*, 2019).

4 Conclusion and Future Work

In this review, various strategies for achieving effective sharing of cloud load were investigated. Certain constraints, such as the throughput maximization problem, the cost minimization problem, and the cloud architecture optimization problem, have been identified (Castelino *et al.*, 2014; Jayaraj & Abdul-Samath, 2019). These limitations stem from the need to implement cloud task scheduling to meet the severe needs of big data settings. High throughput, low response time, low processing cost, and reorganization of the cloud architecture are all required. Previous research did not pay enough attention to optimizing processing costs and cloud architecture, resulting in a significant research gap that must be filled. To address the identified flaws, a central-distributive framework based on throughput maximization is presented in Figure 1.

http://dx.doi.org/10.46792/fuoyejet.v7i1.753 http://journal.engineering.fuoye.edu.ng/

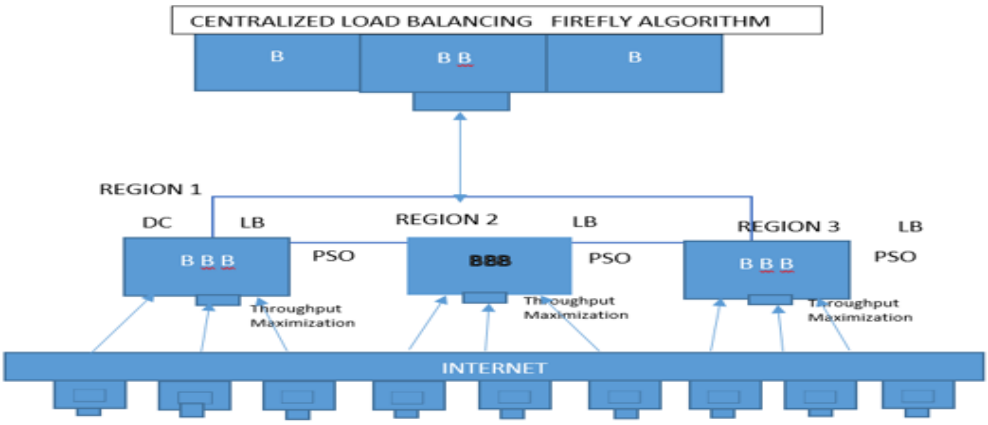

Fig.1: Central-distributive cloud load balancing architecture

The framework's operations are based on the assumptions that there is a central cloud data centre (DC) with up to five regional data centres, and that a user's request will be handled by the data centre in the region from which the request originated. Cloud load will be balanced at two levels by the suggested system: Level 1 load balancing will be done in a dispersed fashion at each DC, whereas Level 2 load balancing will be done by a DC controller in a centralized manner across all DCs. Task requests that match the throughput maximization requirements in their respective regions will be accepted by each DC. The approved tasks will then be separated into two groups: Group A and Group B.

A task will be assigned to Group A if the source and destination nodes are in the same region; otherwise, it will be assigned to Group B. The Group A jobs will be first given to available nodes/servers at their respective DCs, and Level 1 load balancing will be achieved using the Particle Swarm Optimization approach. All of the tasks in Group B must be transmitted to the network's central DC controller for server allocation using the Firefly method across all of the network's available nodes. Because of its ability to find optimal solutions quickly, especially for less complicated optimizations, the PSO algorithm is preferred for regional load balancing. It does so by obtaining its global best solution from local best solutions (Devi & Ryhmend, 2014; Miglani & Sharma, 2019). Because of its high rate of processing jobs, the firefly technique will be used to balance load at the central level (Jayaraj & Samath, 2019; Kumar et al., 2020). This arrangement limits the tasks that must be transferred to those that cannot be handled locally. Hence, response time and costs will be decreased while throughput will be raised as a result of prioritizing the admission of tasks that maximize throughput.

REFERENCES

Achar R., Thilagam, P. S., Soans, N. Vikyath, P. V., Rao, S., Vijeth, A. M. (2013). Load balancing in cloud based on live migration of virtual machines. In: 2013 Annual IEEE India Conference (INDICON), pp. 1–5.

Adhikari, A. & Amgoth, T. (20180. Heuristic-based load-balancing algorithm for IaaS cloud, Future Generation Computer Systems, Vol. 81, pp. 156-165.

Afzal, S. & Kavitha, G. (2019). Load Balancing in Cloud Computing: A Hierarchical Taxonomical Classification. Journal of Cloud Computing: Advances, Systems & Applications. 8(22), pp. 1-24.

Alakeel, A. M. (2010). A guide to dynamic load balancing in distributed computer systems. International Journal of Computer Science and Information Security, 10(6): p. 153-160.

Alexeev, Y., Mahajan, A., Leyffer, S., Eletcher, G. & Fedorov, D. G. (2012). Heuristic static load-balancing algorithm applied to the fragment-molecular orbital method. IEEE International Conference on High Performance Computing, Networking, Storage and Analysis

Alkayal, E. S., Jennings, N. R. & Abulkhair, M. F. (2016). Efficient Task scheduling Multi-Objective Particle Swarm Optimization in Cloud Computing. IEEE 41st Conference on Local Computer Networks Workshops.

Aswini, J., Malarvizhi, N. & Kumanan, T. (2019). A Novel Firefly algorithm based Load Balancing approach for Cloud Computing. International Journal of Innovative Technology and Exploring Engineering (IJITEE). ISSN-2278-3075. 8(2), pp. 91- 96.

Castelino, C., Gandhi, D., Narula, H. G., & Chokshi, N. H. (2014). Integration of Big Data and Clou Computing. International Journal of Engineering Trends and Technology (IJETT), 16(2),100- 102.

Dam, S., Mandal, G., Dasgupta, K, & Dutta P. (2015). Genetic algorithm and gravitational emulation based hybrid load balancing strategy in cloud computing. In; Proceedings of the third international conference on computer, communication, control and information technology C3IT), pp 1–7.

Dar, A. & Ravindran, A. (2018). A Comprehensive Study on Cloud Computing Paradigm. International Journal of Advance Research in Science and Engineering, 7(4), 235-242.

Dave, S. & Maheta, P. (2014). Utilizing round robin concept for load balancing algorithm at virtual machine level in cloud environment. Int J Comput Appl [Internet] 94(4), pp. 23–29.

Dave, A, Patel, B., & Bhatt, G. (2016). Load balancing in cloud computing using optimization techniques: a study. In: International Conference on Communication and Electronics Systems (ICCES), pp 1–6.

Devi, M. & Ryhmend, U. (2014). Particle swarm optimization based node and link lifetime prediction algorithm for route recovery in MANET. Journal of Wireless Communication and Technology, 107(1), 1–10.

Dhinesh, B. L. D., & Venkata, K. P. (2013). Honey bee behavior inspired load balancing of tasks in cloud computing environments Appl Soft Comput J. 13(5), 2292–303.

Jayaraj, T. & Abdul Samath, J. (2019). Process Optimization of Big-Data Cloud Centre Using Nature Inspired Firefly Algorithm and

http://dx.doi.org/10.46792/fuoyejet.v7i1.753 http://journal.engineering.fuoye.edu.ng/

K-Means Clustering. International Journal of Innovative Technology and Exploring Engineering (IJITEE), 8(12), 48-52.

Kapur, R. (2015). A workload balanced approach for resource scheduling in cloud computing. In: eighth international conference on contemporary computing (IC3), pp. 36–41.

Kargar, M. J. & Vakili, M. (2015). Load balancing in Map-Reduce on homogeneous and heterogeneous clusters: an in-depth review. Systems, Vol. 15, pp. 149–168.

Katyal, M. & Mishra, A. (2013). A Comparative Study of Load Balancing Algorithms in Cloud Computing Environment. International Journal of Distributed and Cloud Computing, 1(2), pp. 7-14.

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

Kumar, K. P., Ragunathan, T., Vasumathi, D. & Prasad, P. K. (2020). An Efficient Load Balancing Technique based on Cuckoo Search and Firefly Algorithm in Cloud. International Journal of Intelligent engineering and Systems, 13(3), pp. 422- 432. DOI: 10.22266/IJIES2020.0630.38.

Li, K., Xu, G., Zhao, G., Dongm, Y. & Wang, D. (2011). Cloud task scheduling based on load balancing ant colony optimization. In: Sixth annual China Grid conference, pp. 3–9.

Magalhaes, D., Calheiros, R. N., Buyya, R., & Gomes, D. G. (2015). Workload modelling for resource usage analysis and simulation in cloud computing. Comp Elect Eng, 47:69–81; DOI:10.1016/j.compeleceng.2015.08.016

Mesbahi, M. & Rahmani, A. M. (2016). Load Balancing in Cloud Computing; A State of the Art Survey. I.I. Modern Education and Computer Science, vol. 3, pp. 64-78.

Miglani, N. & Sharma, G. (2019). Modified Particle Swarm Optimization based upon Task categorization in Cloud Environment. International Journal of Engineering and Advanced Technology (IJEAT), 8(4C), 67 – 72.

Mishra, S. K., Sahoo, B., & Parida, P. P. (2018). Load balancing in cloud computing: a big picture. J King Saud Univ Comp Infor Sci: pp. 1–32.

Ngharamike, E., Ijemaru, G., Akinsannmi, O. & Folorunso, O. (2018). Cloud-based Simulation Tools for Cloud Test: A Review. FUOYE Journal of Engineering and Technology, Volume 3, Issue 1, pp.80-85. http://dx.doi.org/10.46792/fuoyejet.v3i1.100

Nuaimi, K. A., Mohamed, N., Nuaimi, M. & Al-Jaroodi, J. (2012). A Survey of Load Balancing in Cloud Computing: Challenges and Algorithms. Second Symposium on Network Cloud Computing and Applications, pp. 137-142, Doi: 10.1109/NCCA.2012.29.

Peng, X. Guimin, H. Zhenhao, L. & Zhongbao, Z. (2018). An efficient load balancing algorithm for virtual machine allocation based on ant colony optimization. International Journal of Distributed Sensor Networks, vol. 14 no 12, pp. 1-9.

Polepally, V. & Chatrapati, K. S. (2017). Dragonfly optimization and constraint measure-based load balancing in cloud computing. Cluster Comp, pp.1–13.

Radojevic, B. & Zagar, M. (2011). Analysis of issues with load balancing algorithms in hosted cloud environments. Proc 34th Int Conv MIPRO pp. 416–420.

Rajput, S. S. & Kushwah, V. S. (2016). A genetic based improved load balanced min-min task scheduling algorithm for load balancing in cloud computing. In: 8th international conference on Computational Intelligence and Communication Networks (CICN), pp. 677–681.

Riakiotakis, I., Clorba, F. M., Androniko, T. & Papakonstantinou, G. (2011). Distributed dynamic load balancing for pipelined computations on heterogeneous systems. Parallel Computing, 37(10): pp. 713-729.

Shi, J., Meng, C. & Ma, L. (2011). The Strategy of Distributed Load Balancing Based on Hybrid Scheduling. Fourth International Joint Conference on Computational Sciences and Optimization, pp. 268-271, Doi: 10.1109/CSO.2011.286.

Suresh, S. & Sakthivel, S. (2017). A novel performance constrained power management framework for cloud computing using an adaptive node scaling approach, Computers & Electrical Engineering, Vol. 60, pp. 30-44.

Tripathi, A. M. & Singh, S. (2017). A literature review on algorithms for the load balancing in cloud computing environments and their future trends. Computer Modelling & New Technologies, 21(1), 64-73.

Vanitha, M. & Marikkannu, P. (2017). Effective resource utilization in cloud environment through a dynamic well-organized load balancing algorithm for virtual machines. Comp Elec Eng, vol. 57, pp. 199–208.

Vasudevan, S. K., Anandaram, S., Menon, A. J & Aravinth, A. (2016). A novel improved honeybee based load balancing technique in cloud computing environment. Asian J Infor Technol, 15(9), pp. 1425–1430.

Wang, S. C., Yan, K. Q., Liao, W. P., & Wang, S. S. (2010). Towards a load balancing in a three-level cloud computing network. In Proceedings: 2010 3rd IEEE International Conference on Computer Science and Information Technology, ICCSIT pp. 108–113.

Xiao, Z., Tong, Z., Li, K. & Li, K. (2017). Learning non-cooperative game for load balancing under self-interested distributed environment. Appl Soft Comput. vol. 52, pp. 376–386